
**Treball de Fi de Grau**

**GRAU D'ENGINYERIA INFORMÀTICA**

**Facultat de Matemàtiques**
**Universitat de Barcelona**

---

# EQUAGRAF: Una aplicació per aprendre a resoldre sistemes d'equacions a través d'un graf.

---

**Joan Gasull Royes**

Director: Mariella Dimiccoli
Realitzat a: Departament de Matemàtica
Aplicada i Anàlisi. UB

Barcelona, 26 de juny de 2015

# Sumari

# 1. Resum

El projecte desenvolupat i documentat a continuació és una aplicació web que resolt sistemes d'equacions de primer grau i mostra de manera gràfica el seu procediment mitjançant el desenvolupament d'un graf.

La funció principal per la qual s'ha creat l'aplicació es buscar d'ajudar a l'usuari a entendre quins son els passos que ha de realitzar per resoldre el seu sistema d'equacions. Per tal de poder realitzar el procediment, l'usuari introduirà les diferents equacions que componen el sistema i ens dirà quins inputs té (variables que en sap el seu valor) i quins outputs busca (variables que busquem). Un cop es coneguin les dades es visualitzarà el graf que ens indicarà els diferents passos a seguir i donarà el resultat per pantalla.

Durant els anys d'aprenentatge de com resoldre un sistemes d'equacions ens han ensenyat com resoldre els sistemes amb explicacions basades en l'aprenentatge de la resolució de problemes mitjançant les tècniques d'igualació, substitució, reducció i mètode de Gauss.

Aquest mètodes es basen en la resolució dels sistemes mitjançant repetir la realització de diferents problemes similars. Es per això que el que es vol aconseguir és una aplicació web per aprendre a resoldre problemes mitjançant la visualització de diferents grafs que ens aniran facilitant entendre el procés de recerca de la solució utilitzant els mètodes de substitució i igualació. Per tal d'aconseguir això, s'utilitzarà el graf com a base per l'aprenentatge.

Aquest projecte està encarat a varis públics objectius. Aquest públics objectius poden ser des d'un alumne que aprèn per primera vegada a resoldre un sistema d'equacions de primer grau fins un problema de física o economia que contenen la mateixa estructura, és a dir, unes equacions de primer grau (fórmules anomenades a física i economia), unes variables d'entrada amb valors i unes variables a buscar.

## 2. Resumen

El proyecto desarrollado y documentado a continuación es una aplicación web que resuelve sistemas de ecuaciones de primer grado y muestra de forma gráfica su procedimiento mediante el desarrollamiento de un grafo.

La función principal por la cual se ha creado la aplicación es buscar de ayudar al usuario a entender cuales son los pasos que tiene que realizar para resolver su sistema de ecuaciones. Para poder realizar el procedimiento, el usuario introducirá las diferentes ecuaciones que componen el sistema y nos dirá que inputs tiene(variables que conoce su valor) i que outputs busca (variables que quiere saber su valor). Una vez se conozcan los datos se visualizará el grafo que nos indicará los diferentes pasos a seguir y nos dará el resultado por pantalla.

Durante los años de aprendizaje de cómo resolver un sistema de ecuaciones nos han enseñado a resolverlo con explicaciones basadas en el aprendizaje mediante los métodos de igualación, substitución, reducción y método de Gauss.

Estos métodos se basan en la resolución de los sistemas mediante la repetición de distintos problemas similares. Es por esto que el que se quiere conseguir es una aplicación web para aprender a resolver problemas mediante la visualización de distintos grados que nos facilitarán poder entender el proceso de recerca de la solución utilizando los métodos de substitución e igualación. Para conseguir esto, se utilizará el grafo como base para crear las imágenes.

Este proyecto estará encarado a varios públicos objetivo. Estos públicos objetivos pueden ser des de un alumno que adquiere los conocimientos por primera vez de resolver un sistema de ecuaciones hasta un problema de física o economía que contengan la misma estructura, es decir, unas ecuaciones de primer grado(fórmulas en términos de física i economía), unas variables que conocemos su valor y unas variables que desconocemos su valor y que queremos saberlo.

## 3. Abstract

This project is a web application that can solve systems of first order equations and graphically shows the procedure by developing a graph.

The main function of this app is help the user to understand what are the steps to be performed to solve his system of equations. In order to run the application, the user has to enter the different equations of his system and also has to indicate the inputs (variable values) and the outputs (variables to find). Once all the data is known, the program will show the graph with all the different steps and the final result (outputs) will be displayed on the screen.

During the learning how to solve a system of equations the teachers have taught us how to solve the systems through the methods of inspection, elimination, substitution and Gauss.

Those methods are based on the resolution of systems of equations through repeat performing various similar problems. The main goal of this web application is solving systems of first order equations ensuring that the user internalizes each step of the resolution. Because that, this app shows different graphs that will help the user to understand the process of finding the solution through the methods of elimination and inspection.

This project is focused on several target audiences. This app can help students that are learning for the first time to solve a system of equations, and also more advanced students that in his physics or economics problems may have to solve systems of equations with a lot of outputs.

# 4. Introducció

En diferents punts de la nostra trajectòria acadèmica ens trobem damunt la taula problemes matemàtics que no sabem ni com començar a resoldre'ls. Sovint aquests problemes matemàtics tenen la mateixa estructura: unes fórmules o equacions que ens dona la teoria i unes dades t'han d'entrada com de sortida que l'enunciat del problema ens estableix i que ens ajuden a resoldre'l.

Tot i tenir la mateixa estructura, un sistema d'equacions, ens podem trobar encallats ja sigui perquè no hem practicat prou(via repetició) o perquè no aconseguim com podem encarar el problema. Tot i així, sovint no ens adonem que simplement es tracta d'un sistema d'equacions.

Per agafar la idea explicaré un exemple pràctic bàsic. Imaginem que nosaltres coneixem les fórmules de la Intensitat (I=Q/t) i la llei d'Ohm(V=I*R) i que volem buscar la resistència elèctrica d'un circuit(R). L'enunciat però com a paràmetres ens estableix que la caiguda de voltatge(V) és 5. També ens dona la carga(Q), que ens diu que és de 14500 i el temps(t), que diu que ha transcorregut 60 segons.

Per tant, per resoldre el problema primer haurem de buscar la intensitat mitjançant l'equació I=Q/t, ja que tenim una equació amb una incògnita d'on extrem la I. Un cop tenim la I podem extreure la resistència elèctrica(R).

Aquest és el cas més bàsic d'un problema amb l'estructura d'un sistema d'equacions, però que succeiria si el problema contingues 10 equacions i no sabéssim quina agafar? I si el sistema és resolt mitjançant el mètode d'igualació però no en sabem relacionar les equacions que coneixem? Explicar-ne com solucionar-ho sovint és complicat i és per això que l'aplicació web vol mostrar visualment com solucionar el problema esmentat.

Segurament nosaltres no som conscients que una visió gràfica o visual ens pot ajudar a obrir la ment i entendre quina és la millor solució. Per aquest motiu, l'aplicació ens presta una altra manera diferent de funcionar i entendre els diferents problemes trobats en el llarg del recorregut acadèmic.

Ens trobem en la creació d'una nova eina d'aprenentatge visual utilitzant els mètodes de substitució i igualació i mostrar mitjançant graf.

## 4.1 Motivació

La motivació es basa en dos pilars principals, la part social i la part tecnològica.

El primer motiu és la necessitat pròpia de realitzar una funció social ajudant altres persones alienes al projecte, en aquest cas, a diferents usuaris que es troben amb un problema estructurat per un sistema d'equacions, explicant el seguiment que ha de realitzar per resoldre'l.

El segon motiu és la creació d'una interfície web completa i des de zero mitjançant un framework obtenint els diferents coneixements tecnològics que això precisa. El fet d'ampliar els meus coneixements és un dels punts que em fa tirar endavant aquest projecte personal ja que, inicialitzar un projecte sense previs coneixements és un repte personal a batre. A més, com ja he mencionat, el fet que sigui un projecte personal elegit per mi augmentar la meva motivació.

Són per aquests dos motius l'elecció i la realització d'aquest treball.

## 4.2 Objectius

L'objectiu principal és donar facilitats a l'usuari per tal que entengui com resoldre el problema de manera més eficient. Per tal que això succeeixi podem diferencia dos objectius primordials, la resolució del problema i el mostreig visual de com arribar a la solució.

Per a la resolució del problema podem definir dins de l'objectiu dos nivells a assolir.

Nivell 1:

Resoldre un sistema d'equacions bàsic mitjançat el mètode de substitució.

- Fórmules: $a+b=c-d$, $a+b=t$

- Variables inputs(variables que coneixem mitjançant l'enunciat):$b$, $c$, $d$

- Variables outputs(variable solució del problema): $t$

La resolució de la qual és buscar primer el valor de '$a$' i seguidament aplicar aquest valor a la segona equació($a+b=t$).

Nivell 2:

Resoldre un sistema d'equacions on desconeixem més d'una variable dins d'una fórmula i les ajuntem per a tractar-les com una única variable, és a dir, utilitzant el mètode d'igualació.

- Fórmules: $a+b=c-d$, $a+b=t$

- Variables inputs: $c$, $d$

- Variables outputs: $t$

Com podem comprovar, la diferència es troba en que el nivell dos no tenim cap equació que té només una variable desconeguda. Al desconèixer tant '$a$' com '$b$' ens és impossible resoldre el problema mitjançant la resolució en cadena de les equacions. Per tal de resoldre això simplement tractarem $a+b$ com una única variable i resoldrem el problema mitjançant substitució: $t=c-d$

Respecte al mostreig visual cal donar molta importància a la forma o manera de mostrar la solució gràfica per tal de donar a l'usuari les màximes facilitats perquè entengui com s'ha resolt.

Per tal d'assolir aquest objectiu, es vol realitzar una aplicació web per tal de visualitzar el procés de resolució d'un sistema d'equacions a través d'un graf.

Cal mencionar que en la realització de l'aplicació web ens limitarem únicament en sistemes de grau 1 i les equacions introduïdes no poden contenir parèntesi.

## 4.3 Missió i visió

### 4.3.1 Missió

La nostra missió és convertí l'aplicació web en una eina d'aprenentatge per l'estudi i la realització de sistemes d'equacions de primer grau utilitzant el graf com a mitjà per resoldre'l.

