Treball de Fi de Grau

GRAU D'ENGINYERIA INFORMÀTICA

Facultat de Matemàtiques
Universitat de Barcelona

DSLR Project
Control de càmeres DSLR en temps real mitjançant dispositius tipus RaspberryPi

---

Ruben Serrano Oller

Director: José Bosch Estrada
Facultat d'Electrònica
Universitat de Barcelona

Índex

Abstract

This project was born in the hands of José Bosch Estrada, teacher of University of Barcelona of Electronic Department. The choice of this project was motivated to have different layers of infrastructure to accomplish the target and acquire knowledge in photography. The project has a powerful target with the characteristic to be unique and to make easy the work of users with few technology knowledge. A further more It will be integrated with another projects to increase usefulness, effectiveness and remote manipulation of a DSLR camera.

The first target is the remote control of a DSLR camera to change any parameter and the manipulation of the camera is avoided. The distance of the control depends of WIFI antenna used to communication between client and server. A normal antenna has a coverage of 90m in outside. The second target is to accomplish 3 edition methods: HDR (High Dynamic Range), It allows you a great dynamic range of luminosity combining different exposures levels; Timelapse, It is a method to capture events with low speed; Focus stacking, It is to get a 100% focused image.

Any person with basic knowledge will be able to use the application and to have access any functionality from any part of the application. To start using the application is necessary to connect hardware with the camera and log on WIFI. When users are connected on network, they can get the user interface of application.

The result of this project is an application can use basic functionalities of a DSLR cam like: LiveView; PreView and Shoot. Image can be viewed in full screen to have a better resolution. The user interface has 4 switchable modes: Normal, to change any parameter of the camera; HDR, to create a HDR with some frames in only one click; Timelapse, to generate a video defining interval seconds between frame and frame, how many frames and how many fps (frames per second in video); and the last mode is Focus stacking, to get a 100% focused image, users have to define a number of frames.

Resum

Aquest projecte va néixer de les mans de José Bosch Estrada, professor de la Universitat de Barcelona del departament d'Electrònica. L'elecció d'aquest projecte va ser motivada per les múltiples capes que disposa la infraestructura per assolir l'objectiu i les ganes d'adquirir uns coneixement mínims en l'àmbit de la fotografia. El projecte té un objectiu poderós amb capacitat de ser únic i facilitar el treball a l'usuari poc experimentat amb la tecnologia. A més a més el projecte està destinat a integrar-se amb altres per incrementar la utilitat, l'eficàcia i la manipulació a distància d'una càmera DSLR.

L'objectiu prioritari del projecte és aconseguir el control remot d'una càmera reflex per poder canviar qualsevol paràmetre a distància, així s'aconseguiria evitar la manipulació de l'objecte. Cal destacar que la distància dependrà de l'antena WIFI utilitzada per la comunicació entre el client i el servidor que controla la càmera, encara que, una antena WIFI bàsica et pot permetre estar fins a 90m en espais oberts. En segon lloc aconseguir 3 tipus d'edició que més s'utilitza en un únic clic: HDR (High Dynamic Range), et permet compensar la il·luminació de la imatge realitzant la combinació d'imatges amb diferent compensació de llum; Timelapse, mètode per capturar esdeveniments amb velocitat lenta; Focus stacking, mètode per aconseguir una imatge enfocada al 100%.

Qualsevol persona amb coneixements molt bàsics serà capaç d'utilitzar tota l'aplicació segons calgui i tenir accés a qualsevol funcionalitat des de qualsevol punt de l'aplicació en un nombre mínim de passos sense perdre una distribució lògica de l'aplicació. Per començar a utilitzar l'aplicació únicament cal connectar el hardware corresponent a la càmera i accedir a la WIFI d'aquest accessori. Un cop connectat a la xarxa de l'aparell, l'usuari amb qualsevol dispositiu podrà accedir a l'aplicació.

El resultat del projecte consta d'una aplicació capaç d'utilitzar funcionalitats bàsiques d'una càmera DSLR com: LiveView, previsualització constant de la imatge capturada per l'objectiu de la càmera; PreView, previsualització d'una imatge sense guardar a memòria; i Shot, realitzar la captura

real. A més a més es realitza un zoom a pantalla completa per realitzar una millor visualització de la imatge.

La interfície disposa de 4 modes intercanviables: Normal, amb capacitat de canviar qualsevol paràmetre que tingui la càmera; HDR, et permet realitzar diferents frames configurables individualment generant un HDR entre ells a un sòl clic; Timelapse, definint un interval de segons entre captures, el nombre de captures i els fps del vídeo et genera un vídeo amb l'efecte Timelapse; i per últim el mode de Focus stacking que et permet, definint una quantitat d'enfocaments, realitzar una imatge completament enfocada.

Agraïments als automagics de NimboTeam
"Només amb codi lliure trencarem els límits"

Capítol 1

Introducció, motivacions, propòsit i objectius

1.1 Introducció i motivacions

Aquest projecte va néixer de les mans de José Bosch Estrada, professor de la Universitat de Barcelona del departament d'Electrònica. L'elecció d'aquest projecte va ser motivada per les múltiples capes que disposa la infraestructura per assolir l'objectiu i les ganes d'adquirir uns coneixement mínims en l'àmbit de la fotografia. El projecte té un objectiu poderós amb capacitat de ser únic i facilitar el treball a l'usuari poc experimentat amb la tecnologia. A més a més el projecte està destinat a integrar-se amb altres per incrementar la utilitat, l'eficàcia i la manipulació a distància d'una càmera DSLR.

1.2 Propòsit

Fins ara tot tipus de fotògraf necessita diferents eines per poder realitzar un producte final amb qualitat. Aquestes modificacions poden derivar de la fase de producció: canvis d'apertura, velocitat d'exposició, compensació d'exposició o qualsevol paràmetre d'una càmera; o bé de postproducció o edició d'imatge: enquadrar imatges, variar l'enfocament, realitzar muntatge de imatges i vídeos i una llarga llista d'opcions creatives que permet obtenir un resultat espectacular i professional.

Aquest motiu ha generat el propòsit general del projecte 'DSLR Project'. En primer lloc aconseguir el control remot d'una càmera reflex per poder canviar qualsevol paràmetre a distància, així s'aconseguiria evitar la manipulació de l'objecte. Cal destacar que la distància dependrà de l'antena WIFI utilitzada per la comunicació entre el client i el servidor que controla la càmera, encara que, una antena WIFI bàsica et pot permetre estar fins a

90m en espais oberts. En segon lloc aconseguir 3 tipus d'edició que més s'utilitza en un únic clic: HDR (High Dynamic Range), et permet compensar la il·luminació de la imatge realitzant la combinació d'imatges amb diferent compensació de llum; Timelapse, mètode per capturar esdeveniments amb velocitat lenta; Focus stacking, mètode per aconseguir una imatge enfocada al 100%.

1.3 Objectius

El principal objectiu d'aquest projecte és que 'DSLR Project' sigui una eina de camp útil i productiva, per això s'ha de tenir especial cura tant en el disseny de l'aplicació com en la implementació. Per tal d'assolir aquest requeriment es plantegen el conjunt d'objectius esmentats a continuació.

El sistema ha de ser molt fàcil de fer servir per qualsevol usuari tant amb experiència fotogràfica com sense, això vol dir que ha de tenir una corba d'aprenentatge el menys pronunciada possible. Qualsevol persona amb coneixements molt bàsics ha de ser capaç d'utilitzar tota l'aplicació segons calgui i tenir accés a qualsevol funcionalitat des de qualsevol punt de l'aplicació en un nombre mínim de passos sense perdre una distribució lògica de l'aplicació, això vol dir que s'ha de dissenyar la interfície gràfica per a aquests efectes. Un altre objectiu important és que es pugui executar en qualsevol tipus de dispositius i resolucions.

El sistema s'ha de dissenyar de forma que l'usuari no hagi de configurar cap aparell, únicament connectar el hardware corresponent a la càmera i accedir a la WLAN d'aquest accessori. Un cop connectat a la xarxa de l'aparell l'usuari podrà accedir a l'aplicació.

Capítol 2

Estudi de viabilitat

2.1 Viabilitat Tecnològica

En els últims anys els ordinadors han evolucionat fins a arribar a una mida petita com pot set 1x1cm. En aquest projecte s'utilitzarà com a part de servidor i controlador de la càmera un dispositiu tipus Rapsberry amb un accessori d'antena WIFI per crear una xarxa WLAN i poder interactuar amb ell. L'elecció del hardware ha estat una sol·licitud del departament d'electrònica de la Universitat de Barcelona per més endavant integrar aquest projecte amb un altre projecte que controla remotament el moviment d'un objecte amb motors. Així es tindria un control absolut.

Una de les principals decisions que han fet possible tecnològicament el projecte 'DSLR Project' ha estat el fet de basar-ho totalment en tecnologies open-source i estàndards oberts. Un dels principals objectius ha estat que el sistema hagi de funcionar en la majoria de dispositius possibles i tenir el màxim grau de portabilitat, això ha fet que triar fer servir estàndards oberts hagi estat una solució lògica. La tria d'eines i llibreries de programari lliure a més és un avantatge, ja que solen estar molt ben documentats i en cas de trobar problemes sempre es poden consultar els codis font per conèixer amb més profundament el funcionament, a part d'estar suportats per comunitats de programadors molt actius i sempre disposats a ajudar.

A escaola de programari lliure s'ha fet una elecció del més eficient del mercat actual i compatible amb el hardware descrit. Per la part de servidor s'utilitzarà: Apache2, per donar servei a les peticions del client per HTTP; Django Framework, per crear un entorn aplicatiu Web on permeti de manera eficient la utilització de: la llibreria de la càmera, algorismes de visió de computació i gestió de les funcionalitats de la web en Python; PostgresSQL, base de dades per treballar amb Django; Foundation, per aconseguir l'adaptació de totes les resolucions amb HTML i CSS.

Per aquestes raons es pot afirmar que la viabilitat tecnològica del sistema 'DSLR Project' es deu en gran part al conjunt d'eines software lliures i protocols oberts que han servit com a base robusta per a aquest. Els principals punts crítics tecnològics del sistema han estat la lògica, la connexió i control de la càmera, juntament amb la interfícia portable per qualsevol dispositiu.

2.2 Viabilitat Econòmica

L'únic requisit indispensable és l'adquisició del hardware Raspberry Pi més una wifi portable que en conjunt pot significar uns 30$. L'aplicació et permet realitzar automàticament tasques de postproducció, sense fer cap mena d'edició per part de l'usuari, això implica que l'usuari no cal que disposi de programari d'edició d'imatges que actualment molts d'ells són privats i per tant tenen un cost econòmic. Per altra banda et permet realitzar tasques que el propi software de la càmera no tingui i això es tradueix en una reducció econòmica important alhora d'escollir el model de la càmera rèflex.

Aquest projecte s'ha dut a terme fent servir tecnologies open-source que han suposat un estalvi en llicències molt gran, ja que tant els sistemes operatius, com els entorns de desenvolupament i els components de software utilitzats han estat triats segons llicències de codi obert que permetin el desenvolupament d'aplicacions comercials. Com a exemple d'alguns dels estalvis més importants aconseguits amb aquesta decisió es podrien indicar els casos del sistema operatiu i la base de dades del servidor on s'ha fet servir un sistema operatiu Linux que a part dels avantatges tecnològics suposa un estalvi econòmic respecte d'altres alternatives com podrien ser Windows Server. Per la base de dades s'ha utilitzat PostgreSQL que també representa un estalvi respecte d'altres alternatives com Oracle o Microsoft SQL Server.

D'aquesta manera s'ha reduït el cost del projecte 'DSLR Project' a nivells mínims fent que sigui un producte de baix cost, accessible i útil per qualsevol usuari.

2.3 Comparativa de mercat

Les estructures trobades en el mercat actual es distingeixen en:

1. Raspberry Pi amb la càmera connectada per USB i utilitzant un wireless o bluetooth per la comunicació. A més a més realitzar una aplicació pel client Android. Principalment aquest mètode trenca el requisit del projecte i no pugui funcionar amb qualsevol dispositiu per part de l'usuari. CamRanger utilitza una estructura similar, font de referència: [1]

2. Android connectat directament amb USB a la càmera utilitzant una funció USB Host d'Android. Un exemple és dslrcontroller. El principal problema torna a ser la dependència de l'usuari amb Android, fonts de referència [2] [3]

3. Ordinador connectant per USB la càmera. Aquest té el inconvenient de la portabilitat i la inexistent distància que et pots allunyar de la càmera. Fonts de referència [4] [5]

4. Un TPLINK connectat per USB a la càmera amb un dispositiu Android per accedir a la xarxa WLAN. El dispositiu Android necessita també una aplicació per poder controlar la càmera, font de referència [6]

5. Dispositiu android que es comunica mitjançant un petit infrarojos. Font de referència [7]

Com es pot observar totes les alternatives de mercat t'obliguen a utilitzar un dispositiu en concret. Es pot dir que l'estructura escollida en aquest projecte farà que sigui la primera infraestructura de control de càmera DSLR amb independència per l'usuari de quin dispositiu tingui per connectar-se.

Capítol 3

Marc de treball i conceptes previs

3.1 Introducció a la fotografia

La fotografia és la tècnica per obtenir imatges mitjançant la llum. Actualment existeix la fotografia digital que consisteix a capturar la llum amb un sensor que divideix el quadre en una matriu de cel·les on cadascuna d'elles representa un píxel digital. Quan es dispara la càmera, per deixar entra llum, fa una conversió de llum a valors digitals (voltatge) perquè la CPU de la càmera pugui processar-la i emmagatzemar les dades.

3.2 Càmeres rèflex digitals

3.2.1 Introducció

El nom de rèflex prové d'aquelles càmeres que utilitzen un mirall per reflectir la llum que entra per l'objectiu i el fa conduir fins al visor. Existeixen les SLR, càmeres amb un objectiu i les TLR, amb dos objectius. En aquest projecte s'utilitzen càmeres completament digitals anomenades DSLR (Digial Single Lens Reflex). Com el seu nom indica són càmeres digitals amb un únic objectiu i tenen la particularitat de tenir moltes funcionalitats automatitzades com: autoenfocament, sincronització de flash; avantatges com: canviar objectius guanyant profunditat, capten millor la llum, poden tenir diferents tipus d'emmagatzement, velocitat d'obturació molt curts i una llarga llista d'eines per software i facilitats per l'usuari.

3.2.2 Paràmetres

Els paràmetres més utilitzats són: l'apertura, definit per la distància que s'obra al diafragma de l'objectiu, això permet calibrar la quantitat d'entrada de llum a l'objectiu; la velocitat d'obturació, un cop fas una fotografia, pots variar el temps d'exposició, això provoca que les coses en moviment es propaguin més en el resultat de la imatge, per exemple si fotografiéssim una cascada d'aigua i el temps d'exposició el definim molt curt podríem veure gotes d'aigua flotant a l'aire, per altra banda si el temps d'exposició fos molt gran pot aparèixer l'aiga com un corrent difuminat; la ISO, la càmera li pots definir la sensibilitat alhora de capturar la imatge, si pugen el valor de la ISO afegim més sensibilitat i per tant captarà més llum; la compensació de l'exposició, aquest paràmetre et permet tenir la imatge subexposada o sobreexposada amb la mateixa configuració a la càmera; molts paràmetres més existeixen, com per exemple: la variació en el format alhora de guardar la imatge, retocs de colors...

3.3 Mètodes d'edició d'imatge

Utilitzant els paràmetres descrits anteriorment es poden realitzar molts efectes que combinats realitzen resultats espectaculars. Els mètodes que utilitzarà aquest projecte són: HDR (High Dynamic Range), es tracta de compensar la llum que existeix a la imatge, bàsicament s'utilitza en paisatges que tenen zones amb diferents graus de llum. Es realitzen diferents fotografies per després combinar-les i aconseguir una imatge compensada amb la mateixa llum; Timelapse, consisteix a realitzar diferents captures al llarg del temps i combinar-les en un vídeo per aconseguir que actes que succeeixen a velocitats molt lentes per la percepció de l'ull humà apareguin en poc temps, com per exemple: veure florir, clarejar, etc; Focus stacking, consisteix en la tècnica de fer diferents captures amb un enfocament diferent per combinar-les i aconseguir una fotografia completament enfocada.

Capítol 4

Anàlisi i Disseny

4.1 Anàlisi de requeriments d'usuari

El projecte està enfocat a usuaris amb un mínim d'experiència fotogràfica, com a mínim que reconegui els paràmetres de la càmera. És possible que un percentatge d'usuaris no disposin d'experiència amb la tecnologia, així que s'ha de facilitar tant el muntatge del hardware a la càmera com la interactuació amb la interfície. L'usuari espera automatització i un control instantani de la càmera, per això s'ha de reduir el màxim possible el temps de resposta entre la càmera i l'usuari.

4.2 Anàlisi de tasques

4.2.1 Control remot càmera rèflex

Disparador i paràmetres

Una de les principals tasques és tenir accés al disparador de la càmera i als paràmetres, tant per llegir-los com per poder modificar qualsevol paràmetre en cas que es necessiti per part de l'usuari.

Com les càmeres rèflex tenen una llista amplia de paràmetres i a més a més l'usuari no té per què enrecordar-se de tots els seus possibles valors de cadascun d'ells, per això l'estructura en la interfície ha de ser una llista ordenada amb les opcions dintre de cada paràmetre encara que sigui de forma oculta, per després desplegar-les.

A la segona iteració del desenvolupament es va observar que l'usuari no utilitza molts dels paràmetres per això es va reduir l'espai de la llista perquè aparegui un percentatge d'ells i amb un scroll poder accedir abaix i adalt.

Previsualització

A més a més, després d'observar l'usuari, una opció imperativa és la previsualització d'una fotografia, és a dir, crear una opció on es dispari una fotografia però que no es guardi a la càmera, només per tenir una primera impressió per començar o continuar canviant paràmetres per aconseguir el resultat desitjat.

També per aquest motiu va néixer, a la tercera iteració, la necessitat de poder veure la imatge a pantalla completa, sobretot quan la resolució del dispositiu utilitzat no era suficientment gran i poder distingir petits errors o perfilar encara més l'efecte que es busca.

LiveView

Per completar totes les funcionalitats de les càmeres rèflex existeix una opció de LiveView, sobretot en models de gama mitja-alta. Consisteix bàsicament en previsualitzar en temps real la imatge que capta l'objectiu de la càmera. Això pot ser útil per exemple: si l'usuari es troba en la cerca d'animals on mai es sap quan apareixerà, per poder llançar la foto ha de veure que està capturant l'objectiu.

4.2.2 HDR: High Dynamic Range

L'usuari podrà seleccionar els paràmetres necessaris per cada frame entre: apertura; velocitat d'obturació, ISO i exposició, un cop s'executa el disparador a l'aplicació, la càmera haurà de realitzar cada fotografia amb cada configuració realitzar el HDR i retornar un arxiu comprimit amb totes les fotografies originals més el resultat de l'algorisme utilitzat per realitzar el HDR.

4.2.3 Timelapse

Aquest mètode necessita saber l'interval de temps per realitzar cada fotografia, la quantitat de fotografies que s'han de fer i el frames per segon que vol que es generi el vídeo. Un cop s'executa el disparador, realitzarà cada fotografia en l'interval de temps acordat i un cop acabat generarà el vídeo i retornarà totes les imatges originals més el vídeo en un fitxer comprimit.

4.2.4 Focus stacking

Per realitzar aquest mètode es facilitarà a l'usuari 64 posicions. Aquestes posicions són posicions que el motor d'enfocament pot aturar-se. Per tant l'usuari podrà especificar en quines posicions del motor vol que es capturi la imatge. Un cop la càmera dispari totes les fotografies, l'aplicació realitzarà la combinació per aconseguir la imatge perfectament enfocada. Per últim la web retornarà un fitxer en zip amb totes les imatges.

4.3 Disseny de software

Com s'ha anomenat anteriorment l'estructura es basa en un client-servidor on el servidor (Rapsberry Pi) s'encarregarà de controlar la càmera. Aquest mateix dispositiu crearà una xarxa perquè el client es connecti i amb una petita aplicació web dintre del mateix servidor pugui interactuar i donar les ordres necessàries a la càmera. Per això el disseny es conforma en diferents capes per poder aconseguir aquesta comunicació.

4.3.1 Control remot

Imperativament ha d'existir un capa, on el sistema operatiu de la Raspberry Pi, en aquest cas un Linux, pugui accedir a la càmera mitjançant un port USB. Aquesta capa ha de poder accedir a les posicions de memòria de la càmera i a la 'SD card', també enviar comandes a la CPU per poder executar ordres de la càmera. Per això s'utilitzarà una llibreria de codi obert anomenada gphoto2 on et permet accedir a la majoria de models de càmeres Canon i Nikon.

4.3.2 Accès a la llibreria de control

Per poder utilitzar el control remot dintre de l'aplicació web s'ha de crear una capa, on et permeti carregar la llibreria de gphoto2. El framework que s'utilitzarà per realitzar l'aplicació web està realitzat amb llenguatge Python, per tant, aquesta capa haurà de ser realitzada també en llenguatge Python.

4.3.3 Servidor web

El client d'aquest sistema pot ser qualsevol navegador per tant, s'ha de gestionar les peticions HTTP amb un servidor web. S'instal·larà l'Apache2 a entorn productiu encara que per realitzar el desenvolupament de l'aplicació el mateix framework et permet utilitzar un servidor propi del framework per debugar. Encara que no hi és el requeriment aquest servidor pot gestionar múltiples peticions alhora.

4.3.4 Aplicació web

L'aplicació serà l'encarregada de transmetre les ordres des de l'usuari fins al controlador de la càmera. En aquesta capa és on recau tot el disseny de la interfície i per tant tot el valor des del punt de vista d'usuari.

Capítol 5

Desenvolupament

5.1 Metodologia de desenvolupament

Segons s'ha vist al capítol 1, un dels principals objectius del sistema és aconseguir el control remot de la càmera. El primer pas és realitzar la configuració del servidor, un cop cobert aquest requeriment, en segon lloc: es podrà realitzar el disseny de la infraestructura del sistema distribuït de client-servidor per crear la comunicació entre ells. El tercer pas és realitzar una interfície amb un bon grau d'usabilitat per la part del client.

Aquests 3 objectius es ramifiquen en diferents tasques, cadascú té la seva part d'estudi previ, disseny, implementació i test. Tenint en compte els recursos de temps i personal i l'objectiu final d'aconseguir funcionalitats de postproducció en l'aplicació, la millor metodologia que s'adapta per aquest desenvolupament de software és l'espiral. Aquesta metodologia et permet fer iteracions de desenvolupament, on cada iteració consisteix en: Anàlisis, Avaluació, Desenvolupament, Test i Planificació de la següent iteració.

Amb cada iteració aconsegueixes una avaluació de riscos, nous requisits, interessos i avantatges o inconvenients de tasques realitzades en l'anterior iteració. Això t'aporta molta flexibilitat en el desenvolupament i et facilita la integració de possibles requisits de sistema realitzant avaluacions constants.

5.2 Implementació

El procés d'implementació del sistema consisteix a construir totes les parts que conformaran el sistema complet. En el projecte actual s'ha dividit en les subtasques d'instal·lació, configuració i programació del sistema detallades a continuació.

5.2.1 Instalació

En aquesta tasca s'engloben tots els processos d'instal·lació, i compilació en cas que calgui, de tots els elements de software necessaris per al funcionament del sistema complet. Això engloba els servidors necessaris per servir la lògica, bases de dades, entorns de programació i llibreries per compilar i executar elements de software del sistema.

En aquest cas concret en fer servir software lliure i protocols oberts no cal gestionar les llicències, més enllà de comprovar que siguin compatibles amb el desenvolupament d'aplicacions per ús comercial i restriccions aplicables, i es simplifica en gran mesura el manteniment de versions i actualitzacions de qualsevol element necessari pel sistema.

Raspberry Pi

Aquest dispositiu consisteix en un ordinador de placa reduïda i de baix cost per potenciar l'ensenyament de les ciències de la computació. Es va crear el 2012 per la Fundació Raspberry Pi, conté una CPU ARM a 700MHz, GPU Broadcom i entre 256-512 MB de memòria RAM. Disposa d'una ranura per una 'SD card' on s'emmagatzemarà el sistema operatiu que es desitgi. El sistema operatiu ha de tenir un kernel Linux ARM. El consum està al voltant de 2.5 a 3.5 W.

5.2.2 Configuració

Posteriorment a la instal·lació, és necessària una tasca de configuració que permeti a tots els elements del sistema funcionar conjuntament. Caldrà configurar els elements software tals com servidors HTTP o de bases de dades.

Sistema Operatiu Debian Linux

Raspberry Pi facilita una distribució de Linux basada en Debian que s'anomena Raspbian. S'ha escollit aquesta distribució per dos factors: compatibilitat amb tots els softwares utilitzats i l'experiència ja coneguda en aquest entorn.

