Treball Fi de Grau
GRAU D'ENGINYERIA INFORMÀTICA
Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

Aplicació web per al seguiment i planificació de tècnics informàtics

Sergi Barrera
Directora: Anna Puig
Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

1. Taula de continguts

1. Taula de continguts 3
2. Introducció 8
2.1. Àmbit del projecte 8
2.2. Motivació 8
2.3. Objectius generals 9
2.4. Tasques Específiques 9
2.4.1. Anàlisi del entorn 9
2.4.2. Definicions del rols principals 9
2.4.3. Consultes mes freqüents i agregacions 9
2.4.4. Disseny de l'arquitectura 10
2.4.5. Desenvolupament i testing 10
2.5. Planificació Temporal 10
2.5.1. Diagrama de Gantt 12
2.6. Organització de la Memòria 13
3. Anàlisi 14
3.1. Anàlisi de Requeriments 14
3.1.1. Tipus d'events 14
3.1.1.1. Incidents 14
3.1.1.2. Canvis i peticions 15
3.1.2. Obertura d'events 15
3.1.3. Organització del departament 16
3.2. Gestió del departament 17
3.3. Tipus d'usuari 18
3.3.1. Casos d'ús 19
3.3.1.1. Casos d'us usuari Tècnic 19
3.3.1.1. Casos d'us de l'usuari Administrador i superadministrador 21
3.4. Anàlisi dels requeriments de la Base de Dades 22
4. Disseny 24
4.1. Arquitectura del Sistema 24
4.1.1. Propostes Analitzades 24
4.1.2. Proposta 25
4.1.2.1. Diagrama de l'arquitectura 25
4.1.2.2. Django 26
4.1.2.3. MariaDB 26
4.1.2.4. HTML + Bootstrap 27
4.1.2.5. Javascript + Chart.js 27

4.2. Disseny de la base de dades 27
4.3. Disseny del modelat de Django 29
4.3.1. Detall de l'arquitectura dels models de Django. 29
4.3.2. Detall del modelat de gestió d'usuaris 30
4.4. Funcionament d'un event del sistema 32
4.5. Vistes i Templates 34
4.5.1. Funcionament de les views 34
4.5.2. Funcionament de les templates 34
4.6. Informació Visual 35
4.6.1. Informació complementaria al gràfics 38
4.7. Sincronització amb la BBDD Oracle 38
5. Resultats i Simulacions 40
5.1. Usuari tècnic 40
5.2. Usuari Administrador 43
5.3. Conclusió 45
6. Referències bibliogràfiques 47
Apèndix A: Manual tècnic 48
A.1 Instal·lació: Requeriments mínims 48
A.2. Manual del desenvolupador 48
Entorn Virtual 48
Organització Estructural del projecte 49
Configuració dels scripts 49
Disseny del back-end 50

ABSTRACT

El projecte que es documenta a continuació anomenat, "Aplicació web per al seguiment i planificació de tècnics informàtics", tracta d'una aplicació web destinada a la gestió i seguiment de tècnics informàtics.

La principal funció a la que està destinada a la plataforma, és donar suport als tècnics informàtics que la utilitzin i especialment als gestors informàtics que hagin de fer el seguiment d'aquests tècnics.

En el present document es detalla el procés de anàlisi, planificació, disseny i desenvolupament de tota l'arquitectura que requereix la plataforma.

ABSTRACT

El proyecto que se documenta a continuación, con el nombre , “Aplicació web per al seguiment i planificació de tècnics informàtics”, trata sobre una aplicación web destinada a la gestión y seguimento de técnicos informaticos.

La principal función a la que el proyecto está destinado, és a dar soporte a los técnicos informáticos que harán uso de la aplicaión y especialmente, a los gestores informàticos que deberán hacer el seguimiento de éstos técnicos.

En el presente documento se especifica el proceso de análisis, planificación, diseño y desarrollo de toda la arqutectura que la plataforma requiere.

ABSTRACT

The he project that is documented below, with the name, “Aplicació web per al seguiment i planificació de tècnics informàtics”, is about a web application dedicated to the management and monitoring IT technicians.

The main function to which the project is intended, is to support the computer technicians who will make use of the application and especially, the department managers who will be in chage of monitoring these technicians.

This document specifies the process of analysis, planning, design and development of all the architecture that the platform requires.

2. Introducció

2.1. Àmbit del projecte

Aquest projecte tracta d'ajudar a la gestió diària d'un departament de Serveis Gestionats. Es vol crear una eina que faci d'enllaç entre el personal tècnic i el personal de gestió, i en faciliti la comunicació i la gestió dels events més importants per al departament.

A més, l'aplicació aportarà visibilitat de l'estat del departament a temps real. El tipus d'usuari que farà ús de l'aplicació web, son usuaris amb coneixements tècnics sobre informàtica elevats.

Tant el personal tècnic com el personal de gestió, esta acostumat a treballar amb diferents tipus d'eines de seguiment i planificació de tasques.

2.2. Motivació

La decisió de dur a terme aquest projecte, la vaig prendre quan em vaig adonar de la dificultat de realitzar un seguiment de 14 tècnics, amb un volum d'aproximadament 5000 tiquets mensuals.

Aquesta situació provoca, el desconeixement de les tasques que s'estan realitzant per part de l'equip de gestió tècnica, que a la vegada provoca, que els tècnics hagin de interrompre la seva tasca normal, per reportar l'estat actualitzat d'un incident o d'un canvi, que possiblement no estiguin realitzant en aquell moment.

Veient aquesta situació decideixo realitzar una aplicació que pugui aportar el nexe entre tècnic i responsable, que sigui el menys problemàtica i feixuga a l'hora de donar feedback pel part del equip tècnic. A la vegada serà necessari que sigui una eina útil per a l'equip de gestió, per tant ha de tenir la informació el més actualitzada possible de cara a tenir una visió general del estat del departament. L'aplicació ha de fer de nexe comunicatiu entre els diferents equips sense necessitat d'haver de interactuar personalment, deixant constància de la feina realitzada o del estat d'una activitat.

Aprofitant, la oportunitat de realitzar aquesta aplicació web, decideixo utilitzar una tecnologia desconeguda per a mi, com és Django, i així poder desenvolupar nous coneixements en l'entorn Web.

2.3. Objectius generals

En aquest projecte, l'objectiu principal és aconseguir una eina, que doni suport via web al departament, i a la vegada faci més eficients, tots els processos d'interlocució entre gestor i tècnic.

L'objectiu de l'eina es que no sigui una càrrega més dins el dia a dia del tècnic, sinó una ajuda a l'hora de prioritzar tasques i donar la millor resposta als clients.

Per la part dels gestors, l'eina ha de proporcionar informació de l'estat general del departament. Dins una mateixa pàgina han de ser capaços d'identificar quins clients generen mes volum de treball o quins tècnics tenen una cua d'incidents més gran, i possibles sobrecarregues de feina. A més ha de ser una eina, en la que es pugui identificar ràpidament si els acords de temps de resolució (SLA) es compleixen per a cada client o no.

L'objectiu principal que vol aconseguir el departament amb la utilització d'aquesta eina és aconseguir que les cues de cada equip siguin cada vegada menors, d'aquesta manera també disminuirà el temps de vida de la incidència fent que la satisfacció del client augmenti.

2.4. Tasques Específiques

2.4.1. Anàlisi del entorn

Es busca millorar la comunicació no verbal, entre el gestor o el cap de departament, i els tècnics. L'aplicatiu ha de proporcionar una forma de comunicació, entre tots dos, que sigui ràpida i fàcil d'utilitzar. Amb aquesta característica, s'evitaran les preguntes recurrents sobre l'estat del incident i les explicacions innecessàries o repetides.

2.4.2. Definicions del rols principals

És necessari identificar quins perfils d'usuari utilitzaran l'eina. Avaluar la informació a tractar i definir les restriccions a l'hora de mostrar dades.

Per altra banda, serà necessari delimitar els permisos d'usuari a l'hora de donar d'alta o baixa usuaris.

2.4.3. Consultes mes freqüents i agregacions

A l'hora de proporcionar informació, s'ha de diferenciar clarament, quina interessa a cada usuari. L'usuari gestor, ha de poder consultar tota la informació dels tiquets, ja que ha de portar el control de tot el departament. En canvi, un tècnic, ha de ser capaç, només, de consultar els tiquets que té assignat i el seu històric. Amb aquesta diferenciació s'aconseguirà que cada tècnic tingui unes dades fiables sobre el propi rendiment, i pugui

identificar quin es el client al que li dedica més esforç. I a la vegada al gestor, tenir aquesta visió diagonal del departament.

2.4.4. Disseny de l'arquitectura

L'eina ha de ser capaç d'autonodri-se de dades, en aquest cas tiquets, per tant ha de tenir connexió a la base de dades principal i poder alimentar-se de tota la informació necessària.

L'eina ha de ser capaç d'actualitzar-se automàticament, mantenint l'estat dels comentaris que ja estan a l'eina, per tal de no perdre informació o haver de fer "reworking".

L'arquitectura de l'eina ha de permetre les connexions simultànies de tots els tècnics de l'àrea.

Cal especificar, que l'objectiu es tenir una eina lleugera, que tingui una dificultat d'utilització molt baixa, per tal de fer més eficient el temps de resolució del tècnic.

2.4.5. Desenvolupament i testing

Per a dur a terme el desenvolupament s'ha d'identificar, quins son els punt clau de l'arquitectura, per poder decidir quines són les eines més adequades a l'hora del desenvolupament de la aplicació.

2.5. Planificació Temporal

El projecte s'ha dividit en quatre grans grups de desenvolupament, i una fase inicial d'aprenentatge. Fase recerca, fase de disseny, fase de desenvolupament, i fase de documentació.

A la fase de recerca, es s'identifiquen el requisits que necessita l'entorn per a ser productiu i com poder millorar el dia a dia. S'analitzen algunes eines de BI com QlikSense, però no aporten el component de comunicació que es busca en aquest projecte.

Dins la fase de disseny, s'estudia com enllaçar les dues bases de dades sense atacar constantment a la base de dades d'origen, ja que podríem causar problemes de rendiment. També s'estudia quins son els components necessaris de la base de dades destí. Es dissenya el front-end de la aplicació i el back-end. S'estudia els models de Django a crear i l'arquitectura.

En la fase de desenvolupament, s'implementa tota la aplicació, a l'hora que es van fent testos de dades, per assegurar que els resultats dels scripts de connexió son correctes.

Es realitzen tres fases intercalades de disseny i desenvolupament. Per tal d'adequar els requisits amb el disseny.

Es fan tests de la carrega de dades, i de la duració de la mateixa. S'implementa una solució per a la lenta actualització dels tiquets.

Fase de documentació, es realitzen els manuals i memòria de la aplicació.

Especificació de les fases i les tasques programades i mostrades també en la figura 1.

Fase d'aprenentatge: 30 dies

Fase Recerca: Total (24 dies)

Identificació de les possibles solucions – 5 dies

Recull de requisits – 6 dies

Identificar les funcionalitats necessàries – 5 dies

Anàlisi d'alternatives (State of the art) – 8 dies

Fase de disseny: Total (22 dies)

Anàlisi de extracció de dades – 10 dies

Disseny del front-end – 5 dies

Disseny del front-end Fase II – 3 dies

Disseny del front-end Fase III – 2 dies

Disseny dels models – 2 dies

Fase de desenvolupament (49 dies)

Implementació dels scripts d'extracció i càrrega -- 5 dies

Test de fiabilitat de dades dels scripts -- 3 dies

Implementació de la Base de dades destí -- 4 dies

Implementació del model de dades de Django -- 7 dies

Tests de càrrega de dades dels scripts -- 5 dies

Desenvolupament del front-end -- 14 dies

Desenvolupament del back-end -- 15 dies)

Desenvolupament del front-end Fase II – 6 dies

Desenvolupament del back-end Fase II – 6 dies

Desenvolupament del front-end Fase III -- 7 dies

Desenvolupament del back-end Fase III -- 5 dies

Fase de documentació i test usuari -- 16 dies

Realització de test d'usuari -- 3 dies

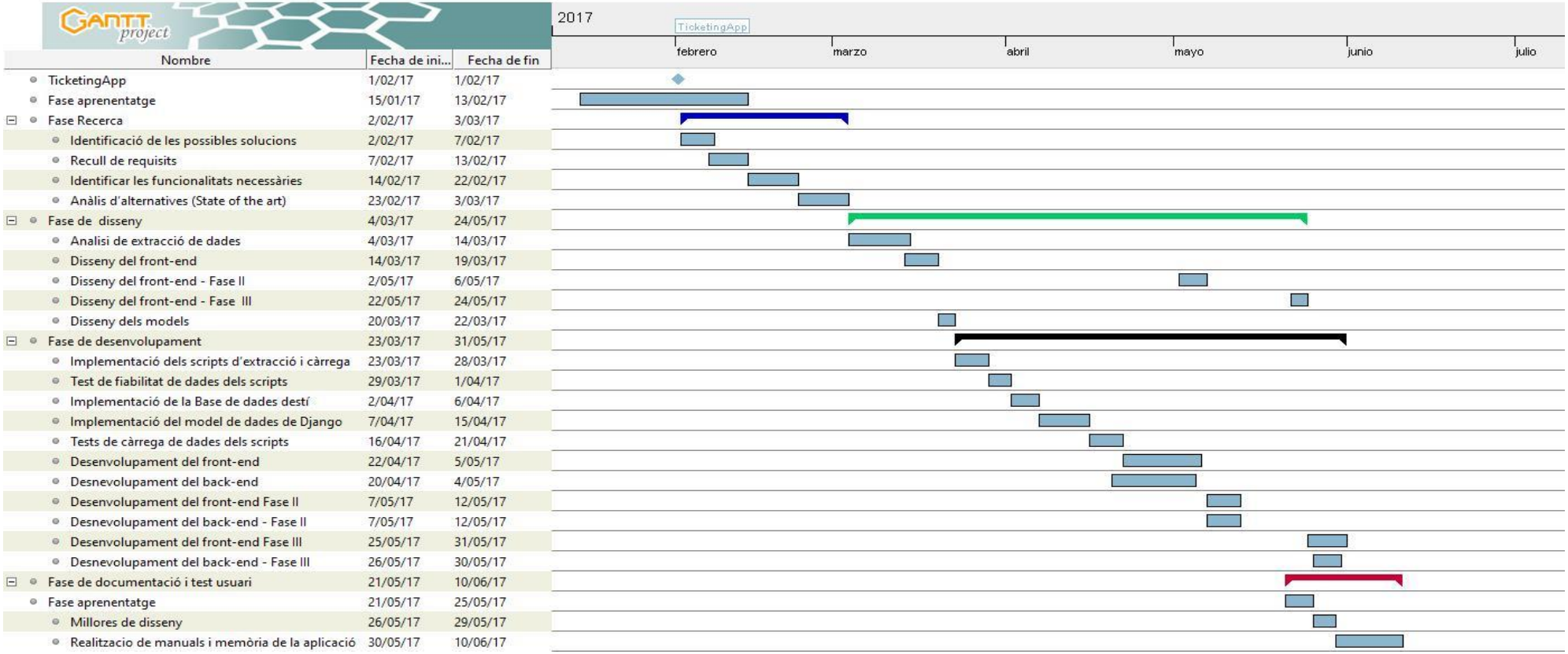
Millores de disseny -- 3 dies

Realització de manuals i memòria de la aplicació -- 10 dies

Estimació total de temps invertit en el projecte: 121 dies.