### 4.3.2 Visió

Volem ser la primera aplicació web dins del mercat en resoldre sistemes d'equacions de primer grau d'una manera gràfica i visual.

# 5. Competidors i anàlisi del mercat

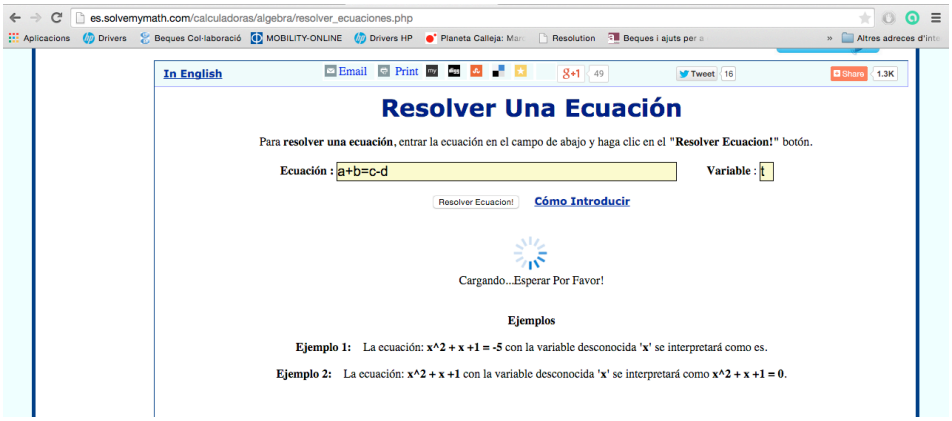
Per tal de poder conèixer quines necessitats manquen actualment i quines propostes ens podem fer diferència més és bàsic tenir un bon estudi sobre les aplicacions similars al projecte. És per això que s'ha realitzat un estudi del mercat, analitzant cadascuns dels punts per tal de tenir una aplicació única.

## 5.1 Competidors i aplicacions semblants.

*Numberempire*[1] és una pàgina web online per poder resoldre diferents equacions. Aquesta utilitza un únic input per afegir totes les equacions i no et permet donar valors. El seu funcionament és difícil d'entendre degut al gran contingut en la zona, és a dir, es poc clar. A més, el disseny visual de la web esta poc treballat. No dóna resultats visuals i no controla errors. Tampoc s'entén el seu funcionament.


Il·lustració 5-1 Pàgina principal de *numberempire*

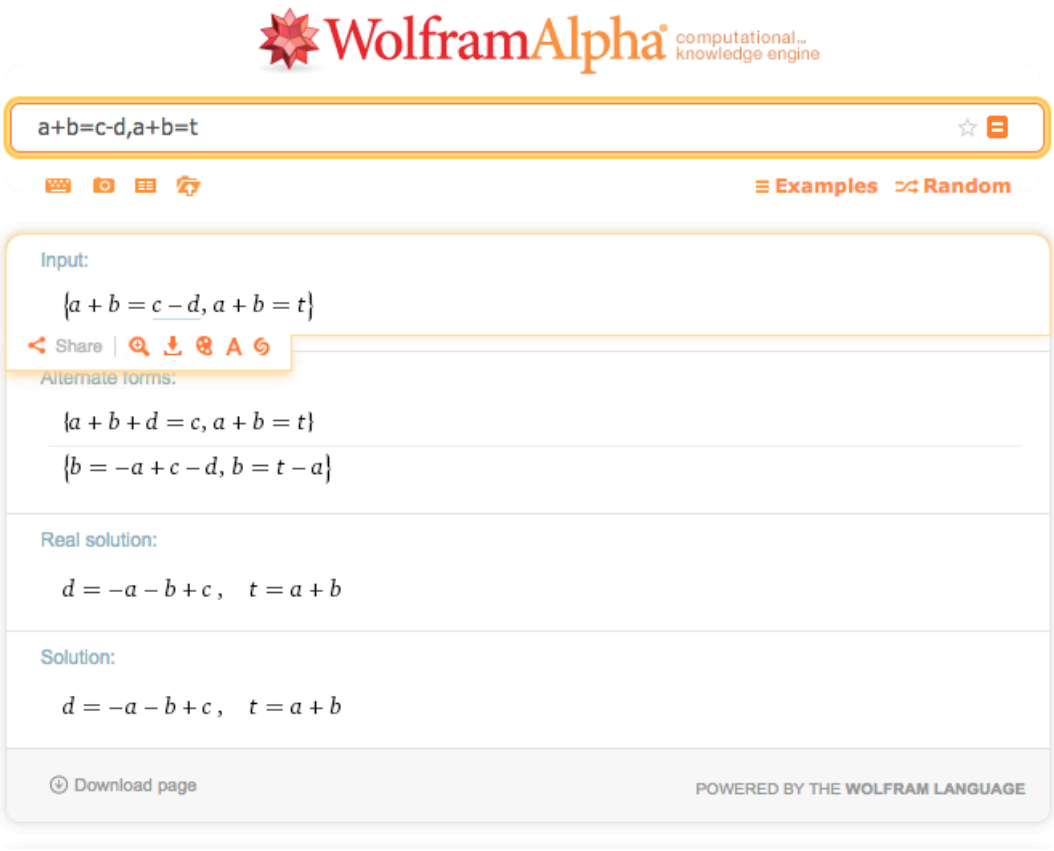
*SolveMyMath*[2] és una aplicació per resoldre equacions. Aquesta només permet resoldre una equació i sempre igualant-la a 0. No permet donar valors i es lenta. No va arribar a donar-me solució.




Il·lustració 5-2 Pàgina principal de SolveMyMath

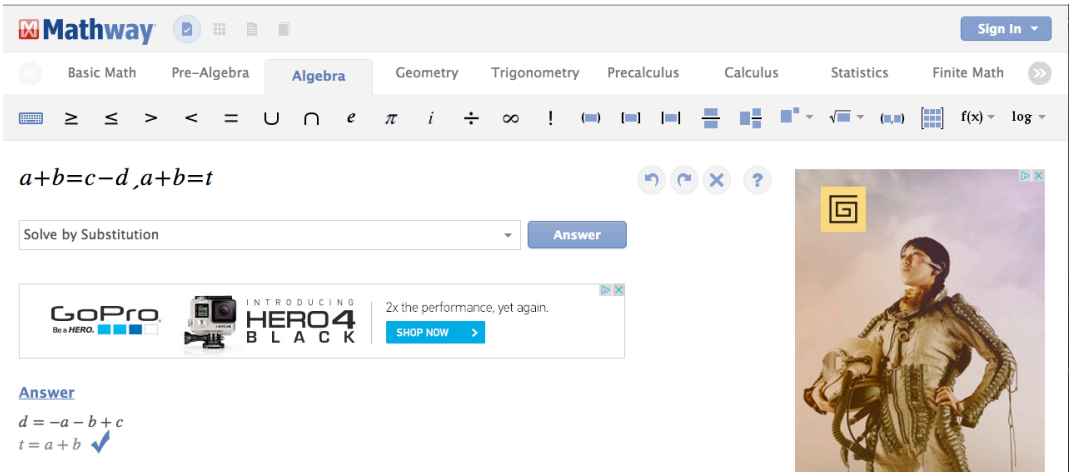
*QuickMath*[3] és una altra aplicació amb la possibilitat de visualitzar el graf de les funcions i per tant, res a veure amb la nostra aplicació. Lo més positiu és que està molt ben documentada, ja que té un apartat *help* per cada opció. Tot i així no resolt l'estructura de problemes mencionada en el projecte. Útil per problemes bàsics però no per les nostres expectatives.

*Wolfram Alpha*[4] és la web online més pròxima a les nostres expectatives. Tot i així te una curba d'aprenentatge considerable i no mostra cap tipus d'aprenentatge visual.




Il·lustració 5-3 Pàgina principal de WolframAlpha

*Mathway*[5] és una aplicació per resoldre problemes bàsics. Tot i així, té un ús complicat. Té solucions pas a pas. No mostra aprenentatge visual. Pot ser útil si aprens a utilitzar-ho però no compleix els nostres objectius.




Il·lustració 5-4 Pàgina principal de Mathway

Com a conclusió de les aplicacions vistes, podem considerar que cap recull les característiques de resoldre problemes amb el format de poder afegir les equacions, els seus inputs i sol·licitant els diferents outputs. A més a més, l'aprenentatge visual es manca en totes les opcions.

## 5.2 Anàlisi DAFO

Aquest anàlisi consisteix en extreure'n les debilitats, les amenaces, les fortaleses i les oportunitats del projecte. D'aquesta manera obtenim un anàlisi extern amb les oportunitats i les amenaces i un anàlisi intern amb les fortaleses i les debilitats.

| | **Aspectes favorables** | **Aspectes desfavorables** |
|---|---|---|
| **Anàlisi intern** | Fortaleses: El projecte té una part d'aprenentatge visual. Utilitzar grafs. | Debilitats: Pocs recursos monetaris. Mercat nou. |
| **Anàlisi extern** | Oportunitats: Es cobreix una necessitat. L'aplicació no es troba en el mercat. Tenim un producte nou en el mercat(aprenentatge visual). | Amenaces: Competidors amb més recursos. No sabem com pot repercutir en el mercat. No tenim marca. |

# 6. Estudi econòmic

En aquest capítol realitzarem un estudi econòmic per saber el valor del cost total de la realització del projecte.

Primerament cal observar que totes les tecnologies i llicències utilitzades són gratuïtes i per tant, no ens resultarà cap cos. Per altra banda, el treball humà realitzat es calcula dividint el projecte en 4 grans apartats, aprenentatge, disseny, desenvolupament i memòria.

| Fase | Hores | Preu/hora | Cost(Euros) |
|---|---|---|---|
| Aprenentatge | 100 | 5 | 500 |
| Disseny | 50 | 25 | 1250 |
| Desenvolupament | 180 | 25 | 4500 |
| Memòria | 120 | 25 | 3000 |
| Total | 450 | - | 9250 |

Pel que fa respecte a la taula, comentar que l'aprenentatge s'ha cobrat a preu de becari de la UB. La resta del treball a 25 euros la hora ja que és el preu a cobrar per un autònom. Si el nostre projecte fos realitzat per una empresa el preu per hora augmentaria fins als 35 euros la hora.