Llibreria Gphoto2

Gphoto2 és una llibreria de codi obert amb suport a més de 1800 models de càmeres realitzada en codi C. Treballa en molts sistemes operatius entre ells Linux Debian. És una llibreria en constant evolució i mantinguda per la comunitat, l'ultima actualització és de Març de 2014, just en plena elaboració d'aquest projecte.

Servidor HTTP Apache2

Apache és un projecte de servidor web HTTP de codi obert molt utilitzat a internet. És perfectament compatible amb el sistema operatiu. A més a més es pot integrar el framework descrit posteriorment pels entorns productius amb una simple configuració al mateix servidor Apache.

Django Framework

Django Framework és un framework de codi obert realitzat en Python. Utilitza una arquitectura de model-view-template, on es defineixen els models, com els objectes necessaris dintre de la base de dades; view, com les possibles vistes que es vulguin generar dintre de l'aplicació, normalment definides per diferents URIs; i per últim templates que són les pàgines amb HTML, CSS, javacript... mostrades a l'usuari. El control de l'aplicació està intrínsec a la lògica del framework, tant per la part de la gestió de peticions com la gestió de la base de dades.

PostgreSQL

PostgreSQL és un sistema de gestió de base de dades objecte-relacional de codi obert. Es pot integrar perfectament amb Django i és compatible amb el sistema operatiu emprat.

FFmpeg Framework

FFmpeg Framework és un software multimèdia de codi obert en llenguatge C. Et permet grabar, transcodificar en diferents formats, realitzar streaming, realitzar muntatges de fitxers multimèdia... En aquest projecte s'utilitzarà exclusivament per realitzar el Timelapse i crear un vídeo a partir d'una sèrie d'imatges.

5.2.3 Programació

La integració de tots els sistemes descrits anteriorment necessita la creació d'una capa per fer la comunicació entre Django on es crearà l'aplicació web i la càmera. Per carregar la llibreria gphoto2 (ligphoto2) es crearà en llenguatge Python un mòdul per poder-lo utilitzar al framework Django. Degut aquest motiu per tot el tractament d'imatge s'utilitzaran llibreries pròpies de Python (PIL). Posteriorment s'haurà de crear una aplicació a Django on estigui l'estructura base de l'aplicació web, vistes i models.

5.3 Prototipats i avaluació

La metodologia seleccionada en espiral està previst que permeti entre 2 a 3 iteracions. Això aportarà els beneficis d'avaluar el prototip en cada cicle, com s'ha vist a l'anàlisi d'usuari la interfície ha d'ajustar-se perfectament a la manera de treballar d'un fotògraf, ja sigui amateur o professional. A la primera iteració es crearà un prototip bàsic per aconseguir funcionalitat, posteriorment s'elaborarà el disseny i l'estudi de la col·locació dels elements per finalitzar amb una interfície adaptativa a totes les pantalles i que tots els elements estiguin ben col·locats i no trenqui el sentit de l'aplicació, així l'usuari no haurà d'aprendre més que un cop, independentment si canvia de pantalla.

Capítol 6

Planificació

En qualsevol projecte amb un volum de feina considerable s'ha de realitzar una planificació per determinar quines són les tasques principals en les quals es descompon, quin temps es necessita per cada tasca i quin és el resultat que hem d'obtenir per cada tasca, per últim també resulta d'utilitat realitzar un diagrama de Gantt per indicar l'ordre d'execució d'aquestes tasques segons siguin necessaris els resultats d'unes per a realitzar les altres.

6.1 Estudi de viabilitat

Tasca	Temps (h)	Resultat de la tasca
Viabilitat tecnològica	16H	Descobriments de tots els software utilitzats
Viabilitat econòmica	16H	Resultat de preus pel desenvolupament del projecte
Comparativa de mercat	24H	Llista de diferents estructures amb el mateix objectiu del projecte per obtenir el millor resultat

6.2 Construcció de l'entorn de treball

Tasca	Temps (h)	Resultat de la tasca
Adaptació del hardware	10H	Adquisició de la Raspberry Pi amb el wifi USB. Intal·lació del sistema operatiu, actualització del sistema i configuració perquè sigui un punt d'accés i enruti tràfic per ethernet.
Instal·lació del software	16H	Instal·lació de servidor Apache2, Django Framework, FFmpeg, Gphoto2, PostgreSQL i dependències creades per elles.
Desenvolupament de la comunicació entre el sistema i la càmera	18H	Configuració de la llibreria Gphoto2, creació de la llibreria en Python i proves interactuant amb la càmera.

6.3 Anàlisi i disseny

Tasca	Temps (h)	Resultat de la tasca
Anàlisi de requeriments d'usuari	8H	Definició dels requeriments per l'usuari final, tant de disseny com de sistema.
Anàlisi de tasques	8H	Definició de les tasques que ha d'assolir l'aplicació
Disseny de software	12H	Definició de l'estructura del sistema per aconseguir el seu propòsit
Prototips i avaluacions	16H	Resultat del disseny de la interfície de l'aplicació web

6.4 Implementació

Tasca	Temps (h)	Resultat de la tasca
Control amb la llibreria Gphoto2	14H	Poder fer crides utilitzant la llibreria des de la Raspberry Pi fins a la càmera rèflex
Aplicació web	36H	Aplicació que permet la interactuació de l'usuari utilitzant un navegador web amb la càmera rèflex
Interficie gràfica	16H	Devenvolupament de la interfície per poder controlar els events creat per l'usuari i crear un temps de resposta adequat, informant l'usuari perquè no es perdi en cap tasca i sigui molt intuïtiva la utilització d'ella.
HDR	16H	Entesa del mètode i realització del mòdul en Python per la seva utilització
Timelapse	12H	Entesa del mètode i realització del mòdul en Python per la seva utilització
Focus stacking	8H	Entesa del mètode i realització del mòdul en Python per la seva utilització

6.5 Validació del sistema

Tasca	Temps (h)	Resultat de la tasca
Control en temps real de la càmera amb l'aplicació web	8H	La càmera és 100% funcional, tot i així, s'han trobat en proves molt llargues que a vegades és necessari reiniciar la càmera.
Funcionament de cada mètode	8H	Els 3 mètodes funcionen perfectament, utilitzant les combinacions que es vulgui dintre d'ells.
Proves	8H	Resultat fotogràfic per cada mètode implementat: HDR, Timelapse i Focus stacking.

Capítol 7

Implementació i proves

En aquest apartat s'explicarà cada punt important en el desenvolupament del projecte. Cada punt té afegit la seva referència a l'annex on es troba el codi. Seguint tots els punts s'obté el projecte sencer, la majoria de punts s'han de seguir ordenadament, ja que en moltes ocasions es creen dependències entre ells.

7.1 Actualització del sistema operatiu

Implementació del codi al punt 10.1.1 del Annex a la pàgina 43

El primer pas a realitzar és actualitzar el sistema operatiu amb el repositori propi. A més a més s'instal·len paquets essencials pel projecte: eines per executar llibreries, eines per accedir fàcilment a l'usb, llibreries per integrar gràfics i utilitzar còdecs.

7.2 Configuració punt d'accés Wireless

Implementació del codi al punt 10.1.2 del Annex a la pàgina 43

Per crear una xarxa WLAN pròpia és imperatiu instal·lar un servei que et permeti crear un punt d'accés. S'utilitza el hostapd en aquest projecte fàcilment configurable i es defineix una IP estàtica en aquest per wlan0.

Per facilitar el desenvolupament s'afegeix l'enrutament de tràfic per la seva interfície de WLAN (per l'antena Wireless) per la interfície ethernet. Això significa que el dispositiu, en cas d'endollar-lo a un router amb internet per ethernet, seria capaç de donar accés a internet a tots els clients dintre de la seva xarxa WLAN.

Per finalitzar la configuració es necessari reiniciar el dispositiu (realitzant abans el punt següent) o bé reiniciar els servies networks i hostapd.

7.3 Mantenir la configuració per cada reboot

Implementació del codi al punt 10.1.3 del Annex a la pàgina 45

Aquest pas únicament és per mantenir l'opció de l'enrutament. Cada cop que es reinicia el dispositiu la configuració de l'enrutament s'esborra, per això s'ha creat un petit script per carregar la configuració cada cop que la màquina s'engegui.

7.4 Instal·lació gphoto2 i llibreria gphoto2

Implementació del codi al punt 10.1.4 del Annex a la pàgina 46

La llibreria encarregada de donar accés per USB a la càmera rèflex digital és gphoto2. Aquest pas realitza la descàrrega, compilació i instal·lació tant de la llibreria nativa com la interfície al command line per realitzar proves.

La llibreria nativa es carregarà amb una llibreria pròpia en Python i es podrà utilitzar a nivell d'aplicació, per altra banda es podrà utilitzar la interfície 'CLI' per poder realitzar comprovacions esporàdiques com comparacions entre resultats de la llibreria i la interfície, detecció de la càmera, llistar paràmetres, etc.

7.5 Instal·lació de llibreries de python i eines de python

Implementació del codi al punt 10.1.5 del Annex a la pàgina 46

Per poder realitzar el tractament d'imatges amb el llenguatge Python es necessari instal·lar el mòdul Pillow, a més a més d'instal·lar una eina per trobar mòduls fàcilment mitjançant un repositori genèric amb easy_install.

7.6 Instal·lació de Django Framework amb PostgreSQL i dependències

Implementació del codi al punt 10.1.6 del Annex a la pàgina 47

Si em realitzat el pas anterior es pot descarregar Django amb la comanda pip. Una alternativa és anar a la web de Django [18] i descarregar la versió desitjada. També és necessari la instal·lació de la base de dades PostgreSQL, tant el servidor, com el client i la llibreria que realitza la comunicació de PostgresSQL amb Python psycopg2.

La versió de Django utilitzada en aquest projecte és 1.6, es podrà trobar tota la informació més detallada a [24]

7.7 Configuració de Django & PostgreSQL

Implementació del codi al punt 10.1.7 del Annex a la pàgina 47

El primer pas és crear el projecte on realitzarem la configuració de l'entorn de l'aplicació. L'estructura de fitxers es crea per defecte i té la següent distribució:

```
mysite/
    manage.py
        mysite/
        __init__.py
        settings.py
        urls.py
        wsgi.py
```

manage.py: és el command-line per poder utilitzar el projecte té diferents utilitats com: sincronitzar la base de dades, agrupar els fitxers estàtics en el directori arrel, engegar un servidor per desenvolupar i altres funcionalitats més específiques.

__init__.py: és un fitxer de Python per indicar que el directori és un Python Package

settings.py: és el fitxer de configuració del projecte

urls.py: s'encarrega de definir les urls que demanda el client a quines aplicacions va dirigides. En el nostra cas si l'usuari no introdueix cap url, només la IP del servidor s'enllaçarà amb l'aplicació dslrapp que més endavant crearem per fer l'aplicació web.

wsgi.py: aquest fitxer s'utilitzarà més endavant per poder integrar el projecte a un servidor web

El segon pas és la creació de l'aplicació dintre del projecte, l'estructura de fitxers que es crea per defecte té la següent distribució:

```
dslrapp/
    \_\_init\_\_.py
    admin.py
    models.py
    tests.py
    views.py
```

admin.py: fitxer per enllaçar els models a l'administració de Django

models.py: fitxer on es defineixen la classe de cada objecte

test.py: fitxer per fer comprovacions de l'aplicació

view.py: fitxer on es defineixen les vistes per cada url que tinguem a l'aplicació

A més a més s'hauria d'afegir el directori 'static' on es guardaren tots els fitxers estàtics, un fitxer urls.py per definir totes aquelles direccions que necessitem i per últim un directori templates per guardar tots els htmls que necessitem per l'aplicació.

Per últim s'haurà de realitzar tota la configuració del projecte i les referències cap a l'aplicació i cap a la base de dades.

7.7.1 Creació de la base de dades

Implementació del codi al punt 10.1.7 del Annex a la pàgina 50

La creació de la base de dades ha d'anar amb concordança amb els fitxers de configuració del projecte a settings.py

7.8 Configuració de l'aplicació web dslrapp

Implementació del codi al punt 10.1.8 del Annex a la pàgina 51

En aquesta secció s'explicarà el desenvolupament de l'aplicació web anomenada dslrapp. Aquesta serà l'encarregada de tota la lògica del projecte i de la gestió de la interfície.

7.8.1 Models de l'aplicació

Implementació del codi al punt 10.1.8 del Annex a la pàgina 51

Els models són aquells objectes que volem guarda a la base de dades. Quan l'usuari connecta la càmera, per no haver d'accedir tota l'estona per obtenir lectures, es registra la càmera en cas de no existir a la base de dades. D'aquesta forma evitem un tràfic innecessari cap a la càmera i només accedirem a ella per enviar-li ordres.

Els models consisteixen en: Camera, model de la càmera; CameraConfig, tots els paràmetres de la càmera; CameraAbility, tots aquells camps informatius que té, com per exemple: artista, propietari, versió de càmera, número de referència...

7.8.2 Afegir models a l'administració

Implementació del codi al punt 10.1.8 del Annex a la pàgina 52

Per defecte el projecte Django crea una administració, per accedir s'afegeix 'admin/' a la url general. L'usuari ve assignat per la configuració dels settings.py.

Un cop tenim els models definits per poder gestionar-los s'han d'afegir en primer lloc a l'admin.py i després sincronitzar el projecte amb els nous models, cada cop que hi hagi un canvi als models s'ha de sincronitzar amb la base de dades.

Per realitzar la sincronització s'ha d'utilitzar el manage.py del projecte i utilitzar la comanda syncdb, és a dir, 'python manage.py syncdb'.

7.8.3 Creació de URLs

Implementació del codi al punt 10.1.8 del Annex a la pàgina 52

Les urls venen en funció de les vistes que necessitem en el nostre cas tenim diferents vistes: en primer lloc tenim l'arrel de la url que dirigeix a l'usuari a l'índex de l'aplicació; en segon lloc tenim una altra vista afegint downloadHDR/fitxer.zip on fitxer.zip és un fitxer hem fet amb el mode HDR de l'aplicació, aquesta vista et retornarà el fitxer per descarregar-lo; en tercer lloc com també existeixen el mode de Timelapse i Focus stacking es va crear, seguint la mateixa metodologia que downloadHDR: downloadTL/fitxer i downloadFS/fitxer.

7.8.4 Vista de l'aplicació

Implementació del codi al punt 10.1.8 del Annex a la pàgina 53

Per realitzar el control de cada vista es defineix la resposta al fitxer views.py. Cada url com s'ha explicat en el punt anterior va dirigit a una view. Cada view és una funció dintre del views.py i per tant allà serà on hi anirà la lògica de la vista i el retorn que sigui en cada cas.

Com hem dit anteriorment existeixen 3 vistes per descarregar fitxers i després una vista molt més important que fa la lògica de l'aplicació.

La lògica de l'aplicació web en la capa de middleware principalment està dividida en 2 parts: la primera part inicial és quan l'usuari demanda amb un mètode GET la vista o pàgina web. En aquest cas la lògica segueix els següents punts:

1. Comprovació de si existeix el registre del model de la càmera a la base de dades

    1.1 Cas negatiu: es registra la càmera a la base de dades

    1.2 En cas de no trobar càmera connectada retorna una splash previ

2. Es llegeix la llista de la configuració de la càmera per inicialitzar l'aplicació

3. Segon aquesta llista es llegeix de la càmera alguna informació rellevant per l'usuari

    3.1 apertura, iso, data, fotografies per fer, velocitat d'obturació, exposició i algunes dades menys rellevants

4. Creació d'un context per inserir tota la informació

5. Creació del template amb HTML de l'aplicació

6. Retorn amb el template renderitzant amb el context al seu interior

La segona part consisteix en una part asíncrona de l'aplicació. Totes les funcionalitats que utilitzarà un cop obtinguda l'aplicació es faran asíncronament al servidor, per tant l'usuari no haurà d'espera cap càrrega a la pantalla i es disminueix tant càrrega de tràfic com temps en el processament de la informació.

Per poder utilitzar funcions asíncrones s'utilitza javascript amb ajax. Es generen crides HTTP al servidor amb el mètode POST i és aquí on la vista fa la distinció si és GET o POST. Quan el controlador de la vista troba que el criden amb POST el seu comportament canvia segons la funció que trobi dintre d'ells. Les funcions que es poden utilitzar són:

1. `config: canvia el valor d'un paràmetre`
2. `captureshot: realitza una fotografia`
3. `liveview: genera el liveview i realitza fotografies només en memòria RAM`
4. `preview: realitza una fotografia només en memòria RAM`
5. `readvalue: llegeix el valor de qualsevol paràmetre que es necessiti`
6. `capturehdr: realitza la crida per utilitzar l'HDR`
7. `capturetimelapse: realitza la crida per utilitzar el Timelapse`
8. `capturefocus: ealitza la crida per utilitzar el Focus stacking`

Totes aquestes funcions realitzen crides al controlador Python que porta l'aplicació. Aquest controlador és l'encarregat d'enviar i rebre la informació de la càmera. A les dues capes hi ha comprovacions d'errors i en cas de que qualsevol procès no hagi funcionat bé l'aplicació informa l'usuari de l'error.

7.8.5 Templates de l'aplicació

Implementació del codi al punt 10.1.8 del Annex a la pàgina 62

Django et permet comunicar variables de l'aplicació amb el codi HTML que es respon a l'usuari. Per realitzar aquest canal s'utilitzen els templates. Principalment un template és un fitxer HTML on pots introduir codi per accedir a variables enviades per Django i poder processar-les per després renderitzar el HTML al navegador. Aquesta funcionalitat és molt importat en aplicacions que tinguin elements dinàmics com poden ser, en aquest cas, els paràmetres de cada càmera.

7.8.6 Fitxers estàtics de l'aplicació

Implementació del codi al punt 10.1.8 del Annex a la pàgina 74

S'entén com a fitxers estàtics aquells que serveixen per les aplicacions web i no es modifiquen després de la seva creació, per exemple: fitxers de CSS, fitxers de Javascript, imatges, etc...

El projecte té configurat un directori específic per tenir tots els estàtics i és imperatiu que tots estiguin allà perquè el servidor web pugui accedir. Per recol·lectar els fitxers estàtics de totes les aplicacions cap al directori 'static' del projecte es pot utilitzar el command line manage.py amb la commanda collectstatic com 'python manage.py collectstatic'.

7.9 Integració Django a producción amb Apache2

Implementació del codi al punt 10.1.9 del Annex a la pàgina 86

7.9.1 Configuració wsgi.py del projecte

Implementació del codi al punt 10.1.9 del Annex a la pàgina 86

WSGI acrònim de Web Server Gateway Interface és la interfície entre el servidor web Apache2 i Django. Ens permet la comunicació entre ells. Per això s'ha de configurar el wsgi.py del projecte tal i com es mostra a l'annex.

7.9.2 Instal·lació del servidor web Apache2 i mòdul WSGI:

Implementació del codi al punt 10.1.9 del Annex a la pàgina 86

La instal·lació del servidor web i del mòdul és semiautomàtic i simplement utilitzant els repositoris especificats el sistema ho instal·la perfectament.

7.9.3 Configuració /etc/apache2/apache2.conf

Implementació del codi al punt 10.1.9 del Annex a la pàgina 87

Per acabar la integració és necessari definir on és l'aplicació Django al fitxer de configuració de l'Apache2 i a més a més s'ha de carregar el mòdul de wsgi.

7.10 Instal·lació FFmpeg

Implementació del codi al punt 10.1.10 del Annex a la pàgina 88

En aquest cas s'ha utilitzat el codi de FFmpeg emmagatzemat al repositori oficial. Només cal descarregar el codi, compilar-ho i instal·lar-ho. Aquest procès pot trigar bastants minuts.

7.11 Modificació de permisos del USB per accedir amb el servidor web Apache2

Implementació del codi al punt 10.1.11 del Annex a la pàgina 88

Un dels erros que pot donar el servidor web és que no pot accedir a l'USB i per tant l'aplicació no podrà communicar-se amb la càmera. Per aquests casos s'ha de modificar els permisos de sistema reflexat a l'annex.

7.12 Llibreria Gphoto2 en Python

Implementació del codi al punt 10.1.12 del Annex a la pàgina 88

Per poder utilitzar la llibreria en C de gphoto2 s'ha creat una llibreria en Python per carregar aquesta i així poder cridar des de Django les funcions necessàries. Per això es va agafar com a referència les següents fonts [22] [23].

7.13 Controlador de la llibreria gphoto2

Implementació del codi al punt 10.1.13 del Annex a la pàgina 111

Per realitzar un control personalitzat per l'aplicació s'ha creat un mòdul en Python amb diferents funcions amb control d'errors. Aquest controlador s'utilitza en la vista de l'aplicació web.

7.14 Controlador dels models de la base de dades

Implementació del codi al punt 10.1.14 del Annex a la pàgina 116

En la secció 7.8.1 de la pàgina pagerefmodelsapp s'explica que s'utilitza els models per poder registrar tota la informació de la càmera. Per poder accedir a la base de dades tant per afegir càmeres com per llegir les seves propietats s'ha definit un mòdul controlador de la base de dades amb els mètodes essencials pel funcionament de l'aplicació.

7.15 Controlador del HDR

Implementació del codi al punt 10.1.15 del Annex a la pàgina 117

El mètode HDR, explicat a la secció 3.3, pàgina 14, té un mòdul dissenyat per ell. El mòdul consisteix per un costat el control de la càmera per realitzar les fotografies amb els paràmetres sol·licitats i per altra banda una classe que realitza les combinacions de les fotografies que genera tot el procès finalitzant amb un fitxer zip amb tot el contingut.

Per realitzar aquest mètode s'utilitza el mòdul Image de la llibreria PIL de Pyhon. Es realitza l'algoritme tenint en compte la saturació i contrast de la imatge retornant 9 possibles resultats.

7.16 Controlador del Timelapse

Implementació del codi al punt 10.1.16 del Annex a la pàgina 125

El mètode Timelapse, explicat a la secció 3.3, pàgina 14, té un mòdul dissenyat per ell. Aquest mòdul realitza totes les fotografies en els intervals especificats. Un cop finalitza de captar imatges, genera el vídeo. Per generar el vídeo s'utilitza FFmpeg amb els fps especificats i amb totes les fotografies capturades per després comprimir tant el vídeo com les imatges en un zip.

7.17 Controlador del Focus stacking

Implementació del codi al punt 10.1.17 del Annex a la pàgina 127

El mètode Focus stacking, explicat a la secció 3.3, pàgina 14, té un mòdul dissenyat per ell. Aquest mòdul realitza totes les fotografies especificades amb el seu enfocament. Un cop finalitza de captar imatges, es realitza una alineació per totes les fotografies, ja que enfocant i desenfocant la imatge varia dintre del mateix marc de píxels i després es combinen tenint en compte

el contrast. Per acabar comprimeix totes les imatges i emmagatzema el zip a disc.

7.17.1 Instal·lació paquet per realitzar el focus stacking

Implementació del codi al punt 10.1.17 del Annex a la pàgina 130

Per realitzar la combinació per l'enfocament de diferents imatges és necessari instal·lar el software enfuse per linux.