2.5.1. Diagrama de Gantt

A continuació es detalla el diagrama de Gantt del procés de realització del projecte.



Nombre	Fecha de ini...	Fecha de fin
TicketingApp	1/02/17	1/02/17
Fase aprenentatge	15/01/17	13/02/17
Fase Recerca	2/02/17	3/03/17
Identificació de les possibles solucions	2/02/17	7/02/17
Recull de requisits	7/02/17	13/02/17
Identificar les funcionalitats necessàries	14/02/17	22/02/17
Anàlis d'alternatives (State of the art)	23/02/17	3/03/17
Fase de disseny	4/03/17	24/05/17
Analisi de extracció de dades	4/03/17	14/03/17
Disseny del front-end	14/03/17	19/03/17
Disseny del front-end - Fase II	2/05/17	6/05/17
Disseny del front-end - Fase III	22/05/17	24/05/17
Disseny dels models	20/03/17	22/03/17
Fase de desenvolupament	23/03/17	31/05/17
Implementació dels scripts d'extracció i càrrega	23/03/17	28/03/17
Test de fiabilitat de dades dels scripts	29/03/17	1/04/17
Implementació de la Base de dades destí	2/04/17	6/04/17
Implementació del model de dades de Django	7/04/17	15/04/17
Tests de càrrega de dades dels scripts	16/04/17	21/04/17
Desenvolupament del front-end	22/04/17	5/05/17
Desnevolupament del back-end	20/04/17	4/05/17
Desenvolupament del front-end Fase II	7/05/17	12/05/17
Desnevolupament del back-end - Fase II	7/05/17	12/05/17
Desenvolupament del front-end Fase III	25/05/17	31/05/17
Desnevolupament del back-end - Fase III	26/05/17	30/05/17
Fase de documentació i test usuari	21/05/17	10/06/17
Fase aprenentatge	21/05/17	25/05/17
Millores de disseny	26/05/17	29/05/17
Realitzacio de manuals i memòria de la aplicació	30/05/17	10/06/17

Figura 1. Diagrama de Gantt

2.6. Organització de la Memòria

La memòria del projecte té una organització dividida en 6 blocs, els quals es subdivideixen en categories.
Al primer bloc, hi podem trobar la introducció al projecte. Aquesta introducció engloba un petit anàlisi de l'àmbit del projecte, així com la motivació i els objectius tant generals com específics del projecte.
A més, s'adjunta la planificació que s'ha seguit per dur-lo a terme dins els termines establerts.

En el segon bloc, s'especifica l'anàlisi del projecte. Es situa el context i les necessitats, fent l'anàlisi de requeriments, tant funcionals com tècnics.
En aquest apartat, també s'especifiquen els casos d'us i els diferents tipus d'usuari que utilitzaran l'aplicació.

Dins el tercer bloc, trobem tot el disseny de l'aplicació. Es fa un model d'arquitectura del projecte. S'analitzen diferents propostes d'implementació i es detalla el funcionament de les tecnologies escollides.

En el quart bloc, es detallen els resultats que s'han obtingut, a partir dels casos d'ús ideats en el primer bloc. D'aquesta manera, es pot decidir si el projecte ha estat realitzat amb èxit, complint el objectius establerts.

En el cinquè bloc podem trobar les referències bibliogràfiques utilitzades.
S'adjunten, també dos annexes, els diferents manuals per a cada perfil d'usuari o desenvolupador.

3. Anàlisi

3.1. Anàlisi de Requeriments

Aquest projecte tracta d'aprofundir en les eines de seguiment dels tècnics d'un departament de Serveis Gestionats, aportant valor a la informació emmagatzemada a l'eina de tiqueting de la empresa.

El volum total de treball d'un equip de Serveis Gestionats tracta, estudia, i resol múltiples incidències, peticions i canvis.

Un exemple de d'incidència, podria ser el nivell elevat d'ocupació d'un disc. Es considerarà un incident, ja que ningú sol·licita una alliberació d'espai, però si no s'aplica es possible que causi problemes a l'entorn de client.

Un exemple de canvi seria l'actualització d'una versió de programari del client, ja que per realitzar aquesta tasca es necessitarà un sol·licitant.

3.1.1. Tipus d'events

Considerem un event, tot tipus de interacció que ens generi una activitat a realitzar per algun dels grups de resolució del departament.

Aquestes interaccions segons el contingut poden derivar en dos grans àrees.

3.1.1.1. Incidents

Events de Monitorització: Aquest tipus d'events es caracteritzen perquè es creen automàticament a l'eina de tíqueting, aportant la informació de l'alarma generada per el sistema de monitoratge.

Dins la informació del tiquet, consta com a sol·licitant aquesta eina. Solen tenir una vida curta, ja que habitualment tenen un procediment adjunt, que fa que la correcció sigui immediata.

Incidents reportats per client o detectats per el tècnic: Aquests problemes solen tenir una vida més amplia, sempre lligada a la criticitat de l'incident. Aquesta criticitat la podem calcular a partir de l'afectació que tingui l'incident (1 – Empresa, 2 – Departament, 3 – Grup d'usuaris, 4 – Usuari únic), juntament amb la urgència que estableixi l'incident, segons el criteri del tècnic, i calculat a partir del entorn afectat. Si la part afectada del sistemes, es determinant per a la producció de client, categoritzarem la urgència com a 1 – Crítica. Solen requerir que el tècnic faci una investigació més profunda per a poder resoldre'l amb satisfacció.

3.1.1.2. Canvis i peticions

Els canvis: Podem definir com a canvi, tota actuació, que comporta un tall en la producció del client. Aquests, es programen setmanalment o cada dues setmanes, conjuntament amb client, i requereixen aprovació prèvia per se executats. Solen realitzar-se fora del horari productiu, per reduir al màxim la afectació a la producció.

Els canvis tenen sempre associat un RFC (Request for Change), on es detalla tota la informació relacionada amb l'actuació, com per exemple, la data i hora d'actuació, qui serà l'encarregat d'executar el canvi, quants equips intervenen, o quin es el rollback del canvi. Un exemple molt clar del que podríem especificar com a canvi amb RFC, son les actualitzacions dels servidors Windows, que solen necessitar un reinici, i per tant, un tall de servei, per a que s'apliquin correctament les actualitzacions.

Les Peticions: Les peticions es defineixen com a sol·licituds per part de client que no tenen una afectació crítica. Un percentatge elevat de les peticions que es reben sol tenir un procediment associat, i acostumen a ser resoltes amb certa rapidesa, ja que no és necessari pactar cap finestra horària amb el client per tal d'aplicar-la. Exemples clars de peticions, englobarien totes les sol·licituds de bloquejos de compte, restauració de fitxers. Alta d'usuaris a Active Directory etc.

3.1.2. Obertura d'events

Aquests events poden sorgir de diverses vies:

Elements de Monitorització: Es produeixen de forma automàtica a través del sistema de monitoratge dels equips dels diferents clients. Solen ser alarmes de indisponiblitat de les diferents plataformes de client o bé alarmes sobre utilització de recursos, CPU, Memòria, Disc, Creixement d'una Base de Dades, etc..

Poden indicar també, caigudes de serveis crítics com per exemple una web i fins i tot, son capaços de determinar si l'accés a un portal és més lent del habitual.

Via telefònica o Mail des de client: Es produeixen quan client, té una petició o detecta qualsevol anomalia en el seu sistema. Aquest tipus d'obertures, moltes vegades son conjuntes, via mail i telefònica a la vegada, ja que, si es tracta d'un incident amb una afectació important per al client, requerirà la màxima prioritat i velocitat d'actuació, per part del departament. Lentitud, recuperació de dades, mal funcionament dels aplicatius o talls en el servei, solen ser les que més es tracten a traces d'aquests mitjans.

A traves de millores als entorns: Mensualment els tècnics fan una revisió exhaustiva dels entorns d'alguns clients, com a resultat dels informes mensuals, el tècnics, proposen canvis, millores o correccions que s'han detectat als sistemes de client, normalment en forma de canvi.

3.1.3. Organització del departament

Actualment, el departament es divideix en 2 capes diferenciades, com podem veure a la figura 2.

La capa tècnica, que esta formada per tres nivells de tècnics que es dediquen a resoldre tots els incidents i canvis que hem definí a l'apartat anterior.

Els diferents nivells que podem trobar a la capa tècnica son els següents.

Nivell 1, format per l'equip d'Operacions. Aquest equip dona un servei 24x7. Està format per 6 tècnics en torns de 8 hores. S'encarrega de realitzar la recepció i escalat de tiquets, incidències o canvis que vagin entrant al departament. Responen al telèfon i tenen el primer contacte amb client quan s'obre un incident. Aquest equip, s'encarrega també d'aplicar els procediments de resolució d'incidents o canvis recurrents establerts.

Nivell 2, format per només 2 tècnics actualment, s'encarrega de tasques de resolució de canvis i incidents, amb certa complexitat. Si dins un temps prudencial no han resolt un incident, l'escalen als tècnics de Nivell 3.

Dins aquest equip no hi ha diferenciació de tecnologies, ja que està destinat a la formació del tècnic per a que adquireixin nivells alts de coneixements en tots els entorns.

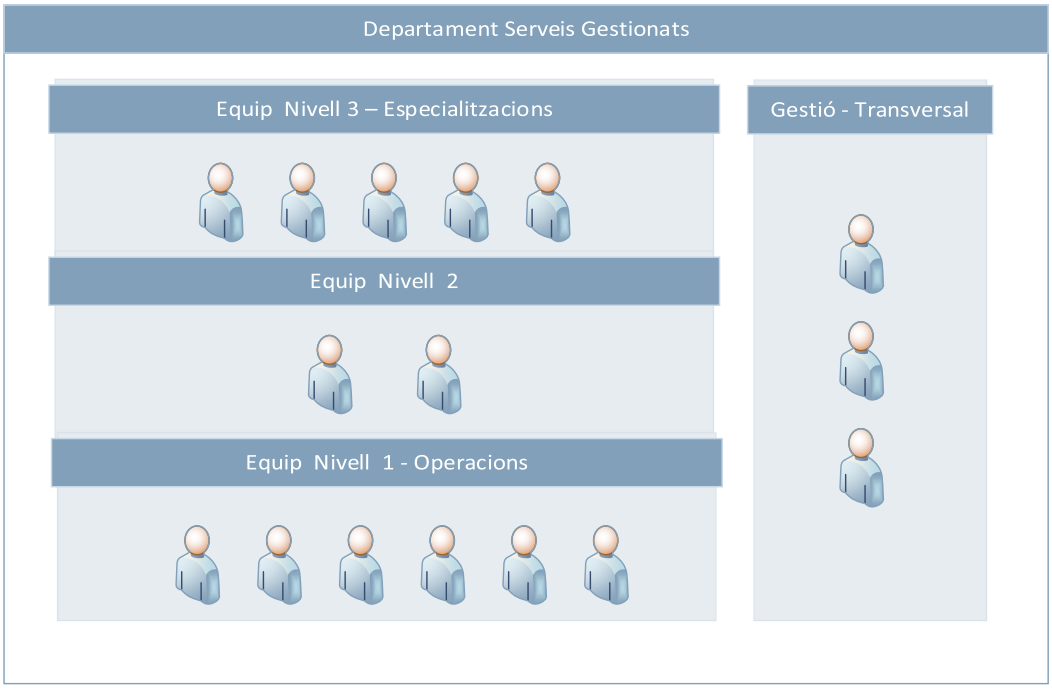


Figura 2. Organització departamental

Nivell 3, equip format per 4 tècnics, polivalents, a la vegada que especialistes en una o varies tecnologies. Aquest equip, està clarament dividit per tecnologies, i s'encarrega de la resolució d'incidents complexes, i a realitzar tasques proactives de millora a les infraestructures dels clients.

A banda de la capa tècnica, trobem la capa de gestió, una capa tranversal als tres nivells de tècnics que s'encarrega de la gestió del departament.

Gestió, equip format per 3 gestors, que s'encarreguen del seguiment del tiquets, assegurar el compliment dels nivells de servei acordats (SLA) amb els clients, assignar les tasques als diferents equips amb la prioritat adequada, gestionar les peticions i canvis que sol·liciten els clients.

Aquests canvis, solen comportar l'actuació de mes d'un equip de resolució per dur a terme correctament la sol·licitud del client, i és, el gestor, la figura en qui recau la coordinació dels diferents equips.