# 7. Planificació

Per tal de planificar el projecte es va mirar de quantes setmanes es disposava. En total es disposava d'unes 20 setmanes. Dins d'aquestes setmanes es van mirar de distribuir totes les tasques per tal de portar a terme el projecte.
Era important saber quantes hores productives es podia disposar setmanalment per tal de distribuir les tasques. Degut a que les 15 primeres setmanes les hores eren més limitades ja que es compaginaven amb feina laboral i classes es va dividir el projecte en dos grans parts.

La primera té una duració de quinze setmanes, les quals si han invertit un total de 4 hores diàries durant 5 dies de la setmana. Aquest 5 dies no necessàriament tenien que ser de dilluns a divendres sinó que s'elegien depenen la setmana, triant els 5 dies més adients per anar procedint a la realització del projecte. Aquestes 15 setmanes completen un total de 300 hores.

La segona part té una duració de 4 setmanes. En aquestes 4 setmanes ja s'ha finalitzat classes i exàmens i a més, ja no s'ha tingut que compaginar amb feina laboral. Degut a que es disposava de tots els dies, la jornada invertida era el doble. En total sumaven 160 hores distribuïdes en 4 setmanes amb una duració de 5 dies laborals realitzant 8 hores diàries.

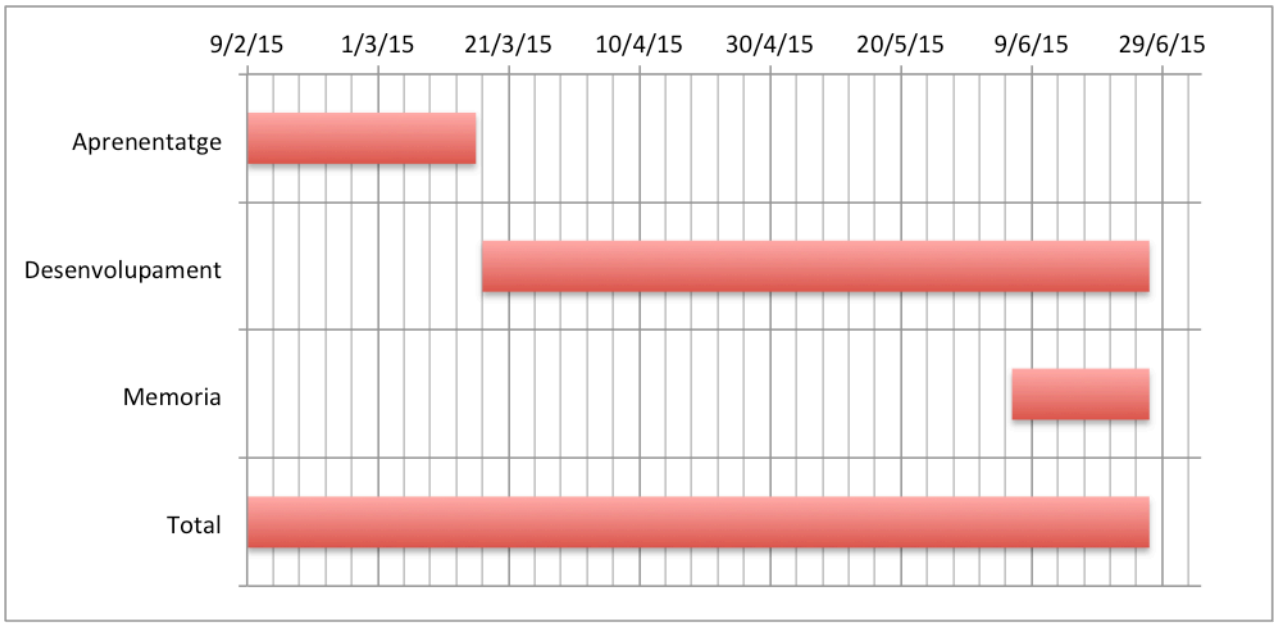
| 15 setmanes (300h) | 4 setmanes (160h) |
|---|---|
| 9 febrer – 15 febrer | 31 maig – 6 juny |
| 16 febrer – 22 febrer | 7 juny – 13 juny |
| 23 febrer – 1 març | 14 juny – 20 juny |
| 2 març – 8 març | 21 juny – 27 juny |
| 9 març – 15 març | |
| 16 març – 22 març | |
| 23 març – 29 març | |
| 30març – 5 abril | |
| 6 abril – 12 abril | |
| 13 abril – 19 abril | |
| 20 abril – 26 abril | |
| 27 abril – 3 maig | |
| 4 maig – 10 maig | |
| 11 maig – 17 maig | |
| 18 maig – 24 maig | |

## 7.1 Diagrama de Gantt

Per tal de tenir una millor visualització de com s'han distribuït les tasques s'han realitzat dos diagrames de Gantt.

### 7.1.1 Diagrama Gantt general

En aquest diagrama es pot observar les tasques separades per aprenentatge, desenvolupament i memòria. A la part inferior a més he afegit el total per poder comparar millor visualment durant quin temps s'ha realitzat cada apartat.




Il·lustració 7-1 Diagrama de Gantt general

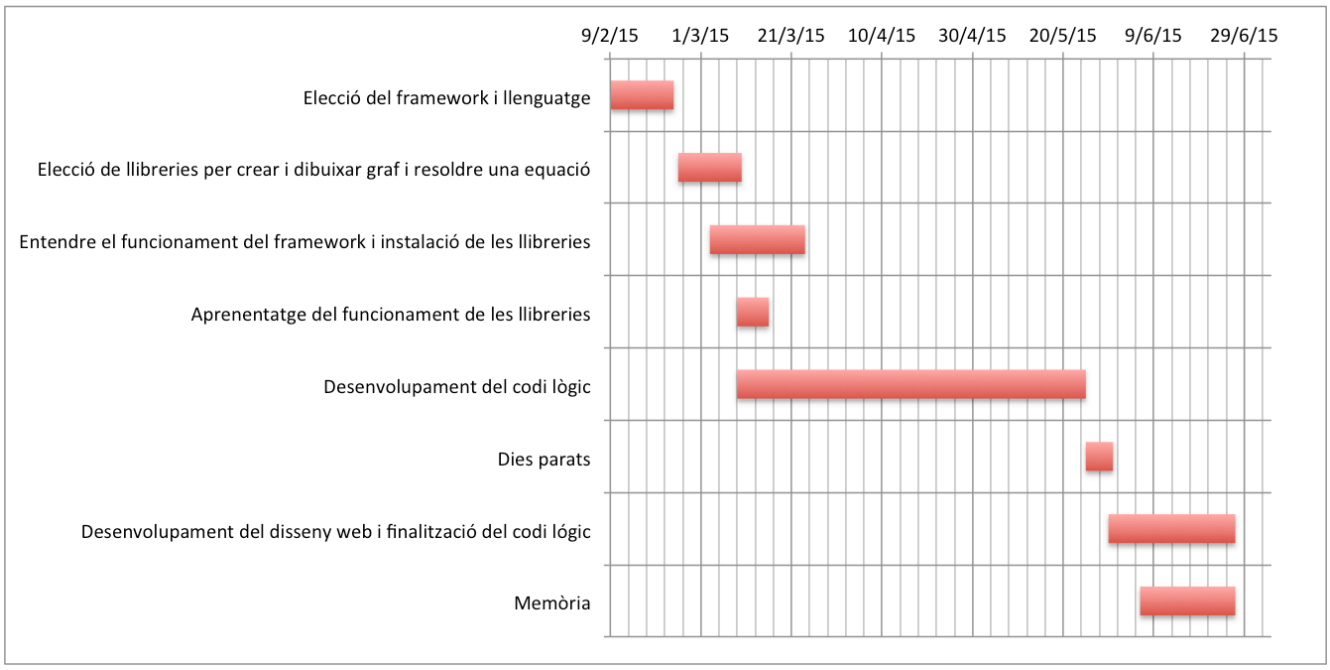
Com podem comprovar a la taula, l'aprenentatge ha durat 5 setmanes, que són un total de 100 hores. El desenvolupament ha durat un total de 14 setmanes i 2 dies. Cal comentar que dins aquest interval es va parar uns dies a causa d'exàmens finals.
Finalment podem observar la realització de la memòria que són aproximadament 21 dies, dels quals han estat compartits amb desenvolupament. En total de hores s'ha invertit unes 120 hores en la realització de la memòria. Cal recordar que les últimes setmanes s'ha invertit el doble de temps, i per aquest motiu s'ha realitzat més hores.

| Activitat | Data Inici | Hores | Duració | Data Final |
|---|---|---|---|---|
| Aprenentatge | 9/2/15 | 100 | 35 | 16/3/15 |
| Desenvolupament | 17/3/15 | 230 | 102 | 27/6/15 |
| Memòria | 6/6/15 | 120 | 21 | 27/6/15 |
| Total | 9/2/15 | 450 | 138 | 27/6/15 |

### 7.1.2 Diagrama de Gantt per tasques

En el següent diagrama de Gantt podem observar les tasques realitzades més específicament.

- **Elecció del *framework* i llenguatge**: Durant aquest període es va esta mirant les diferents opcions per poder dur a terme el projecte. Entre elles diferents frameworks i bàsicament dos llenguatges, java i python.
- **Elecció de llibreries per crear i dibuixar graf i resoldre una equació**: En aquest període, el qual ja havíem elegit Django com a framework web, es va inicia la cerca de llibreries per poder realitzar el projecte.
- **Entendre el funcionament del framework i instal·lació de les llibreries**: Descobrir el funcionament de Django i les seves opcions i limitacions.
- **Aprenentatge del funcionament de les llibreries**: Un cop les llibreries les tenim elegides i sabem que podem tirar endavant el projecte, s'ha invertit temps en entendre-les i comprovar com encaixarien a l'aplicació web. Aquesta tasca s'ha compartit amb la d'entendre el funcionament de Django ja que van agafades de la mà.
- **Desenvolupament del codi lògic**: Període de temps que s'ha invertit en realitzar tota la part funcional del projecte. És la part del projecte que engloba el llenguatge python. També s'ha realitzat les diferents proves per comprovar el funcionament del projecte.
- **Dies parats**: Exàmens
- **Desenvolupament del disseny web i finalització del codi lògic**: Aquest període de temps ha estat invertit en el disseny de la web, és a dir, la part visual del projecte. És la part que engloba els llenguatges html, css i jquery. En ella també si engloba la part de test corresponent al disseny.




Il·lustració 7-2 Diagrama de Gantt específic.