7.18 Prototips

Com s'ha explicat en la metodologia espiral es fan diferents iteracions per aconseguir un millor resultat avaluant en cada iteració el desenvolupament realitzat. En aquest cas es mostrarà el procès de canvi dels prototips partint d'una base simplificada on només es realitzava previsualitzacions i canvis de paràmetre sense cap fitxer de CSS a la web, figura 7.1. El següent pas va ser millorar la percepció de l'usuari des de les posicions dels elements com la millora gràfica, figura 7.2. L'avaluació d'aquesta segona iteració no va ser correcta en totes les resolucions de pantalla i la seva imatge resultava poc intuïtiva per l'usuari. Per aquest motiu es va utilitzar la llibreria Foundation que permet aconseguir una interfície responsive, col·locació dels elements per cada resolucions de pantalla, i uns colors més definits i opacs. Per altra banda es va aconseguir una millora visual emmarcant i realitzant marges als elements creant visualment 2 columnes i 3 files de forma global. Una altra millora és realitzar els canvis de mode sempre al mateix espai de la web perquè s'intueixi perfectament en quin mode ens troben sense veure el botó seleccionat, figura 7.3

Figura 7.1: Prototip de la interfície en la primera iteració del projecte

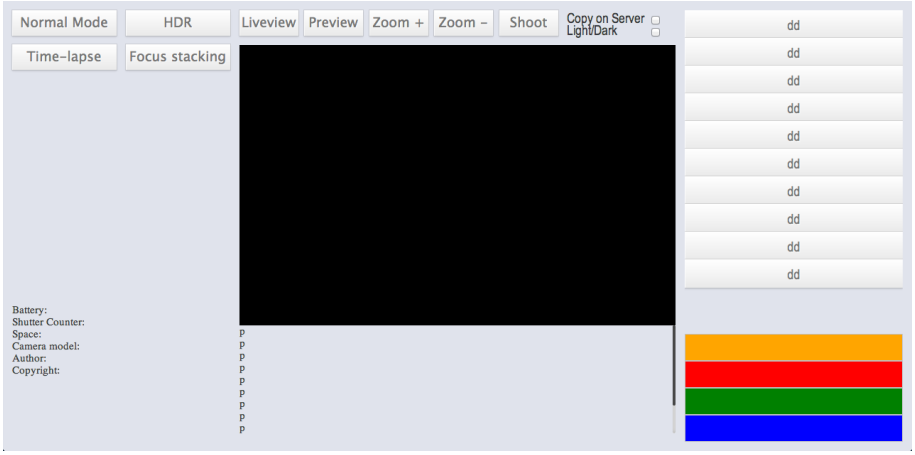

Figura 7.2: Prototip de la interfície en la segona iteració del projecte

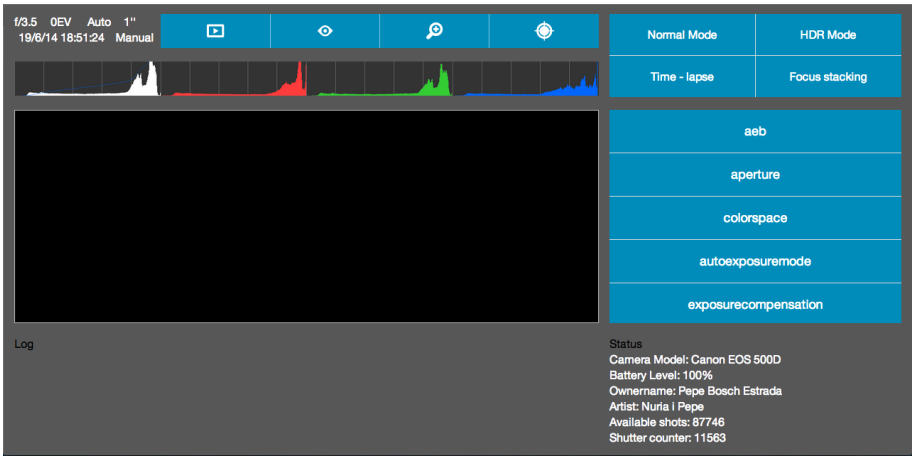

Figura 7.3: Prototip de la interfície en la tercera iteració del projecte

Capítol 8

Conclusions

En el començament del projecte el principal objectiu era aconseguir el control en temps real de la càmera. Aquest objectiu ha estat assolit perfectament. També incorpora 3 mètodes bàsics de fotografia com són: HDR, Timelapse i Focus stacking. El codi actual del projecte ha sigut una versió estable, però el desenvolupament es pot anar incrementant, incloent-hi millores i funcionalitats. Tota l'experiència del desenvolupament del projecte ha estat satisfactòria i una feina molt gratificant.

8.1 Treball futur

En el cas de voler continuar el projecte es podria començar per millorar el LiveView, actualment el LiveView consta de previsualitzacions consecutives. Una solució pot ser realitzar 25 fps amb la càmera, generar un microvídeo en mp4 amb aquestes 25 fotografies, utilitzant l'FFmpeg, i cada segon realitzar aquest procès, retornant cada vídeo a l'aplicació web. Existirà un retard d'un segon que serà el buffer que tindrem per començar a realitzar el procès. Altra solució és generar un streaming directament amb els frames encara que no es suportat per tots els navegadors actualment.

Una altra funcionalitat a incorporar pot ser afegir la utilització del bulb, mantenint el capturador obert de la càmera segons demani l'usuari. A la interfície d'usuari hi ha moltes possibles millores i amb l'ajuda d'un usuari experimentat és pot realitzar fins i tot millores per aconseguir bons resultats per imatges en l'àmbit de l'astronomia o àmbits específics que es necessari tècniques més concretes.

Capítol 9

Bibliografia

Bibliografia

[1] CamRanger (2014). *Wireless Camera Control*. http://camranger.com/features/

[2] J.B. Jongma Holding B.V (2014). *Dslcontroller*. http://dslrcontroller.com/

[3] Dslrsystem (2014). *Usb Dslr Camera Controller*. http://dslrsystems.com/?p=404

[4] digiCamControl (2014). *Remote control camera DSRL*. http://digicamcontrol.com/

[5] REMOTE DSLR CONTROL (2014). *Remote Dslr Control*. http://www.remotedslrcontrol.com/

[6] DeeJay Scharton (2014). *DSLR Film Noob*. http://www.dslrfilmnoob.com/2013/12/11/tp-link-tl-mr3040-wireless-field-monitor-dslr-controller/

[7] PhotoIRmote (2014). *PhotoIRmote*. http://www.wegroo.com/photoirmote/photoirmote/

[8] Wikimedia Foundation, Inc. (2014). *Wikipedia Photography*. http://en.wikipedia.org/wiki/Photography

[9] Wikimedia Foundation, Inc. (2014). *Wikipedia Single-lens Reflex Camera*. http://en.wikipedia.org/wiki/Single-lens_reflex_camera

[10] Wikimedia Foundation, Inc. (2014). *Wikipedia Digital Single-lens Reflex Camera*. http://en.wikipedia.org/wiki/Digital_single-lens_reflex_camera

[11] Wikimedia Foundation, Inc. (2014). *Wikipedia Hight Dynamic Range*. http://en.wikipedia.org/wiki/High-dynamic-range_imaging

[12] Montessori Muddle (2013). *Montessori Muddle Focus stacking*. http://montessorimuddle.org/2013/06/21/focus-stacking-combining-images-for-a-sharper-view/

[13] Incanus Ltd.(2014). *APT - Astro Photography Tool*. http://www.ideiki.com/astro/Default.aspx

[14] Raspberry Pi Foundation (2014). *Raspberry Pi*. http://www.raspberrypi.org/

[15] Mike Thompson and Peter Green (2014). *Raspbian SO*. http://www.raspbian.org/

[16] Gphoto Team (2014). *Gphoto2*. http://www.gphoto.org/

[17] The Apache Software Foundation (2014). *Apache Http Server Project*. http://httpd.apache.org/

[18] Django Software Foundation (2014). *Django Framework*. https://www.djangoproject.com/

[19] Django Software Foundation (2014). *Django documentation*. https://docs.djangoproject.com/en/1.6/

[20] The PostgreSQL Global Development Group (2014). *PostgreSQL*. http://www.postgresql.org/

[21] FFmpeg Team (2014). *FFmpeag*. http://www.ffmpeg.org/

[22] Jacob Marble (2011). *Github Piggyphoto*. https://github.com/alexdu/piggyphoto/

[23] PySnippet (2009). *PySnippet Ctypes*. http://pysnippet.blogspot.com.es/2009/12/when-ctypes-comes-to-rescue.html

[24] Graham Dumpleton (2014). *Python WSGI adapter module for Apache*. https://code.google.com/p/modwsgi/wiki/QuickInstallationGuide

Capítol 10

Annexos

10.1 Implementació pas a pas

10.1.1 Actualització del sistem operatiu

```
apt-get update
apt-get upgrade
apt-get install libtool libltdl-dev zlib1g-dev
   libusb-dev \
  libgmp-dev pkg-config libexif-dev libjpeg8-dev \
    graphviz libdbus-1-dev libgd2-xpm-dev
       libpopt-dev
```

10.1.2 Configuració punt d'accés Wireless

```
apt-get install hostapd
```

```
root@raspberrypi:~# cat /etc/network/interfaces
auto lo

iface lo inet loopback
iface eth0 inet dhcp

allow-hotplug wlan0
#iface wlan0 inet manual
#wpa-roam /etc/wpa_supplicant/wpa_supplicant.conf
#iface default inet dhcp
```

```
iface wlan0 inet static
  address 192.168.0.1
    netmask 255.255.255.0
    up iptables-restore < /etc/iptables.ipv4.net


root@raspberrypi:~# cat /etc/hostapd/hostapd.conf
interface=wlan0
driver=nl80211
ssid=RaspAP
hw_mode=g
channel=8
wpa=2
wpa_passphrase=Pii.2013
wpa_key_mgmt=WPA-PSK
wpa_pairwise=CCMP
rsn_pairwise=CCMP
beacon_int=100
auth_algs=3
wmm_enabled=1

cat /etc/resolv.conf
nameserver 8.8.8.8

cat /etc/sysctl.conf
# Uncomment the next line to enable packet
   forwarding for IPv4
net.ipv4.ip_forward=1



*filter
:INPUT ACCEPT [38:2428]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [34:3432]
-A FORWARD -i eth0 -o wlan0 -m state --state
   RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -i wlan0 -o eth0 -j ACCEPT
COMMIT
*nat
:PREROUTING ACCEPT [6:588]
:INPUT ACCEPT [6:588]
:OUTPUT ACCEPT [1:116]
```

```
:POSTROUTING ACCEPT [0:0]
-A POSTROUTING -o eth0 -j MASQUERADE
COMMIT
```

10.1.3 Mantenir la configuració per cada reboot

```
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each
   multiuser runlevel.
# Make sure that the script will "exit 0" on
   success or any other
# value on error.
#
# In order to enable or disable this script just
   change the execution
# bits.
#
# By default this script does nothing.

# Print the IP address
_IP=$(hostname -I) || true
if [ "$_IP" ]; then
  printf "My IP address is %s\n" "$_IP"

exit 0
```

10.1.4 Instal·lació gphoto2 i llibreria gphoto2

```
wget
   http://downloads.sourceforge.net/project/gphoto/gphoto/2.5.4/gphoto

wget
   http://downloads.sourceforge.net/project/gphoto/libgphoto/2.5.4/lib


tar xjf libgphoto2-2.5.4.tar.bz2
cd libgphoto2-2.5.4
./configure --with-prefix=/usr
make
make install


tarx zvf gphoto2-2.5.4.tar.gz
cd gphoto2-2.5.4
./configure --with-libgphoto2=/usr
make
make install
```

10.1.5 Instal·lació de llibreries i eines de python

```
apt-get install python-setuptools

apt-get install python-dev

easy_install pip

pip install Pillow
```

10.1.6 Instal·lació de Django Framework amb PostgreSQL i dependències

```
pip install Django

apt-get install postgresql postgresql-client
   postgresql-server-dev-9.1

pip install psycopg2
```

10.1.7 Configuració de Django & PostgreSQL

```
mkdir /var/www/ ; cd /var/www/

django-admin.py startproject dslrproject

cd /var/www/dslrproject

python manage.py startapp dslrapp
------
create app on django

python manage.py startapp dslrapp


"""
Django settings for dslrproject project.

For more information on this file, see
https://docs.djangoproject.com/en/1.6/topics/settings/

For the full list of settings and their values, see
https://docs.djangoproject.com/en/1.6/ref/settings/
"""

# Build paths inside the project like this:
   os.path.join(BASE_DIR, ...)
import os
```

```
BASE_DIR =
    os.path.dirname(os.path.dirname(__file__))


# Quick-start development settings - unsuitable for
    production
# See
    https://docs.djangoproject.com/en/1.6/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in
    production secret!
SECRET_KEY =
    '3dur#=*gq7+1p&vza#@ltr0*ntjp=qg_88=-787-7^f%w6gxq8'

# SECURITY WARNING: don't run with debug turned on
    in production!
DEBUG = True

TEMPLATE_DEBUG = True

ALLOWED_HOSTS = []


# Application definition

INSTALLED_APPS = (
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'dslrapp',
)

MIDDLEWARE_CLASSES = (
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
)
```

```
ROOT_URLCONF = 'dslrproject.urls'

WSGI_APPLICATION = 'dslrproject.wsgi.application'


# Database
#
   https://docs.djangoproject.com/en/1.6/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE':
           'django.db.backends.postgresql_psycopg2',
        'NAME': 'dslrproject',
        'USER': 'pi',
        'PASSWORD': 'Pi.2013',
        'HOST': 'localhost',
        'PORT': '',
    }
}

# Internationalization
# https://docs.djangoproject.com/en/1.6/topics/i18n/

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'UTC'

USE_I18N = True

USE_L10N = True

USE_TZ = True


# Static files (CSS, JavaScript, Images)
#
   https://docs.djangoproject.com/en/1.6/howto/static-files/

STATIC_URL = '/static/'

STATIC_ROOT = os.path.join(BASE_DIR, 'static')

STATICFILES_DIRS = (
```

```
        "/var/www/dslrproject/dslrapp/static/",
)

TEMPLATE_DIRS = (
        os.path.join(BASE_DIR, 'templates'),
        '/var/www/dslrproject/dslrapp/templates/',
)
```

Creació base de dades PostgreSQL

```
su - postgres
    > psql acces to shell

    createdb dslrproject

    postgres@raspberrypi:~$ createuser -P
    Ingrese el nombre del rol a agregar: pi
    Ingrese la contrasena para el nuevo rol:
    Ingresela nuevamente:
    Sera a el nuevo rol un superusuario? (s/n) n
    Debe permitirsele al rol la creacion de bases
       de datos? (s/n) n
    Debe permitirsele al rol la creacion de otros
       roles? (s/n) n

    postgres@raspberrypi:~$ psql
    psql (9.1.12)
    Digite help para obtener ayuda.

    postgres=# GRANT ALL PRIVILEGES ON DATABASE
       dslrproject TO pi;
    GRANT
```

10.1.8 Creació i configuració de l'aplicació web dslrapp

Creació de models per l'aplicació dslrapp

```
from django.db import models

# Create your models here.
class Camera(models.Model):
        name = models.CharField(max_length=64)

        def __unicode__(self):
                return self.name

        class Meta:
                ordering = ['name']
class CameraConfig(models.Model):
        container = models.ForeignKey(Camera,
           db_index=True)
        key = models.CharField(max_length=64,
           db_index=True)
        value = models.CharField(max_length=1024,
           db_index=True)

        def __unicode__(self):
                return str(self.container.name) +"
                   - "+ self.key

        class Meta:
                ordering = ['container','key']

class CameraAbility(models.Model):
        container = models.ForeignKey(Camera,
           db_index=True)
        key = models.CharField(max_length=64,
           db_index=True)
        value = models.CharField(max_length=1024,
           db_index=True)

        def __unicode__(self):
                return str(self.container.name) +"
                   - "+ self.key
```

```
        class Meta:
                ordering = ['container','key']
```

Afegir models a l'administració del projecte Django

```
from django.contrib import admin
from dslrapp.models import *
# Register your models here.

admin.site.register(Camera)
admin.site.register(CameraConfig)
admin.site.register(CameraAbility)
```

Creació de URLs del projecte i de l'aplició

```
from django.conf.urls import patterns, include, url

from django.contrib import admin

admin.autodiscover()

urlpatterns = patterns('',
    # Examples:
    # url(r'^$', 'dslrproject.views.home',
       name='home'),
    # url(r'^blog/', include('blog.urls')),

    url(r'^admin/', include(admin.site.urls)),
    url(r'^', include('dslrapp.urls')),
)


from django.conf.urls import patterns, url

from dslrapp import views

urlpatterns = patterns('',
    url(r'^$', views.index, name='index'),
```

```
    url(r'^downloadHDR/(?P<namefile>.*)',
       views.downloadHDR, name='hdr'),
    url(r'^downloadTL/(?P<namefile>.*)',
       views.downloadTL, name='timelapse'),
    url(r'^downloadFS/(?P<namefile>.*)',
       views.downloadFS, name='focusstacking'),
)
```

Vista de l'aplicació

```
from django.shortcuts import render
from django.template import Context, loader
from django.http import HttpResponse
from django.middleware.csrf import get_token
from django.views.decorators.csrf import
   ensure_csrf_cookie
from django.shortcuts import render_to_response
import time
import camlib as pd
import os
import dbcontroller
import controller
import imgcontroller
import hdrmode
import timelapse
import focus

autodetect = False
liveviewon = False

@ensure_csrf_cookie
def index(request):

    if not controller.autodetect():
        return HttpResponse(loader.get_template(
           "dslrapp/unplug.html" ).render(
           Context({}) ) )
    if request.method == "POST":
        if request.POST['function'] == 'config':
            result =
               controller.set_value_on_config_by_name(
```

```
            request.POST['value'],
            request.POST['param'] )
        if result:
            return HttpResponse(
                request.POST['param']+" has been
                modified to
                "+request.POST['value'] )
        else:
            return HttpResponse(
                request.POST['param']+" has NOT
                been modified" )
    elif request.POST['function'] ==
       'captureshot':
        now =
            time.strftime("_%d_%m_%y_%H_%M_%S")
        mpath =
            '/static/dslrapp/images/preview'+now+'.jpg'
        try:
            path =
                controller.capture_image_on_sd_hd(
                '/var/www/dslrproject'+mpath )
            if path:
                if 'None' not in path:
                    imgcontroller.gethistogram('/var/www/dslrproje
                    return HttpResponse(
                        ".."+mpath+";Camera made
                        shoot and saved on
                        "+str(path) )
        except:
            pass
        return HttpResponse( "Camera failed
            making shoot" )

    elif request.POST['function'] == 'liveview':
        #path =
            controller.capture_view(liveviewon)
        #path =
            '/static/dslrapp/images/liveview.jpg'
        #if path is None:
        #liveviewon = False
        return HttpResponse( "Camera cannot
            take more capture views")
        #else:
        #liveviewon = True
```

```
        #return HttpResponse( path )

    elif request.POST['function'] ==
       'liveviewstop':
        #result = controller.capture_view_stop()
        #result = True
        #liveviewon = False
        #if result:
        return HttpResponse( "Liveview has been
           stopped" )
        #else:
        #return HttpResponse( "Error appears
           stopping Liveview" )

    elif request.POST['function'] == 'preview':
        timenow =
           time.strftime("_%d_%m_%y_%H_%M_%S")
        mpath =
           '/var/www/dslrproject/static/dslrapp/images/preview'+ti
        staticpath =
           '../static/dslrapp/images/preview'+timenow+'.jpg'
        path = None
        try:
            path = controller.capture_preview(
               mpath )
        except:
            return HttpResponse( 'Error appears
               doing Preview' )

        if path is not None:
            imgcontroller.gethistogram(mpath)
            return HttpResponse( staticpath )
        else:
            return HttpResponse( 'Error appears
               doing Preview' )

    elif request.POST['function'] ==
       'readvalue':
        value =
           controller.get_config_value_by_name(
           request.POST['param'] )
        if value:
            return HttpResponse( value )
        return HttpResponse(' ')
```

```
elif request.POST['function'] ==
   'capturehdr':

    arrayconfig = request.POST['array']
    arrayconfig = [ str(x) for x in
       arrayconfig.split(",") ]

    try:
        namefile =
           hdrmode.getHDR(nframeshdr,
           arrayconfig)
        if namefile:
            return HttpResponse(namefile)
    except:
        pass

    return HttpResponse( "Error appears
       doing HDR" )
elif request.POST['function'] ==
   'capturetimelapse':
    print
       request.POST['frame']+request.POST['inter']+request.POS
    intervals = str(request.POST['inter'])
    nframeslapse =
       str(request.POST['frame'])
    fps = str(request.POST['fps'])
    print nframeslapse+" "+intervals+" "+fps
    print
       'timelapse.getTimelapse(nframeslapse,
       intervals, fps)'
    try:
        namefile =
           timelapse.getTimelapse(nframeslapse,
           intervals, fps)
        if namefile:
            return HttpResponse(namefile)
    except:
        pass

    return HttpResponse( "Error appears
       doing Timelapse" )
elif request.POST['function'] ==
   'capturefocus':
```

```
            framestep =
               str(request.POST['framestep'])

            print framestep
            print
               'focus.getFocusstacking(framestep)'
            try:
                namefile =
                   focus.getFocusstacking(framestep)
                if namefile is not None:
                    return HttpResponse(namefile)
            except:
                pass

        else:
            return HttpResponse(
               request.POST['function']+" doesn't
               exist yet" )

    if request.method == "GET":

        model = controller.get_name_of_camera()
        if model is None:
            return
               HttpResponse(loader.get_template("dslrapp/unplug.html")
        listconfig =
           dbcontroller.get_config_by_name(model)
        if not listconfig:
            #set autoexposuremode to Manual to get
               all widgets
            camisregistered =
               dbcontroller.registercam()

            if camisregistered:
                listconfig =
                   dbcontroller.get_config_by_name(model)

        #init lists of camera widgets
        listtext = []
        listrange = []
        listtoggle = []
        listradio = []
        listmenu = []
        listbutton = []
```

```
listdate = []

listradiomod = []
canon = False
nikon = False
#this contents will be shown on view
if 'Canon' in model:
    listfilter = [ 'aeb','aperture',
       'colorspace',
       'exposure','focusmode',
       'imageformat', 'iso','manualfocus',
       'meter', 'shutterspeed',
       'whitebalance']
    canon = True
if 'Nikon' in model:
    listfilter = [ 'aeb','aperture',
       'colorspace',
       'exposure','focus','f-number',
       'imageformat', 'iso', 'meter',
       'shutterspeed', 'whitebalance']
    nikon = True

for filter in listfilter:
   get_element_by_tag( filter, listradio,
   listradiomod )
#Get values of camera ( ap, ex, iso and ss )
ap=None
mode = None
if canon:
    ap =
       controller.get_config_value_by_name(
       'aperture' )
    mode =
       controller.get_config_value_by_name(
       'autoexposuremode' )
if nikon:
    ap =
       controller.get_config_value_by_name(
       'f-number' )
    mode =
       controller.get_config_value_by_name('expprogram')
ex = controller.get_config_value_by_name(
   'exposurecompensation' )
```

```
iso = controller.get_config_value_by_name(
   'iso' )
ss = controller.get_config_value_by_name(
   'shutterspeed' )
bl = controller.get_config_value_by_name(
   'batterylevel' )
on = controller.get_config_value_by_name(
   'ownername' )
at = controller.get_config_value_by_name(
   'artist' )
ass = controller.get_config_value_by_name(
   'availableshots')
sc = controller.get_config_value_by_name(
   'shuttercounter')

if not ap : ap = ''
if not ex : ex = ''
if not iso: iso = ''
if not ss: ss = ''
if not mode: mode = ''
if not bl: bl = ''
if not on: on = ''
if not at: at = ''
if not ass: ass = ''
if not sc: sc = ''
date = time.strftime("%d/%m/%y  %H:%M:%S")

#Create array for HDRmode from DB
if 'Canon' in model:
    apvalues =
       dbcontroller.get_values_by_widgetname(model,'aperture')
if 'Nikon' in model:
    apvalues =
       dbcontroller.get_values_by_widgetname(model,'f-number')

exvalues =
   dbcontroller.get_values_by_widgetname(model,'exposurecomper
isovalues =
   dbcontroller.get_values_by_widgetname(model,'iso')
ssvalues =
   dbcontroller.get_values_by_widgetname(model,'shutterspeed')

template =
   loader.get_template("dslrapp/index.html")
```

```
        context = Context({
            'listtext': listtext,
            'listtoggle': listtoggle,
            'listradio': listradiomod,
            'model':model,
            'ap':ap,
            'ex':ex,
            'iso':iso,
            'ss':ss,
            'mode':mode,
            'date':date,
            'apvalues':apvalues,
            'exvalues':exvalues,
            'isovalues':isovalues,
            'ssvalues':ssvalues,
            'canon':canon,
            'nikon':nikon,
            'bl':bl,
            'on':on,
            'at':at,
            'ass':ass,
            'sc':sc,
        })
        return
           HttpResponse(template.render(context))


def get_element_by_tag( tag, list, listdest ):
    for i in list:
        if tag in i[0]:
            listdest.append(i)

def set_value_on_config_by_name( value, namewidget):
    try:
        result =
           controller.set_value_on_config_by_name(
           value, namewidget)
        cameraname = controller.get_name_of_camera()
        if result:
            return time.strftime("%d_%m_%y_%H_%M_%S
               :")+' Value '+value+ ' of
               '+namewidget+' has been changed on
               '+cameraname+' successfully'
```

```
            return time.strftime("%d_%m_%y_%H_%M_%S
               :")+' Value '+value+ ' of '+namewidget+'
               has not been changed on '+cameraname+',
               an error appears'
        except:
            return time.strftime("%d_%m_%y_%H_%M_%S
               :")+' Value '+value+ ' of '+namewidget+'
               has not been changed on '+cameraname+',
               an error appears'

    def downloadHDR(request, namefile):
        if namefile:
            hdrurl=
               '/var/www/dslrproject/static/dslrapp/images/hdrmode/'+namef
            response = HttpResponse( open(hdrurl,
               'rb').read(),
               content_type='application/zip')
            response['Content-Disposition'] =
               'attachment; filename='+namefile
            return response
        return HttpResponse('')

    def downloadTL( request, namefile):
        if namefile:
            hdrurl=
               '/var/www/dslrproject/static/dslrapp/images/timelapse/'+nam
            response = HttpResponse( open(hdrurl,
               'rb').read(),
               content_type='application/zip')
            response['Content-Disposition'] =
               'attachment; filename='+namefile
            return response
        return HttpResponse('')

    def downloadFS( request, namefile):
        if namefile:
            hdrurl=
               '/var/www/dslrproject/static/dslrapp/images/focus/'+namefil
            response = HttpResponse( open(hdrurl,
               'rb').read(),
               content_type='application/zip')
            response['Content-Disposition'] =
               'attachment; filename='+namefile
            return response
```

```
    return HttpResponse('')
```