D'altre banda es fan càrrec de la gestió de l'equip humà, tenir control dels dies festius dels tècnics, procurar sempre cobrir les vacants tecnològiques en períodes vocacionals, i en definitiva del bon funcionament el departament.

També s'encarrega de la gestió de la eina de tiqueting, realitzant les altes i baixes de clients, personal i personal extern i de la realització dels informes de seguiment dels clients.

L'equip de gestió es fa càrrec de la part econòmica del departament, fent el càlcul de la rentabilitat dels diferents clients, mesurant l'esforç que es dedica a cada client.

A partir d'aquesta situació departamental, s'han extret els casos d'ús bàsics per a que l'aplicació fos suficientment completa per tractar tota la informació correctament, com es mostra a la figura 3 i 4.

3.2. Gestió del departament

El projecte està orientat cap a tots els integrants de l'equip de Serveis Gestionats, encara que tindrà una repercussió més elevada en l'equip de gestió, ja que tindrà una visibilitat molt realista del departament, i podrà distingir clarament els següents aspectes:

El gestor podrà veure ràpidament a quin client, se li esta dedicant mes esforç, aquesta dada molt important a l'hora de fer càlculs de rentabilitat al departament. Ja que amb les hores invertides per tècnic es pot aconseguir un cost aproximat del client.

Quin client té la mitjana de vida de tiquet més elevada, quan un client que té una vida molt tiquet molt elevada, serà segur, un problema a curt o mitja termini. D'una banda el client tindrà la sensació que no s'està treballant suficient per resoldre els seus problemes. I per altra, si no es compleixen els SLA de resolució, el

preu de la facturació baixarà, ja que no s'estan respectant els acords de temps de resolució.

L'equip de gestors, podrà distingir, quins tècnics donen millor rendiment, ja que per a que el departament funcioni, idealment, la resolució de tiquets hauria de ser bastant esglaonada, seguin la següent mètrica.

Un tècnic de nivell 1, ha de ser el tècnic que més tiquets resol, amb una mitjana de temps mes baixa, ja que s'encarregarà de la resolució dels events de monitorització i d'aplicar els procediments establerts a les alertes.

Un tècnic de nivell 2 ha de ser el pas intermig, Ha de tenir una mitjana de temps resolució relativament baixa, i una tassa de tancament de tiquet alta. Com que és tracta d'un nivell de formació, els tiquets que no es solucionen en un temps determinat els escalen al tècnics especialistes.

Un tècnic de nivell 3, ha de aportar l'experiència i resoldre els tiquets de més complexitat, aportant valor a l'administració dels sistemes del client, per tant el temps mig de tancament serà més alt, i resoldrà un menor nombre de tiquets.

Aquests indicadors, mostraran l'estat del dimensionament de l'equip. Si no es compleixen aquestes tres premisses, significarà que l'equip no té el suficient personal.

També serà necessari saber si la càrrega de treball està ben distribuïda a tots els tècnics, el volum de tiquets assignats a cada tècnic, hauria de ser similar entre nivells de resolució, sinó és un indicador clau que demostra que no s'està repartint bé l'esforç en el departament.

D'altra banda, un gestor ha de tenir la capacitat, també, de poder donar d'alta a nous usuaris, ja siguin gestors (Administradors) o nous tècnic que s'incorporin al departament.

3.3. Tipus d'usuari

Per tal de que l'aplicació pugui donar suport tant a la capa de gestió del departament com al diferents nivells de l'equip tècnic, s'han identificat dos tipus d'usuari diferents, que han de fer ús de l'aplicació.

Usuari Tècnic: Aquest usuari serà qui tingui el perfil més restringit, tenint només accés a la informació corresponent als tiquets que se li han assignat. Tindrà permisos per a modificar el seu password.

Per a poder prioritzar les seves tasques pot consultar la seva cua de tiquets, pot consultar l'estat dels tiquets que té assignats, i podrà afegir comentaris als tiquets que té assignats.

A més per a tenir una visió de la càrrega de treball al que està sotmès podrà consultar el seu històric de tancaments i assignacions, i així també saber el seu

rendiment mensual. A la figura 3 podem distingir els diferents casos d'ús que ha de tenir aquest tipus d'usuari.

Usuari Administrador : Usuari amb tots els permisos de creació i edició i accés de l'aplicació, generalment correspondrà amb la capa de gestió, ja que pot crear usuaris nous, tenir accés a tots els tiquets registrats i actualitzar l'estat de tots els tiquets de la base de dades. Aquest usuari obtindrà una visió general de l'estat del departament, tant dels tiquets que encara no s'han pogut resoldre, com l'històric d'obertures i tancaments general del departament.
Per dur a terme la tasca de control de tècnics, podrà consultar l'estat de les cues de resolució de cada tècnic, i prioritzar les seves tasques.

Usuari Superadministrador: Com a tot projecte es contempla crear també el perfil de Superusuari, serà l'únic que tingui accés al backend de la tecnologia, on podrà donar de baixa tècnics, que hagin deixat el departament. Té tots els privilegis de l'usuari Administrador, i a més, te accés al backend de l'arquitectura Django.

No es contempla l'opció d'un usuari públic, ja que es tracta informació privada de l'empresa, i es vol restringir l'accés a l'aplicació donant, només, accés personal contractat per la companyia.

3.3.1. Casos d'ús

3.3.1.1. Casos d'us usuari Tècnic

Després d'estudiar els requeriments de cada tipus d'usuari, a continuació es detallen els casos d'ús extrets de l'anàlisi d'aquests

Per a l'usuari tècnic, s'han identificat set casos d'ús que hem dividit en 3 grans blocs, Accés al Sistema, Consulta al dashboard i llistats i consultes d'indicadors, com es pot distingir a la figura 3.

A més, per al tipus usuari tècnic, es restringiran totes les dades, que no tinguin relació amb aquest. D'aquesta manera, l'usuari tècnic no podrà consultar cap tipus d'informació que no li hagi estat assignada prèviament. Per tant totes les consultes o edicions de tiquet que realitzi, sempre estaran filtrades per el seu usuari.

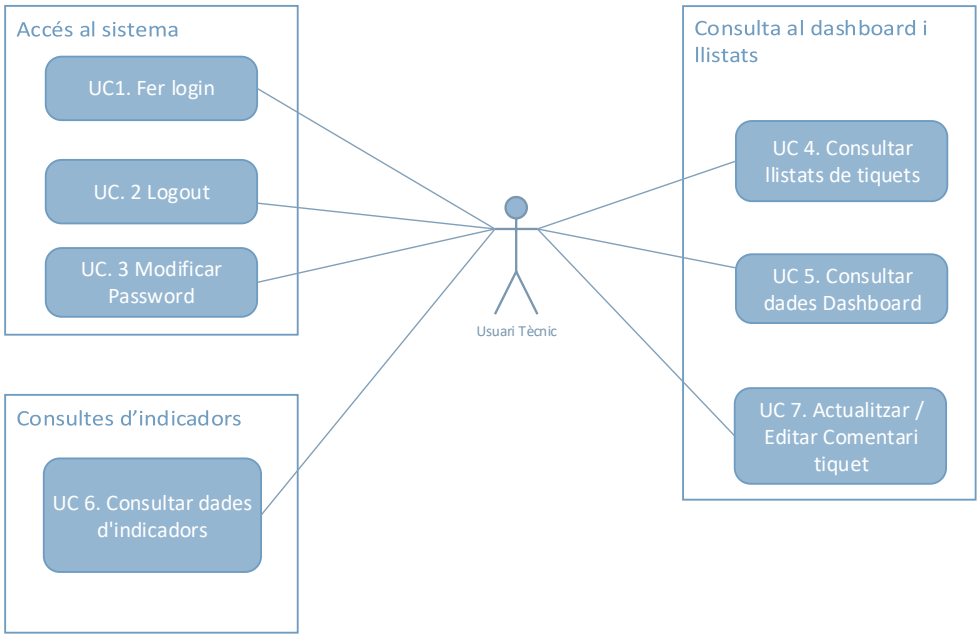


Figura 3. Casos d'ús – usuari Tècnic.

- Accés al sistema

1. Fer login

L'usuari tècnic ha de poder identificar-se en la aplicació per tal de consultar la informació

2. Logout

L'usuari ha de poder desconnectar-se del sistema

3. Modificar Password

L'usuari podrà modificar la seva password, tantes vegades com desitgi, omplint el formulari que se li proporciona

- Consultes al dashboard i llistats

4. Consultar llistats de tiquets

L'usuari té accés a les llistes on s'indica quins tiquets té oberts, i el seu detall en el moment de la consulta.

5. Consultar dades del Dashboard

El dashboard ofereix una visió general de l'estat del tècnic en el moment de la consulta, mostrant-li gràfiques i taules del tiquets assignats sense resoldre.

7. Actualitzar / Editar Comentari tiquet

L'usuari te l'opció de deixar un comentari descriptiu del estat en que es troba el tiquet.

- Consultes d'indicadors

6. Consultar dades d'indicadors

Permetrà a l'usuari tenir una visió del ser rendiment, mostrant-li gràfiques dels tiquets que se li han assignat, així com la relació de temps de tancament per client.

3.3.1.1. Casos d'us de l'usuari Administrador i superadministrador

L'usuari Administrador té els mateixos casos d'ús que l'usuari tècnic. L'única diferencia entre aquests usuaris és que l'administrador té permisos per veure la informació sense filtrar. És a dir, un usuari administrador te la visió de tots els usuaris del departament, permetent-li d'aquesta manera poder controlar el departament.

A banda dels casos d'ús abans esmentats, s'afegeix un nou grup, Gestió d'usuaris que engloba el cas d'us UC 8. Afegir Usuaris Tècnics i Administradors, ja que com a gestor de la plataforma, l'usuari Administrador, té permisos per afegir als nous tècnics que s'incorporin al departament, o a nous gestors, com es pot veure en la figura 4.

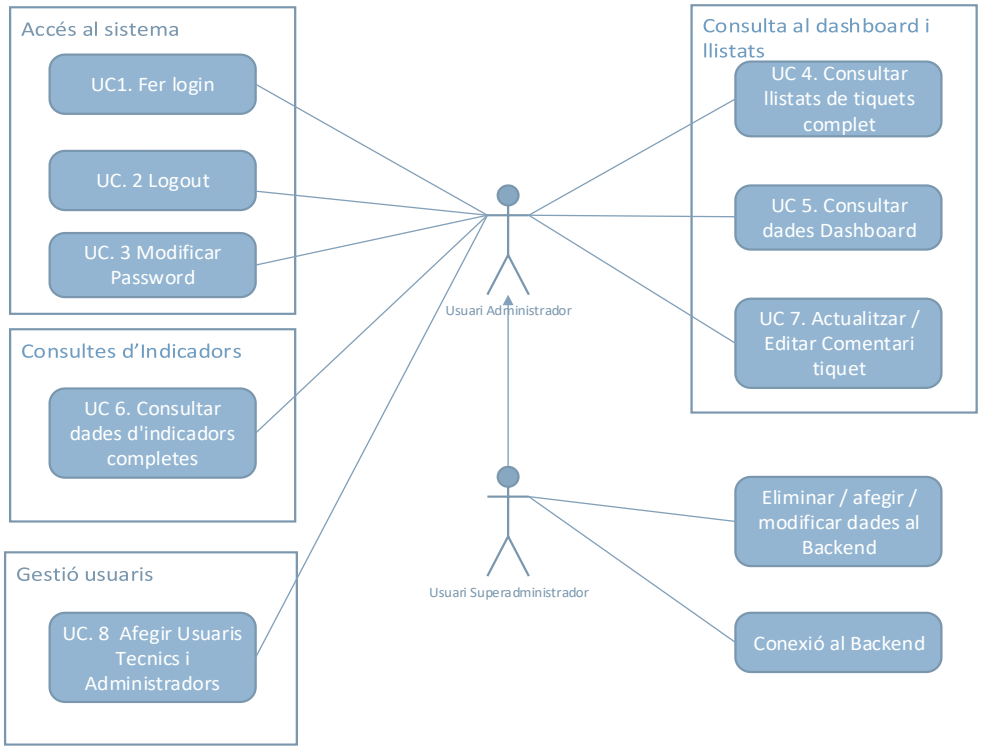


Figura 4. Casos d'ús – usuari Administrador.

3.4. Anàlisi dels requeriments de la Base de Dades

Donats els casos d'ús estudiats en l'apartat anterior, sorgeix la necessitat de que l'aplicació sigui capaç d'emmagatzemar totes les dades dels tiquets, clients i tècnic, i pet tant, per tal de poder realitzar el projecte, serà necessària la creació d'una base de dades, que s'encarregarà de guardar de forma persistent totes les dades que s'hagin de consultar o editar.

Per això serà necessària una arquitectura que ens permeti realitzar aquest procés.

Com es pot veure a la figura 5, serà necessària una connexió unidireccional, només de consulta, a una base de dades Oracle. Aquesta connexió, realitzarà les querys necessàries per a aconseguir tota la informació que requerim, ja que la base de dades esmentada, pertany a l'aplicació de tiqueting de l'empres i que conté tota la informació necessària per a implementar l'aplicació.