A la taula següent podem observar les dades de l'anterior diagrama de Gantt. Cada tasca, amb la seva data d'inici i final, les setmanes que ha durat i per tant, el número de dies que s'ha invertit.

| Tasca | Data inici | Setmanes | Duració | Data final |
|---|---|---|---|---|
| Elecció del framework i llenguatge | 9/2/15 | 2 | 14 | 23/2/15 |
| Elecció de llibreries per crear i dibuixar graf i resoldre una equació | 24/2/15 | 2 | 14 | 10/3/15 |
| Entendre el funcionament del framework i instalació de les llibreries | 3/3/15 | 3 | 21 | 24/3/15 |
| Aprenentatge del funcionament de les llibreries | 9/3/15 | 1 | 7 | 16/3/15 |
| Desenvolupament del codi lògic | 9/3/15 | 11 | 77 | 25/5/15 |
| Dies parats | 25/5/15 | 0,857142857 | 6 | 31/5/15 |
| Desenvolupament del disseny web i finalització del codi lógic | 30/5/15 | 4 | 28 | 27/6/15 |
| Memòria | 6/6/15 | 3 | 21 | 27/6/15 |

# 8. Arquitectura

Enfoc utilitzat i entorn de desenvolupament. En aquest apartat veurem quin framework és l'escollit i perquè. També veurem les diferents llibreries utilitzades per a realitzar l'aplicació

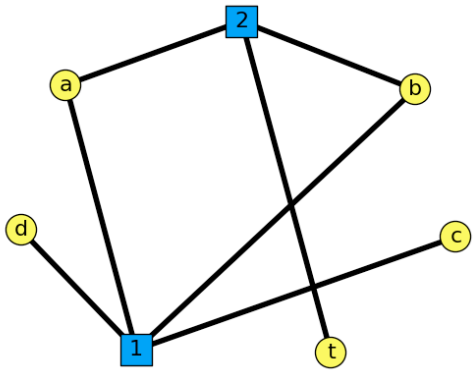
## 8.1 Enfoc

Tal i com em mencionat en l'apartat objectius es representarà el sistema d'equacions mitjançant un graf, el qual estarà composat per dos tipus de nodes.

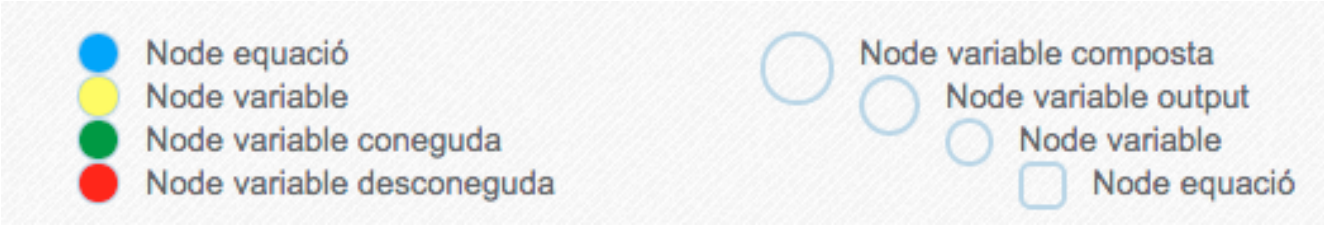
El primer tipus de node, és el node equació que representa una de les equacions introduïdes per l'usuari i el segon tipus de node, és el node variable que representa cadascuna de les variables que composen una equació.

Cada node equació es troba enllaçat amb els seus respectius nodes variables que apareixen a l'equació, així d'aquesta manera es pot visualitzar la relació entre equacions a través de les variables.

Els nodes equació estaran representats per el seu identificador, on es podrà comprovar en una taula mostrada per l'aplicació de quina equació estem parlant.




Il·lustració 8-1 Graf inicial. És el graf creat inicialment, on encara l'usuari no ha afegit els valors de les variables que coneix i no s'ha cercat el resultat.



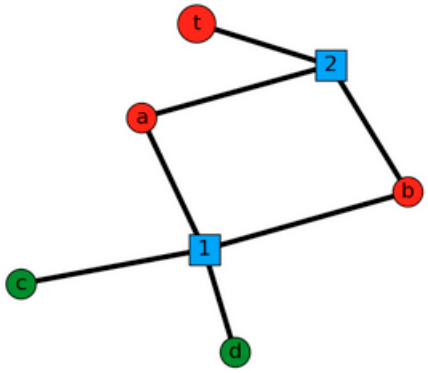

Il·lustració 8-2 Llegendes. Llegenda de colors i llegenda de formes.

A partir d'aquesta representació podrem mostrar clarament els diferent nodes coneguts, els desconeguts i en quines equacions tenen relació. Primerament comentar que l'aplicació es basarà en dos llegendes, la primera és la representació mitjançant colors, que indicarà si coneixem o no coneixem el node. La segona llegenda està realitzada utilitzant la forma com a font d'informació. Els quadrats ens indicaran que es tracta d'un node equació. Les rodones, segons la mida indicaran que es una variable, una variable composta per altres variables o si es tracta d'una variable que volem saber-ne el resultat(variable output).

Degut a que per resoldre una equació de primer grau, únicament hi ha d'haver un node variable desconegut, podem visualitzar fàcilment per quina equació s'ha de començar per aplicar el mètode de substitució. D'altra banda, si no hi ha cap equació que té només una variable desconeguda podrem

visualitzar fàcilment a quines equacions, com es mostra en l'il·lustració 8.3, podem aplicar el mètode d'igualació.

Aquest mostreig visual és el que ens diferencia de les altres aplicacions que resolen problemes similars.




Il·lustració 8-3 Graf amb els valors indicats segons les llegendes. En aquest cas coneixem el valor de "c" i "d" i desconeixem el valor de "a", "b" i "t". A més, sabem que "1" i "2" són equacions i que "t" és una variable que volem saber-ne el resultat ja que la rodona és lleugerament més gran.

## 8.2 Elecció del framework Django

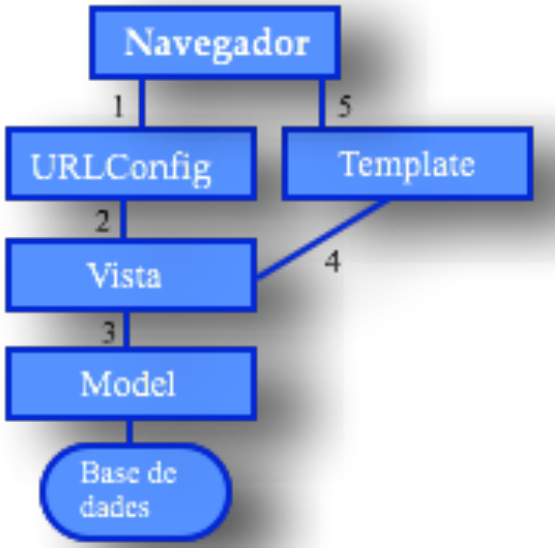
Una de les decisions més importants per a la creació del projecte és l'elecció del llenguatge de programació i el seu corresponent framework amb el que el realitzaríem. Per tal de poder elegir vam tenir que mirar quins aspectes importants ens podien fer decantar.

- Utilització de graf: Un dels aspectes importants era la possibilitat de poder crear un graf i no només poder-lo crear sinó també poder-lo visualitzar d'una manera simple i eficient. Per tant, el llenguatge de programació emprat tenia que poder instal·lar una llibreria per el tractament de les dades. En aquest apartat ja vam delimitar els llenguatges a dos, java o python.

- Llibreries vistes en anterioritat: En una assignatura de Python s'ha realitzat alguna petita pràctica amb la llibreria NetworkX la qual et dóna possibilitats per poder treball amb el graf. En canvi, en java no s'havia treballat en cap assignatura.

Un cop vist aquests dos aspectes l'elecció es trobava en el framework a elegir i per tant, quin llenguatge. Degut a la mancança de coneixements del funcionament de cap framework, ja que no s'havia treballat amb cap en anterioritat, és va processar a realitzar una petita reunió amb la tutora per tal de decidir quin framework elegir. Finalment, el framework elegit va ser Django degut a la seva estructura i perquè el llenguatge emprat és Python, i per tant, es pot utilitzar la llibreria NetworkX.

## 8.3 Estructura de Django




Il·lustració 8-4 Estructura Django

Django segueix l'estructura mostrada a l'imatge. A diferencia de la estructura model-vista-controlador, en Django s'ha creat una estructura per facilitar al màxim el funcionament. Una diferencia molt important per entendre l'estructura de Django és saber que la vista fa de controlador i el template fa de vista. Per entendre la comunicació amb cadascuna de les parts les explicarem individualment.

- Navegador: Programa informàtic que ens permetrà visualitzar la part continguda al template.

- URLConfig: Conté totes les direccions on es guarden els templates, és a dir, es l'arxiu que redirecciona les rutes. La seva funció és llegir la url del navegador, trobar la vista apropiada per la seva sol·licitud i passar alguna variable en cas que la vista la necessiti. Llenguatge Python.

- Vista: Conté totes les funcions pel tractament de dades, mostreig dels templates o accés a la base de dades. Aquestes funcions són creades mitjançant el llenguatge Python. Aquí es on es troben els algoritmes del treball per tal de que funcioni correctament.

- Model: L'arxiu model inclou les diferents classes necessàries per poder treballar a la vista. També és l'arxiu que fa d'enllaç amb la base de dades. En el nostre cas, no ha estat necessari realitzar aquest enllaç ja que no em considerat necessari tenir una base de dades dels nostres usuaris ni dels problemes indicats. També és llenguatge Python.

- Template: En el template indicarem tots els arxius HTML. Aquest arxius són les diferents pàgines web que mostrarem al navegador. Aquest arxius també poden contenir altres llenguatges com Javascript, CSS, CSV, etc. En el nostre cas utilitzarem el HTML i el CSS.