Templates de l'aplicació

Template quan no es detecta cap càmera connectada:

```
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="refresh" content="7">
        <link rel="stylesheet"
           href="../static/dslrapp/css/foundation-icons.css">
        <link rel="stylesheet"
           href="../static/dslrapp/css/foundation.css">
        <link rel="stylesheet"
           href="../static/dslrapp/css/lightstyle.css">
    </head>
    <body>
        <div "row" style="text-align:
           center;margin-top: 5% !important;">
           <h1>DSLR Project</h1>
           <h3>University of Barcelona</h2>
           <h5>designed and implemented by RSO</h3>
           <br><br>
           <ul class="button-group">
                <li style="width:100%
                   !important;"><a class="button
                   expand fi-camera buttonicon"
                   href=""><br>PLEASE, PLUG A
                   CAMERA AND ENJOY</a></li>
            </ul>
        </div>
    </body>
</html>
```

Template de la interfície:

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet"
   href="../static/dslrapp/css/foundation-icons.css">
```

```
<link rel="stylesheet"
   href="../static/dslrapp/css/foundation.css">
<link rel="stylesheet"
   href="../static/dslrapp/css/lightstyle.css">
<script
   src="../static/dslrapp/js/vendor/modernizr.js"></script>
</head>
<body>
<div class="row fullWidth">
        <div class="large-8 columns">
                <div class="row">
            <div class="large-3 columns
               nopaddingmargin">
                <div class="row">
                    <div class="large-3 columns
                       nopaddingmargin widthquart">
                        {% if canon %}
                        <p id="statusap"
                           >f/{{ap}}</p>
                        {% endif %}
                        {% if nikon %}
                        <p id="statusap" >{{ap}}</p>
                        {% endif%}
                    </div>
                    <div class="large-3 columns
                       nopaddingmargin widthquart">
                        <p
                           id="statusex">{{ex}}EV</p>
                    </div>
                    <div class="large-3 columns
                       nopaddingmargin widthquart">
                        <p
                           id="statusiso">{{iso}}</p>
                    </div>
                    <div class="large-3 columns
                       nopaddingmargin widthquart"
                       style="margin-left: -10px
                       !important;">
                        {% if canon %}
                        <p
                           id="statusss">{{ss}}&#39;&#39;</p>
                        {% endif %}
                        {% if nikon %}
                        <p id="statusss">{{ss}}</p>
```

```
                {% endif%}
            </div>
        </div>
        <div class="row" style="text-align:
           center;">
            <div class="large-8 columns
               nopaddingmargin"
               style="width:65% !important">
                <p
                   id="statusdate">{{date}}</p>
            </div>
            <div class="large-4 columns
               nopaddingmargin widthquart"
               style="width:35% !important">
                <p
                   id="statusmode">{{mode}}</p>
            </div>
        </div>
    </div>
    <div class="large-9 columns
       nopaddingmargin">
        <ul class="button-group">
        <li><a class="button expand
           fi-play-video buttonicon"
           data-dropdown="liveviewhover"
           data-options="is_hover:true"
           onclick="buttonLiveView()"></a></li>
        <li><a class="button expand fi-eye
           buttonicon"
           data-dropdown="previewhover"
           data-options="is_hover:true"onclick="buttonPreView(
        <li><a
           data-reveal-id="screenimgmodal"
           class="button expand fi-zoom-in
           buttonicon"
           data-dropdown="zoomhover"
           data-options="is_hover:true"></a></li>
        <li><a class="button expand
           fi-target-two buttonicon"
           onclick="makeShot();"
           data-dropdown="makeshothover"
           data-options="is_hover:true"></a></li>
            <ul id="liveviewhover"
               class="f-dropdown"
```

```
                    data-dropdown-content>
                     <li style="width:auto
                        !important;"
                        ><a>Liveview</a></li>
                  </ul>
                  <ul id="previewhover"
                     class="f-dropdown"
                     data-dropdown-content>
                     <li style="width:auto
                        !important;"><a>Preview</a></li>
                  </ul>
                  <ul id="zoomhover"
                     class="f-dropdown"
                     data-dropdown-content>
                     <li style="width:auto
                        !important;"><a>Zoom
                        In</a></li>
                  </ul>
                  <div id="screenimgmodal"
                     class="reveal-modal full
                     nopaddingmargin noborder"
                     data-reveal>
                     <img src="" width="100%"
                        height="100%"/>
                     <a
                        class="close-reveal-modal">&#215;</a>
                  </div>
                  <ul id="makeshothover"
                     class="f-dropdown"
                     data-dropdown-content>
                     <li style="width:auto
                        !important;"><a>Shot!</a></li>
                  </ul>
                  <div id="workinprogress"
                     class="reveal-modal tiny"
                     data-reveal>
                     <h2>Work in progress</h2>
                     <p
                        style="color:black">Please
                        wait a few seconds...</p>
                     <a
                        class="close-reveal-modal">&#215;</a>
                  </div>
               </ul>
```

```
    </div>
        </div>
        <div class="row">
    <ul class="button-group">
    <li><img id="histl" class="button
       expand nopaddingmargin histogram
       noborder"
       src="../static/dslrapp/images/histogramgrey.png?r=1"></
    <li><img id="histr" class="button
       expand nopaddingmargin histogram
       noborder"
       src="../static/dslrapp/images/histogramred.png?r=1"></l
    <li><img id="histg" class="button
       expand nopaddingmargin histogram
       noborder"
       src="../static/dslrapp/images/histogramgreen.png?r=1"><
    <li><img id="histb" class="button
       expand nopaddingmargin histogram
       noborder"
       src="../static/dslrapp/images/histogramblue.png?r=1"></
    </ul>
</div>
        <div class="row">
                <div class="screen" />
                <img id="screen_img"
                   onload="onLoadScreen();"
                   style="width:100%;height:100%"></img>
                </div>
        </div>
        <div class="row log">
                <div id="log">
                        <p
                           style="color:black">Log</p>
    </div>
</div>
</div>
<div class="large-4 columns">
                <dl class="tabs" data-tab>
                        <dd
                           class="nactive"><a
                           href="#panel2-1"
                           onclick="setMode(0);">Normal
                           Mode</a></dd>
```

```
            <dd
               class="border-left"><a
               href="#panel2-2"
               onclick="setMode(1);">HDR
               Mode</a></dd>
            <dd
               class="border-up"><a
               href="#panel2-3"
               onclick="setMode(2);">Time
               - lapse</a></dd>
            <dd
               class="border-up
               border-left" ><a
               href="#panel2-4"
               onclick="setMode(3);">Focus
               stacking</a></dd>
</dl>
<div class="tabs-content">
          <div
             class="content
             active control"
             id="panel2-1">
                 <dl
                    class="accordion"
                    data-accordion>
                          {%
                             for
                             name,values
                             in
                             listradio
                             %}
                          <dd>
    <a
       href="#{{name}}">{{name}}</a>
                          {%
                             for
                             value
                             in
                             values
                             %}
                                <div
                                   id="{{name}
                                   class="cont
                                   ucontent">
```

```
                <a name="{{name}}"
                   onclick="change_value_widget(this);
            </div>
                                {%
                                   endfor
                                   %}
                                </dd>
                                {%
                                   endfor
                                   %}
                <div class="content
                   control"
                   id="panel2-2">
                        <div
                           class="row"
                           >
                                <div
                                   class="large-2
                                   columns
                                   nopaddingmargin
                                   centerhdr">
                                        <a>ID</a>
                                </div>
<div class="large-2 columns
   nopaddingmargin
   centerhdr">
                                        <a>AP</a>
                                </div>
                                <div
                                   class="large-2
                                   columns
                                   nopaddingmargin
                                   centerhdr">
                                        <a>EXP</a>
                                </div>
                                <div
                                   class="large-2
                                   columns
                                   nopaddingmargin
                                   centerhdr">
                                        <a>ISO</a>
                                </div>
```

```
                                <div
                                   class="large-2
                                   columns
                                   nopaddingmargin
                                   centerhdr">
                                        <a>SS</a>
                                </div>
<div class="large-2 columns
   nopaddingmargin
   centerhdr">
                                        <a>Del</a>
                                </div>
                        </div>
                        <div
                           id="hdrelements"
                           class="row">
                        </div>
                        <div
                           class="row">
                                <a
                                   class="button
                                   expand
                                   fi-plus
                                   nopaddingmargin
                                   centerhdr"
                                   style="padding-top:
                                   6px
                                   !important;"onclick
                        </div>
                </div>
                <div class="content
                   control"
                   id="panel2-3">
                        <p>Interval
                           in
                           seconds</p><input
                           id="intervaltimelapse"
                           type="text"
                           required
                           pattern="number">
<p>Number of
   frames</p><input
   id="nframestimelapse"
   type="text" required
```

```
   pattern="number">
<p>FPS</p><input
   id="fpstimelapse"
   type="text" required
   pattern="number">
        </div>
        <div class="content
           control"
           id="panel2-4">

<p>Distance:</p>
<br>
<div class="row">
    <div class="large-4
       columns
       nopaddingmargin"
       style="width: 33.33%
       !important">
        <input type="radio"
           name="distance"
           value="1"
           id="dist1"
           class="focus">
        <label for="dist1"
           class="focus
           fi-mountains"><p>Far</p></label>
    </div>
    <div class="large-4
       columns
       nopaddingmargin"
       style="width: 33.33%
       !important" >
        <input type="radio"
           name="distance"
           value="2"
           id="dist2"
           class="focus">
        <label for="dist2"
           class="fi-home
           focus"><p>Middle</p></label>
    </div>
    <div class="large-4
       columns
       nopaddingmargin"
```

```
                                    style="width: 33.33%
                                    !important" >
                                     <input type="radio"
                                        name="distance"
                                        value="3"
                                        id="dist3"
                                        class="focus">
                                     <label for="dist3"
                                        class="fi-torsos-all
                                        focus"><p>Near</p></label>
                                </div>
                            </div>
                                    </div>
                            </div>
                     <div class="row status">
                <p style="color:black">Status</p>
                <p id="model">Camera Model:
                   {{model}}</p>
                <p id="blvl">Battery Level: {{bl}}</p>
                <p id="oname">Ownername: {{on}}</p>
                <p id="artist">Artist: {{at}}</p>
                <p id="ashots">Available shots:
                   {{ass}}</p>
                <p id="scounter">Shutter counter:
                   {{sc}}</p>
            </div>
            </div>
<div>
</body>
<script
   src="../static/dslrapp/js/dslrajaxcontroller.js"></script>
<script
   src="../static/dslrapp/js/vendor/jquery.js"></script>
<script
   src="../static/dslrapp/js/foundation/foundation.js"></script>
<script
   src="../static/dslrapp/js/foundation/foundation.accordion.js"></scr
<script
   src="../static/dslrapp/js/foundation/foundation.tab.js"></script>
<script
   src="../static/dslrapp/js/foundation/foundation.reveal.js"></script
<script
   src="../static/dslrapp/js/foundation/foundation.dropdown.js"></scri
```

```
<script
   src="../static/dslrapp/js/foundation/foundation.abide.js"></script>
<script>
    $(document).foundation({
        accordion: {
            // specify the class used for active
               (or open) accordion panels
            active_class: 'active',
            // allow multiple accordion panels to
               be active at the same time
            multi_expand: false,
            // allow accordion panels to be closed
               by clicking on their headers
            // setting to false only closes
               accordion panels when another is
               opened
            toggleable: true
        }
    });
</script>
<script>
    $(document).foundation({
        tab: {
            callback : function (tab) {
                console.log(tab);
            }
        }
    });
</script>
<script>
     $(document).foundation({
        dropdown: {
            // specify the class used for active
               dropdowns
            active_class: 'open'
        }
    });
</script>
<script>
    $(document).on('ready',function(){
        $("div.ucontent").click( function(){
            $(this).parent().children('a')[0].click();
         });
    });
```

```
</script>
<script>
    $(document).foundation({
        abide : {
            patterns: {
                number: /^[-+]?[1-9]\d*$/,
            }
        }
    });
</script>
<script>
var canon = "{{canon}}";
var nikon = "{{nikon}}";
    function addElementHDR(){
        var divelement =
           document.createElement('div');
        divelement.id = 'hdrelement'+hdrelements;
        divelement.className = 'row';
        divelement.innerHTML = '<div class="large-2
           columns nopaddingmargin centerhdr"><p
           style="margin-top:5px
           !important;">'+hdrelements+'</p></div><div
           class="large-2 columns nopaddingmargin
           centerhdr"><select>{% for val in
           apvalues%}<option
           value="{{val}}">{{val}}</option>{%
           endfor%}</select></div><div
           class="large-2 columns nopaddingmargin
           centerhdr"><select >{% for val in
           exvalues%}<option
           value="{{val}}">{{val}}</option>{%
           endfor%}</select></div><div
           class="large-2 columns nopaddingmargin
           centerhdr"><select>{% for val in
           isovalues%}<option
           value="{{val}}">{{val}}</option>{%
           endfor%}</select></div><div
           class="large-2 columns nopaddingmargin
           centerhdr"><select>{% for val in
           ssvalues%}<option
           value="{{val}}">{{val}}</option>{%
           endfor%}</select></div><div
           class="large-2 columns nopaddingmargin
           centerhdr"><a class="button expand
```

```
            fi-minus nopaddingmargin centerhdr"
            style="padding-top: 8px !important;"
            onclick="deleteElementHDR(this);"></a></div>';
        hdrelements +=1;
        document.getElementById('hdrelements').appendChild(divelement)
    }
</script>
<script>
window.setInterval(function(){setTime();},1000);
</script>
<script>
function workEnable(){
    $('#workinprogress').foundation('reveal',
       'open', '');
}
function workDisable(){
    $('#workinprogress').foundation('reveal',
       'close', '');
}
</script>
</html>
```

Fitxers estàtics de l'aplicació

A més a més del fitxer mostrar és necessari tenir la llibreria de Foundation 5.

```
var liveview = null;
var imageliveview = null;
var mode = 0;
var loadingimg = false;
var completedimg = true;
var hdrelements = 1;

function isNumber(n) {
      return !isNaN(parseFloat(n)) && isFinite(n);
}

function buttonPreView(){
    var xmlhttp;
    if (window.XMLHttpRequest)
    {
        /* code for IE7+, Firefox, Chrome, Opera,
           Safari*/
```

```
        xmlhttp=new XMLHttpRequest();
    }
    else
    {/* code for IE6, IE5*/
        xmlhttp=new
           ActiveXObject("Microsoft.XMLHTTP");
    }

    xmlhttp.onreadystatechange=function()
    {
        if (xmlhttp.readyState==4 &&
           xmlhttp.status==200)
        {
            if
               (xmlhttp.responseText.indexOf("preview")
               > -1){
                document.getElementById("screen_img").src
                   = xmlhttp.responseText;
                document.getElementById("screen_img").style.visibility
                   = "visible";
                document.getElementById("screenimgmodal").childNodes[1
                   = xmlhttp.responseText;
                updatehistrograms();
                document.getElementById("log").innerHTML+=
                   "Preview is done:
                   "+xmlhttp.responseText+"<br>";
            }else{
                document.getElementById("log").innerHTML+=
                   xmlhttp.responseText+"<br>";
            }
            setShots();
            workDisable();
        }
    };
    xmlhttp.open("POST",document.URL,true);
    xmlhttp.setRequestHeader("Content-type","application/x-www-form-ur
    xmlhttp.setRequestHeader("X-CSRFToken",getCookie('csrftoken'));
    xmlhttp.send("function=preview");
    workEnable();
}

function updatehistrograms(){
```

```
    document.getElementById('histl').src =
       document.getElementById('histl').src+'1';
    document.getElementById('histr').src =
       document.getElementById('histr').src+'1';
    document.getElementById('histg').src =
       document.getElementById('histg').src+'1';
    document.getElementById('histb').src =
       document.getElementById('histb').src+'1';

}

function buttonLiveView(){
    if ( liveview !== null ){
        document.getElementById("blv").style.color="red";
        window.clearInterval(liveview);
        document.getElementById("screen").style.backgroundColor="black
        document.getElementById("screen_img").style.visibility="hidden
        liveview = null;
        stop_capture_image();
        imageliveview = null;
        loadingimg = false;
    }else{
        document.getElementById("blv").style.color="green";
        document.getElementById("screen_img").style.visibility="visibl
        liveview =
           window.setInterval(function(){load_image();},2000);
    }
}

function stop_capture_image(){
    var xmlhttp;
    if (window.XMLHttpRequest)
    {
        /* code for IE7+, Firefox, Chrome, Opera,
           Safari*/
        xmlhttp=new XMLHttpRequest();
    }
    else
    {/* code for IE6, IE5*/
        xmlhttp=new
           ActiveXObject("Microsoft.XMLHTTP");
    }

    xmlhttp.onreadystatechange=function()
```

```
    {
        if (xmlhttp.readyState==4 &&
           xmlhttp.status==200)
        {
            document.getElementById("log").innerHTML+=xmlhttp.response
        }
    };
    xmlhttp.open("POST",document.URL,true);
    xmlhttp.setRequestHeader("Content-type","application/x-www-form-ur
    xmlhttp.setRequestHeader("X-CSRFToken",getCookie('csrftoken'));
    xmlhttp.send("function=liveviewstop");
}

function onLoadScreen() {
    completedimg = true;
}

function load_image(){
    if ( loadingimg === true ){
        if ( completedimg === true){
            completedimg = false;
            var d = new Date();
            /*var randomnumber =
               Math.floor(Math.random()*1000+1);*/
            if (imageliveview !== null ){
                /*document.getElementById("screen_img").src
                   =
                   imageliveview+"?time="+d.getTime();*/
                document.getElementById("log").innerHTML+=
                   "NEW SNAP WITH LIVEVIEW<br>";
            }
        }
    }else{
        capture_image();
        loadingimg = true;
    }
}
function capture_image(){
    var xmlhttp;
    if (window.XMLHttpRequest)
    {
        /* code for IE7+, Firefox, Chrome, Opera,
           Safari*/
        xmlhttp=new XMLHttpRequest();
```

```
    }
    else
    {/* code for IE6, IE5*/
        xmlhttp=new
           ActiveXObject("Microsoft.XMLHTTP");
    }

    xmlhttp.onreadystatechange=function()
    {
        if (xmlhttp.readyState==4 &&
           xmlhttp.status==200)
        {
            if
               (xmlhttp.responseText.indexOf("liveview")
               > -1){
                /*var randomnumber =
                   Math.floor(Math.random()*1000+1);*/
                var d = new Date();
                imageliveview =
                   xmlhttp.responseText;
                document.getElementById("screen_img").src
                   =
                   imageliveview+"?time="+d.getTime();
                document.getElementById("log").innerHTML+=
                   "LiveView is started <br>";
            }else{
                document.getElementById("log").innerHTML+=
                   xmlhttp.responseText+"<br>";
                buttonLiveView();
            }
        }
    };
    xmlhttp.open("POST",document.URL,true);
    xmlhttp.setRequestHeader("Content-type","application/x-www-form-ur
    xmlhttp.setRequestHeader("X-CSRFToken",getCookie('csrftoken'));
    xmlhttp.send("function=liveview");
}

function makeShot(){

    var xmlhttp;
    if (window.XMLHttpRequest)
    {
```

```
    /* code for IE7+, Firefox, Chrome, Opera,
       Safari*/
    xmlhttp=new XMLHttpRequest();
}
else
{/* code for IE6, IE5*/
    xmlhttp=new
       ActiveXObject("Microsoft.XMLHTTP");
}

xmlhttp.onreadystatechange=function()
{
    if ( xmlhttp.readyState==4  ){
        workDisable();
    }
    if (xmlhttp.readyState==4 &&
       xmlhttp.status==200)
    {
        switch(mode){
            case 1:
                if(
                   !(xmlhttp.responseText.indexOf("Error")
                   > -1) &&
                   !(xmlhttp.responseText.indexOf("fail")
                   > -1) ){
                    window.open("downloadHDR/"+xmlhttp.responseTex
                }
                document.getElementById("log").innerHTML+=
                   "HDR Completed:
                   "+xmlhttp.responseText+"<br>";
                break;
            case 2:
                if(
                   !(xmlhttp.responseText.indexOf("Error")
                   > -1) &&
                   !(xmlhttp.responseText.indexOf("fail")
                   > -1)){
                    window.open("downloadTL/"+xmlhttp.responseText
                }
                document.getElementById("log").innerHTML+=
                   "Timelapse Completed:
                   "+xmlhttp.responseText+"<br>";
                break;
            case 3:
```

```
                    if(
                       !(xmlhttp.responseText.indexOf("Error")
                       > -1) &&
                       !(xmlhttp.responseText.indexOf("fail")
                       > -1)){
                        window.open("downloadFS/"+xmlhttp.responseText
                    }
                    document.getElementById("log").innerHTML+=
                       "Focus stacking Completed:
                       "+xmlhttp.responseText+"<br>";
                    break;
                default:
                    if(
                       !(xmlhttp.responseText.indexOf("fail")
                       > -1) &&
                       !(xmlhttp.responseText.indexOf("Error")
                       > -1) ){
                        document.getElementById("screen_img").src
                           =
                           xmlhttp.responseText.split(";")[0];
                        document.getElementById("screen_img").style.vi
                           = "visible";
                        document.getElementById("screenimgmodal").chil
                           =
                           xmlhttp.responseText.split(";")[0];
                        updatehistrograms();
                        document.getElementById("log").innerHTML+=
                           xmlhttp.responseText.split(";")[1]+"<br>";
                    }else{
                        document.getElementById("log").innerHTML+=
                           xmlhttp.responseText+"<br>";
                    }
                    break;
            }
            setShots();
            setConfig();
        }
};
xmlhttp.open("POST",document.URL,true);
xmlhttp.setRequestHeader("Content-type","application/x-www-form-ur
xmlhttp.setRequestHeader("X-CSRFToken",getCookie('csrftoken'));
switch(mode){
    case 1:
        var frames = getElementsHDRSize();
```

```
        if ( frames > 0 ){
            var array = getElementsHDR();
            workEnable();
            xmlhttp.send("function=capturehdr&frames="+frames+"&ar
        }else{
            alert( 'Error in input value' );
            return;
        }
        break;
    case 2:
        var inter =
           document.getElementById('intervaltimelapse').value;
        var num =
           document.getElementById('nframestimelapse').value;
        var fps =
           document.getElementById('fpstimelapse').value;
        if ( isNumber(inter) && isNumber(num)
           && isNumber(fps) ){
            workEnable();
            xmlhttp.send("function=capturetimelapse&inter="+inter+
        }else{
            alert( 'Error in input value' );
            return;
        }
        break;
    case 3:
        var dist1 =
           document.getElementById('dist1');
        var dist2 =
           document.getElementById('dist2');
        var dist3 =
           document.getElementById('dist3');

        if ( dist1.checked ){
            workEnable();
            xmlhttp.send("function=capturefocus&framestep=1");
        }else if (dist2.checked){
            workEnable();
            xmlhttp.send("function=capturefocus&framestep=2");
        }else if (dist3.checked){
            workEnable();
            xmlhttp.send("function=capturefocus&framestep=3");
        }else{
            alert( 'Error in input value' );
```

```
                return;
            }
            break;
        default:
            workEnable();
            xmlhttp.send("function=captureshot");
            /*changepreview = 1;*/
            break;
        }

}
function getCookie(name) {
    var cookieValue = null;
    if (document.cookie && document.cookie !== '') {
        var cookies = document.cookie.split(';');
        for (var i = 0; i < cookies.length; i++) {
            var cookie = cookies[i].trim();
            if (cookie.substring(0, name.length +
               1) == (name + '=')) {
                cookieValue =
                   decodeURIComponent(cookie.substring(name.length
                   + 1));
                break;
            }
        }
    }
    return cookieValue;
}

function change_value_widget(element)
{
    var xmlhttp;
    if (window.XMLHttpRequest)
    {
        /* code for IE7+, Firefox, Chrome, Opera,
           Safari*/
        xmlhttp=new XMLHttpRequest();
    }
    else
    {/* code for IE6, IE5*/
        xmlhttp=new
           ActiveXObject("Microsoft.XMLHTTP");
    }
```