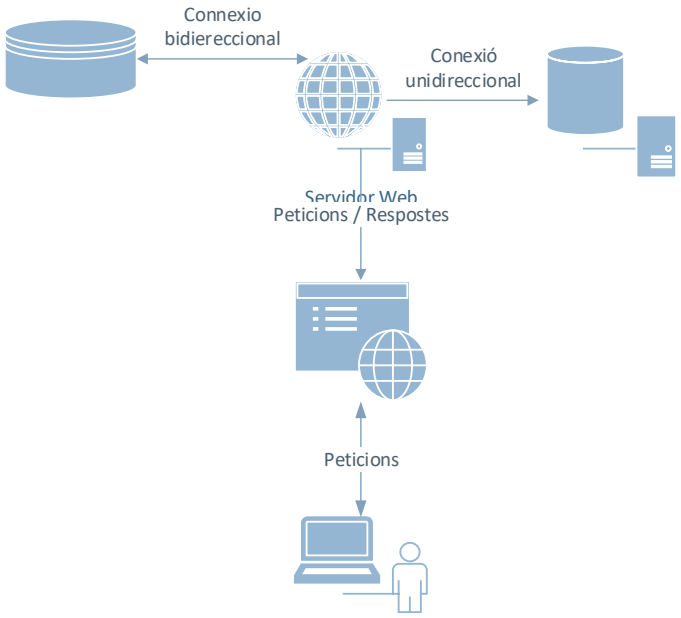


Figura 5. Diagrama funcional.

Un cop extretes les dades, requerirem d'un nexe entre aquestes dades i la base de dades pròpia de l'aplicació. Aquest nexe ha de tenir la capacitat de llegir la informació que extraiem de les consultes a l'origen de la informació, la base de dades Oracle, i transformar-la per a poder, després guardar-la en la base de dades pròpia de la aplicació que s'està dissenyant.

Per tal de poder servir les pàgines web, l'aplicació ha d'estar muntada dins un servidor Web, capaç de respondre amb rapidesa les peticions dels usuaris de l'aplicació. S'ha de tenir en compte també la carrega de processament que suposarà

al servidor realitzar les consultes i el tractament de dades cada vegada que es faci l'extracció d'informació de la base de dades Oracle. Per tant el servidor haurà de tenir connexió amb el servidor de base de dades Oracle, sinó serà impossible, realitzar aquestes consultes.

Cal tenir en compte també, on s'allotjarà la base de dades de l'aplicació. Per decisió d'infraestructura i qüestions de velocitat, també estarà dins el servidor web, ja que formarà part de l'entorn de l'aplicació, i serà més fàcil per a l'aplicació tractar les dades.

4. Disseny

4.1. Arquitectura del Sistema

4.1.1. Propostes Analitzades

Per tal de poder donar una solució al problema plantejat, s'han analitzat diferents tecnologies, per a realitzar el projecte.

Per una part s'ha estudiat la possibilitat d'utilitzar el CMS Wordpress [5]. La arquitectura de Wordpress, suporta una base de dades, i amb una diversitat de plug-ins molt amplia. La premisa de Wordpress és crear molt ràpidament, una web funcional, utilitzant els plug-ins que s'ofereixen. El fet de que es recomani la utilització de plug-ins dificultarà molt la tasca de mantenir l'aplicació, ja que s'hauran d'actualitzar aquests plug-ins-

Enfocat cap a realitzar webs de tipus informatiu, com poden ser blogs o pàgines amb un contingut molt estàtic, no complia els requeriments a l'hora de mostrar gràfics i actualitzar l'estat dels tiquets amb rapidesa.

Per altra banda, s'ha estudiat abordar la solució del problema amb Django[1], que és un framework per a desenvolupar aplicacions web, open source programat completament amb Python, que promou la codificació neta i àgil. Django es un framework, que comparat amb la opció de Wordpress, un CMS, ens dona eines més útils a l'hora de construir una web complexa, amb moltes consultes a bases de dades.

Finalment s'ha estudiat la possibilitat d'utilitzar una eina ja implementada com és QlickSense [6], que és una eina de Business Inteligence, que pot connectar directament a la Base de Dades Oracle. És molt potent i capaç de realitzar anàlisi de KPI i altres indicadors. Aporta una gran capacitat gràfica i moltes utilitats per mostrar gràfics.

El problema d'aquesta eina, es que no aporta la comunicació que es busca entre tècnic i gestor.

Es podria contemplar com a eina complementaria.

S'ha escollit Django, com a eina per desenvolupar la aplicació perquè ens permet tenir un control total sobre les dades.

Wordpress, ens aporta velocitat a l'hora de la creació de continguts estàtics, amb molts plugins que ajuden a crear continguts de forma molt rapida.

Django aporta una solució on la premissa és un codi, net i ordenat, mentre que Wordpress, aposta per plugins, i el poc desenvolupament de la aplicació.

Amb Wordpress es pot aconseguir la mateixa aplicació, en termes de front-end, que en Django, però serà molt més difícil de mantenir. Ja que on Django ofereix eines per construir aplicacions completes, wordpress aposta per instal·lar dins el CMS les aplicacions necessàries per mostrar la informació que es requereixi.

4.1.2. Proposta

4.1.2.1. Diagrama de l'arquitectura

Aquest projecte consta de dues parts clarament diferenciades en quant al disseny de l'arquitectura, com es mostra en la figura 6.

Per una banda tenim un servidor que allotja una base de dades Oracle. Aquesta Base de dades, es la que recull tota la informació de l'eina de tiqueting. Consta de més de 150 taules, de les quals utilitzarem tan sols 9.

Per a realitzar aquesta extracció s'han programat uns scripts que recullen totes les dades necessàries per a dur a terme el projecte. Son scripts programats en python integrats dins de la aplicació dins l'apartat de middleware. Aquests scripts s'encarreguen de realitzar consultes a la base de dades oracle, parsejar tota la informació, i guardar-la en un arxius XML, que ens serviran per a poblar la base de dades pròpia de Django, que es MariaDB.

La base de dades de Django es troba muntada sobre una maquina Ubuntu 14.04.5 LTSc, sense entorn gràfic en el CPD de l'empresa. Aquest servidor accepta connexions via a SSH i FTP.

Només és accessible a través de la plataforma VDI de la companyia, això fa que tant el desenvolupament del projecte com consultar l'aplicació web, sigui possible des de qualsevol ordinador connectat a internet del mon. L'entorn VDI es un servei de IaaS (Infrastructure as a Service) sota la tecnologia Citrix, que ofereix una solució d'escriptoris virtuals via web.

Amb aquesta solució podem connectar directament al Web, per el port que li especifiquem al servidor integrat de Django. Els ports recomanats son el 80 o 8080 i 8081.

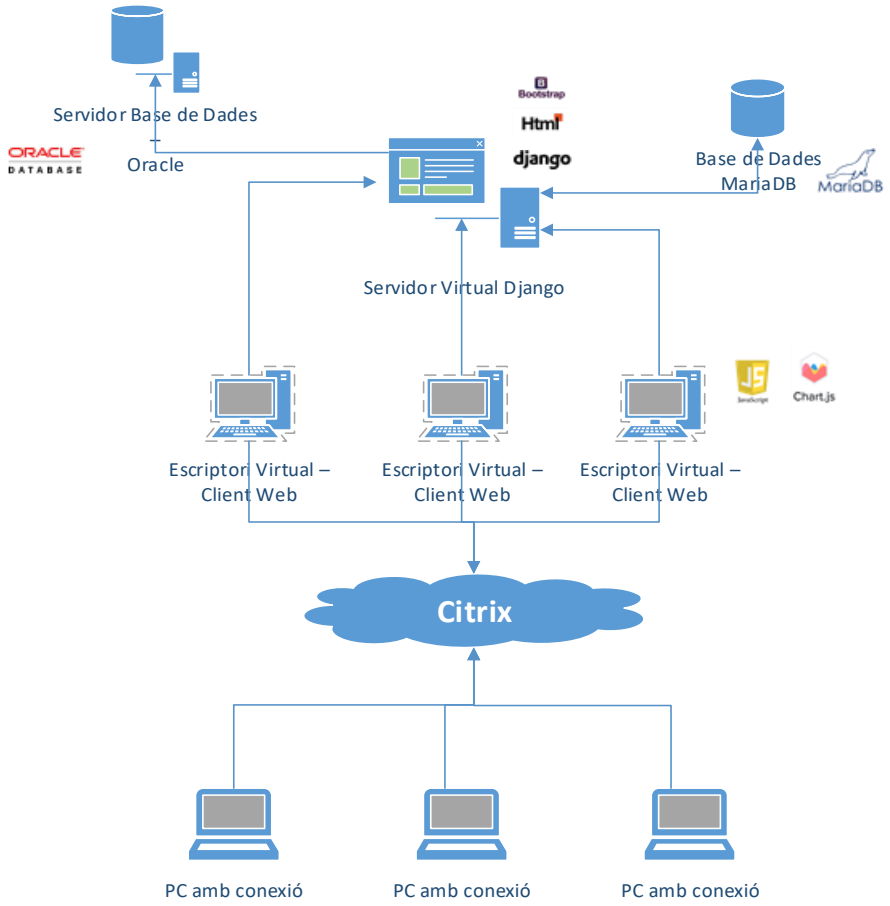


Figura 6. Definicio de l'arquitectura

4.1.2.2. Django

La tecnologia escollida per desenvolupar aquest projecte ha estat Django. [1]

Ofereix la posiblitat de connexió a base de dades, tant Postgres[4], SQLite3[2] o MariaDB[3].

A través de la API pot proporcionar una abstracció de la base de dades implementada, que permet crear i fer modificacions i eliminar objectes, també anomenats models.

Integrat en el framework també aporta un servidor que es capaç de suportar al carrega de les peticions de l'aplicació.

4.1.2.3. MariaDB

MariaDB es un sistema de gestió de bases de dades de programari lliure, basat en Mysql.

S'ha escollit MariaDB, ja que es una solució recomanada per django, i ofereix una integració molt robusta.

4.1.2.4. HTML + Bootstrap

Per al front-end s'ha utilitzat Django, i HTML, i per als estils Bootrstap 3.

Bootrstrap, també és un framework de desenvolupament CSS, que aporta una gran usabilitat a l'hora de dissenyar la interfície de la web, gràcies al seu sistema GRID de 12 columnes, que permet distribuir els elements del web amb facilitat i elegància.

A més aporta un gran nombre d'elements com Panells, barres de navegació, taules o diferents tipus de botó per a cada situació del web.

4.1.2.5. Javascript + Chart.js

Per a poder realitzar les gràfiques a temps real, s'ha utilitzat una solució en javascript, anomenada Chart.js. Aquesta solució ens permetrà realitzar un gran nombre de gràfics diferents, amb els que es pot interactuar i presentar un gran volum de dades sense problemes de rendimen

4.2. Disseny de la base de dades

La base de dades recull els camps necessaris per a que tíquet, la taula principal, sobre la que es centrarà el projecte, com es pot veure en la figura 7.

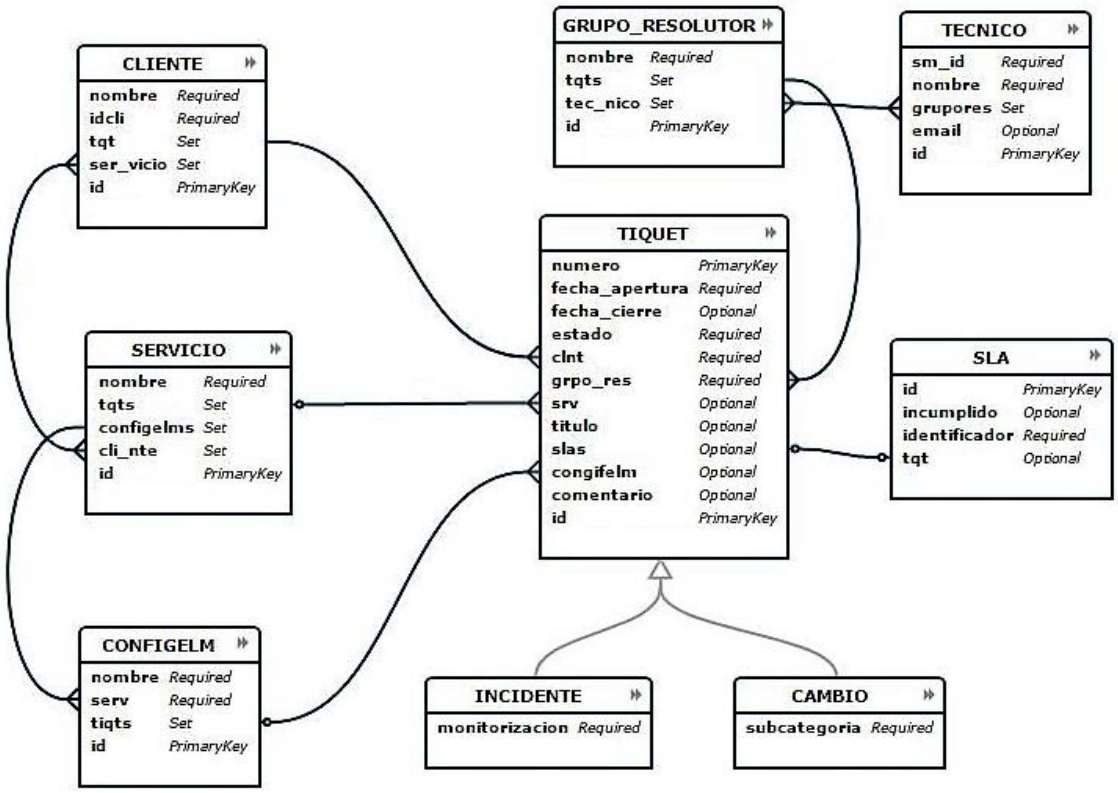


Figura 7. Diagrama E-R de la Base de Dades – MariaDB

Consta de 6 taules, relacionades totes entre elles i amb tiquet.