Un cop vistes cadascuna de les parts procedim a l'explicació de la comunicació entre les diferents parts. Primerament, com hem dit en anterioritat, el navegador rep una url escrita per l'usuari. En aquest punt ens trobem en el punt 1 de l'imatge. El navegador connecta amb la URLConfig, la qual agafa les variables en cas que siguin necessàries i crida la vista. Un cop això, anem al punt 2 de l'imatge. La vista executa la funció corresponent a la url indicada i comença l'algoritme. Aquí ens trobem en la possibilitat de dos camins. El primer és que aquesta pot ser que cridi a l'arxiu model(punt 3). Si es produeix això, l'arxiu model pot cridar la base de dades, o simplement retornar les dades necessàries per l'algoritme de la vista, tal i com succeeix en el nostre treball. El segon camí es que la vista

directament executi un algoritme el qual no necessàriament passa per l'arxiu model. Aquí es on ens trobem en el pas 4, la qual la vista crida un fitxer HTML el qual li passa les dades corresponents perquè aquest les tracti. Per tal de que l'usuari pugui visualitzar les dades, ens hem de posar al punt 5, el qual el template s'executa a través del navegador de tal manera que tanquem el cicle, per tal de poder tornar a començar l'execució de l'estructura explicada en Django.

## 8.4 Llenguatges utilitzats al framework Django.

Python: El llenguatge principal del framework és el Python. És per aquest motiu per el qual s'ha elegit Django tal i com s'ha mencionat anteriorment. Python és un llenguatge d'alt nivell utilitzat a l'universitat en varies assignatures. Com ja sabem el seu funcionament ens ha ajudat a entendre el funcionament de Django. Aquest es troba en els arxius vista, URLConf i model. A través d'ell s'ha realitzat tots els algoritmes necessaris per poder crear, controlar i executar totes les funcions.
Per tal de poder executar els fitxers Python és imprescindible l'instal·lació en el servidor de la web. L'instal·lació és senzilla. Primerament s'ha de descarregar Python, en el nostre cas l'última versió, la 3.4 la qual es troba a la direcció https://www.python.org/downloads/. Un cop descarregada s'ha d'instal·lar clicant damunt l'arxiu descarregat.

HTML: Per tal de poder visualitzar les dades Django utilitza el llenguatge HTML. Aquest es troba als arxius dins de la carpeta Template, els quals tracten les dades rebudes per tal que l'usuari les pugui interpretar amb facilitat. La versió utilitzada és la HTML5, la qual es compatible amb els grans navegadors universals.

CSS: El HTML ens ajudar a interpretar les dades, però el llenguatge que li donar l'estructura final i el color és el CSS. Aquest es troba també a dins els arxius HTML. El CSS és un llenguatge no molt treballat a l'universitat però que té molta informació externa per tal de poder-lo treballar. La majoria de codi CSS es troba dins la carpeta *static* en l'arxiu style.css

```css
body {
    background-image: url("/static/bg.png");
    padding: 30px;
    font: 13px normal Helvetica, Arial, sans-serif;
    color: #4F5155;
}
```

Il·lustració 8-5 Codi CSS del cos general de la web on es carrega una imatge per el fondo de la pàgina, així com un marge interior i el color i tipus de lletra utilitzats.

JQUERY: El JQUERY només s'utilitza en una pàgina. Únicament s'utilitza per realitzar un control en les entrades de dades, fent desaparèixer la zona d'entrada de dades si és seleccionada la variable com a output.

```
<script>
function buttonChecked(item){
    $('#text-'+item).val('');
    $('#text-'+item).toggle();
}
</script>
```

Il·lustració 8-6 Funció JQUERY

## 8.5 Llibreries utilitzades

En aquest apartat explicarem les tres llibreries extres utilitzades. Aquestes llibreries són alienes al framework però igual de necessàries per la realització del projecte. Primerament explicarem la llibreria NetworkX, la qual és necessària per la interpretació de les dades mitjançant un graf. Aquesta llibreria és la que ens diferenciarà dels competidors ja que això no està realitzat en cap de les pàgines que treballen amb fórmules o equacions. Per tal de completar aquesta llibreria s'utilitza la llibreria matplotlib, la qual ens ajudar en mostreig del graf mitjançant una imatge. La tercera llibreria és la Sympy, una llibreria trobada mitjançant cerques per internet. Aquesta llibreria va ser l'elegida per poder resoldre equacions simbòliques. Ambdues llibreries són compatibles amb Python.

### 8.5.1 NetworkX 1.9.1

NetworkX és una llibreria per a Python utilitzada per la creació i manipulació de xarxes complexes. En el nostre cas la xarxa estarà composta per les fórmules i les seves variables corresponents. Aquestes variables a més poden trobar-se en altres fórmules, de manera que la llibreria ens ajuda a través d'un graf a tractar la xarxa i manipular-la segons les nostres necessitats.
La llibreria ens permet per tant tenir nodes, que són les variables i les fórmules i els seus respectius enllaços. D'aquesta manera podem guardar una imatge del graf creat per poder entendre la relació de les diferents fórmules per tal de poder resoldre el problema. A més a més, ens permet dibuixar els nodes i els enllaços de diferents formes i colors.
Aquest treball d'estudi s'ha realitzat mitjançant estudis a través de la xarxa internet i anant fent proves per tal de entendre del tot el funcionament de la llibreria.
Per tal de poder-la utilitzar primerament s'ha d'instal·lar dins el servidor on s'executarà Django, on ho expliquem a continuació.

#### 8.5.1.1 Instal·lació de la llibreria NetworkX

- Primerament s'ha de descarregar la llibreria, la qual és troba a l'enllaç https://networkx.github.io/download.html.
- Un cop descarregat s'introdueixen els fitxers a dins la carpeta de Python prèviament instal·lada. En cas d'utilitzar un servidor, s'introdueix els fitxers a dins la carpeta Python de dins el servidor.
- Ja amb els arxius preparats per a la instal·lació, a través de consola ens situem on es troba l'arxiu setup.py i executem la comanda: sudo python3.4 setup.py install
- Ja tenim la llibreria NetworkX instal·lada.

#### 8.5.1.2 Funcionament de la llibreria NetworkX

Per tal de recordar una mica el seu funcionament s'ha revisat un arxiu entregat a una assignatura de Python la qual explica les funcions principals per a tractar-la[6] i també la pàgina web on s'hi troben totes les funcions amb una petita explicació de cadascuna[7].

Les funcions principals de la llibreria són:

- Creació del graf. Per tal de crear el graf s'ha de realitzar l'import de la llibreria amb la línia de codi *import networkx as nx*. Un cop ja la tenim importada es crea amb la línia de codi *G=nx.Graph(),* on G seria el nostre graf, nx la llibreria importada i Graph() la funció de dins al llibreria que el crea. Ja tenim el graf creat.

```
import networkx as nx
import matplotlib.pyplot as plt
```

Il·lustració 8-7 Importació de les llibreries

```
#Grafs
GI = nx.Graph()#Graf inicial que contindrà les equacions introduides per l'usuari
G = nx.Graph()#Graf complert
GV = nx.Graph()#Graf unicament amb variables per a tractarles
GF = nx.Graph()#Graf pas a pas
```

Il·lustració 8-8 Creació dels diferents grafs utilitzats a l'aplicació.

- Tractament dels nodes. El Graf creat ens serveis per crear la nostra xarxa i manipular-la, és per això que és important conèixer les funcions importants per poder manipular els nodes. Per tal d'afegir un node s'utilitza la funció *G.add_node(X)*, on G és el graf creat en anterioritat, add_node() la funció de la llibreria per afegir el node al graf i X és el valor del node afegit.
  També és pot afegir una llista de nodes amb la funció *G.add_nodes_from([X,Y])*, la qual afegim un seguit de nodes en una mateixa crida.
  Tal i com creem el node també em de poder eliminar-lo. Això es realitzar amb la funció *G.remove_node(X)*, on la funció remove_node() eliminar el node X.
  La nostra xarxa pot tenir varios nodes i em de poder-los conèixer en qualsevol moment. Per poder saber quants nodes tenim i quins tenim utilitzarem dos crides. La primera és *G.number_of_nodes()* la qual ens retorna el número de nodes que té el graf. La segona és *G.nodes()* la qual retorna tots els nodes en una llista.

```
equacio.G.add_node(varFinal)
```

Il·lustració 8-9 Afegint el node anomenat "varFinal" al graf G

```
equacio.G.remove_node(var)
```

Il·lustració 8-10 Eliminant el node *var* del graf G

- Enllaços. La nostra xarxa no té sentit si no podem enllaça els nodes. Per tal de tenir un enllaç entre nodes s'utilitza la crida *G.add_edge(X,Y)*, on add_edge() és la funció de la llibreria per a crear l'enllaç entre els nodes X i Y. Ara ja sabem crear nodes i enllaçar-los. També cal comentar que si executes la comanda anterior sense haver afegit els nodes, aquests es creen automàticament a dins el graf i per tant, ens podem estalviar de realitzar dos crides ja que al realitzar la crida per enllaçar dos nodes, si un o els dos no existeix els crea automàticament.
  Tal i com poder crear els enllaços també els podem eliminar. Per a realitzar la següent funció utilitzem la comanda *G.remove_edge(X,Y)* o *G.remove_edge_from(X)*.
  Com hem comentat amb els nodes, en els enllaços també és important saber quants enllaços tenim i quins són. Per poder obtenir aquesta informació s'utilitza les comandes *G.number_of_edges()* la qual retorna el número d'enllaços del graf i *G.edges()* per tal de saber quins enllaços són.
  Una altra funció important i molt utilitzada en el projecte és *G.neighbors(X)*, la qual retorna els enllaços del node X. Aquesta funció és molt important ja que ens permet anar a un node en concret i saber quina xarxa l'envolta.

```
equacio.G.add_edge(valor1Ca,varFinal)
equacio.G.add_edge(valor2Ca,varFinal)
equacio.labels[varFinal]=varFinal
```

Il·lustració 8-11 Afegint els l'enllaços del node anomenat "*varFinal"* amb els nodes anomenats "*valor1Ca"* i "*valor2Ca"*. També es pot observar com es guarda al diccionari el valor que es mostrarà del node "*varFinal"*, que en aquest cas és el mateix valor que té el node "*varFinal"*.

```
equacio.G.remove_edge(valor1Ca,var)
equacio.G.remove_edge(valor2Ca,var)
```

Il·lustració 8-12 Eliminació dels enllaços de dins el parèntesis de la funció.

```
llista1 = GNF.neighbors(node)
llista2 = GNF.neighbors(node)
```