```
    xmlhttp.onreadystatechange=function()
    {
        if (xmlhttp.readyState==4 &&
           xmlhttp.status==200)
        {
            document.getElementById("log").innerHTML+=xmlhttp.response
            setConfig();
        }
    };
    xmlhttp.open("POST",document.URL,true);
    xmlhttp.setRequestHeader("Content-type","application/x-www-form-ur
    xmlhttp.setRequestHeader("X-CSRFToken",getCookie('csrftoken'));
    var value = new String(element.innerHTML);
    value = value.substring(2, value.length-1 );
    xmlhttp.send("function=config&param="+element.name+"&value="+value

}
function setValueStatus(id, param, htmltext)
{
    var xmlhttp;
    if (window.XMLHttpRequest)
    {
        /* code for IE7+, Firefox, Chrome, Opera,
           Safari*/
        xmlhttp=new XMLHttpRequest();
    }
    else
    {/* code for IE6, IE5*/
        xmlhttp=new
           ActiveXObject("Microsoft.XMLHTTP");
    }

    xmlhttp.onreadystatechange=function()
    {
        if (xmlhttp.readyState==4 &&
           xmlhttp.status==200)
        {
            if ( xmlhttp.responseText !== ' ' ){
                if (htmltext === 'EV' || htmltext
                   === '\'\'' ){
                    document.getElementById(id).innerHTML=
                       xmlhttp.responseText+htmltext;
                }else{
```

```
                    document.getElementById(id).innerHTML=
                        htmltext+xmlhttp.responseText;
                }
            }
        }
    };
    xmlhttp.open("POST",document.URL,true);
    xmlhttp.setRequestHeader("Content-type","application/x-www-form-ur
    xmlhttp.setRequestHeader("X-CSRFToken",getCookie('csrftoken'));
    xmlhttp.send("function=readvalue&param="+param);

}

function setTime(){
    var now = new Date();
    var dd = now.getDate();
    var mm = now.getMonth()+1;
    var yy =
       now.getFullYear().toString().substring(2);
    var HH = now.getHours();
    var MM = now.getMinutes();
    var SS = now.getSeconds();

    document.getElementById('statusdate').innerHTML
       = dd+"/"+mm+"/"+yy+" "+HH+":"+MM+":"+SS;

}

function setConfig(){

    if (canon){
        setValueStatus('statusap','aperture', 'f/');
    }else if (nikon){
        setValueStatus('statusap','f-number', '');
    }
    sleep(1750);
    setValueStatus('statusex','exposurecompensation',
       'EV');
    sleep(1750);
    setValueStatus('statusiso','iso', '');
    sleep(1750);
    setValueStatus('statusss','shutterspeed',
       '\'\'');
    sleep(1750);
```

```
    if (canon){
        setValueStatus('statusmode','autoexposuremode','');
    }else if (nikon){
        setValueStatus('statusmode','expprogram','');
    }
}

function setShots(){
    setValueStatus('ashots','availableshots','Available
       shots: ');
    sleep(1750);
    setValueStatus('scounter','shuttercounter','Shutter
       counter: ');
    sleep(1750);
    setValueStatus('blvl','batterylevel', 'Battery
       Level: ');
}

function sleep(ms)
{
    var dt = new Date();
    dt.setTime(dt.getTime() + ms);
    while (new Date().getTime() < dt.getTime());
}

function deleteElementHDR(element){
    if (hdrelements > 1 ){
        document.getElementById('hdrelements').removeChild(element.par
        hdrelements -= 1;
    }
}
function setMode(n){
    mode = n;
}
function getElementsHDRSize(){
    return
       document.getElementById('hdrelements').childNodes.length-1;
}
function getElementsHDR(){
    var n = getElementsHDRSize();
    var array = new Array();
    for ( var i = 1 ; i <= n ; i++ ){
        var hdrelement =
           document.getElementById('hdrelement'+i);
```

```
        var selectap =
           hdrelement.childNodes[1].childNodes[0];
        array.push(selectap.options[selectap.selectedIndex].value);
        var selectex =
           hdrelement.childNodes[2].childNodes[0];
        array.push(selectex.options[selectex.selectedIndex].value);
        var selectiso =
           hdrelement.childNodes[3].childNodes[0];
        array.push(selectiso.options[selectiso.selectedIndex].value);
        var selectss =
           hdrelement.childNodes[4].childNodes[0];
        array.push(selectss.options[selectss.selectedIndex].value);
    }
    return array;
}
```

10.1.9 Integració Django a producción amb el servidor web Apache2

Configuració wsgi.py del projecte

```
"""
WSGI config for dslrproject project.

It exposes the WSGI callable as a module-level
   variable named ``application``.

For more information on this file, see
https://docs.djangoproject.com/en/1.6/howto/deployment/wsgi/
"""

import os
os.environ.setdefault("DJANGO_SETTINGS_MODULE",
   "dslrproject.settings")

from django.core.wsgi import get_wsgi_application
application = get_wsgi_application()
```

Instal·lació del servidor web Apache2 i mòdul WSGI:

```
apt-get install apache2
```

```
root@raspberrypi:~/mod_wsgi-3.4# apt-get install
   apache2-threaded-dev


wget
   https://modwsgi.googlecode.com/files/mod_wsgi-3.4.tar.gz
root@raspberrypi:~# tar xvfz mod_wsgi-3.4.tar.gz
mod_wsgi-3.4/
mod_wsgi-3.4/.hgignore
mod_wsgi-3.4/.hgtags
mod_wsgi-3.4/configure
mod_wsgi-3.4/configure.ac
mod_wsgi-3.4/LICENCE
mod_wsgi-3.4/mod_wsgi.c
mod_wsgi-3.4/posix-ap1X.mk.in
mod_wsgi-3.4/posix-ap2X.mk.in
mod_wsgi-3.4/README
mod_wsgi-3.4/win32-ap22py26.mk
mod_wsgi-3.4/win32-ap22py27.mk
mod_wsgi-3.4/win32-ap22py31.mk


root@raspberrypi:~# cd mod_wsgi-3.4
root@raspberrypi:~/mod_wsgi-3.4# ./configure
compile
root@raspberrypi:~/mod_wsgi-3.4#make
root@raspberrypi:~/mod_wsgi-3.4#make install
```

Configuració /etc/apache2/apache2.conf

```
LoadModule wsgi_module
   /usr/lib/apache2/modules/mod_wsgi.so

WSGIScriptAlias /
   /var/www/dslrproject/dslrproject/wsgi.py
WSGIPythonPath /var/www/dslrproject

<Directory /var/www/dslrproject/dslrapp>
<Files wsgi.py>
Allow from all
Order deny,allow
```

```
</Files>
</Directory>
```

10.1.10 Instal·lació FFmpeg

```
install ffmpeg an compiling in rasp

sudo apt-get install git
git clone git://source.ffmpeg.org/ffmpeg.git
cd ffmpeg
./configure
sudo make && make install
```

10.1.11 Modificació de permisos del USB per accedir amb l'Apache2

```
Changed the permission on file:
   /lib/udev/rules.d/91-permissions.rules

# usbfs-like devices
#SUBSYSTEM=="usb", ENV{DEVTYPE}=="usb_device",
   MODE="0664"
SUBSYSTEM=="usb", ENV{DEVTYPE}=="usb_device",
   MODE="0666"
```

10.1.12 Llibreria Gphoto2 en Python

```
# Copyright (C) 2014 Ruben Serrano
# Based on:
# - a small code example by Mario Boikov,
   http://pysnippet.blogspot.com/2009/12/when-ctypes-comes-to-rescue.h
# - libgphoto2 Python bindings by David PHAM-VAN
   <david@ab2r.com>
```

```
# - ctypes_gphoto2.py by Hans Ulrich Niedermann
   <gp@n-dimensional.de>
# - piggyphoyo by Alexdu
   https://github.com/alexdu/piggyphoto

# Some functions return errors which can be fixed
   by retrying.
# For example, capture_preview on Canon 550D fails
   the first time, but subsequent calls are OK.
# Retries are performed on: camera.capture_preview,
   camera.capture_image and camera.init()
retries = 2

# This is run if gp_camera_init returns -60 (Could
   not lock the device) and retries >= 1.
#unmount_cmd = 'gvfs-mount -s gphoto2'

# With this script you can reset a usb if you know
   its ID_Vendor and ID_Product:
unmount_cmd =  '/var/www/dslrproject/resetusb'

libgphoto2dll = 'libgphoto2.so'

import re
import ctypes
gp = ctypes.CDLL(libgphoto2dll)
context = gp.gp_context_new()

def library_version(verbose = True):
    gp.gp_library_version.restype =
       ctypes.POINTER(ctypes.c_char_p)
    if not verbose:
        arrText =
           gp.gp_library_version(GP_VERSION_SHORT)
    else:
        arrText =
           gp.gp_library_version(GP_VERSION_VERBOSE)

    v = ''
    for s in arrText:
        if s is None:
            break
        v += '%s\n' % s
    return v
```

```
import os, string, time
from ptp import *

PTR = ctypes.pointer

# gphoto structures
""" From 'gphoto2-camera.h'
typedef struct {
        char name [128];
        char folder [1024];
} CameraFilePath;
"""
class CameraFilePath(ctypes.Structure):
    _fields_ = [('name', (ctypes.c_char * 128)),
                ('folder', (ctypes.c_char * 1024))]

class CameraText(ctypes.Structure):
    _fields_ = [('text', (ctypes.c_char * (32 *
       1024)))]

#cdef extern from "gphoto2/gphoto2-port-version.h":
#  ctypedef enum GPVersionVerbosity:
GP_VERSION_SHORT = 0
GP_VERSION_VERBOSE = 1

#cdef extern from
   "gphoto2/gphoto2-abilities-list.h":
#  ctypedef enum CameraDriverStatus:
GP_DRIVER_STATUS_PRODUCTION = 0
GP_DRIVER_STATUS_TESTING = 1
GP_DRIVER_STATUS_EXPERIMENTAL = 2
GP_DRIVER_STATUS_DEPRECATED = 3

#  ctypedef enum CameraOperation:
GP_OPERATION_NONE = 0
GP_OPERATION_CAPTURE_IMAGE = 1
GP_OPERATION_CAPTURE_VIDEO = 2
GP_OPERATION_CAPTURE_AUDIO = 3
GP_OPERATION_CAPTURE_PREVIEW = 4
GP_OPERATION_CONFIG = 5

#  ctypedef enum CameraFileOperation:
GP_FILE_OPERATION_NONE = 0
```

```
GP_FILE_OPERATION_DELETE = 1
GP_FILE_OPERATION_PREVIEW = 2
GP_FILE_OPERATION_RAW = 3
GP_FILE_OPERATION_AUDIO = 4
GP_FILE_OPERATION_EXIF = 5

#  ctypedef enum CameraFolderOperation:
GP_FOLDER_OPERATION_NONE = 0
GP_FOLDER_OPERATION_DELETE_ALL = 1
GP_FOLDER_OPERATION_PUT_FILE = 2
GP_FOLDER_OPERATION_MAKE_DIR = 3
GP_FOLDER_OPERATION_REMOVE_DIR = 4

#cdef extern from
   "gphoto2/gphoto2-port-info-list.h":
#  ctypedef enum GPPortType:
GP_PORT_NONE = 0
GP_PORT_SERIAL = 1
GP_PORT_USB = 2


class CameraAbilities(ctypes.Structure):
    _fields_ = [('model', (ctypes.c_char * 128)),
                ('status', ctypes.c_int),
                ('port', ctypes.c_int),
                ('speed', (ctypes.c_int * 64)),
                ('operations', ctypes.c_int),
                ('file_operations', ctypes.c_int),
                ('folder_operations', ctypes.c_int),
                ('usb_vendor', ctypes.c_int),
                ('usb_product', ctypes.c_int),
                ('usb_class', ctypes.c_int),
                ('usb_subclass', ctypes.c_int),
                ('usb_protocol', ctypes.c_int),
                ('library', (ctypes.c_char * 1024)),
                ('id', (ctypes.c_char * 1024)),
                ('device_type', ctypes.c_int),
                ('reserved2', ctypes.c_int),
                ('reserved3', ctypes.c_int),
                ('reserved4', ctypes.c_int),
                ('reserved5', ctypes.c_int),
                ('reserved6', ctypes.c_int),
                ('reserved7', ctypes.c_int),
                ('reserved8', ctypes.c_int)]
```

```
class PortInfo(ctypes.Structure):
    _fields_ = [
            ('type', ctypes.c_int), # enum is 32
               bits on 32 and 64 bit Linux
            ('name', (ctypes.c_char * 64)),
            ('path', (ctypes.c_char * 64)),
            ('library_filename', (ctypes.c_char *
               1024))
            ]

# gphoto constants
# Defined in 'gphoto2-port-result.h'
GP_OK = 0
# CameraCaptureType enum in 'gphoto2-camera.h'
GP_CAPTURE_IMAGE = 0
# CameraFileType enum in 'gphoto2-file.h'
GP_FILE_TYPE_NORMAL = 1



GP_WIDGET_WINDOW = 0  # Window widget This is the
   toplevel configuration widget. It should likely
   contain multiple GP_WIDGET_SECTION entries.
GP_WIDGET_SECTION = 1 # Section widget (think Tab).
GP_WIDGET_TEXT = 2    # Text widget.
GP_WIDGET_RANGE = 3   # Slider widget.
GP_WIDGET_TOGGLE = 4  # Toggle widget (think check
   box).
GP_WIDGET_RADIO = 5   # Radio button widget.
GP_WIDGET_MENU = 6    # Menu widget (same as RADIO).
GP_WIDGET_BUTTON = 7  # Button press widget.
GP_WIDGET_DATE = 8    # Date entering widget.
widget_types = ['Window', 'Section', 'Text',
   'Range', 'Toggle', 'Radio', 'Menu', 'Button',
   'Date']

class CameraWidget(ctypes.Structure):
    _fields_ = [('type', ctypes.c_int),
                ('label', (ctypes.c_char * 256)),
                ('info', (ctypes.c_char * 1024)),
                ('name', (ctypes.c_char * 256)),
                ('parent', (ctypes.c_void_p)),
                ('value_string', ctypes.c_char_p),
```

```
                ('value_int', ctypes.c_int),
                ('value_float', ctypes.c_float),
                ('choice', (ctypes.c_void_p)),
                ('choice_count', ctypes.c_int),
                ('min', ctypes.c_float),
                ('max', ctypes.c_float),
                ('increment', ctypes.c_float),
                ('children', (ctypes.c_void_p)),
                ('children_count', (ctypes.c_int)),
                ('changed', (ctypes.c_int)),
                ('readonly', (ctypes.c_int)),
                ('ref_count', (ctypes.c_int)),
                ('id', (ctypes.c_int)),
                ('callback', (ctypes.c_void_p))]


class libgphoto2error(Exception):
    def __init__(self, result, message):
        self.result = result
        self.message = message
    def __str__(self):
        return self.message + ' (' +
           str(self.result) + ')'

def check(result):
    if result < 0:
        gp.gp_result_as_string.restype =
           ctypes.c_char_p
        message = gp.gp_result_as_string(result)
        raise libgphoto2error(result, message)
    return result

def check_unref(result, camfile):
    if result!=0:
        gp.gp_file_unref(camfile._cf)
        gp.gp_result_as_string.restype =
           ctypes.c_char_p
        message = gp.gp_result_as_string(result)
        raise libgphoto2error(result, message)

class camera(object):
    def __init__(self, autoInit = True):
        self._cam = ctypes.c_void_p()
        # var created to use _get_port_info:
```

```
        self._info = PortInfo()
        self._leave_locked = False
        check(gp.gp_camera_new(PTR(self._cam)))
        self.initialized = False
        if autoInit:
            self.init()

    def init(self):
        if self.initialized:
            print "Camera is already initialized."
            return
        ans = 0
        for i in range(1 + retries):
            ans = gp.gp_camera_init(self._cam,
               context)
            if ans == 0:
                break
            elif ans == -60:
                ab = self._get_abilities()
                id_vend = str(ab.usb_vendor)
                id_prod = str(ab.usb_product)
                print 'Exec '+unmount_cmd+'
                   '+id_vend+' '+id_prod
                os.system(unmount_cmd+' '+id_vend+'
                   '+id_prod)
                time.sleep(2)
                print "camera.init() retry #%d..."
                   % (i)
        check(ans)
        if ans == 0:
            self.initialized = True
        else:
            self.reinit()

    def reinit(self):
        gp.gp_camera_free(self._cam)
        self.__new__()
        self.init()

    def __del__(self):
        if not self._leave_locked:
            check(gp.gp_camera_exit(self._cam))
            check(gp.gp_camera_free(self._cam))
```

```
def leave_locked(self):
    self._leave_locked = True

def ref(self):
    check(gp.gp_camera_ref(self._cam))

def unref(self):
    check(gp.gp_camera_unref(self._cam))

def exit(self):
    check(gp.gp_camera_exit(self._cam, context))

def _get_summary(self):
    txt = CameraText()
    check(gp.gp_camera_get_summary(self._cam,
       PTR(txt), context))
    return txt.text
summary = property(_get_summary, None)

def _get_manual(self):
    txt = CameraText()
    check(gp.gp_camera_get_manual(self._cam,
       PTR(txt), context))
    return txt.text
manual = property(_get_manual, None)

def _get_about(self):
    txt = CameraText()
    check(gp.gp_camera_get_about(self._cam,
       PTR(txt), context))
    return txt.text
about = property(_get_about, None)

def _get_abilities(self):
    ab = cameraAbilities()
    check(gp.gp_camera_get_abilities(self._cam,
       PTR(ab._ab)))
    return ab
def _set_abilities(self, ab):
    check(gp.gp_camera_set_abilities(self._cam,
       ab._ab))
abilities = property(_get_abilities,
   _set_abilities)
```

```
def _get_config(self):
    window = cameraWidget(GP_WIDGET_WINDOW)
    check(gp.gp_camera_get_config(self._cam,
       PTR(window._w), context))
    window.populate_children()
    return window

def _set_config(self, window):
    check(gp.gp_camera_set_config(self._cam,
       window._w, context))
config = property(_get_config, _set_config)

def _get_port_info(self):
    #raise NotImplementedError
    check(gp.gp_camera_get_port_info(self._cam,
       PTR(self._info)))
    return self._info

def _set_port_info(self, info):
    check(gp.gp_camera_set_port_info(self._cam,
       info))
    port_info = property(_get_port_info,
       _set_port_info)

def capture_image(self, destpath = None,
   delete=False):
    path = CameraFilePath()

    ans = 0
    for i in range(1 + retries):
        ans = gp.gp_camera_capture(self._cam,
           GP_CAPTURE_IMAGE, PTR(path), context)
        if ans == 0: break
        else:
            ab = self._get_abilities()
            id_vend = str(ab.usb_vendor)
            id_prod = str(ab.usb_product)
            print 'Exec '+unmount_cmd+'
               '+id_vend+' '+id_prod
            os.system(unmount_cmd+' '+id_vend+'
               '+id_prod)
            time.sleep(2)
            print "capture_image(%s) retry
               #%d..." % (destpath, i)
```

```
    check(ans)

    if destpath:
        self.download_file(path.folder,
           path.name, destpath)
        if delete:
            check(
               gp.gp_camera_file_delete(self._cam,
               path.folder, path.name, context)
               )
            return destpath

    if not delete:
        return (path.folder, path.name)

def capture_preview(self, destpath = None):
    path = CameraFilePath()
    cfile = cameraFile()

    ans = 0
    for i in range(1 + retries):
        ans =
           gp.gp_camera_capture_preview(self._cam,
           cfile._cf, context)
        if ans == 0: break
        else: print "capture_preview(%s) retry
           #%d..." % (destpath, i)
    check(ans)

    if destpath:
        cfile.save(destpath)
        return destpath
    else:
        return cfile


def download_file(self, srcfolder, srcfilename,
   destpath):
    cfile = cameraFile(self._cam, srcfolder,
       srcfilename)
    cfile.save(destpath)
    gp.gp_file_unref(cfile._cf)

def trigger_capture(self):
```

```
        check(gp.gp_camera_trigger_capture(self._cam,
            context))

    def wait_for_event(self, timeout):
        raise NotImplementedError

    def list_folders(self, path = "/"):
        l = cameraList()
        check(gp.gp_camera_folder_list_folders(self._cam,
            str(path), l._l, context));
        return l.toList()

    def list_files(self, path = "/"):
        l = cameraList()
        check(gp.gp_camera_folder_list_files(self._cam,
            str(path), l._l, context));
        return l.toList()

    def _list_config(self, widget, cfglist, path):
        children = widget.children
        if children:
            for c in children:
                self._list_config(c, cfglist, path
                    + "." + c.name)
        else:
            print path, "=", widget.value
            cfglist.append(path)

    def list_config(self):
        cfglist = []
        cfg = self.config
        self._list_config(cfg, cfglist, cfg.name)
        return cfglist

    def ptp_canon_eos_requestdevicepropvalue(self,
        prop):
        params = ctypes.c_void_p(self._cam.value +
            12)
        gp.ptp_generic_no_data(params,
            PTP_OC_CANON_EOS_RequestDevicePropValue,
            1, prop)

    # TODO: port_speed, init, config
    # Pending review
```

```
    def del_file( self, srcfolder, srcfile ):
        check( gp.gp_camera_file_delete( self.cam,
           srcfolder, srcfile, context ) )

    # def read_file( self): Maybe it is not
       necessary

class cameraFile(object):
    def __init__(self, cam = None, srcfolder =
       None, srcfilename = None):
        self._cf = ctypes.c_void_p()
        self._data = PTR(ctypes.c_char_p())
        self._size = PTR(ctypes.c_void_p())
        check(gp.gp_file_new(PTR(self._cf)))
        if cam:
            check_unref(gp.gp_camera_file_get(cam,
               srcfolder, srcfilename,
               GP_FILE_TYPE_NORMAL, self._cf,
               context), self)


    def open(self, filename):
        check(gp.gp_file_open(PTR(self._cf),
           filename))

    def save(self, filename = None):
        if filename is None: filename = self.name
        check(gp.gp_file_save(self._cf, filename))

    def ref(self):
        check(gp.gp_file_ref(self._cf))

    def unref(self):
        check(gp.gp_file_unref(self._cf))

    def clean(self):
        check(gp.gp_file_clean(self._cf))

    def copy(self, source):
        check(gp.gp_file_copy(self._cf, source._cf))

    def __dealoc__(self, filename):
        check(gp.gp_file_free(self._cf))
```

```
    def _get_name(self):
        name = ctypes.c_char_p()
        check(gp.gp_file_get_name(self._cf,
           PTR(name)))
        return name.value
    def _set_name(self, name):
        check(gp.gp_file_set_name(self._cf,
           str(name)))
    name = property(_get_name, _set_name)

    # TODO: new_from_fd (?), new_from_handler (?),
       mime_tipe, mtime, detect_mime_type,
       adjust_name_for_mime_type, data_and_size,
       append, slurp, python file object?
    #Added by me
    def _get_data_and_size(self):
        check(gp.gp_file_get_data_and_size(self._cf,
           self._data, self._size))
        return self._data,self._size
class cameraAbilitiesList(object):
    _static_l = None
    def __init__(self):
        if cameraAbilitiesList._static_l is None:
            cameraAbilitiesList._static_l =
               ctypes.c_void_p()
            check(gp.gp_abilities_list_new(PTR(cameraAbilitiesList._st
            check(gp.gp_abilities_list_load(cameraAbilitiesList._stati
               context))
        self._l = cameraAbilitiesList._static_l

    def __del__(self):
        # don't free, since it is only created once
        #check(gp.gp_abilities_list_free(self._l))
        pass

    def detect(self, il, l):
        check(gp.gp_abilities_list_detect(self._l,
           il._l, l._l, context))

    def lookup_model(self, model):
        return
           check(gp.gp_abilities_list_lookup_model(self._l,
           model))
```

```
    def get_abilities(self, model_index, ab):
        check(gp.gp_abilities_list_get_abilities(self._l,
           model_index, PTR(ab._ab)))

class cameraAbilities(object):
    def __init__(self):
        self._ab = CameraAbilities()

    def __repr__(self):
        return "Model : %s\nStatus : %d\nPort :
           %d\nOperations : %d\nFile Operations :
           %d\nFolder Operations : %d\nUSB
           (vendor/product) : 0x%x/0x%x\nUSB class
           : 0x%x/0x%x/0x%x\nLibrary : %s\nId :
           %s\n" % (self._ab.model,
           self._ab.status, self._ab.port,
           self._ab.operations,
           self._ab.file_operations,
           self._ab.folder_operations,
           self._ab.usb_vendor,
           self._ab.usb_product,
           self._ab.usb_class,
           self._ab.usb_subclass,
           self._ab.usb_protocol, self._ab.library,
           self._ab.id)

    model = property(lambda self: self._ab.model,
       None)
    status = property(lambda self: self._ab.status,
       None)
    port = property(lambda self: self._ab.port,
       None)
    operations = property(lambda self:
       self._ab.operations, None)
    file_operations = property(lambda self:
       self._ab.file_operations, None)
    folder_operations = property(lambda self:
       self._ab.folder_operations, None)
    usb_vendor = property(lambda self:
       self._ab.usb_vendor, None)
    usb_product = property(lambda self:
       self._ab.usb_product, None)
    usb_class = property(lambda self:
       self._ab.usb_class, None)
```