- **Cliente** – Taula que conté la informació bàsica de cada client.
- **Servicio** -- Conté les dades de les tecnologies que té contractactades un client.
- **Configelm** – Aquesta taula guarda la informació de la infraestructura de cada client. És a dir, guarda cada servidor o element que estigui dins el contracte. Esta directament relacionat amb els serveis, ja que un element de configuració pertany coma mínim a un servei del client. Per exemple el sistema operatiu windows de un element de configuració, pertanyeria al servei Microsoft. Però el mateix servidor, pot allotjar un aplicació de ERP, com pot ser Navision, en aquest cas també estaria relacionat amb el servei ERP.
- **Grupo_resolutor** – Es tracta dels diferents grups de tècnic que hi ha al departament. No només estan dividits per nivells, sinó que també per especialitzacions. Així que un mateix tècnic que tingui coneixement de Linux i Virtualització d'entorns, podrà pertànyer perfectament als dos grups de resolució.
- **Tecnico** – Conté la informació bàsica dels tècnics de l'empresa.
- **SLA** – Aquesta taula fa referencia als acords de servei, que es signen amb client. Permetrà controlar quan un tiquet ha superat el temps de resolució establert en el contracte.
- **Tíquet** – La taula més important de tota la base de dades, recull l'estat en que es troba un tiquet en tot moment, i permet afegir comentaris, per tenir-ne mes informació.
    - Dins d'aquesta taula, es defineix la informació més rellevant a l'hora de realitzar consultes. La taula tiquet, recull l'estat, la prioritat, el client i l'assignatari, entre d'altres.
    - S'ha definit un camp booleà, que defineix, si el tiquet ha estat obert per el sistema te monitorització o no
    - Un altre camp útil, és data d'apertura. Ens permetrà calcular quants dies fa que s'ha obert i calcular quant temps s'ha tardat en resoldre, juntament amb el camp data de tancament.
    - Per últim, l'atribut comentario, que ens mostrarà la informació que actualitzin tant tècnics, com gestors.

4.3. Disseny del modelat de Django

4.3.1. Detall de l'arquitectura dels models de Django.

En el diagrama es veuen representades, les entitats o models (com Django els anomena) que s'utilitzaran per linkar les views amb la base de dades, com mostra la figura 9.

Aquests models agafen el nom de les entitats de la base de dades, es creen relacions entre els models (objectes python) que es tractaran desde les views.

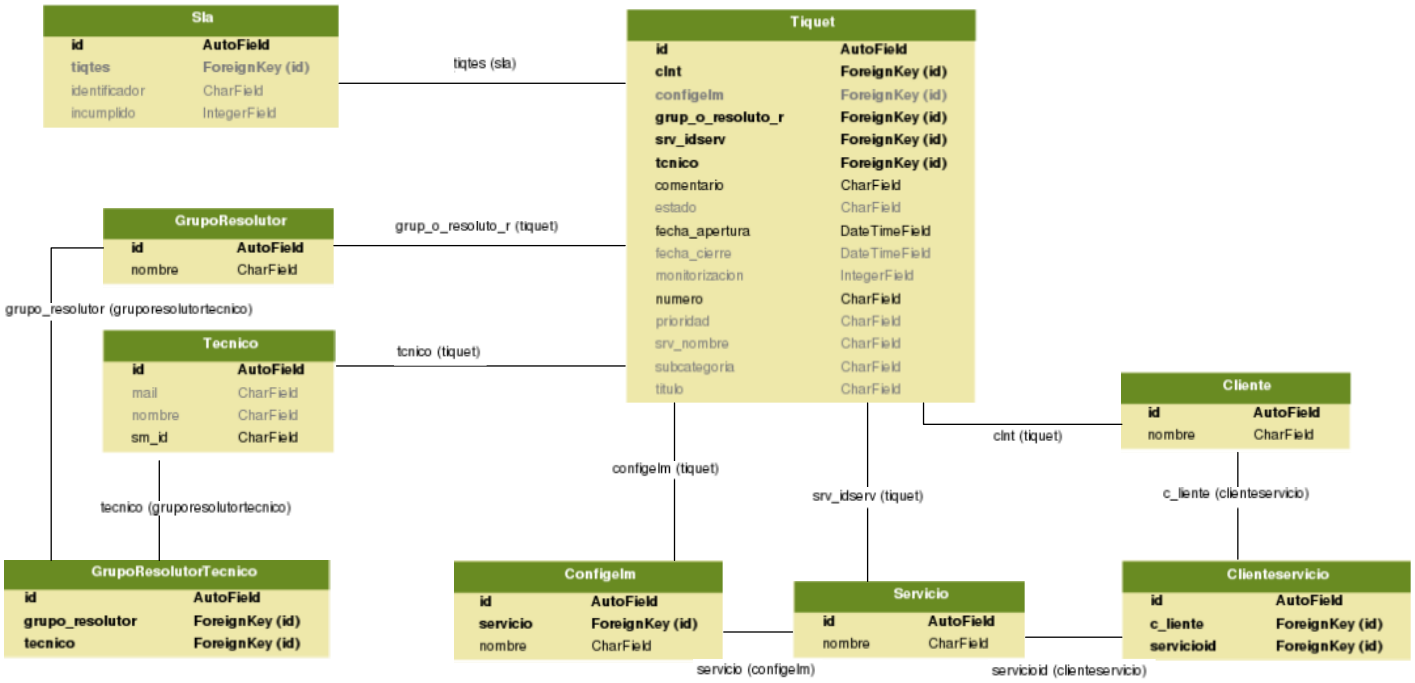


Figura 8. Disseny del modelat de l'apliació Django

A la figura 8, es mostra la representació dels models que Django crea en el moment de utilitzar la comanda "python manage.py InspectDB".

Aquesta comanda, utilitza la base de dades que hem representat a la figura 7, i la transforma a objectes python que seran els utilitzats per l'aplicació com a enllaça entre les views de l'aplicació (back-end) i la base de dades, com es mostra a la figura 9.

Com s'observa, a la figura 9, la view tracta les dades que s'han de mostrar, i els models son qui s'encarrega de connectar amb la base de dades, en el cas del projecte, MariaDB.

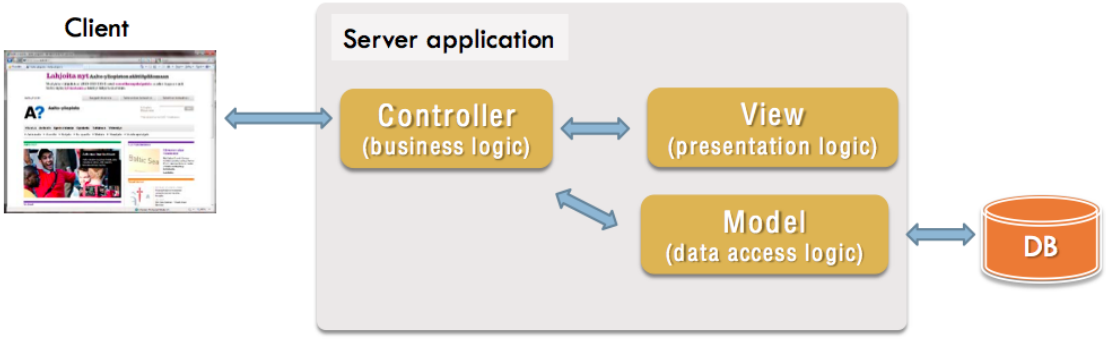


Figura 9. Diagarama acces a Base de Dades.

4.3.2. Detall del modelat de gestió d'usuaris

Al mateix temps que la comanda crea els models de la Base de Dades, crea les taules necessàries per a realitzar la gestió des de el backend d'administració. Serà necessari, per això, llançar dues comandes, "python manage.py makemigrations", que sincronitza la base de dades amb els models creats, i "python manage.py migrate", que aplica les configuracions que hem afegit als models, per exemple mètodes per mostrar atributs, o funcions que fan càlculs automàtics per tal d'evitar fer-los a les views. Es necessari executar aquestes comandes, cada vegada que apliquem una modificació al fitxer models.py ja que s'està modificant la configuració dels objectes.

Aquestes comandes, també creen automàticament, noves entitats, com es pot apreciar la figura 10, les taules que es generen fan les funcions de Gestió d'Usuaris, inclouen servei d'autentificació, gestió de permisos, i control de versions. Aquest control de versions es basa en la comanda i "python manage.py migrate". Cada vegada que es llança aquesta comanda, si hi ha hagut canvis al model, es crea un nou registre, tenint d'aquesta manera un historic de migracions, o el que es el mateix un petit control de canvis sobre el projecte.

La entitat DjangoSessin guarda informació de les sessions dels usuaris connectats, per tant agilitza les restriccions de continguts segons usuaris.

Taules utilitzades per Django, per poder gestionar usuaris, permisos i sessions.

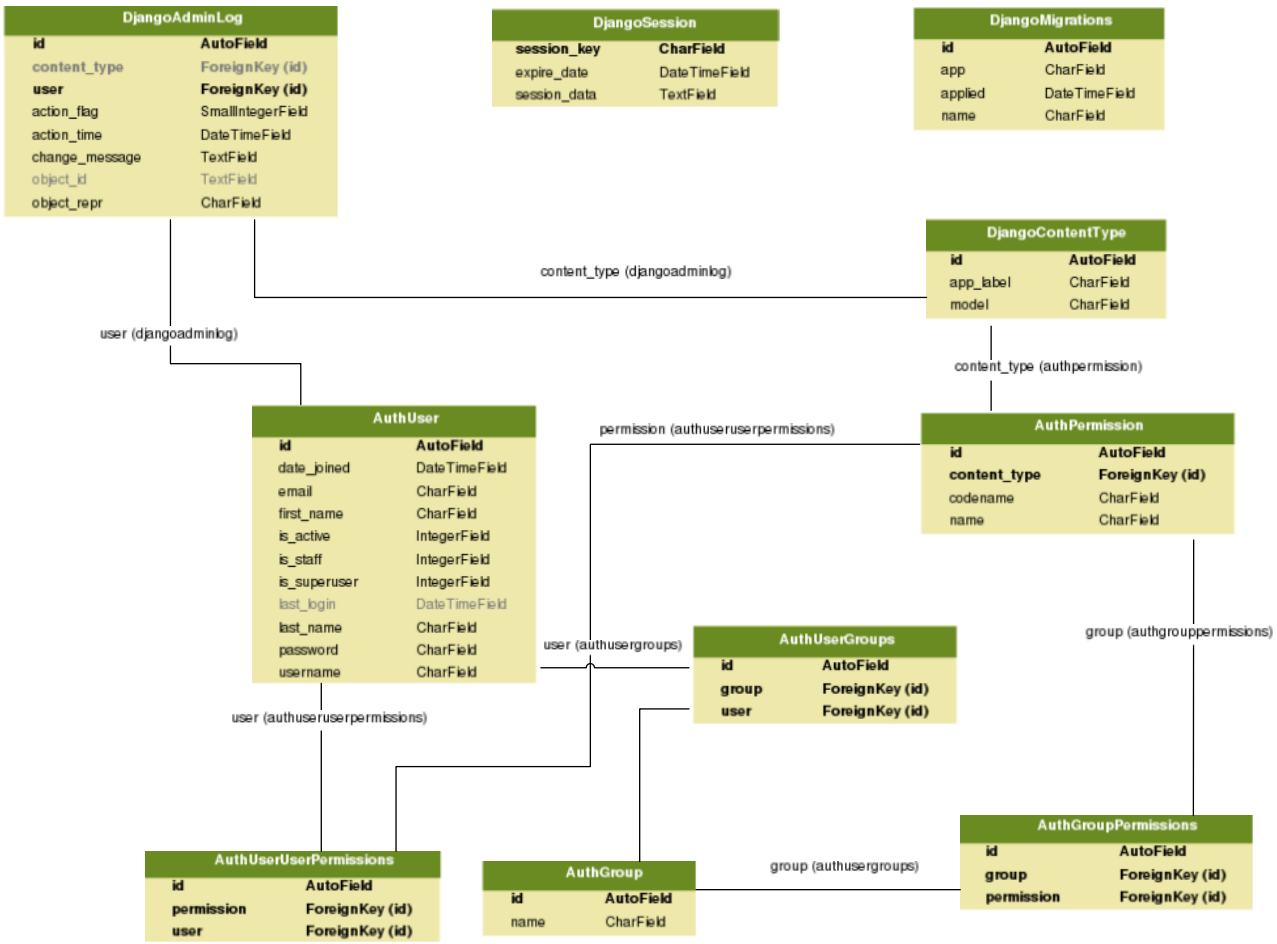


Figura 10. Modelat de Django – Gestió d'usuari

4.4. Funcionament d'un event del sistema

Per a cada cas d'us identificat en les figures 3 i 4, es produeix un event de sistema des del front-end dels clients.

A continuació es detalla i s'explica, un event de sistema genèric, a l'arquitectura proposada.

El funcionament del framework Django, és bastant similar al MVC (model vista controlador), ja que, Django utilitza el MVT (Model-View-Template), veure figura 11 on:

Un usuari fa una petició, és a dir un request, a una URL de l'aplicació web que recollirà un webserver i la passara al URL Mapping. Un cop s'URL mapping l'hagi tractat l'enviarà a la View corresponent. Aquesta s'encarrega de preparar les dades que s'hauran de visualitzar, de manera que a través del connector dels models, pot fer consultes a la base de dades. Cal especificar, que no sempre serà necessari que la vista faci alguna consulta contra la base de dades, per exemple, si la pàgina web és estàtica i nomes es retorna contingu HTML. Un cop la view, ha processat totes les dades que s'han requerit, les passarà a la template, que serà el que finalment l'usuari que ha fet el request vegi per pantalla.