Il·lustració 8-13 Guardant els enllaços del node *node*

- Altres. Dins de la llibreria hi ha moltes altres funcions que no utilitzarem. Tot i així si que s'utilitzaran funcions per a tal de donar forma al graf. El que hem fet a estat donar color als nodes i als enllaços, donar forma als nodes i als enllaços i guardar el graf o eliminar-lo en una imatge png. Per poder guardar aquests retocs s'utilitza la llibreria *matplotlib*.
  La llibreria NetworkX no s'havia utilitzat en anterioritat respecte el que fa al disseny del graf. Només en vam veure algun detall en alguna assignatura però no la vaig utilitzar per a la realització de pràctiques. Per tal de saber-ne el funcionament he utilitzat l'arxiu Networkx Tutorial i he indagat a la web per tal de poder entendre les seves funcionalitats. Al principi va ser costos degut a que si hi ha exemples però estan poc explicats. Els exemples revisats són els indicats a la pàgina de Networxk Developers[7] a la secció Drawing, ja que té un apartat exclusiu de dibuixar grafs. Després de realitzar varies proves vaig aconseguir entendre'n el seu funcionament però com es pot comprovar hi ha el codi per realitzar el dibuix però no té cap explicació. Per tal de entendre millor el codi explicarem les parts del codi de l'il·lustració 8-14.

```
pos=equacio.nx.spring_layout(equacio.G)
equacio.nx.draw_networkx_nodes(equacio.G,pos,nodelist=equacio.listConegudes,node_color='g',node_size=500,alpha=1)#Blau
equacio.nx.draw_networkx_nodes(equacio.G,pos,nodelist=equacio.listCheckOut,node_color='r',node_size=800,alpha=1)
equacio.nx.draw_networkx_nodes(equacio.G,pos,nodelist=equacio.listDesconegudes,node_color='r',node_size=500,alpha=1)
equacio.nx.draw_networkx_nodes(equacio.G,pos,nodelist=equacio.nodelistFormules,node_color='#1E90FF',node_size=500,node_shape='s',alpha=1)
equacio.nx.draw_networkx_nodes(equacio.G,pos,nodelist=equacio.listNovesVariablesFuncions,node_color='r',node_size=2000 ,alpha=1)
equacio.nx.draw_networkx_edges(equacio.G,pos,edgelist=equacio.edgelist,width=4,alpha=1,edge_color='black')
equacio.nx.draw_networkx_labels(equacio.G,pos,equacio.labels,font_size=16)
equacio.plt.axis('off')
equacio.plt.savefig(path)
```

Il·lustració 8-14 Codi on es dona forma i color al graf.

Primerament cal explicar que has de separar cadascun dels nodes en grups segons com els vols dibuixar. Per exemple, en aquest cas diferenciem els nodes coneguts, els nodes a cercar, els nodes desconeguts, els nodes fórmula i els nodes creats per igualació.

Tal i com podem s'observa a l'il·lustració 8-12, aquestes variables es troben en una llista i s'agafen amb la comanda "nodeList=equacio.listConegudes". Això es degut a que la llibreria no em permetia dibuixar els nodes individualment, sinó que ho he tingut que fer per grups. Dins d'aquestes línies trobem diferents característiques. Primerament trobem l'atribut "node_color", on com l'exemple de l'il·lustració 8-12 "node_color='g'", ens indica de quin color seran els nodes de la llista, en aquest cas verds. Els colors es poden donar en hexadecimal. També podem indicar a través de "node_size" i "alpha" la mida dels nodes i la transparència. La forma

s'indica mitjançant la comanda *node_shape='s'*, on s és perquè es visualitzi quadrada i 'o' perquè es visualitzi rodona.

El cas dels enllaços és similar però amb les funcions pertinents als enllaços, és a dir, amb "edge_color" per donar color a l'enllaç i "width" per donar llargada.

Finalment trobem la comanda de dibuixa els labels(línia 8 de l'il·lustració 8-14). Aquesta comanda ens serveix per indicar quins valors tindran els nodes. Aquests valors es troben en una llista, en aquest cas la llista és "equacio.labels". També es pot indicar la mida amb la funció "font_size". L'última línia de codi de l'il·lustració 8-14 ens guarda el graf en una imatge png. Per tal de poder realitzar aquesta funció em de veure la llibreria Matplotlib explicada a continuació.

### 8.5.2 Matplotlib

Aquesta llibreria va molt enllaçada a la de NetworkX, el motiu és perquè ens permet guardar el graf.

#### 8.5.2.1 Instal·lació de la llibreria Matplotlib

La instal·lació de la llibreria és similar que la Networkx, lo que en aquest cas ho hem realitzat tot per terminal.

- Primerament descarregar la llibreria. Per tal de descarregar la llibreria, ens situarem per consola a la carpeta de Python3.4. Un cop ens trobem a dins executarem la comanda per terminal "sudo git clone git://github.com/matplotlib/matplotlib.git".
- Un cop descarregada entrarem a la carpeta descarregada: cd matplotlib.
- Un cop dins la carpeta matplotlib observarem que també conté un setup.py. Aquí es on executarem la mateixa comanda que la utilitzada per instal·lar la NetworkX, és a dir, executem *sudo python3.4 setup.py install*.
- Ja tenim instal·lada la llibreria Matplotlib.

#### 8.5.2.2 Funcionament de la llibreria Matplotlib[8]

Les funcions primordials són:

- Importació de la llibreria. Per tal de poder utilitzar la llibreria s'ha de realitzar l'importació amb la comanda "import matplotlib.pyplot as plt" on plt és el nom donat per a tractar-la.
- Guardar el graf com a imatge. Per tal de poder realitzar això, tal i com hem explicat en anterioritat s'utilitza la funció "plt.savefig(ruta)", on plt és la llibreria importada, savefig() és la funció de la llibreria i ruta és la direcció on volem que ens guardi l'imatge.

### 8.5.3 Sympy 0.7.6

En el nostre projecte tractem equacions simbòliques. Per tal de tractar-les necessitem una llibreria que ens resolgui el valor de l'equació en cas que es pugui resoldre. Per tal de poder aconseguir aquest objectiu utilitzem Sympy, una llibreria Python que no requereix de cap llibreria externa i que és fàcil d'utilitzar. La llibreria Sympy ens permet passar-li la equació i el "output" que volem trobar i ens retorna una llista amb totes les solucions. L'únic inconvenient és que necessita les equacions en format

canònic. Per tal de passar les equacions en format canònic existeix una funció (canònica()) que ens transforma totes les equacions a canòniques.

#### 8.5.3.1 Instal·lació de Sympy 0.7.6

La instal·lació de la llibreria és idèntica a la MatplotLib:

- Primerament s'ha de descarregar la llibreria, la qual és troba a l'enllaç https://github.com/sympy/sympy/releases
- Seguidament posarem la carpeta descarregada a dins la carpeta de Python3.4.
- Un cop descarregada entrarem a la carpeta descarregada: cd sympy-0.7.6.
- Un cop dins la carpeta sympy-0-7-6 observarem que també conté un setup.py. Aquí es on executarem la mateixa comanda que la utilitzada per instal·lar les altres llibreries, *sudo python3.4 setup.py install*.
- Ja tenim instal·lada la llibreria Sympy 0.7.6.

#### 8.5.3.2 Funcionament de la llibreria Sympy 0.7.6[9]

El funcionament és molt senzill en el nostre cas ja que només utilitzem una funció, la solve, per tal de resoldre l'equació.

```
d=solve(posCami,out)#Fem el càlcul de la fòrmula
```

Il·lustració 8-15 Funció de la llibreria Sympy per resoldre una equació canònica.

Com podem observar en la figura 6, la funció solve() té dos paràmetres d'entrada, un és el possible camí, és a dir, la fórmula. L'altre és el valor que volem buscar, el que nosaltres coneixem com a output. Aquesta funció retorna una llista, ja que pot ser que l'output tingui més d'un valor possible. Seguidament, com la llibreria esta en Python, tractem les dades com millor ens sembla.

# 9. Anàlisi i disseny

## 9.1 Anàlisis de requeriments i funcionalitats

En l'apartat d'anàlisi i disseny es van analitzar les funcionalitats ha implementar en l'aplicació i un seguit de requeriments que havia de complir perquè l'aplicació funcioni a la perfecció en els diferents navegadors.

La funció principal del projecte és que l'usuari pugui introduir les diferents equacions que disposa, que en pugui introduir les dades i que pugui observar els resultats obtinguts tan mitjançant imatges realitzades pel graf com mitjançant les diferents taules on s'informarà de les dades mostrades en les imatges.

## 9.2 Requisits funcionals

En el nostre projecte al ser una aplicació que no utilitza bases de dades ni registres només es pot identificar un únic usuari, l'usuari anònim. Tot i així, aquest usuari no pot accedir a segons quina pàgina degut a que necessita realitzar un seguit d'accions a la web per accedir-hi a obtenir el resultat final.

Aquests usuari va augmentant el nombre de casos d'ús que pot realitzar a mesurar que realitza un seguit d'accions i per tant, l'usuari podrà realitzar tots els casos d'ús un cop hagi realitzat les accions d'introduir fórmules i d'afegir les dades.

Per tal de poder entendre millor els casos d'ús de l'usuari s'ha procedit a la realització del diagrama de casos d'ús, on es pot comprovar que l'usuari va augmentat les accions a poder realitzar. Abans però, realitzarem una petita explicació de cada pas.

1. Primer pas: L'usuari pot realitzar les accions informatives, és a dir, les d'observar el *help* i mirar qui ha realitzat la pàgina web. També pot realitzar les accions d'afegir equacions, observar el graf que es va realitzant a mesura que va afegint les equacions i reiniciar els valors de l'aplicació.