```
    usb_subclass = property(lambda self:
       self._ab.usb_subclass, None)
    usb_protocol = property(lambda self:
       self._ab.usb_protocol, None)
    library = property(lambda self:
       self._ab.library, None)
    id = property(lambda self: self._ab.id, None)

class portInfoList(object):
    _static_l = None
    def __init__(self):
        if portInfoList._static_l is None:
            portInfoList._static_l =
               ctypes.c_void_p()
            check(gp.gp_port_info_list_new(PTR(portInfoList._static_l)
            check(gp.gp_port_info_list_load(portInfoList._static_l))
        self._l = portInfoList._static_l

    def __del__(self):
        # don't free, since it is only created once
        #check(gp.gp_port_info_list_free(self._l))
        pass

    def count(self):
        c = gp.gp_port_info_list_count(self._l)
        check(c)
        return c

    def lookup_path(self, path):
        index =
           gp.gp_port_info_list_lookup_path(self._l,
           path)
        check(index)
        return index

    def get_info(self, path_index):
        info = PortInfo()
        check(gp.gp_port_info_list_get_info(self._l,
           path_index, PTR(info)))
        return info

class cameraList(object):
    def __init__(self, autodetect=False):
        self._l = ctypes.c_void_p()
```

```
        check(gp.gp_list_new(PTR(self._l)))

        if autodetect == True:
            if hasattr(gp, 'gp_camera_autodetect'):
                gp.gp_camera_autodetect(self._l,
                   context)
            else:
                # this is for stable versions of
                   gphoto <= 2.4.10.1
                xlist = cameraList()
                il = portInfoList()
                il.count()
                al = cameraAbilitiesList()
                al.detect(il, xlist)
                for i in xrange(xlist.count()):
                    model = xlist.get_name(i)
                    path = xlist.get_value(i)
                    if re.match(r'usb:\d{3},\d{3}',
                       path):
                        self.append(model, path)
                del al
                del il
                del xlist

    def ref(self):
        check(gp.gp_list_ref(self._l))

    def unref(self):
        check(gp.gp_list_ref(self._l))

    def __del__(self):
        # this failed once in gphoto 2.4.6
        check(gp.gp_list_free(self._l))
        pass

    def reset(self):
        #check(gp.gp_list_free(self._l))
        #check(gp.gp_list_new(PTR(self._l)))
        check(gp.gp_list_reset(self._l))

    def append(self, name, value):
        check(gp.gp_list_append(self._l, str(name),
           str(value)))
```

```
def sort(self):
    check(gp.gp_list_sort(self._l))

def count(self):
    return check(gp.gp_list_count(self._l))

def find_by_name(self, name):
    index = ctypes.c_int()
    check(gp.gp_list_find_by_name(self._l,
       PTR(index), str(name)))
    return index.value

def get_name(self, index):
    name = ctypes.c_char_p()
    check(gp.gp_list_get_name(self._l,
       int(index), PTR(name)))
    return name.value

def get_value(self, index):
    value = ctypes.c_char_p()
    check(gp.gp_list_get_value(self._l,
       int(index), PTR(value)))
    return value.value

def set_name(self, index, name):
    check(gp.gp_list_set_name(self._l,
       int(index), str(name)))

def set_value(self, index, value):
    check(gp.gp_list_set_value(self._l,
       int(index), str(value)))

def __str__(self):
    header = "cameraList object with %d
       elements:\n" % self.count()
    contents = ["%d: (%s, %s)" % (i,
       self.get_name(i), self.get_value(i))
        for i in range(self.count())]

    return header + string.join(contents, "\n")

def toList(self):
    return [(self.get_name(i),
       self.get_value(i)) for i in
```

```
            xrange(self.count())]
        xlist = []
        for i in range(self.count()):
            n, v = self.get_name(i),
               self.get_value(i)
            if v is None:
                xlist.append(n)
            else:
                xlist.append((n, v))
        return xlist

    def toDict(self):
        return dict(self.toList())

class cameraWidget(object):

    def __init__(self, type = None, label = ""):
        self._w = ctypes.c_void_p()
        if type is not None:
            check(gp.gp_widget_new(int(type),
               str(label), PTR(self._w)))
            check(gp.gp_widget_ref(self._w))
        else:
            self._w = ctypes.c_void_p()

    def ref(self):
        check(gp.gp_widget_ref(self._w))

    def unref(self):
        check(gp.gp_widget_unref(self._w))

    def __del__(self):
        # TODO fix this or find a good reason not to
        #print "widget(%s) __del__" % self.name
        #check(gp.gp_widget_unref(self._w))
        pass

    def _get_info(self):
        info = ctypes.c_char_p()
        check(gp.gp_widget_get_info(self._w,
           PTR(info)))
        return info.value
    def _set_info(self, info):
```

```
    check(gp.gp_widget_set_info(self._w,
       str(info)))
info = property(_get_info, _set_info)

def _get_name(self):
    name = ctypes.c_char_p()
    check(gp.gp_widget_get_name(self._w,
       PTR(name)))
    return name.value
def _set_name(self, name):
    check(gp.gp_widget_set_name(self._w,
       str(name)))
name = property(_get_name, _set_name)

def _get_id(self):
    id = ctypes.c_int()
    check(gp.gp_widget_get_id(self._w, PTR(id)))
    return id.value
id = property(_get_id, None)

def _set_changed(self, changed):
    check(gp.gp_widget_set_changed(self._w,
       str(changed)))
def _get_changed(self):
    return gp.gp_widget_changed(self._w)
changed = property(_get_changed, _set_changed)

def _get_readonly(self):
    readonly = ctypes.c_int()
    check(gp.gp_widget_get_readonly(self._w,
       PTR(readonly)))
    return readonly.value
def _set_readonly(self, readonly):
    check(gp.gp_widget_set_readonly(self._w,
       int(readonly)))
readonly = property(_get_readonly,
   _set_readonly)

def _get_type(self):
    type = ctypes.c_int()
    check(gp.gp_widget_get_type(self._w,
       PTR(type)))
    return type.value
type = property(_get_type, None)
```

```
    def _get_typestr(self):
        return widget_types[self.type]
    typestr = property(_get_typestr, None)

    def _get_label(self):
        label = ctypes.c_char_p()
        check(gp.gp_widget_get_label(self._w,
           PTR(label)))
        return label.value
    def _set_label(self, label):
        check(gp.gp_widget_set_label(self._w,
           str(label)))
    label = property(_get_label, _set_label)

    def _get_value(self):
        value = ctypes.c_void_p()
        ans = gp.gp_widget_get_value(self._w,
           PTR(value))
        if self.type in [GP_WIDGET_MENU,
           GP_WIDGET_RADIO, GP_WIDGET_TEXT]:
            value = ctypes.cast(value.value,
               ctypes.c_char_p)
        elif self.type == GP_WIDGET_RANGE:
#            value = ctypes.cast(value.value,
   ctypes.c_float_p)
            value = ctypes.cast(value.value,
               ctypes.c_void_p)
        elif self.type in [GP_WIDGET_TOGGLE,
           GP_WIDGET_DATE]:
            #value = ctypes.cast(value.value,
               ctypes.c_int_p)
            pass
        else:
            return None
        check(ans)
        return value.value

    def _set_value(self, value):
        if self.type in (GP_WIDGET_MENU,
           GP_WIDGET_RADIO, GP_WIDGET_TEXT):
            value = ctypes.c_char_p(value)
        elif self.type == GP_WIDGET_RANGE:
```

```
            value = ctypes.c_float_p(value) # this
               line not tested
        elif self.type in (GP_WIDGET_TOGGLE,
           GP_WIDGET_DATE):
            value = PTR(ctypes.c_int(value))
        else:
            return None # this line not tested
        check(gp.gp_widget_set_value(self._w,
           value))
    value = property(_get_value, _set_value)

    def append(self, child):
        check(gp.gp_widget_append(self._w,
           child._w))

    def prepend(self, child):
        check(gp.gp_widget_prepend(self._w,
           child._w))

    def count_children(self):
        return gp.gp_widget_count_children(self._w)

    def get_child(self, child_number):
        w = cameraWidget()
        check(gp.gp_widget_get_child(self._w,
           int(child_number), PTR(w._w)))
        check(gp.gp_widget_ref(w._w))
        return w

    def get_child_by_label(self, label):
        w = cameraWidget()
        check(gp.gp_widget_get_child_by_label(self._w,
           str(label), PTR(w._w)))
        return w

    def get_child_by_id(self, id):
        w = cameraWidget()
        check(gp.gp_widget_get_child_by_id(self._w,
           int(id), PTR(w._w)))
        return w

    def get_child_by_name(self, name):
        w = cameraWidget()
        # this fails in 2.4.6 (Ubuntu 9.10)
```

```
    check(gp.gp_widget_get_child_by_name(self._w,
       str(name), PTR(w._w)))
    return w

def _get_children(self):
    children = []
    for i in range(self.count_children()):
        children.append(self.get_child(i))
    return children
children = property(_get_children, None)

def _get_parent(self):
    w = cameraWidget()
    check(gp.gp_widget_get_parent(self._w,
       PTR(w._w)))
    return w
parent = property(_get_parent, None)

def _get_root(self):
    w = cameraWidget()
    check(gp.gp_widget_get_root(self._w,
       PTR(w._w)))
    return w
root = property(_get_root, None)

def _set_range(self, range):
    """cameraWidget.range = (min, max,
       increment)"""
    float = ctypes.c_float
    min, max, increment = range
    check(gp.gp_widget_set_range(self._w,
       float(min), float(max),
       float(increment)))
    return w
def _get_range(self, range):
    """cameraWidget.range => (min, max,
       increment)"""
    min, max, increment = ctypes.c_float(),
       ctypes.c_float(), ctypes.c_float()
    check(gp.gp_widget_set_range(self._w,
       PTR(min), PTR(max), PTR(increment)))
    return (min.value, max.value,
       increment.value)
range = property(_get_range, _set_range)
```

```
    def add_choice(self, choice):
        check(gp.gp_widget_add_choice(self._w,
           str(choice)))

    def count_choices(self):
        return gp.gp_widget_count_choices(self._w)

    def get_choice(self, choice_number):
        choice = ctypes.c_char_p()
#         check(gp.gp_widget_add_choice(self._w,
   int(choice_number), PTR(choice)))
        check(gp.gp_widget_get_choice(self._w,
           int(choice_number), PTR(choice)))
        return choice.value

    def createdoc(self):
        label = "Label: " + self.label
        info = "Info: " + (self.info if self.info
           != "" else "n/a")
        type = "Type: " + self.typestr
        #value = "Current value: " + str(self.value)
        childs = []
        for c in self.children:
            childs.append("  - " + c.name + ": " +
               c.label)
        if len(childs):
            childstr = "Children:\n" +
               string.join(childs, "\n")
            return label + "\n" + info + "\n" +
               type + "\n" + childstr
        else:
            return label + "\n" + info + "\n" + type

    def _pop(widget, simplewidget):
        #print widget
        for c in widget.children:
            simplechild = cameraWidgetSimple()
            if c.count_children():
                setattr(simplewidget, c.name,
                   simplechild)
                simplechild.__doc__ = c.createdoc()
                c._pop(simplechild)
            else:
```

```
                setattr(simplewidget, c.name, c)

            #print c.name, simplewidget.__doc__
        #print dir(simplewidget)

    def populate_children(self):
        simplewidget = cameraWidgetSimple()
        setattr(self, self.name, simplewidget)
        simplewidget.__doc__ = self.createdoc()
        self._pop(simplewidget)

    def __repr__(self):
        return "%s:%s:%s:%s:%s" % (self.label,
           self.name, self.info, self.typestr,
           self.value)

class cameraWidgetSimple(object):
    pass
```

10.1.13 Controlador de la llibreria gphoto2

```
import camlib as pd
import time
import threading
from PIL import Image


def capture_preview( destpath=None ):
    if destpath is None:
        destpath =
           '/var/www/dslrproject/static/dslrapp/images/preview.jpg'
    try:
        path = pd.camera().capture_preview(destpath)
        return path
    except:
        return None

#Capture a view from camera without saving into
   camera sdcard
#Function returns filepath where image has been
   saved on raps flash
```

```
def capture_view(
   capture_view_on=False,destpath=None):
    pass
    #global thread_capture_view

    #try:
    #if destpath is None:
    #destpath =
       '/var/www/dslrproject/static/dslrapp/images/liveview.jpg'

    #if not capture_view_on:
    #thread_capture_view = LiveViewThread(
       self.camera,destpath )
    #thread_capture_view.start()

    #if 'dslrproject' in destpath and 'liveview' in
       destpath:
    #destpath =
       '/static/dslrapp/images/liveview.jpg'
    #if not thread_capture_view.isstopped():

    #return destpath
    #except:
    #if thread_capture_view is not None:
    #thread_capture_view.stop()
    #return None

#Capture view from camera saving the image into
   camera sdcard and raps flash
#Function returns filepath where image has been
   saved on camera sdcard
def capture_image_on_sd_hd(
   destpath=None,delete=False):
    try:
        if destpath:
            return pd.camera().capture_image(
               destpath, delete )
        else:
            return
               pd.camera().capture_image('/var/www/dslrproject/static/
               delete)
    except:
        return None
```

```
#Capture view from camera saving the image into
   camera sdcard
#Function returns filepath
def capture_image_on_sd( ):
    try:
        return pd.camera().capture_image()
    except:
        return None

#Returns True if a camera is connected
def autodetect():
    try:
        if pd.cameraList(True).count() == 1:
            return True
    except:
        return False

#Function returna a dict with Camera Abilities
def get_abilities( ):
    try:
        ab = pd.camera().abilities
        dicta={}
        for i in dir(ab._ab):
            if not '_' in i and hasattr(ab._ab,i):
                dicta[i]= getattr(ab._ab,i)
        return dicta
    except:
        return None

#Function returns CameraWidget searched by name
def get_config_by_name( name ):
    try:
        child =
           pd.camera().config.get_child_by_name(
           name )
        return child
    except:
        return None

#Function returns value of any CameraWidget by name
def get_config_value_by_name( name ):
    try:
        value =
           pd.camera().config.get_child_by_name(
```

```
            name ).value
        return value
    except:
        return None

#Functions returns a dictionary with all names and
   choices of CameraWidgets of camera
def get_all_config( ):
    try:
        dictw ={}
        list_config( pd.camera().config, dictw, cam
           )
        return dictw
    except:
        return None

#Functions inflate dictionary with names and
   choices of a CameraWidget and his children (
   recursive method )
def list_config( widget, dictw):
    try:
        if widget.children:
            for c in widget.children:
                list_config( c, dictw)
        else:
            dictw[ widget.name ] =
               get_all_choices_from_widget( widget)
    except:
        return None

#Function returns a list with all choices from
   CameraWidget
def get_all_choices_from_widget( widget):
    try:
        listw = []
        listw.append ( widget.type )
        if widget.type == pd.GP_WIDGET_RADIO or
           widget.type == pd.GP_WIDGET_MENU:
            for i in range( widget.count_choices()
               ):
                listw.append( widget.get_choice(i) )
        else:
            listw.append( widget.value )
        return listw
```

```
    except:
        return None

#Function returns True if a new config is set on
   camera
def set_config( window=None):
    if window is None:
        window = get_config()
    try:
        pd.camera()._set_config(window)
        return True
    except:
        return False

#Function returns True if value of widget has been
   changed and the new config is set on camera
def set_value_on_config_by_name( value, namewidget):
    try:
        config = pd.camera().config
        widget = config.get_child_by_name(
           namewidget )
        listw = get_all_choices_from_widget(widget)

        if any(str(value).lower() ==
           str(val).lower() for val in listw):
            widget.value = value
            return set_config( cam, config )
        return False
    except:
        return False

#Function returns a list of configuration of camera
def show_config():
    try:
        return pd.camera().list_config()
    except:
        return None


#Function returns name of Camera model
def get_name_of_camera():
    try:
        return pd.camera().abilities.model
    except:
```

```
        return None
```

10.1.14 Controlador dels models de la base de dades

```
import controller
from dslrapp.models import *


def registercam():
        try:
                camname =
                   controller.get_name_of_camera()

                if camname is None:
                        return False

                for camera in Camera.objects.all():
                        if camera.name == camname:
                                return True

                cam =
                   Camera.objects.create(name=camname)
                cam.save()
                registerconfigs(cam)
                registerabilities(cam)
                return True
        except:
                return False

def registerconfigs(camera):
        dictc = controller.get_all_config()
        for i,j in dictc.iteritems():
                config =
                   CameraConfig.objects.create(container=camera,key=i,
                config.save()

def registerabilities(camera):
        dicta = controller.get_abilities()
        for i,j in dicta.iteritems():
                ab =
                   CameraAbility.objects.create(container=camera,key=i
```

```
                ab.save()

def get_config_by_name(camname):

        listc = []
        for config in
           CameraConfig.objects.filter(container=Camera.objects.filter
                listc.append( [ config.key,
                   config.value[1:-1].split(',') ] )
        return listc

def get_values_by_widgetname( camname, widgetname):
        try:
            listunicode =
               list(CameraConfig.objects.filter(container=Camera.objec
            return [ str(x.split("'")[1]) for x in
               listunicode]
        except:
            return None
```

10.1.15 Controlador del HDR

```
from os.path import basename
import controller
import time
import dbcontroller
import math
from PIL import Image
import os, os.path
import sys, string
from copy import copy
import zipfile

# param: nframe number of frames
#        config list with [ap,ex,iso,ss]
def getHDR(nframes, config):

    try:
        listtime = []
        lastap = None
        lastex = None
```

```
lastiso = None
lastss = None
exindex = 0
model = controller.get_name_of_camera()
if not model: return "Connection failed"

print "Model: " + str(model)
for i in range( int(nframes) ):
    ap = config.pop(0)
    ex = config.pop(0)
    iso = config.pop(0)
    ss = config.pop(0)

    if lastap != ap:
        res = None
        if 'Canon' in model:
            res =
               controller.set_value_on_config_by_name(ap,
               'aperture')
            print "Aperture: "+str(ap)+"
               "+str(res)
        if 'Nikon' in model:
            res =
               controller.set_value_on_config_by_name(ap,
               'f-number')
            print "Aperture: "+str(ap)+"
               "+str(res)
        lastap = ap
        if res is None: return "Connection
           Failed"
        if res is False: return "Aperture
           could not be changed"
    if lastex != ex:
        res =
           controller.set_value_on_config_by_name(
           ex, 'exposurecompensation')
        print "Exposure: "+str(ex)+"
           "+str(res)
        lastex = ex
        if res is None: return "Connection
           Failed"
        if res is False: return "Exposure
           could not be changed"
```

```
            if lastiso != iso:
                res =
                   controller.set_value_on_config_by_name(iso,
                   'iso')
                print "ISO: "+str(iso)+" "+str(res)
                lastiso = iso
                if res is None: return "Connection
                   Failed"
                if res is False: return "ISO could
                   not be changed"
            if lastss != ss:
                res =
                   controller.set_value_on_config_by_name(ss,
                   'shutterspeed')
                print "Shutterspeed: "+str(ss)+"
                   "+str(res)
                lastss = ss
                if res is None: return "Connection
                   Failed"
                if res is False: return
                   "Shutterspeed could not be
                   changed"

            now = time.strftime("%d_%m_%y_%H_%M_%S")
            path =
               controller.capture_image_on_sd_hd(
               '/var/www/dslrproject/static/dslrapp/images/hdrmode/sna
               delete=True)
            print "Image "+str(path)+" created"
        project = hdr(case='hdrmode', resize=0.5,
           img_type='jpg',
           cur_dir='/var/www/dslrproject/static/dslrapp/images/')
        project.get_hdr()
        os.system('rm
           /var/www/dslrproject/static/dslrapp/images/hdrmode/snap*')
        filez =
           zipdir('/var/www/dslrproject/static/dslrapp/images/hdrmode/
        os.system('rm -r
           /var/www/dslrproject/static/dslrapp/images/hdrmode/out')
        return filez
    except:
        return None

def zipdir(path):
```

```
    now = time.strftime("%d_%m_%y_%H_%M_%S")
    zipf =
       zipfile.ZipFile('/var/www/dslrproject/static/dslrapp/images/hdr
       'w')
    for root, dirs, files in os.walk(path):
        for filex in files:
            zipf.write(os.path.join(root,
               filex),basename(os.path.join(root,
               filex)))
    zipf.close()
    return 'hdr'+now+'.zip'

class hdr:
    """
    a collection of images to merege into HDR

    blend an arbitrary number of photos into a
       single HDR image
    or several images with various combinations of
       HDR parameters

    it is assumed that project folders contain all
       the images you want to merge
    case = folder name
    cur_dir = folder where the project folders are
       located
    images are auto-sorted by degree of exposure
       (image brightness)
    """

    def get_imgs(self):
        """
        load a list of images from folder
        sort them from lightest to darkest
        """
        drks = []
        print [ os.path.join(self.indir,fn) for fn
           in self.fns]
        imgs =
           [Image.open(os.path.join(self.indir,fn))
           for fn in self.fns]
        print imgs
        for img in imgs:
```

```
            samp = img.resize((50,50))
               #crop(box=((10,10,20,20)))
            drks.append(sum(samp.getdata(band=0)))
        if self.resize:
            newsize = tuple([int(x*self.resize) for
               x in img.size])
            imgs = [img.resize(newsize) for img in
               imgs]

        newdrks = sorted(drks)
        idxs = [drks.index(x) for x in newdrks]
        imgs = [imgs[x] for x in idxs]
        self.fns = [self.fns[x] for x in idxs]

        self.drks = drks
        self.idxs = idxs

        print 'got',len(imgs),'images'
        return imgs


    def
       __init__(self,case='',resize=None,img_type='.jpg',cur_dir='/var
       ):

        """
        load a project
        all images of [img_type] are loaded from
           folder [case]
        and resized by a factor [resize]
        """
        if not case:
            print 'case is required'
            sys.exit(1)
        self.resize = resize
        self.ext = img_type.strip('.')
        self.case = case
        self.cur_dir = os.path.normpath(cur_dir)
        self.indir   = os.path.join(self.cur_dir,
           case)
        self.outdir  = os.path.join(self.indir ,
           'out')
        print self.indir
```

```
    if not os.path.isdir(self.outdir):
        os.makedirs(self.outdir)
    f = lambda x:
       x.upper().endswith(self.ext.upper())
    self.fns = filter(f,os.listdir(self.indir))
    self.imgs = self.get_imgs()
    self.fin_set = {}


def get_masks(self,imgs):
    """
    create a set of masks from a list of images
    (one mask for every adjacent pair of images
    """
    masks = []
    mask_ct = len(imgs)-1
    imgs = [self.bal(img.convert(mode='L')) for
       img in imgs]
    for i in range(mask_ct):
        blend_fraction = .5     #1. -
           (float(i)+.5)/float(mask_ct)
        m =
           Image.blend(imgs[i],imgs[i+1],blend_fraction)
        masks.append(m)
        #print blend_fraction
    print 'blending using',mask_ct,'masks'
    return masks


def lev(self,im):
    #im =
       ImageEnhance.Brightness(im).enhance(self.bri)
    #im =
       ImageEnhance.Contrast(im).enhance(self.con)
    return self.bal(im, self.str)


def bal(self,im):
    """
    adjust the balance of the mask
    (re-distribute the histogram so that there
       are more
    extreme blacks and whites)
```

```
        like increasing the contrast, but without
           clipping
        and maintains overall average image
           brightness
        """
        h = im.histogram()
        ln = range(len(h))
        up = [sum(h[0: i]) for i in ln]
        lo = [sum(h[i:-1]) for i in ln]
        ct = sum(h)
        st = int(self.cur_str*255.)

        lut = [i+st*up[i]*lo[i]*(up[i]-lo[i])/ct**3
           for i in ln]
        for i in ln:
            if lut[i]<  1:lut[i]=  1
            if lut[i]>255:lut[i]=255
        return im.point(lut)


    def save_im(self,im,name):
        """
        save an image
        """
        print 'saving',name
        im.save(os.path.join(self.outdir,
           name+'.jpg'), format='JPEG')

    def merge(self,imgs):
        """
        combine a set images into a smaller set by
           combinding all
        adjacent images
        """
        masks = self.get_masks(imgs)
        imx = lambda
           i:Image.composite(imgs[i],imgs[i+1],masks[i])
        return [imx(i) for i in range(len(masks))]


    def merge_all(self,imgs):
        """
        iteratively merge a set of images until
           only one remains
```