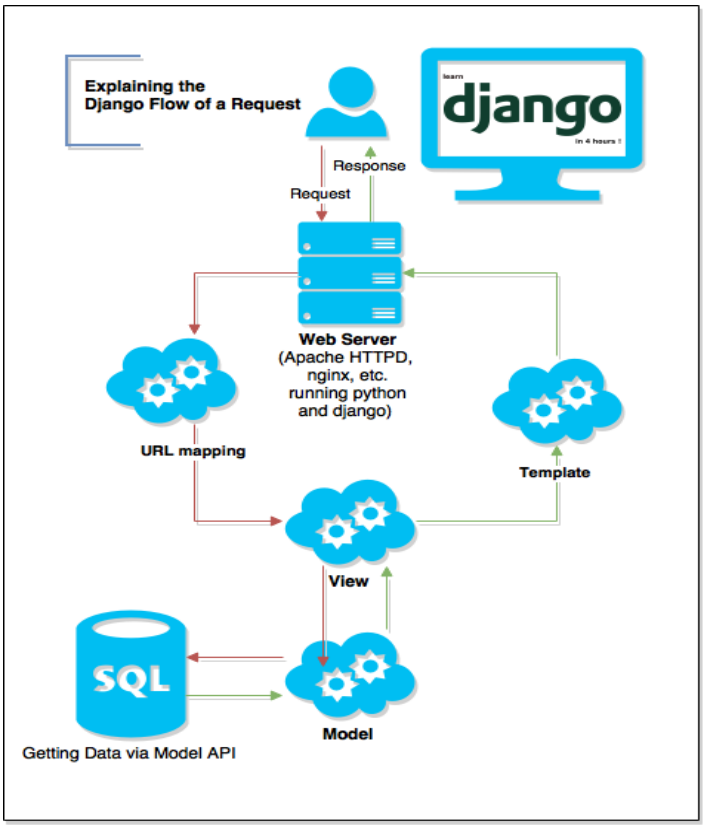


Figura 11. Gestió d'una petició

Entrant en detall, seguint la Figura 11, es realitza el següent procés quan tenim un event al dashboard.

Petició al Dashboard:

L'usuari fa click al link Dashboard.

A partir d'aquí el l'URLDispatcher, recull la petició, i busca de forma seqüencial, dins el fitxer urls.py. En aquest cas, quan trobés la següent línia de codi:

```
url(r'^dashboard', dashboardView.dashboard, name='dashboard'),
```

Es redirigirà el request a la vista (view) anomenada dashboardView, concretament a la funció dashboard.

Un cop el dashboard, reculli la request processarà les dades, i farà les consultes que tingui implementades a la base de dades.Un cop obtingudes els dades, es guarden en llistes o diccionaris que es passaran a la template amb la instrucció:

return render_to_response('dashboard/dashboard.html', context_dict, context)

enviarà la un diccionari de context (contex_dict), variable que guarda tota la informació que s'envia a la template.

Un cop la template ho rep, renderitzarà les dades i les mostrarà al usuari.

4.5. Vistes i Templates

4.5.1. Funcionament de les views

Una view es la part que recull tota la lògica per a poder retornar una resposta. Es poden generar diferents fitxers de vista, per tal d'organitzar el codi correctament. L'únic requisit per a que no hi hagi errors, és que a la definició de urls, les rutes i noms de funcions siguin correctes.

La vista sol·licitarà les dades al model, que li retornarà un diccionari amb les consultes realitzades.

```
@login_required(login_url='/login/')
def dashboard(request):
    # Obtain the context from the HTTP request.
    context = RequestContext(request)
    usuarioActivo = request.user
    tiquets = utils.filterTiquetsByRole(usuarioActivo)
    tiquetsCliente = Tiquet.objects.annotate(numtiquclnt = Count('clnt')).order_by('clnt')
    tiquetsCliente=  tiquetsCliente.filter(Q(fecha_apertura__year="2017"))
    tiquetsCliente = tiquetsCliente.exclude(Q(estado = 'Cierre') | Q(estado = 'Closed') | Q(estado = 'Revision') | Q(estado ='Resolved'))
```

Figura 12. Exemple de la definició d'una view

Una view pot integrar entre altres components, els decorators.

Aquest components permeten restringir l'accés a la vista, per exemple si l'usuari no esta logat.

En aquests casos, si l'usuari no te permís per accedir a la vista, el decorator li fa una redirecció automàtica cap a la URL que s'indiqui en el mètode.

4.5.2. Funcionament de les templates

Una template per Django, és un fitxer HTML, amb incrustacions de python, gràcies a les template tags.

Els tags de template, no son res més que instruccions que Django interpreta, i que permeten realitzar les següents operacions:

Hi ha 4 tipus de tags.

Mostrar variables {{ variable }}

Una variable, sempre es mostrarà entre "{{ }}". La template la substituirà per l'element del diccionari que tingui el mateix nom i existeixi.

```
<tr >
<th data-title="Cliente"><a href="/ticketing/tiquetsList?idCliente={{clientes}}">{{clientes}}</a></th>
<td data-title="Totales"><a href="/ticketing/tiquetsList?idCliente={{clientes}}">{{values.todo}}</a></td>
<td data-title="Incidentes"><a href="/ticketing/tiquetsList?idCliente={{clientes}}">{{values.incidentes}}</a></td>
<td data-title="Cambios"><a href="/ticketing/tiquetsListCM?idCliente={{clientes}}">{{values.cambios}}</a> </td>
</tr>
```

Figura 13. Exemple de part d'un template

Filtres – {{ variable|filtre}}

Per tal de poder modificar les variables, Django ens ofereix la estructura de filtre.

El tag if – {% if %}

Ens dona la posiblitat d'utilitzar la sentencia if dins la template.
També poden ser niuades, utilitzant {% elif %} i {% else%}.
Per finalitzar el tag if, es necessari acabar-lo amb: {%endif%}

El tag for – {% for %}

Aquesta comanda, permet recórrer els diccionaris o llistes que trobem en el context.

Per finalitzar el for, també serà necessari una comanda de tancament {%endfor%}

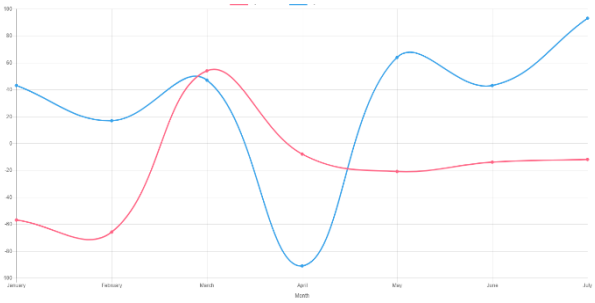
El tag block -- {%block%}

Amb aquest tag es té la possibilitat de fer herències de templates. Així podem construir una template amb, per exemple una barra lateral, fer que sempre sigui visible, sense necessitat de posar tot el codi en cada template.

4.6. Informació Visual

A l'aplicació es volen mostrar diferents gràfics, per tal de poder mostrar les dades, per a que siguin el més entenedores possible. Dins de les opcions que l'eina de javascript, chart.js es disposa dels següents tipus de gràfiques:

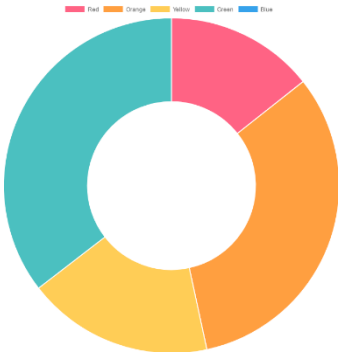
Gràfics de linies



El gràfic lineal podria ser una molt bona opció a tenir en compte, ja que dona la sensació d'evolució, a l'hora de representar les dades. Però en aquest cas, a l'hora de tendir a zero, no es pot truncar, i les dades queden mal representades .

Figura 14. Exemple gràfic lineal

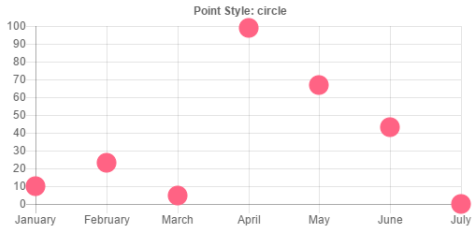
Gràfics en forma de corona



Aquest tipus de gràfic, pot ser útil si, els camps a mostrar son pocs. En el cas de l'aplicació, no seria efectiu, ja que les dades amb els valors mes baixos, no quedarien representades amb claredat.

Figura 15. Exemple gràfic corona

Gràfiques de punts

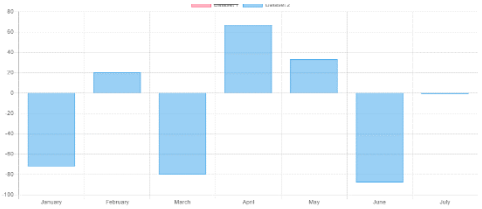


Mostra tota la informació com es desitja. Un cop implementat, es va detectar, que si la pantalla era massa petita, no s'aconseguia distingir la llegenda.

Per a representar les dades mensuals, podria ser útil, però no admet més d'un punt a la mateixa dada en l'eix de les X.

Figura 16. Exemple gràfic de punts

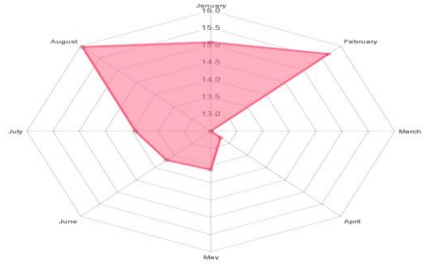
Gràfica de tipus Barra



Aquest tipus de gràfic, mostra de forma clara, la informació que es vol representar. A més te la opció de tooltip. Això farà que cada vegada que es passi el cursor per sobre d'una barra, es mostrarà un pop-up amb la informació detallada.

Figura 17. Exemple gràfic de barra

Gràfica tipus Radar



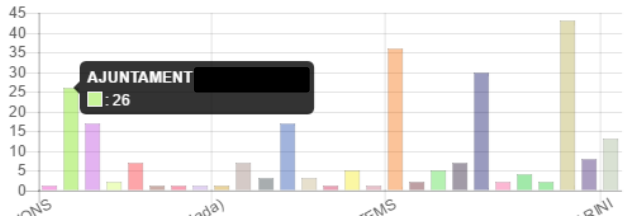
Aquest gràfic mostra per una sèrie limitada de items, el resultat, en una àrea. Fa que els diferents items siguin fàcilment comparables. Si la llista es massa gran, es perd certa visibilitat de les dades.

Figura 18. Exemple gràfic de radar

S'han estudiat les gràfiques ja que podien donar la informació més clarament. Chart.js ofereix moltes més opcions, però no totes eren útils per representar la informació tractada en els tiquets, com per exemple la gràfica logarítmica, que va quedar descartada, ja que no representa una informació veraç.

Els diferents tipus de disseny de gràfiques escollits per a representar les dades desitjades, s'exposen a continuació:

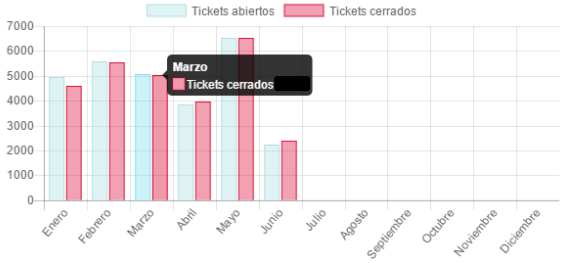
Gràfica de barres



- Les gràfiques més útils per a l'aplicació son les de tipus "barra" , ja que permeten fer una molt bona diferenciació entre clients.

Figura 19. Exemple gràfic de barres

Gràfica de barres doble



El gràfic de barra, admet per a cada dada de l'eix X, més d'una dada al Y. D'aquesta manera es poden representar dades, amb la finalitat de obtenir una comparació fàcil.

Figura 20. Exemple gràfic de doble barra

Aquest tipus de gràfic ens dona l'opció d'ocultar una de les sèries de dades que introduïm, per tal de veure el resultat amb més claredat.

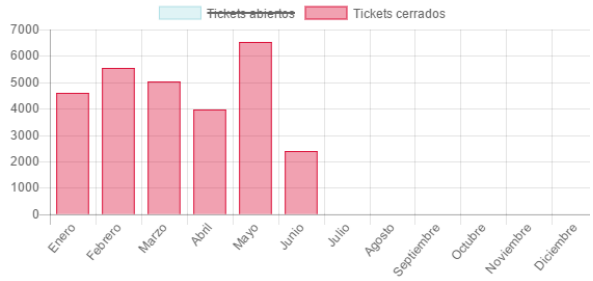
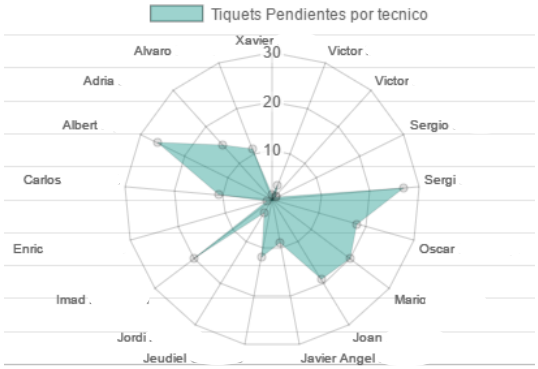


Figura 21. Exemple gràfic de doble barra – filtrat

Gràfic tipus Radar



Amb el gràfic de radar, s'obté una comparativa clara i amb, només un sol cop d'ull veure, en aquest cas quins son els tècnics amb més càrrega.

Aquest tipus de gràfic, a més, també ens mostrarà la informació detallada, quan el cursor sigui sobre la dada desitjada.

Figura 22 Exemple gràfic de radar

4.6.1. Informació complementaria al gràfics

A més per a cada gràfica, de l'apartat dashboard, es decideix adjuntar la taula amb les dades, per a tenir la informació detallada.

Així quan una de les dades sigui alarmant, podrem visualment adonar-nos-en. A la figura 21, es pot veurà la taula de tiquets sense resoldre desglossat per client.

Si el nombre de tiquets supera els 15, es modificarà el color de la fila afectada, de color groc. Si els tiquets acumulats superen els 25 el color que adopta la taula serà el vermell.