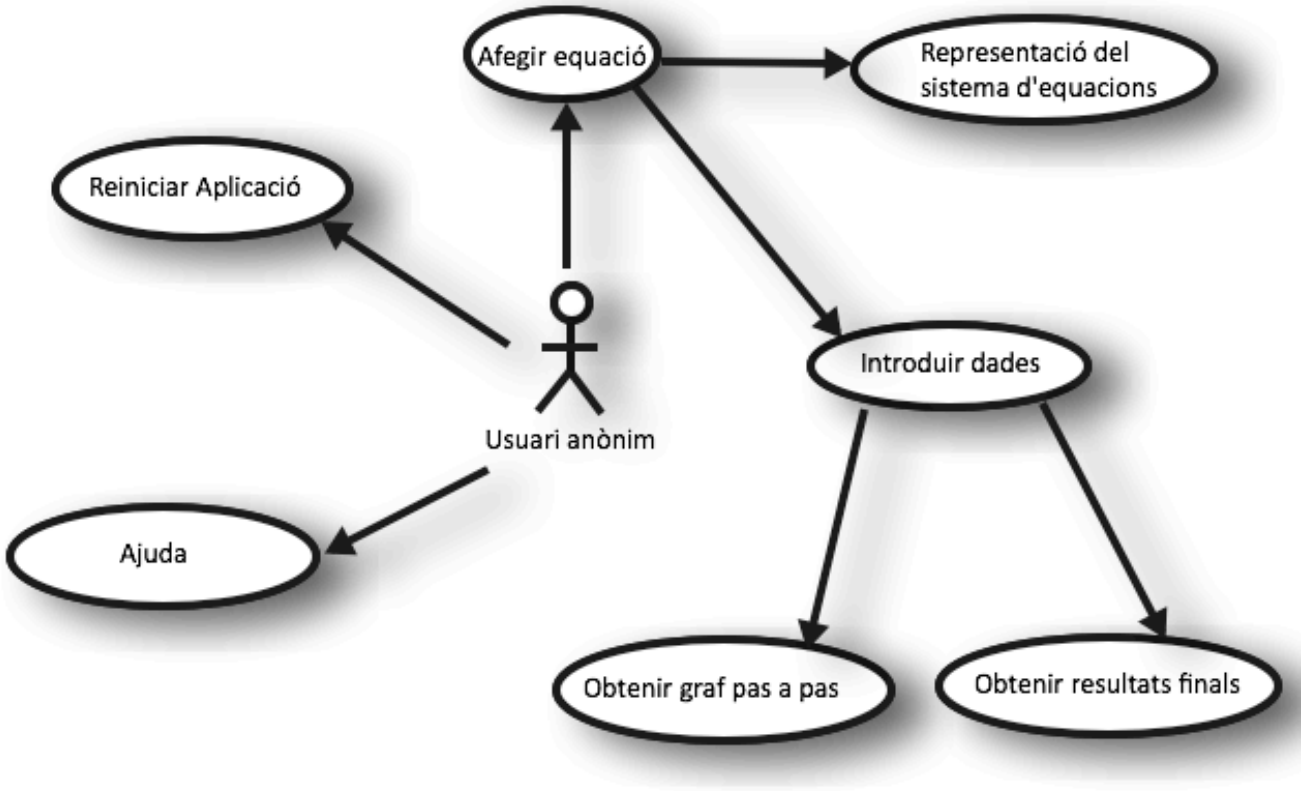
2. Segon pas: A mesura que l'usuari afegeix les equacions, l'aplicació detectarà les diferents variables de les equacions. Per tant, com l'aplicació ja ha detectat variables ja pot introduir dades, és a dir, dona valors a les variables i indicar quines variables en volem cercar el resultat.

3. Tercer pas: Com l'usuari ja ha introduït les equacions i les dades l'usuari ja pot realitzar:

    a. Ajuda
    b. Reiniciar l'aplicació
    c. Afegir equació
    d. Representació del sistema d'equacions
    e. Observar graf pas a pas
    f. Obtenir resultats finals
    g. Introduir dades

Cal comentar finalment que qualsevol usuari que realitzi l'acció de reiniciar l'aplicació passa automàticament a ser un usuari anònim bàsic.

## 9.3 Diagrama de cas d'ús




Il·lustració 9-1 Diagrama de cas d'ús.

## 9.4 Descripció dels casos d'ús

| **UC1. Observar pàgina funcionament** |
|---|
| **Objectiu:** Observar les pàgines informatives del funcionament i de qui ha realitzat la pàgina. |
| **Actor:** Usuari anònim bàsic, usuari anònim amb fórmules, usuari anònim amb fórmules i dades |
| **Pre-condicions:** ----- |
| **Procediment:**<br>1. Usuari sol·licita accedir a la pàgina inicial<br>2. El sistema processa al mostreig i mostra la pàgina principal<br>3. L'usuari sol·licita accedir a la pàgina funcionament<br>4. El sistema processa el mostreig i mostra la pàgina funcionament |
| **Procediment alternatiu:**<br>1. L'usuari sol·licita accedir a la pàgina funcionament des de qualsevol altra pàgina que tingui l'opció a la part superior.<br>2. Sistema processa el mostreig i mostra la pàgina funcionament |
| **Post-condicions:** L'usuari podrà visualitzar l'explicació informativa de com funciona l'aplicació web. |

| **UC2. Afegir fórmula** |
|---|
| **Objectiu:** Afegir una equació a l'aplicació |
| **Actor:** Usuari anònim bàsic, usuari anònim amb fórmules, usuari anònim amb fórmules i dades |
| **Pre-condicions:** ----- |
| **Procediment:**<br>1. Usuari sol·licita accedir a la pàgina inicial per poder afegir una equació<br>2. El sistema processa al mostreig i mostra la pàgina principal<br>3. L'usuari escriu l'equació a l'input i sol·licita afegir l'equació<br>4. El sistema processa l'equació i crea i dibuixa el graf, detectant les diferents variables de l'equació<br>5. El sistema mostra a la pàgina principal el graf dibuixat i l'equació afegida. |
| **Procediment alternatiu:**<br>4. El sistema processa l'equació i modifica el graf si l'usuari ja havia afegit anteriorment alguna altra equació.<br>5. El sistema mostra a la pàgina principal el graf modificat i totes les equacions afegides per l'usuari. |
| **Post-condicions:** L'usuari podrà visualitzar el graf creat per l'equació i totes les equacions afegides. |

| **UC3. Observar graf complert** |
|---|
| **Objectiu:** Visualitzar a través d'una imatge creada per un graf les diferents relacions de les equacions afegides per l'usuari. |
| **Actor:** Usuari anònim bàsic, usuari anònim amb fórmules, usuari anònim amb fórmules i dades |
| **Pre-condicions:** L'usuari ha d'haver passat per UC2. |
| **Procediment:**<br>1. El sistema li mostra a través de la pantalla principal el graf mitjançant una imatge on s'hi pot observar les diferents relacions de les equacions afegides i les variables detectades.<br>2. L'usuari observa el graf complert. |
| **Procediment alternatiu:**<br>1. L'usuari no ha passat per UC2.<br>2. El sistema mostra per pantalla un feedback indicant que té que afegir equacions a l'aplicació per tal de poder visualitzar el graf.<br>3. L'usuari es redirecciona a UC2. |
| **Post-condicions:** L'usuari podrà visualitzar el graf creat per l'equació i totes les equacions afegides. |

| **UC4. Reiniciar aplicació.** |
|---|
| **Objectiu:** Reiniciar les dades de l'aplicació web. |
| **Actor:** Usuari anònim bàsic, usuari anònim amb fórmules, usuari anònim amb fórmules i dades |
| **Pre-condicions:** ----- |
| **Procediment:**<br>1. Usuari sol·licita reiniciar les dades de l'aplicació.<br>2. El sistema reinicia als valors inicials les variables utilitzades en la lògica del projecte per poder realitzar a terme el funcionament de l'aplicació.<br>3. El sistema mostra la pàgina principal.<br>4. L'usuari passa automàticament a ser usuari anònim bàsic.<br>5. L'usuari visualitza la pàgina principal tal i com es mostra al iniciar la pàgina. |
| **Procediment alternatiu:** ----- |
| **Post-condicions:** L'usuari passa a ser usuari anònim bàsic i el sistema no té cap dada guardada de l'usuari. |

| **UC5. Introduir dades** |
|---|
| **Objectiu:** Indica al sistema els diferents valors coneguts de les variables i indicar al sistema quines són les variables que volem trobar el seu valor. |
| **Actor:** Usuari anònim amb fórmules i usuari anònim amb fórmules i dades |
| **Pre-condicions:** L'usuari ha d'haver passat per UC2. |
| **Procediment:**<br>1. L'usuari sol·licita accedir a la pàgina per introduir les dades.<br>2. El sistema processa l'acció i mostra la pàgina per introduir les dades.<br>3. L'usuari visualitzar les diferents variables detectades pel sistema.<br>4. L'usuari introdueix el valor de les variables que coneix i l'usuari indica quines variables són les que vol trobar-ne el valor.<br>5. El sistema processa l'informació.<br>6. L'usuari passa de ser usuari anònim amb fórmules a usuari anònim amb fórmules i dades.<br>7. UC6 i UC7. |
| **Procediment alternatiu:**<br>1. L'usuari sol·licita accedir a la pàgina per introduir les dades.<br>2. El sistema processa l'acció i no troba cap tipus de dada en el sistema.<br>3. El sistema mostra un missatge a l'usuari indicant que primer ha de passar per UC2.<br>4. El sistema mosta l'opció directa per anar a UC2. |
| **Post-condicions:** L'usuari passa a ser usuari anònim amb fórmules i dades on podrà visualitzar-ne els resultats finals i els passos a seguir per resoldre el sistema d'equacions establert al introduir les dades a l'aplicació. |

| **UC6. Observar graf pas a pas** |
|---|
| **Objectiu:** El sistema mostra per pantalla els diferents passos a seguir mitjançant imatges dibuixades a través de graf per poder resoldre el sistema amb les dades introduïdes per l'usuari. El sistema a més, mostra el graf complert inicialment. |
| **Actor:** Usuari anònim amb fórmules i dades |
| **Pre-condicions:** L'usuari ha d'haver passat per UC5. |
| **Procediment:**<br>1. L'usuari sol·licita visualitzar els resultats finals.<br>2. El sistema processa l'acció i calcula el resultat final.<br>3. El sistema mostra per pantalla el graf complert i les diferents imatges del graf per ordre de procediment per resoldre el sistema d'equacions.<br>4. L'usuari observar les imatges mostrades. |
| **Procediment alternatiu:**<br>1. L'usuari sol·licita visualitzar els resultats finals.<br>2. El sistema processa l'acció i però l'usuari o no ha introduït cap dada o el sistema no pot resoldre el sistema d'equacions.<br>3. El sistema no mostra cap imatge. |
| **Post-condicions:** L'usuari passa a ser usuari anònim amb fórmules i dades on podrà visualitzar-ne els resultats finals i els passos a seguir per resoldre el sistema d'equacions establert al introduir les dades a l'aplicació. |

| **UC7. Obtenir resultats finals.** |
| --- |
| **Objectiu:** El sistema mostra per pantalla els diferents passos a seguir mitjançant taules informatives per poder resoldre el sistema amb les dades introduïdes per l'usuari. |
| **Actor:** Usuari anònim amb fórmules i dades. |
| **Pre-condicions:** L'usuari ha d'haver passat per UC5. |
| **Procediment:**<br>1. L'usuari sol·licita visualitzar els resultats finals.<br>2. El sistema processa l'acció i calcula el resultat final.<br>3. El sistema mostra per pantalla les taules informatives de com resoldre el sistema d'equacions.<br>4. L'usuari observar les taules mostrades. |
| **Procediment alternatiu:**<br>1. L'usuari sol·licita visualitzar els resultats finals.<br>2. El sistema processa l'acció i però l'usuari o no ha introduït cap dada o el sistema no pot resoldre el sistema d'equacions.<br>3. El sistema no mostra cap taula. |
| **Post-condicions:** L'usuari passa a ser usuari anònim amb fórmules i dades on podrà visualitzar-ne els resultats finals i els passos a seguir per resoldre el sistema d'equacions establert al introduir les dades a l'aplicació. |

## 9.5 Requisits no funcional

Aquest són els requisits no necessaris pel funcionament de l'aplicació però si que en doner un alt valor al resultat final obtingut.
Bàsicament els podem classificar amb escalabilitat, accessibilitat i simplicitat.