```
        """
        while len(imgs)>1:
            imgs = self.merge(imgs)
        return imgs[0]


    def get_hdr(self, strength=[0.0,1.0,2.0],
       naturalness=[0.8,0.9,1.0]):
        """
        process the hdr image(s)
        strength - a list or a float that defines
           how strong the hdr
        effect should be
        - a value of zero will combine images by
           using a
        greyscale image average
        - a value greater than zero will use higher
           contrast
        versions of those greyscale images
        naturalness- values between zero and one
        - zero will be a very high-contrast image
        - 1.0 will be a very flat image
        - 0.7 to 0.9 tend to give the best results
        """
        contrast=1.0
        brightness=1.0
        self.con = contrast # not used
        self.bri = brightness # not used
        self.nat = self.to_list(naturalness)
        self.str = self.to_list(strength)
        self.fin_set = {}

        for s in self.str:
            self.cur_str = s
            print 'getting saturation image,
               strength',str(s)
            imgs = copy(self.imgs)
            sat_img = self.merge_all(imgs)

            print 'getting contrast image'
            imgs.reverse()
            con_img = self.merge_all(imgs)

            self.final_blend(con_img,sat_img)
```

```
        self.print_set(self.fin_set)
        ori_set = dict(('orig_'+str(i),
           self.imgs[i]) for i in
           range(len(self.imgs)))
        self.print_set(ori_set)
        print self.indir


    def final_blend(self,im1,im2):
        """
        combines a saturated image with a contrast
           image
        and puts them in a dictionary of completed
           images
        """
        for nat in self.nat:
            n_s = '_nat' + str(round(nat,2))
            s_s = '_str' +
               str(round(self.cur_str,2))
            s = self.case + s_s + n_s
            self.fin_set[s] =
               Image.blend(im1,im2,nat)


    def print_set(self, im_dict):
        """
        print all rendered images
        """
        for k in im_dict.keys():
           self.save_im(im_dict[k],k)


    def to_list(self, val):
        if type(val)!=type(list()):
            val = [float(val)]
        else:
            val = [float(v) for v in val]
        return val
```

10.1.16 Controlador del Timelapse

```
from os.path import basename
import time
import controller
import os, os.path
import zipfile

def getTimelapse(nframes, intervals, fps):

#    try:
    os.system('rm -r
       /var/www/dslrproject/static/dslrapp/images/timelapse/out')
    os.makedirs('/var/www/dslrproject/static/dslrapp/images/timelapse/
    dateini = time.strftime("%d_%m_%y_%H_%M_%S")
    init = int(time.time())
    sizenframes = len(nframes)
    nframes = int(nframes)
    intervals = int(intervals)
    aux = 0
    count = 1
    while (nframes > 0):

        now = int(time.time())

        if ( (now - init) >= intervals):
            codi = str(count)
            lencount = len(codi)
            if lencount < sizenframes:
                dif = sizenframes - lencount
                for i in range(dif):
                    codi = '0'+codi
            time.sleep(1)
            res =
               controller.capture_image_on_sd_hd('/var/www/dslrproject
            if res is None: return None
            init = now
            count += 1
            nframes -= 1

    #get all images to movie
    #return movie
    datefi = time.strftime("%d_%m_%y_%H_%M_%S")
    namefile =
       "/var/www/dslrproject/static/dslrapp/images/timelapse/out/timel
```

```
    command = "ffmpeg -f image2 -i
       /var/www/dslrproject/static/dslrapp/images/timelapse/out/timela
       -r "+fps+" "+namefile

    os.system(command)
    filez =
       zipdir('/var/www/dslrproject/static/dslrapp/images/timelapse/ou
       dateini+"__"+datefi)
    os.system('rm -r
       /var/www/dslrproject/static/dslrapp/images/timelapse/out')
    return filez
 #   except:
  #      return None

def zipdir(path, now):
    zipf =
       zipfile.ZipFile('/var/www/dslrproject/static/dslrapp/images/tim
       'w')
    for root, dirs, files in os.walk(path):
        for filex in files:
            zipf.write(os.path.join(root,
               filex),basename(os.path.join(root,
               filex)))
    zipf.close()
    return 'timelapse'+now+'.zip'
```

10.1.17 Controlador del Focus stacking

```
from os.path import basename
import time
import camlib
import os, os.path
import zipfile

def getFocusstacking(framestep):

    #try:

    rootdir =
       '/var/www/dslrproject/static/dslrapp/images/focus/out'
    os.system('rm -r '+rootdir)
```

```
os.makedirs( rootdir )
rootpath = rootdir+'/';
cam = camlib.camera()
cam.leave_locked()
cam.capture_preview()

for i in range(5):
    set_value_on_config_by_name( cam,'Near
       3','manualfocusdrive')
    time.sleep(1)

if ( framestep == '1' ): #Distance mountain
   8frames

    for i in range(8+1):
        cam.capture_preview(rootpath+"focus_"+str(i)+".jpg")
        time.sleep(1)
        set_value_on_config_by_name( cam,'Far
           2','manualfocusdrive')
        time.sleep(1)
        set_value_on_config_by_name( cam,'Far
           2','manualfocusdrive')
        time.sleep(1)

elif ( framestep == '2'): #Distance home 32
    for i in range(32+1):
        codi = str(i)
        if i<10:
            codi = '0'+codi
        cam.capture_preview(rootpath+"focus_"+codi+".jpg")
        time.sleep(1)
        set_value_on_config_by_name( cam,'Far
           2','manualfocusdrive')
        time.sleep(1)

elif ( framestep == '3'): #Distance pp 64
    for i in range(64+1):
        codi = str(i)
        if i<10:
            codi = '0'+codi
        cam.capture_preview(rootpath+"focus_"+codi+".jpg")
        time.sleep(1)
        set_value_on_config_by_name( cam,'Far
           1','manualfocusdrive')
```

```
            time.sleep(1)


    cam.exit()
    os.system('align_image_stack -a
       '+rootpath+'aligned_ -v -m -d -i -g 10 -C
       '+rootpath+'focus*.jpg')
    os.system('enfuse -o '+rootpath+'result.tiff
       --exposure-weight=0 --saturation-weight=0
       --contrast-weight=1 --hard-mask
       '+rootpath+'aligned_*')
    filez = zipdir(rootdir)
    os.system('rm -r '+rootdir)
    return filez
    #except
        #return None

def zipdir(path):
    now = time.strftime("%d_%m_%y_%H_%M_%S")
    zipf =
       zipfile.ZipFile('/var/www/dslrproject/static/dslrapp/images/foc
       'w')
    for root, dirs, files in os.walk(path):
        for filex in files:
            zipf.write(os.path.join(root,
               filex),basename(os.path.join(root,
               filex)))
    zipf.close()
    return 'focus'+now+'.zip'

def set_config( cam, window ):
    cam._set_config(window)


def set_value_on_config_by_name( cam, value,
   namewidget ):
    config = cam.config
    widget = config.get_child_by_name( namewidget )
    widget.value = value
    return set_config( cam, config )
```

Instal·lació paquet per realitzar el focus stacking