Mes	Totales	Incidentes	Cambios
	1	1	0
	26	16	10
	17	1	16
	2	1	1
	7	4	3

Figura 23. Taula alertes

4.7. Sincronització amb la BBDD Oracle

S'han creat diversos scripts es connecten al port 1521 de la base de dades, que és el port estàndard de comunicació de d'Oracle, i realitzen diferents query's. Cadascuna d'aquestes consultes, generen un arxiu XML, que es guarda dins la carpeta /../middleare/xml-data. Un cop recollides les dades, un altre script, UpdateDB.py, s'encarrega de llegir l'arxiu XML que prèviament s'ha creat, el

parseja, n'extreu les dades i amb la comanda update_or_create(), de Django, omple tota la informació sobre Tecnics, Clients i Equips i les seves relacions, que es necessiten per a la identificació de tiquets.

Un cop poblada la base de dades, procedim a executar el updateTickets.py. Aquest script només poblarà la taula Tiquet, de la base de dades.

Aquest scipt només es llançarà la primera vegada que poblem la base de dades, ja que recull tota la informació de tiquets des de començament d'any, juntament amb els que encara hi ha oberts. Degut a que conté tanta informació, a mode d'històric, la carrega de dades pot teniu una duració de fins a 20 minuts.

Un cop tenim tota la informació a la base de dades de l'aplicació, s'ha creat un script per tal de mantenir-la actualitzada automàticament. Aquest script, no té en compte els tiquets tancats, ja que un cop tancats el seu estat no varia. Amb això aconseguim que el temp de càrrega es redueixi fins a 1 minut. Per tal de que sigui el totalment automàtic es programa, dins el crontab de linux, que la tasca s'executi cada 10 minuts. Temps suficient per a tenir la visió actualitzada que es requereix del departament, i per a que no hi hagi problemes ni de rendiment, ni de saturació

5. Resultats i Simulacions

El resultat de l'aplicació proporciona una visió molt fidel, de l'estat del departament.

Es buscava poder saber l'estat de les cues del tècnics per a poder saber la seva càrrega de treball. L'aplicació, es una web privada, en la que no podem consultar dades, si l'usuari no està logat. Si un usuari intenta connectar-se a qualsevol pàgina de l'aplicació, el retornarà a la pàgina de login.

Seguint els objectius marcats amb els casos d'us de les figures 3 i 4, estudiarem si el projecte ha arribat al objectiu proposat.

5.1. Usuari tècnic

Segons els casos d'ús proposats per a l'usuari tècnic, s'han creat diferents pàgines.

La informació que es mostra en aquestes pàgines, sempre estarà filtrada per l'usuari tècnic que estigui conectat a l'apliació.

D'aquesta manera, un usuari tècnic mai podrà veure informació de cap altre tècnic, tampoc editar ni modificar cap tiquet, que no li hagi estat assignat.

Cas d'ús 1 – UC1. Fer Login

Com es mostra a la figura 22, s'ha creat una pantalla de login, en la que l'usuari ha de logar-se, si vol ser capaç de entrar a l'aplicació.

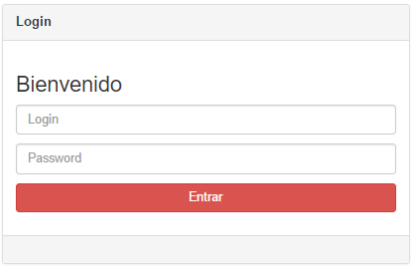


Figura 24. Pantalla de Login

Es requereixen el nom d'usuari i la password per poder entrar. La web no contempla domini públic ja que conté informació privada de l'empresa i tota persona aliena a aquesta, no hi pot tenir accés, per tant tampoc es contempla la opció de registre.

Cas d'ús 2 – UC2. Logout

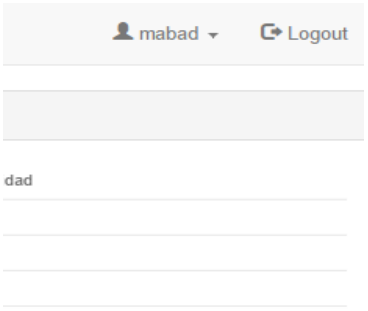


Figura 26. Botó Logout

Un cop l'usuari s'ha logat. Per tal de sortir de l'aplicació, només haurà de fer un clic a la part superior dreta, on trobarà el botó de Logout.

Aquest boto el portarà directament a la pantalla de Login de nou. Tancant-li la sessió.

Cas d'ús 3 – UC3. Modificar Password

Figura 26. Modificar password

L'usuari pot modificar la password entrant al seu perfil. Aquí se li mostra un formulari que ha de complimentar seguint les restriccions que la password ha de complir. Si la password no s'adequa a aquestes restriccions l'usuari rebrà un missatge, indicant-li el motiu perquè no s'ha actualitzat la seva password.

En cas que l'usuari modifiqui la password correctament, se li mostrarà un missatge d'actualització.

Cas d'ús 4 – UC4. Consultar llistats de tiquets

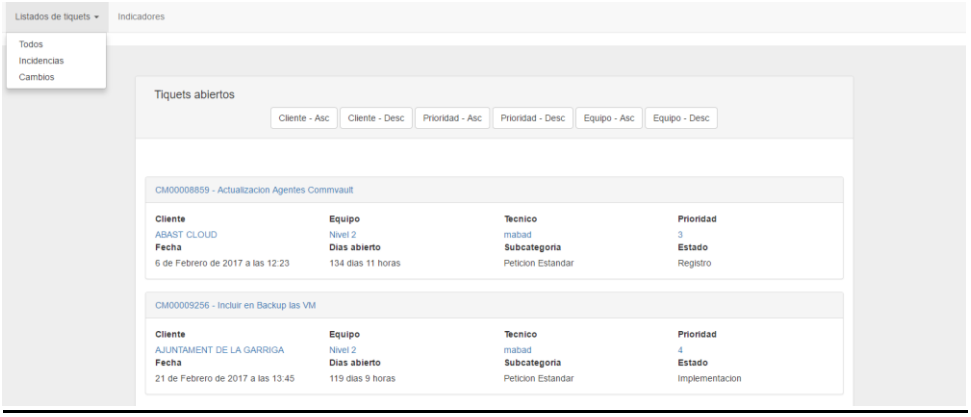


Figura 27. Consultar llistats de tiquets

Com es pot observar, s'han creat diferents llistats de tiquets que l'usuari pot consultar.

Aquests llistats es divideixen en Incidents, Canvis i una conjunció dels dos. Tots els llistats, es poden filtrar per l'equip, la companyia a la que pertanyin o per prioritat.

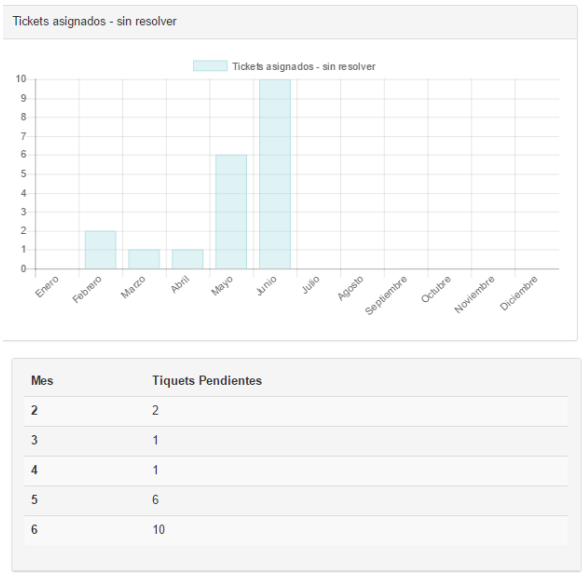
Es mostra la informació detalla del tiquet i el temps que fa que s'ha creat.

Si el tiquet tingués un comentari també es mostraria a la pàgina. Aquesta vista, està filtrada per el tècnic que està consultant-la en el cas de que l'usuari sigui de tipus Tècnic.

Cas d'ús 5 – UC5. Consultar dades Dashboard

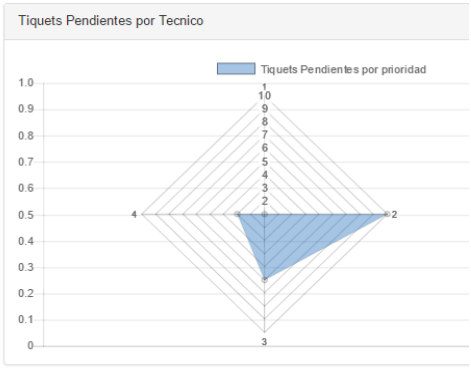
Per a tenir la visió global de l'estat del tècnic que es desitja, l'usuari un cop ha fet el login, és redirigit, cap al dashboard. En aquesta pàgina, se li mostren 3 gràfiques, juntament amb tres gràfiques amb les que pot consultar la cua de tiquets que té

assignat i el mes en que es van obrir, a quin client pertanyen els tiquets, i amb quina prioritat se li han assignat.



Mes	Tiquets Pendientes
2	2
3	1
4	1
5	6
6	10

Figura 28. Tiquets oberts assignats per mes



Prioridad	Tiquets Pendientes
1	1
2	10
3	6
4	3

Figura 29. Tiquets oberts assignats per prioritat

Cas d'ús 6 – UC6. Consultar dades d'indicadors

L'usuari te la opció de coneixer el seu rendiment, en la pàgina d'indicadors.
En aquesta pàgina es mostra un detall dels tiquets que ha tancat fins al moment. Així com la mitjana en hores, de resolució per tiquet que emplea a cada client. També podrà veure a quin client li està dedicant més esforços, o a quin hauría de controlar, per tal de que no tingui un volum d'incidents elevats.
Per poder aprofundir encara més en el detall dels tiquets, el tècnic pot saber, del total assignats, quants pertanyen a tiquets provinents del sistema de monitorització i quins son incidències que client ha reportat, com es pot veure a la figura 29

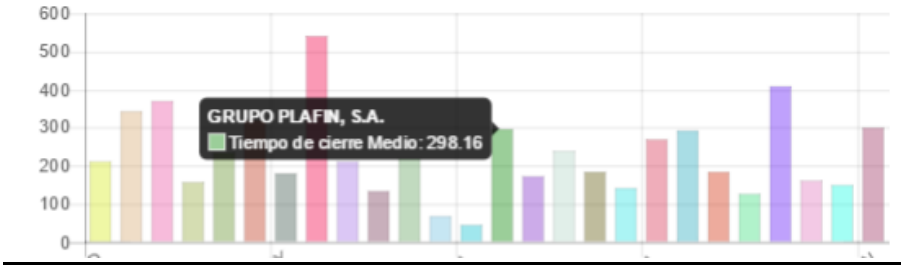


Figura 30 . Mitjana d'hores empleades en la resolució

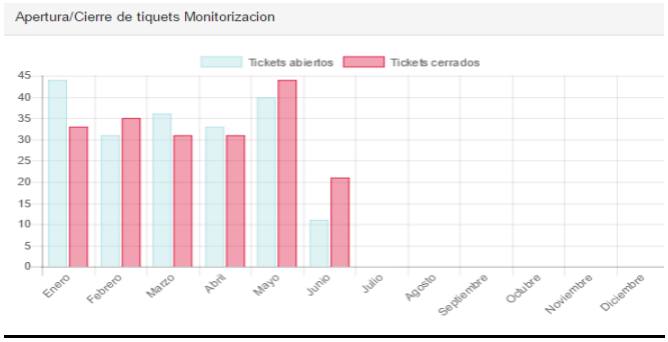


Figura 31 Tiquets oberts/tancats de monitorització assignats per mes

Cas d'ús 7 – UC7. Actualitzar / Editar comentari de tiquet

Com hem vist a la figura 25, des de els llistats de tiquets, podem accedir a un tíquet en concret. Des d'aquesta vista, podrem actualitzar la informació d'un comentari, com es mostra a la figura 30.

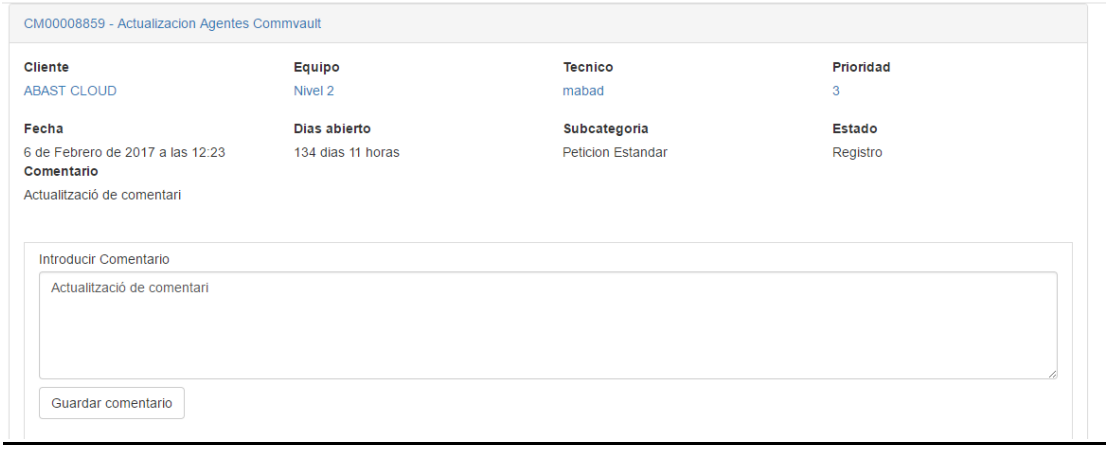


Figura 32. Actualització de comentari

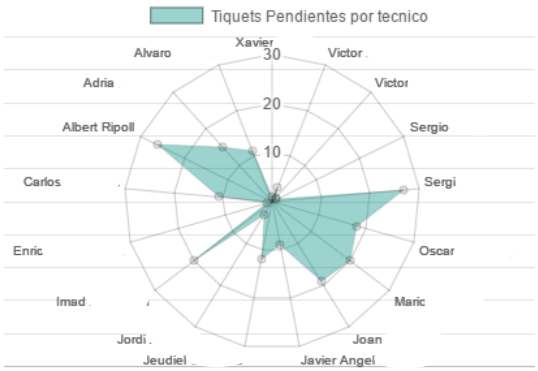
En aquesta vista, si l'usuari modifica la URL per entrar a un tíquet que no li ha estat assignat, rebrà un missatge especificant que no té permisos per accedir al tiquet, introduït.