```
apt-get install enfuse
```

10.2 Manual d'usuari

10.2.1 Connexió del dispositiu Raspberry

El dispositiu Raspberry Pi disposa de 2 connectors USB i un connector microUSB per l'alimentació. El primer pas és connectar el dispositiu wireless tenim com accessori a un dels USB. Després endollar amb alguna bateria o alimentació de corrent amb un cable amb sortida microUSB.

Un cop el sistema operatiu estigui engegati, observarem al nostre smartphone, PC, portatil una xarxa WIFI anomenada RaspAP. La contrasenya d'aquesta xarxa és 'Pii.2013' i accedirem a la xarxa de la Raspberry Pi.

Per poder accedir a l'aplicació escriurem a qualsevol navegador web l'adreça http://192.168.0.1, si es desitja en el fitxer /etc/hosts de qualsevol linux, android es pot descriure un hostname per no haver d'enrecordarse de la IP sempre. Si afegim al fitxer /etc/hosts la línia '192.168.0.1 www.dslrproject.com' amb el nom especificat accedirem automàticament a l'aplicació i es mostrarà:

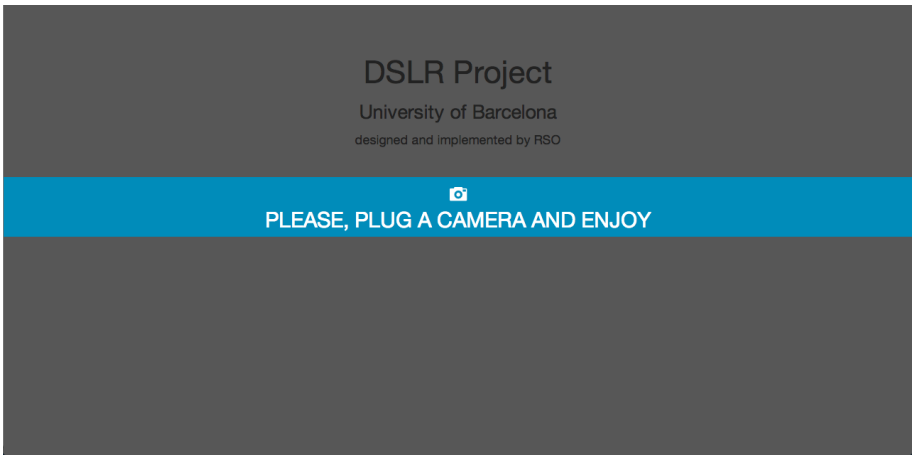

Figura 10.1: Pantalla inicial quan l'aplicació no detecta cap càmera connectada

Aquesta pantalla inicial s'actualitza en un interval d'uns segons i just quan es connecta la càmera automàticament detectarà la càmera. En cas que el model de la càmera mai hagi sigut registrat ho farà en aquest pas, abans d'ensenyar la interfície gràfica de l'aplicació.

10.2.2 Connexió de la càmera DSLR

Un cop tenim tots els dispositius connectats, endollem la càmera amb el seu cable USB a la segona ranura de la Raspberry Pi. L'aplicació detectarà automàticament la càmera i apareixerà la interfície gràfica per poder començar a utilitzar la càmera.

La interfície està estructurada de la següent manera: en primer lloc informació tècnica de la càmera a dalt a l'esquerre, seguit de les 4 opcions bàsiques LiveView, PreView, Zoom, Shot; en segon lloc veurem 4 modes possibles i sempre a l'iniciar l'aplicació es començara per el mode normal; en tercer lloc a sota de la informació tècnica i les funcions bàsiques veurem les fotografies realitzades; just a la dreta i sota del panell de modes les funcionalitats del mode en aquell cas seleccionat; per últim en la part inferior disposa d'un log on anem trobant feedback de l'aplicació i informació no tan essencial de la càmera:

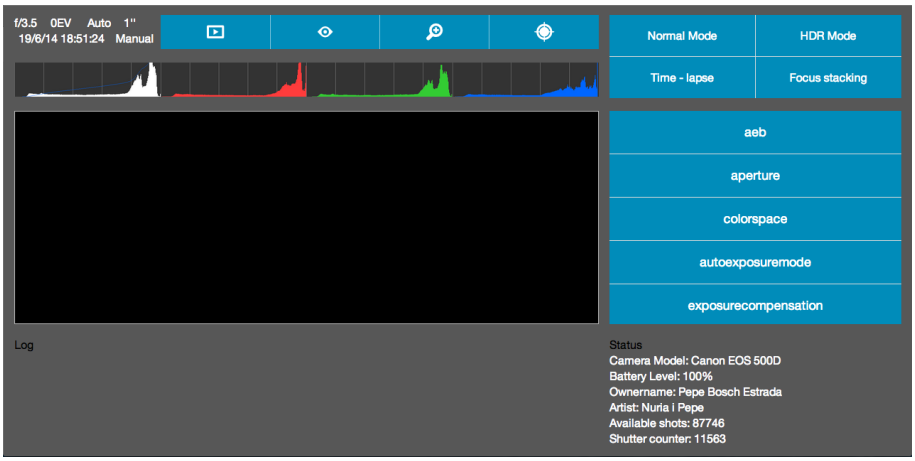

Figura 10.2: Primera vista de la interfície gràfica un cop que l'aplicació detecta una càmera

10.2.3 Funcionalitats bàsiques

Realitzar una fotografia

Com hem dit en el pas anterior existeixen quatre funcionalitats bàsiques: LiveView, et permet veure imatges en temps real realitzant previsualitzacions; PreView, realitzar una fotografia sense guardar a disc; Zoom, en cas d'haver captat una imatge i tenir la previsualització a la pantalla, mostrarà la imatge a pantalla completa; i per últim Shot, que serveix per disparar la càmera en qualsevol mode.

Una seqüència d'una captura normal es reproduiria de la següent manera: realitzem un clic al botó Shot! com la figura 10.3, sempre que s'utilitzi aquesta funció haurem d'esperar que l'aplicació faci el seu procès, figura 10.4, en aquest cas com només és realitzar una fotografia, el temps de computació és mínim; per finalitzar l'aplicació et dirà el nom de la imatge guardada i et mostrarà la imatge per pantalla, figura 10.5.

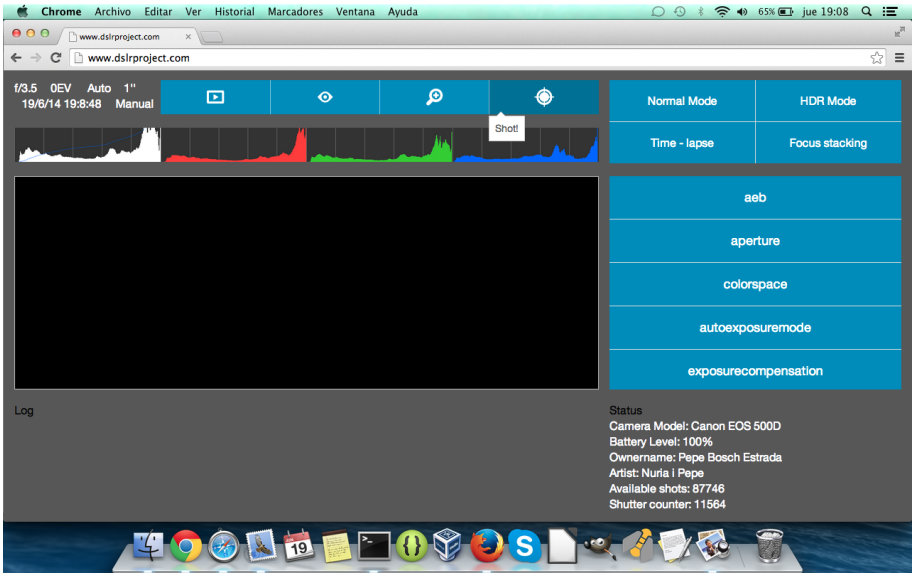

Figura 10.3: Exemple per clicar el botó amb la funció Shot!

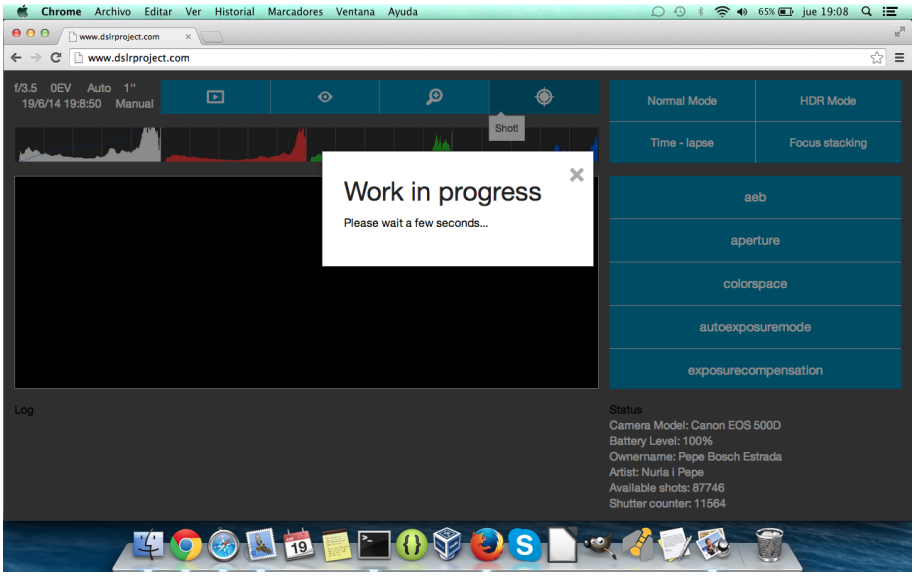

Figura 10.4: Pantalla on mostrà la finestra emergent mentre l'aplicació està treballant internament

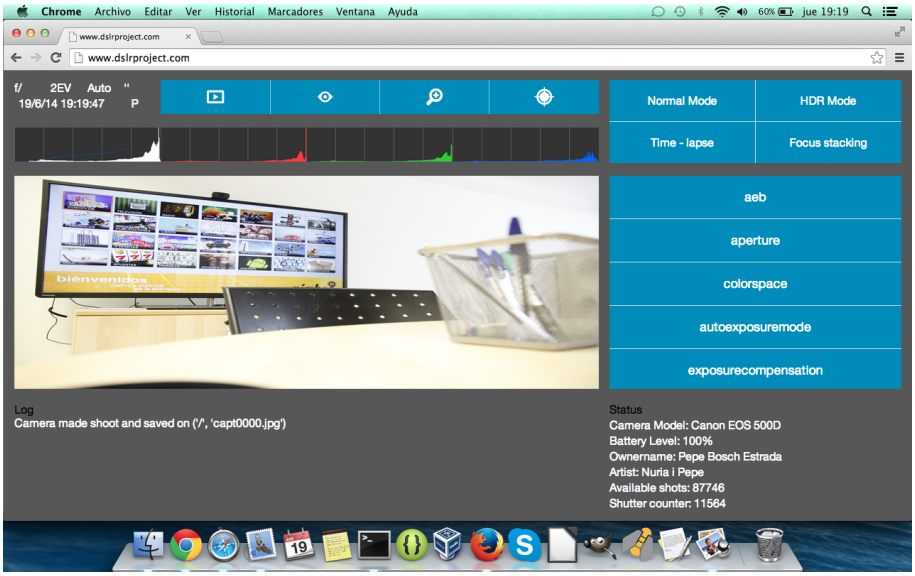

Figura 10.5: Finalització de la captura i previsualització de la imatge

Realitzar un Zoom

Quan tenim qualsevol previsualització a la interfície gràfica es pot utilitza la funció de Zoom per veure la fotografia a pantalla completa:

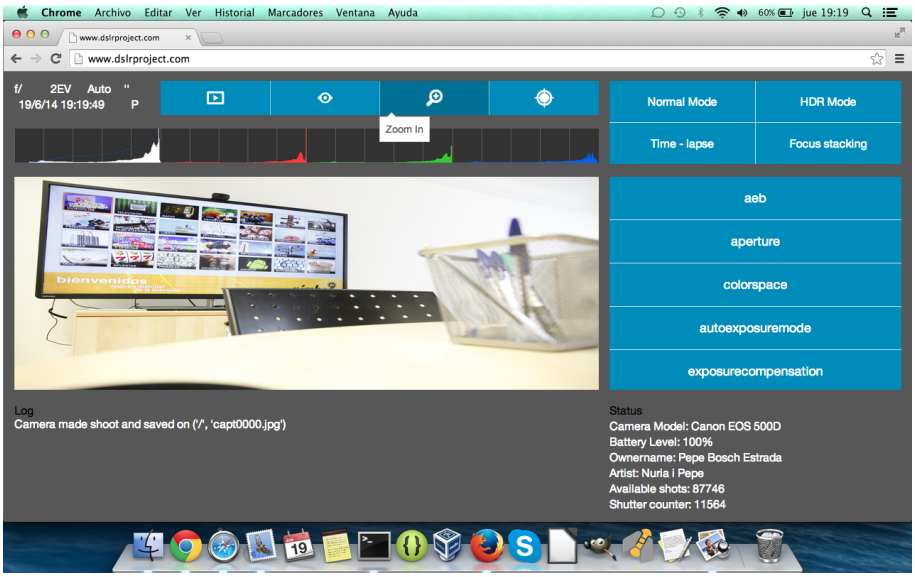

Figura 10.6: Realització del clic al botó Zoom per maximitzar la imatge previsualitzada

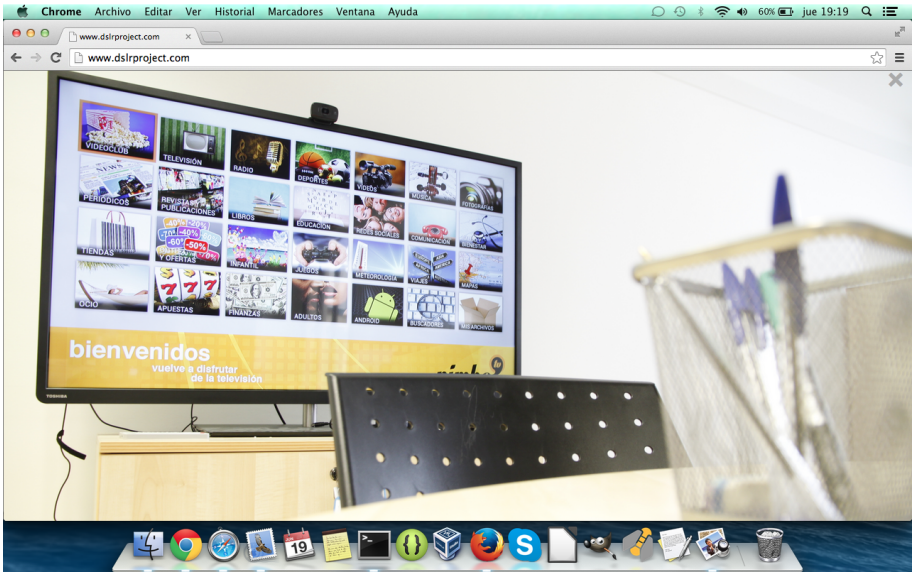

Figura 10.7: Imatge previsualitzada a pantalla completa utilitzant el Zoom

Realitza un canvi de paràmetre en el mode Normal

Per realitzar qualsevol canvi en un dels paràmetres que disposem en el mode Normal únicament s'ha de pitjar al paràmetre que vulguem, figura 10.8; un cop seleccionat apareixerà totes les possibles opcions que dona la càmera endollada, figura 10.9; i en cas de voler realitzar un canvi pitjarem al valor escollit; per últim l'aplicació informarà en cas afirmatiu o erroni si ha realitzat el procès. Per exemple hem canviat el paràmetre autoexposuremode a Manual, figura 10.10.

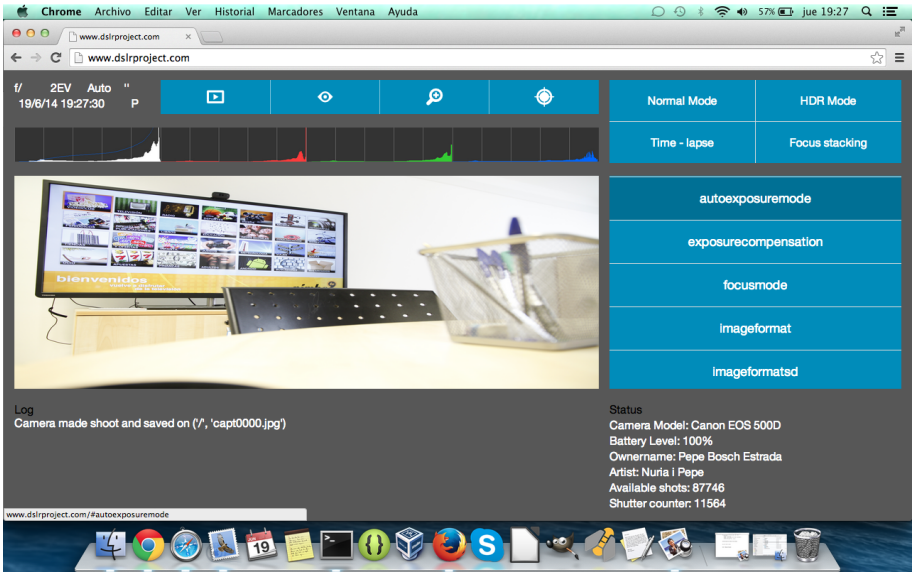

Figura 10.8: Selecció del paràmetre autoexposure, s'observa com el color canvia un cop estem per sobre

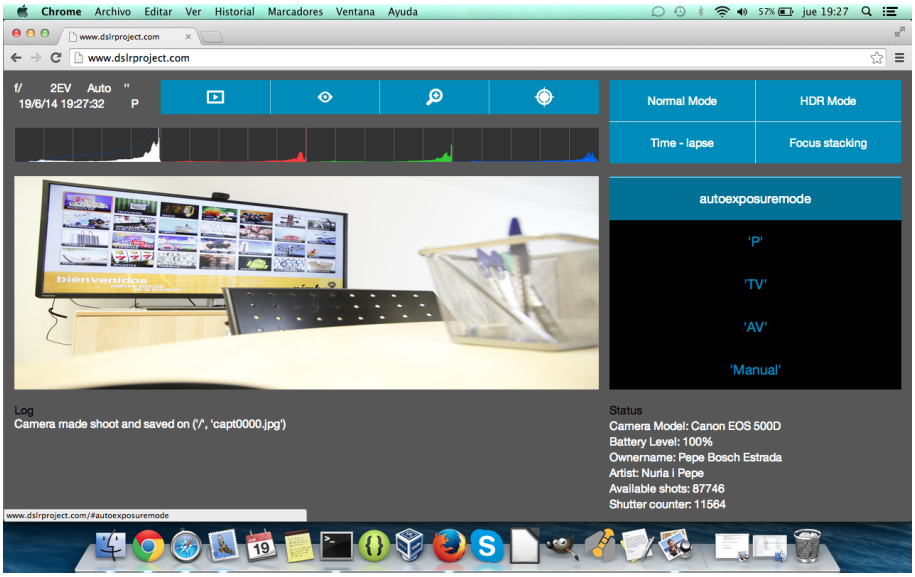

Figura 10.9: Desplegament dels possibles valors que existeixen dintre d'un paràmetre

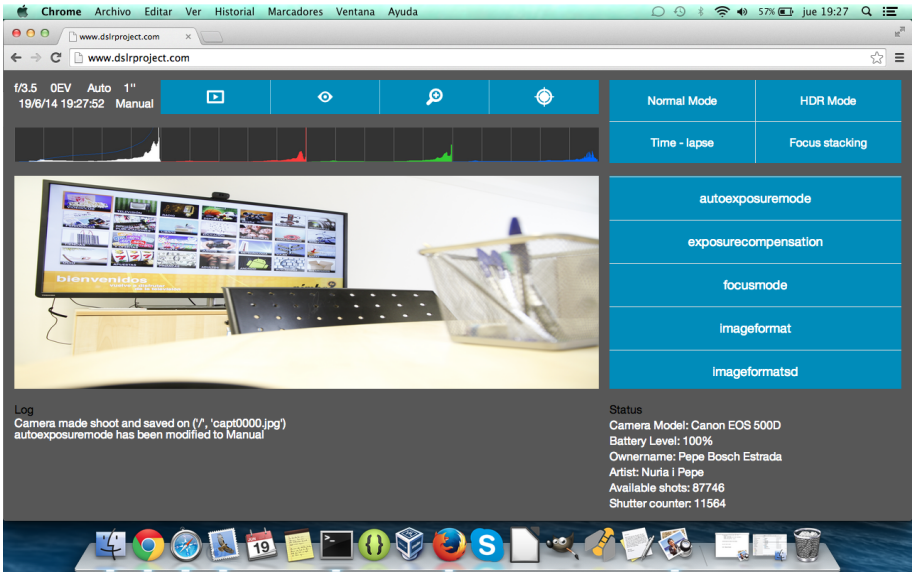

Figura 10.10: Visualització de la resposta en el log i actualització del mode per la modificació del paràmetre autoexposuremode

10.2.4 Modalitat HDR

La modalitat HDR et permet configurar cada frame amb els paràmetres desitjats. Per poder afegir captures només cal fer clic al botó que es mostra un signe més, figura 10.11; cada cop que pitgem apareixerà un frame, figura 10.12 10.13; i podrem configurar cada frame com vulguem, figura 10.14; en cas d'error utilitzant el botó amb un menys es podrà eliminar el frame, figura 10.15; per últim un cop configurats tots els frames podem disparar amb el botó Shot i realitzarà tot el procès com si estiguessis el mode Normal.

Quan el procès finalitza, l'aplicació et retorna un fitxer zip amb totes les fotografies, tant el resultat HDR com les utilitzades per fer-ho. Tant els modes Timelapse com el Focus stacking també retornen el fitxer un cop finalitzat el procès, figura 10.16.

Es possible que tinguem el pop-up bloquejats i aparegui el següent missatge, figura 10.17, per poder descarregar el fitxer només cal seleccionar la url que ha bloquejat el navegador.

A l'interior del fitxer zip trobarem tots elements com es mostra a la figura 10.18

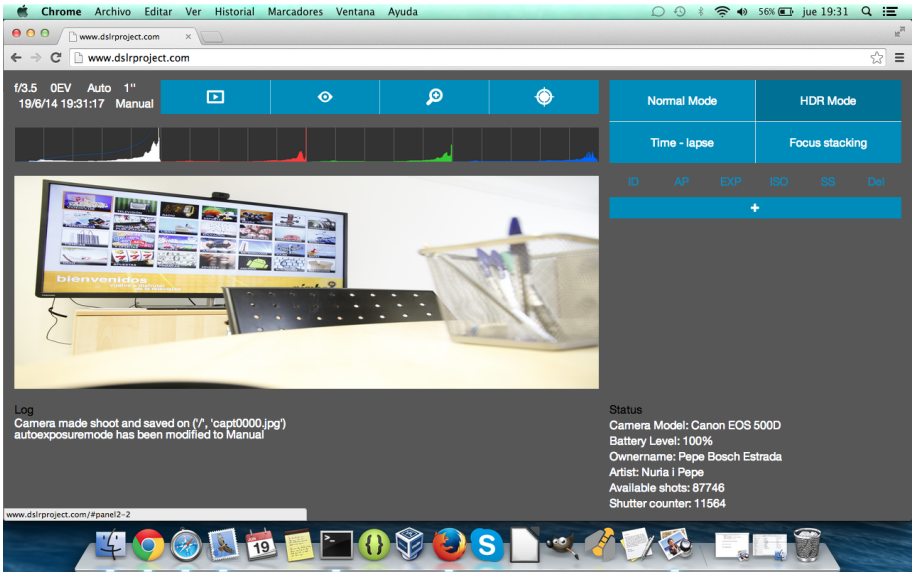

Figura 10.11: Visualització del mode HDR

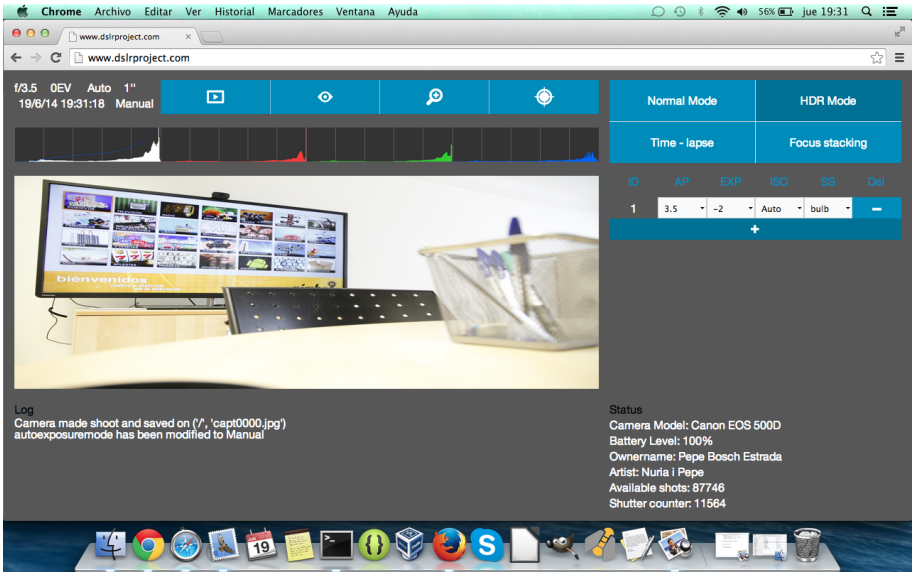

Figura 10.12: Frame afegit utilitzant el botó 'més' en el mode HDR

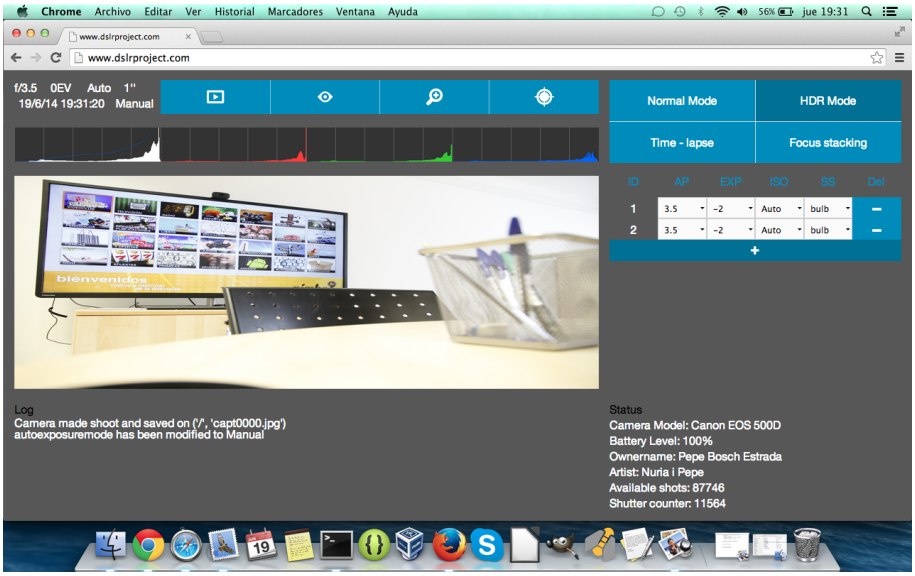

Figura 10.13: Cada cop que utilitzem el botó + s'afegeix un nou frame

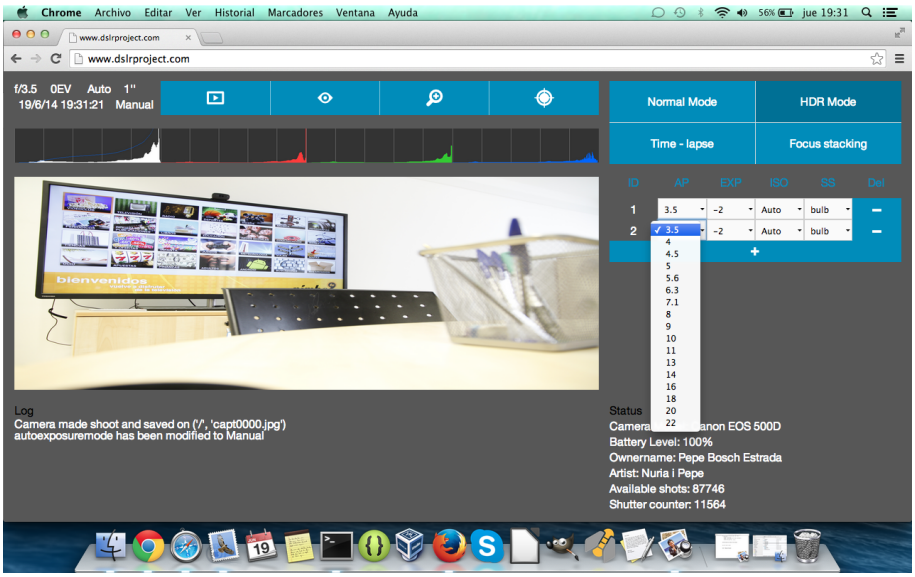

Figura 10.14: Visualització dels valors possibles del primer paràmetre (apertura) utilitzant una Canon EOS 500D

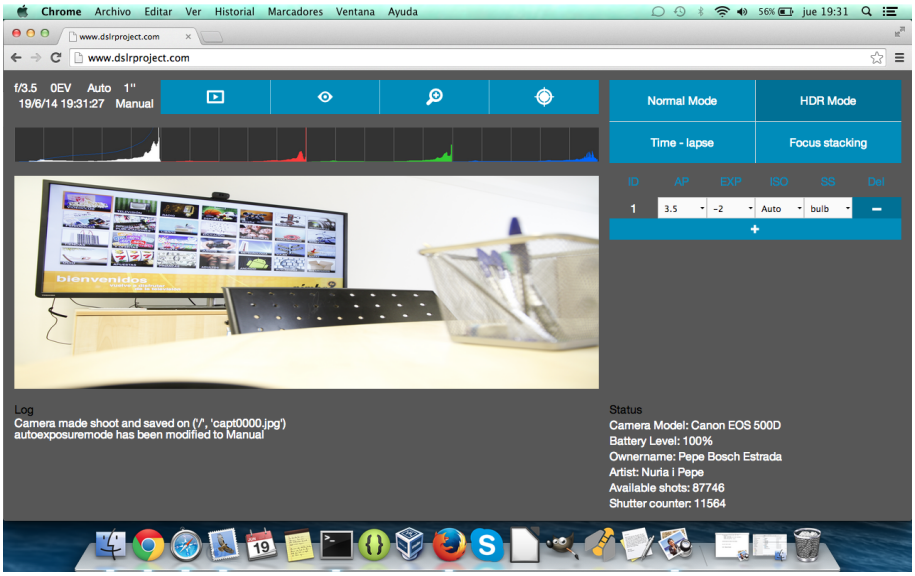

Figura 10.15: Eliminació d'un frame utilitzant el botó - del mode HDR

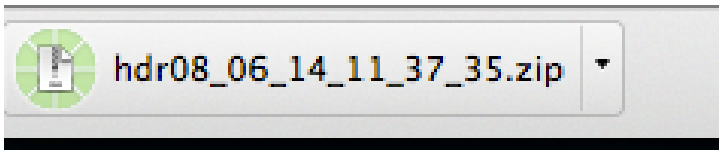

Figura 10.16: Vista d'un fitxer descarregat

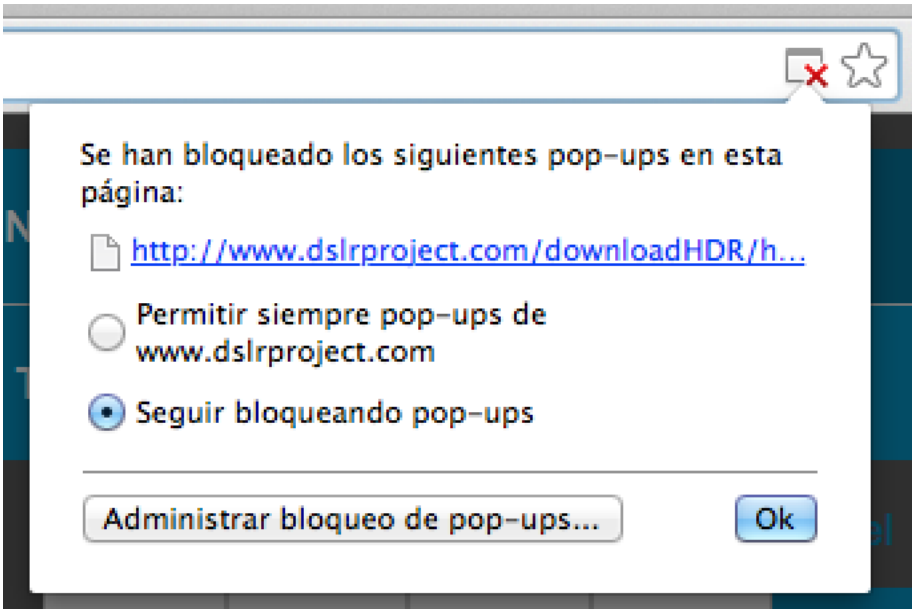

Figura 10.17: Missatge de pop-up bloquejat en les descàrregues del fitxer zip

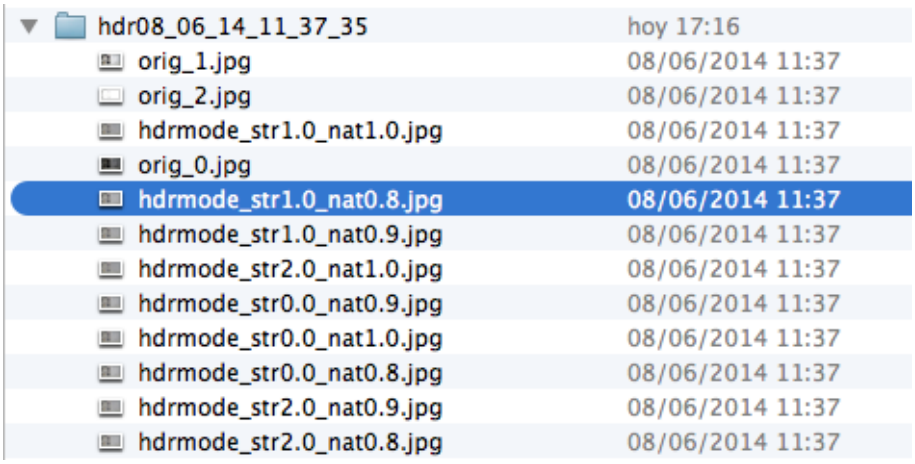

Figura 10.18: Vista del fitxers que conté el zip retornat del mode HDR

10.2.5 Modalitat Timelapse

El mode Timelapse necessita 3 paràmetres per poder realitzar-se: en primer lloc l'interval de segons entre les fotografies, en segon lloc, el nombre de fotografies que ha de fer la càmera i per últim quants frames per segons es realitza el vídeo, figura 10.19. Com s'ha explicat en el mètode anterior totes s'executen utilitzant el botó Shot, figura 10.20, i en cas de no omplir correctament les dades apareixerà un error emergent, figura 10.21.

Un cop el procès es finalitza, et retorna el fitxer zip amb totes les imatges i el vídeo generat.

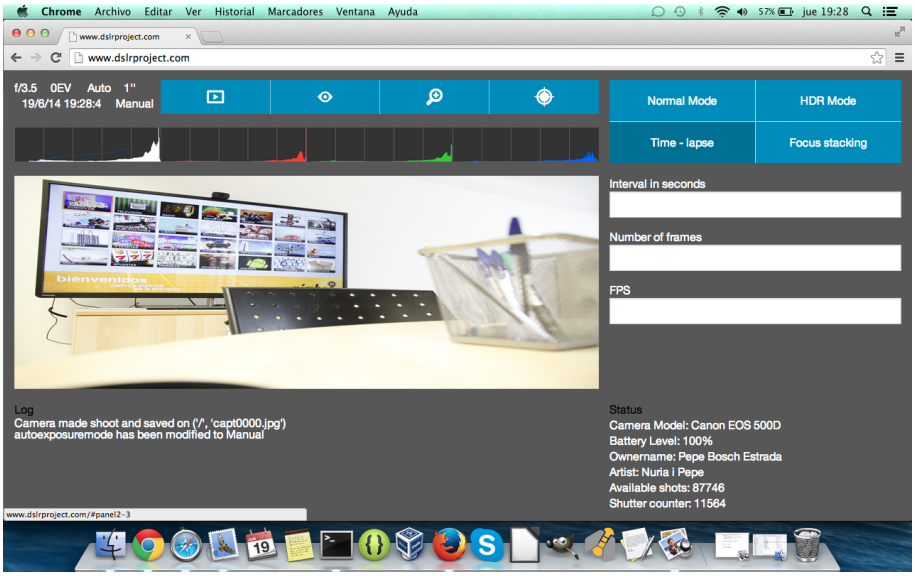

Figura 10.19: Visualització del mode Timelapse

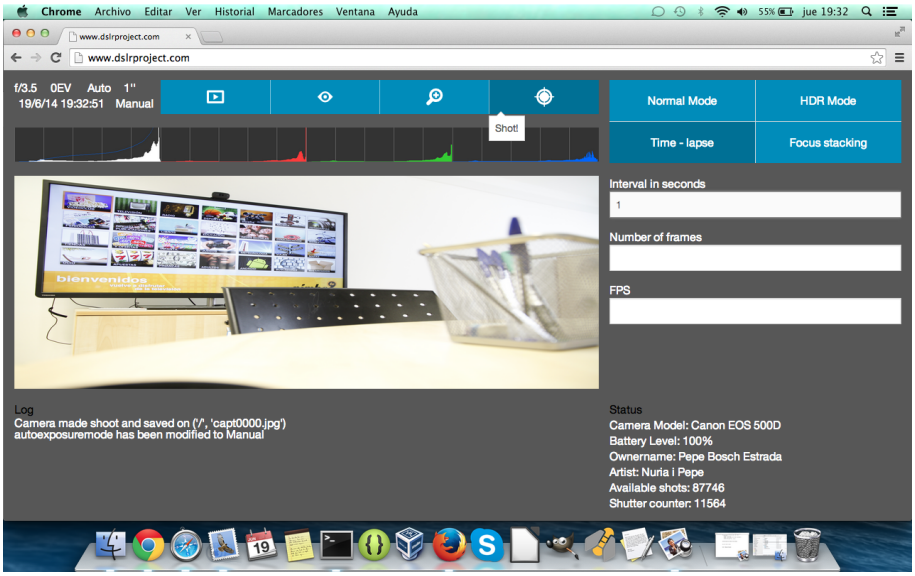

Figura 10.20: Qualsevol mode es dispara amb el mateix botó Shot!

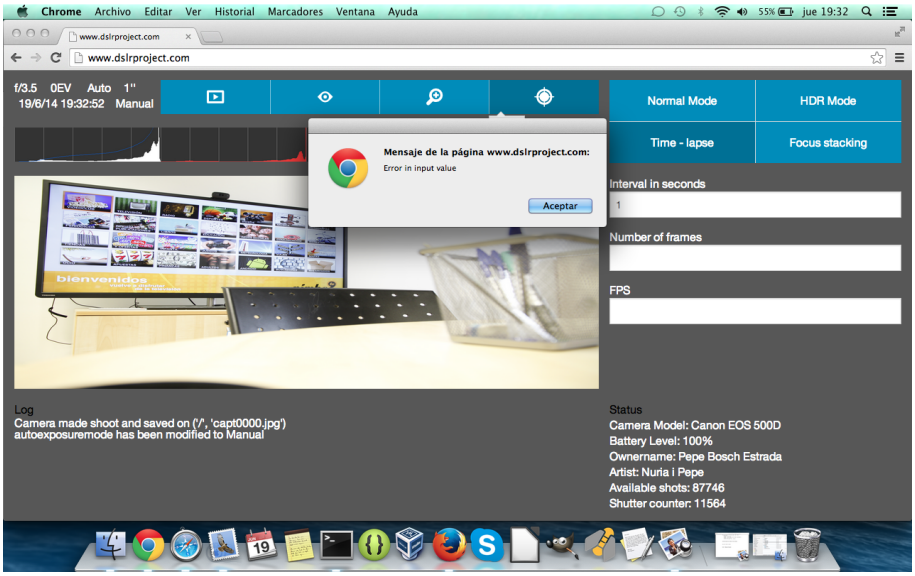

Figura 10.21: Visualització d'error quan existeix manca de dades

10.2.6 Modalitat Focus stacking

Aquest mètode únicament necessita el número d'enfocament que es vol fer, figura 10.22. Com els altres mètodes té comprovacions d'erros en la

introducció del número i un cop l'aplicació realitza el procès et retorna un fitxer zip amb totes les imatges.

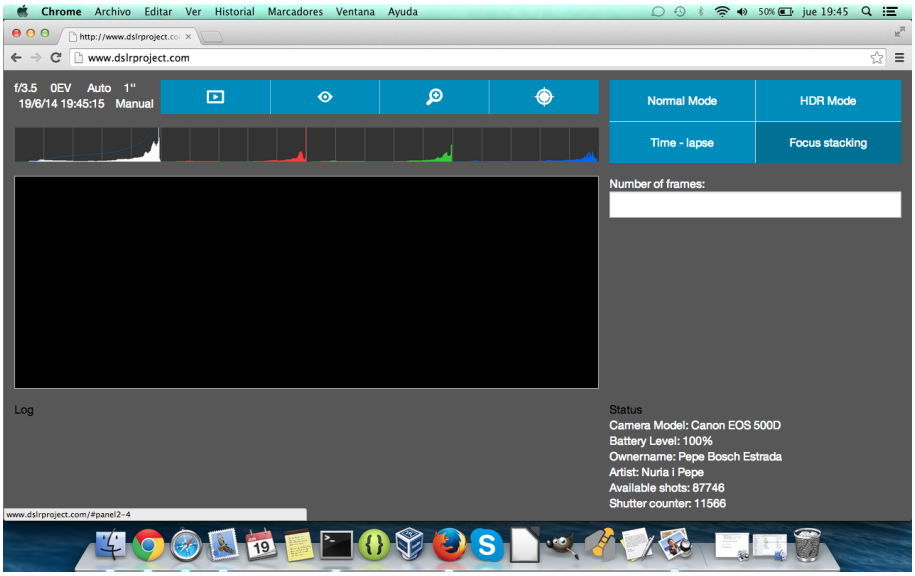

Figura 10.22: Visualització del mode Focus stacking