Si el tiquet no existeix, la informació que es mostrarà serà un panell informatiu especificant que no existeix.

5.2. Usuari Administrador

Al contrari que amb l'usuari tècnic, l'administrador té accés a la informació de tots els tiquets de la base de dades.

Per això els casos d'ús UC.4, UC.5, UC.6, i UC.7, seran el mateixos però sense filtratge d'informació.



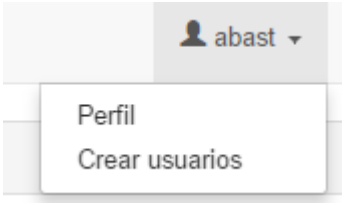
L'única diferencia que trobarem entre el tècnic i l'usuari en aquests casos d'us, és que a l'usuari administrador, se li ha afegir un gràfic en el Dashboard, on pot consular les cues de tiquets assignats als tècnics, per així portar el control del departament. Com es mostra en la figura 31.

En aquest apartat, també es mostra una taula amb la especificació de cada tècnic i el nombre de tiquets.

Figura 33. Cúes per tècnic

Els casos d'ús UC.1, UC.2, UC.3, tenen el mateix comportament per tots els usuaris, ja que es tracta de logar-se i desconnectar-se.

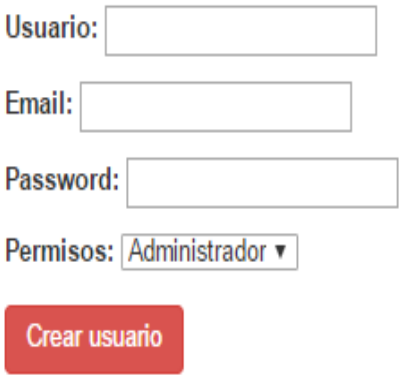
Cas d'ús 8 – UC. 8 Afegir usuaris tècnics o Administradors.



Com es mostra en la figura 32, l'usuari Administrador, té una opció més que el tècnic en el desplegable d'usuari.

Amb aquesta opció obté accés a donar d'alta usuaris en el sistema, fent click, a Crear Usuarios.

Figura 32. Opcions Administrador

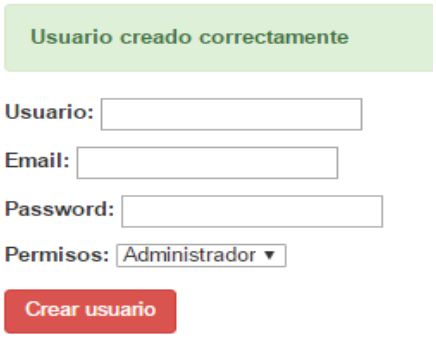


Un cop ha clicat, es mostra un formulari com el de la figura 33. Serà necessari el nom d'usuari, mail, password i saber si es tracta d'un usuari Administrador o Tècnic.

Si es tracta d'un tècnic, el seu nom d'usuari, ha de ser igual que el nom d'usuari de la Base de dades Oracle.

Si en el moment del registre, hi ha algun camp que no es correcte, es mostrarà un missatge d'error, especificant, quin camp no es correcte. Per exemple, *"Introduzca una dirección de correo electrónico válida"*.

Figura 34. Formulari afegir usuari



Quan l'alta d'usuari, es realitza correctament, l'aplicació mostra un missatge de creació d'usuari correcte.

Figura 35. Missiatge de confrimació.

5.3. Conclusió

El projecte ha assolit tots els casos d'ús proposats, aconseguint una eina que proporcionarà una visió del departament que fins ara no es podia tenir.

Fa que la comunicació entre gestor i tècnic sigui senzilla i ràpida, sense necessitat de ser oral. L'aplicació aporta una nova forma de treballar, en la que no es necessària la comunicació verbal. Aquesta nova forma de treballar, aporta als tècnics una major concentració a l'hora de estar resolent tiquets d'alta complexitat.

Ha repercutit també en l'ambient del departament, fent que el silenci sigui una eina més a l'hora de treballar.

Per la part dels gestors, tenen el control total sobre el departament, obtenen gràfiques de rendiment i poden fer avaluacions dels tècnics. Tenir constància per escrit de l'estat d'un tiquet, fa que tots els gestors puguin gestionar l'equip sense necessitat d'haver estat prèviament informats. D'aquesta manera son capaços de prioritzar tasques als tècnics i fer el departament més eficient.

Tant gestor com tècnic tenen dades fiables del propi rendiment, en el cas del tècnic, i del rendiment de l'equip en el cas del gestor. Es fa més fàcil la planificació diària d'un tècnic i a l'hora la priorització de tiquets per a la seva resolució.

Les gràfiques d'indicadors han aportat, una visió diferent dels clients, ja que ha quedat pal·les que s'estava aportant molt d'esforç a clients que no en requerien tant.

S'ha aconseguit que es balancegi la càrrega que tenen els tècnics ja que a l'inici del projecte, hi havia cues molt desiguals. Ara, els tècnics tenen una carrega de treball similar, depenent del nivell de resolució en el que es trobin.

Gracies a les gràfiques de tiquets assignats, s'han pogut reduir les cues d'incidents, d'aquesta manera, els tècnics especialistes, poden dedicar part del seu temps a realitzar millores en els entorns del client. Aquestes millores han comportat una millor satisfacció del client de cara al departament.

De cara a millorar la aplicació s'ha estudiat la forma de que pugui connectar bi-direccionalment amb l'eina de tiqueting del departament. D'aquesta manera, si un tècnic actualitza el comentar d'un tiquet a l'aplicació també es pugui veure a l'eina.

Una altre millora que s'ha proposat és afegir una eina de planificació automàtica, que el gestor controli i pugui modificar, si s'escau. Aquesta eina, hauria de construir un horari de tasques, segons la prioritat, el client i el temps mitja de resolució del tècnic.

6. Referències bibliogràfiques

[1] .- Tango with Django 1.9 - Leif Azzopardi and David Maxwell

[2] - Tango with Django 1.9 - Leif Azzopardi and David Maxwell – pàg. 45

[3] - https://medium.com/code-zen/django-mariadb-85cc9daeeef8

[4] - https://www.digitalocean.com/community/tutorials/how-to-use-postgresql-with-your-django-application-on-ubuntu-14-04

[5] - Lanzamiento de un sitio web con WordPress - Peter Fischer

[6] - Learning Qlik Sense: The Official Guide - Christopher Ilacqua, Henric Cronstrom, James Richardson

Apèndix A: Manual tècnic

A.1 Instal·lació: Requeriments mínims

Per a la instal·lació del projecte, es necessari una màquina Linux, amb un mínim de 2 GB de memòria i 20 GB d'espai lliure al disc.

Serà necessària la connexió a la base de dades d'origen.

A.2. Manual del desenvolupador

Entorn Virtual

Es crea un entorn virtual, que permet diferents configuracions, a diferents ubicacions del sistema. Son útils per a poder provar, per exemple varies versions de python o diferents versions de Django, sense que hi hagi conflicte entre elles, ja que l'entorn virtual, es un entorn tancat.

Aquest entorn es crea sobre el directori especificat, i guarda les característiques de les instal·lacions dins del directori i sub-directoris.

Es crea amb la comanda:

```
virtualenv "Nom_del_projecte"
```

Aquesta comanda crearà una estructura amb el següent aspecte:

```
Nom_del_projecte/
bin/
include/
lib/
```

i s'activa amb la comanda següent, un cop creat.

```
Cd Nom_del_projecte
source bin/activate
```

Sabrem que l'entorn està activat ja que la línia de comandes variarà, adoptant la següent formula:

```
(Nom_del_projecte)$
```

Per tornar a l'estat normal, nomes caldrà entrar la comanda :

```
(Nom_del_projecte)$ deactivate
```

Organització Estructural del projecte

A continuació es detalla la organització de directoris del projecte.

```
entornosvirtualesbin/
   /ticketing
        /bin        -->  instalació de Venv
        /include          -->  instalació de Venv
        /lib      -->  instalació de Venv
        /local    -->  instalació de Venv
        /ticketing -->  directori del projecte Django
           Manage.py --> fitxer de gestió del projecte Django
           /ticketing  --> directori de fitxers de configuració
        /ticketingapp  --> directori de l'aplicació
```

Configuració dels scripts

Els scripts es connecten al port 1521 de la base de dades, que és el port estàndard de comunicació de d'Oracle, i realitzen diferents query's. Cadascuna d'aquestes consultes, generen un arxiu XML, que es guarda dins la carpeta /../middleare/xml-data. Un cop recollides les dades, un altre script, UpdateDB.py, s'encarrega de llegir l'arxiu XML que prèviament s'ha creat, el parseja, n'extreu les dades i amb la comanda update_or_create(), de Django, omple tota la informació sobre Tècnics, Clients i Equips i les seves relacions, que es necessiten per a la identificació de tiquets.

Un cop poblada la base de dades, procedim a executar el updateTickets.py. Aquest script només poblarà la taula Tiquet, de la base de dades.

Aquest scipt només es llançarà la primera vegada que poblem la base de dades, ja que recull tota la informació de tiquets des de començament d'any, juntament amb els que encara hi ha oberts. Degut a que conté tanta informació, a mode d'històric, la carrega de dades pot teniu una duració de fins a 20 minuts.

Un cop tenim tota la informació a la base de dades de l'aplicació, s'ha creat un script per tal de mantenir-la actualitzada automàticament. Aquest script, no té en compte els tiquets tancats, ja que un cop tancats el seu estat no varia. Amb això aconseguim que el temps de càrrega es redueixi fins a 1 minut. Per tal de que sigui el totalment automàtic es programa, dins el crontab de Linux, que la tasca s'executi cada 10 minuts. Temps suficient per a tenir la visió actualitzada que es requereix del departament, i per a que no hi hagi problemes ni de rendiment, ni de saturació.

Disseny del back-end

Dins el projecte Django, podrem trobar al menys dos directoris

Com s'ha vist a la figura anterior trobem el directori de fitxers de configuració i el de la pròpia aplicació.

```
/ticketing  --> directori de fitxers de configuració
        __init__.py
        roles.py
        settings.py
        urls.py
        wsgi.py
/ticketingapp  --> directori de l'aplicació
        /migrations
        /middleware
        /static
        /templates
        /views
        _init__.py
        Admin.py
        Apps.py
        Context_processors.py
        Forms.py
        Models.py
        Tests.py
        Urls.py
```

El directori ticketing, conté l'arxiu settings.py, és en aquest arxiu on especificarem totes les configuracions que ha de tenir el projecte. Des de quines aplicacions conté, fins a l'idioma o quin connector de base de dades ha d'utilitzar.

En el fitxer urls.py tan sols s'ha d'incloure el fitxer d'urls de l'aplicació. En el cas de que hi hagués més d'una aplicació dins el mateix projecte, s'inclourien tots.

En la part d'aplicació, dins el directori s'hi poden trobar diferents sub-directoris. Migrations, es un registre de modificacions que es fan a la base de dades, cada cop que es modifica el fitxer models.py.

Aquest fitxer conté tota la informació dels models de la aplicació. Es poden afegir mètodes, que faran més senzill el càlcul de dades a la aplicació.

Aquest fitxer es pot auto-generar, a partir de la comanda:

```
Python manage.py inspectdb
```

El que farà aquesta comanda, serà connectar a la base de dades que li haurem especificat al fitxer settings.py i crearà automàticament els models de dades que prèviament havíem carregat a la base de dades, en aquest cas MariaDB.

```python
class Tecnico(models.Model):
    sm_id = models.CharField(max_length=255)
    nombre = models.CharField(max_length=255, blank=True, null=True)
    mail = models.CharField(max_length=255, blank=True, null=True)

    def __unicode__(self):
        return self.sm_id

    class Meta:
        managed = True
        db_table = 'tecnico'


class Tiquet(models.Model):
    numero = models.CharField(max_length=255)
    fecha_apertura = models.DateTimeField()
    estado = models.CharField(max_length=255, blank=True, null=True)
    clnt = models.ForeignKey(Cliente, models.DO_NOTHING, db_column='clnt')
    tcnico = models.ForeignKey(Tecnico, models.DO_NOTHING, db_column='tcnico')
    srv_nombre = models.CharField(max_length=255, blank=True, null=True)
    srv_idserv = models.ForeignKey(Servicio, models.DO_NOTHING, db_column='srv_idserv')
    titulo = models.CharField(max_length=255, blank=True, null=True)
    configelm = models.ForeignKey(Configelm, models.DO_NOTHING, blank=True, null=True)
    prioridad = models.CharField(max_length=255, blank=True, null=True)
    grup_o_resoluto_r = models.ForeignKey(GrupoResolutor, models.DO_NOTHING, db_column='grup_o_resoluto_r')
    fecha_cierre = models.DateTimeField(blank=True, null=True)
    comentario = models.CharField(max_length=255)
    subcategoria = models.CharField(max_length=255, blank=True, null=True)
    monitorizacion = models.IntegerField(blank=True, null=True)

    def __unicode__(self):
        return self.numero

    class Meta:
        managed = True
        db_table = 'tiquet'
```

Fig. Exemple del model Tecnico i Tiquet

En el sub-directori /middleware, és on es pot trobar tota la part d'extracció de dades de la base de dades Oracle.

Al directori /static, s'emmagatzemen tots els elements estàtics de les templates. Com poden ser els CSS, funcions javascript o totes les imatges que vulguem mostrar al web. Aquest directori també s'especifica al fitxer settings.py i és totalment configurable,

Al directori templates, hi ha cadascuna de les templates, que s'han dissenyat per a cada url-request que es faci a l'aplicació.

Les urls es mappejen al fitxer urls.py i aquest, redirigeix la petició cap a una vista o View, que es troba dins el directori /views.