1. Escalabilitat: El codi realitzat ha de mantenir un ordre. Aquest ordre s'aconsegueix encapsulant els mètodes segons les diferents funcions que realitza. D'aquesta manera el projecte pot tenir un creixement continu ja que cada funció esta realitzada en el seu mètode sent "independent" dels altres mètodes.

2. Accessibilitat: L'usuari ha de poder accedir fàcilment a l'aplicació web mitjançant els diferents navegadors web existents.

3. Simplicitat: L'usuari ha de poder entendre bé el seu funcionament d'una manera fàcil, és a dir, sense tenir problemes en l'ús de l'aplicació de tal manera que la corba d'aprenentatge no sigui molt elevada.

## 9.6 Servidor mínim per l'aplicació

Per tal de que l'aplicació no tingui problemes en la creació del graf i dibuix del graf el servidor ha de tenir un tractament de les imatges emmagatzemades.

Cada imatge creada pel graf és emmagatzemada al servidor per tal de poder-la mostrar. Amb les proves realitzades em pogut comprovar que de mitjana una imatge de l'aplicació ocupa 31KB i per cada sistema d'equacions el sistema crea una mitjana d'entre 6 i 7 imatges. Per tant, podem dir que un usuari utilitza 217KB.

Com podem comprovar estem parlant d'una capacitat força elevada i per tant, haurem de realitzar un tractament d'eliminació de les imatges.

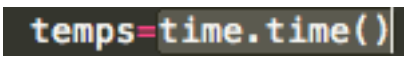
Primerament cal comentar que el sistema està preparat perquè elimini les imatges que ja no ens serviran a mesura que va creant noves imatges. El sistema guarda una llista de rutes, on podem accedir a totes les imatges.
Quan l'usuari reinicia l'aplicació, el sistema elimina automàticament les imatges que té accés mitjançant la llista de rutes, d'aquesta manera realitzem la primera neteja d'imatges del servidor.
També quan es sobreescriu una imatge, l'anterior és eliminada, com per exemple quan anem modificant el graf complert.
Per tant podem dir que tenim dos tipus de neteja d'imatges no utilitzables.

Tot i així, això no es suficient ja que anirien quedant residualment imatges al servidor. Per tal de que això no sigui un problema com a treball futur s'hauria d'implementar un mètode al servidor que cada X temps elimini les imatges més antigues de X temps. D'aquesta manera el sistema podria treballar amb comoditat ja que tindria espai i el servidor no s'ompliria d'imatges residuals. Això és molt important, ja que si el servidor està ple, l'aplicació no funcionarà.

Un cop realitzat un tractament per l'eliminació de les imatges residuals i sabent quina mida ocupen, s'hauria de buscar un servidor d'una mida suficient perquè no s'ompli.
Actualment, no es considera que l'aplicació tingués moltes visites i per tant, considerem que un servidor de 3 GB és suficient conjuntament amb els tractament mencionats anteriorment.

Un servidor de 3 GB són 1.048.576KB. Sabent que el projecte ocuparà uns 800KB, podem dir que tenim 1047776KB pel tractament de dades. Si dividim l'espai lliure obtingut(1047776KB) per la mitjana d'ocupació d'una imatge(31KB) obtenim que podem realitzar un total de 33.799 imatges. Suposant que un usuari deixa 7 imatges de rebuig poden interactua amb la web un total de 4828 usuaris amb el temps que nosaltres desitgem fer la neteja en el servidor.

Les imatges estan guardades segons la data en que es realitza en un valor molt aproximat al moment en que es creen i això facilita que no hi hagi conflictes de noms en les imatges del servidor.




Il·lustració 9-2 Funció que retorna el temps en un valor numèric.

Podem afirmar que un servidor de 3 GB és suficient per l'aplicació web en aquests moments.

# 10. Desenvolupament

Capítol on explicarem tota la part corresponent a la realització del codi i del treball final.

## 10.1 Entorns de desenvolupament utilitzats

En aquest punt explicarem els entorn utilitzats per el desenvolupament de l'aplicació web.

### 10.1.1 Servidor per execució en local

Per tal de poder executar el projecte, Django inclou un servidor interior. Aquest ens facilita poder treballar en local per tal de després poder-ho pujar a un servidor extern. Aquest servidor intern s'executa mitjançant terminal.
Per tal de poder executar el servidor ens em de moure per terminal fins l'arxiu manage.py de dins el projecte.

```
1  #!/usr/bin/env python
2  import os
3  import sys
4
5  if __name__ == "__main__":
6      os.environ.setdefault("DJANGO_SETTINGS_MODULE", "myproject.settings")
7
8      from django.core.management import execute_from_command_line
9
10     execute_from_command_line(sys.argv)
```

Il·lustració 10-1 Arxiu manage.py

Un cop ens trobem a la mateixa alçada que l'arxiu s'ha d'executar per terminal la comanda "python manage.py runserver". Aquesta inicia el servidor, la qual ens dóna algunes dades, com per exemple la direcció i el port del servidor.

```
Last login: Tue Jun 16 09:40:14 on ttys000
MacBook-MacBook-Pro-de-Joan:~ joangasullroyes$ cd resolution/wsgi/myproject/
MacBook-MacBook-Pro-de-Joan:myproject joangasullroyes$ ls
db.sqlite3      manage.py       myproject
MacBook-MacBook-Pro-de-Joan:myproject joangasullroyes$ python manage.py runserver

Performing system checks...

System check identified no issues (0 silenced).
June 17, 2015 - 17:37:52
Django version 1.7.5, using settings 'myproject.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

Il·lustració 10-2 Execució del servidor per terminal.

Com podem comprovar a l'imatge, a través de terminal s'ha anat fins al projecte(en el nostre cas es la carpeta *myproject*). Un cop ha dins s'ha comprovat que hi inclou l'arxiu manage.py. Un cop comprovat s'ha executat la comanda. Aquesta ens ha donat la ruta web que és http://127.0.0.1:8000/. Aquesta ruta és la que cridarem mitjançant els navegadors webs per tal de que es mostri l'aplicació. Per tancar el server només s'ha de teclejar CONTROL-C per terminal.

### 10.1.2 Navegador utilitzat

Per tal de poder comprovar que el treball realitzat és correcte és necessari un navegador on mostrar els arxius html perquè ens mostrin que el treball realitzat està funcionant perfectament. Els navegadors utilitzats han estat Google Chrome  Versió 43.0.2357.124 (64-bit) i Safari Versió 8.0.4 (10600.4.10.7). Els navegadors són una peça clau del projecte, ja que són ells els que ens mostraran l'aplicació i les dades a mostrar.

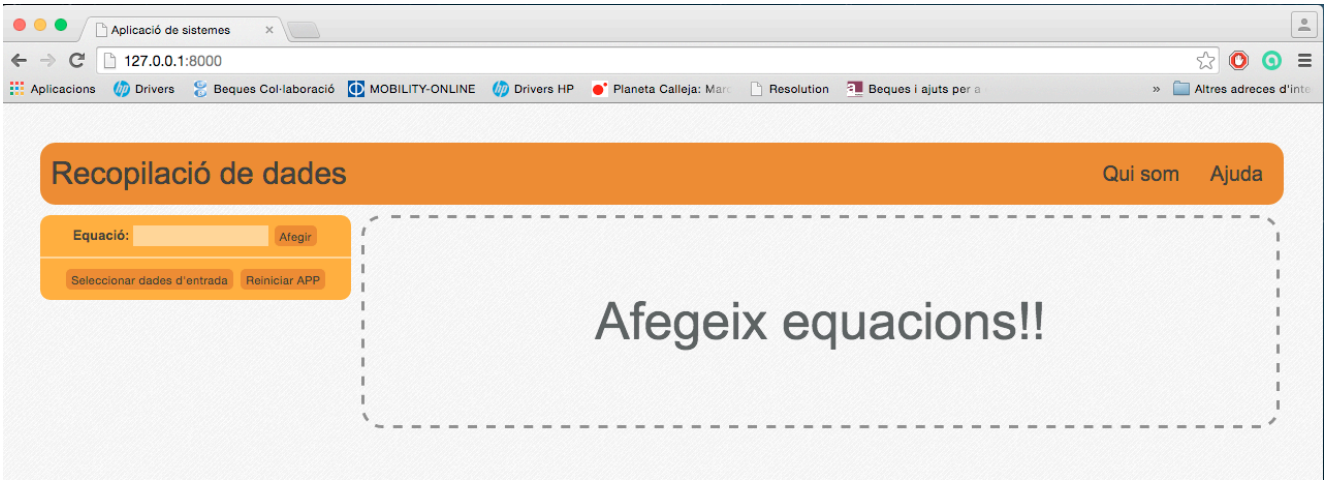
## 10.2 Modularitat

En aquest apartat explicarem la modularitat del projecte. El projecte es pot dividir en tres gran parts. La part gràfica, que inclou tot el desenvolupament de html perquè la pàgina web tingui una bona estructura. La part d'estructura de dades que inclou tot el tractament interior de dades, des de quan s'introdueix una equació fins que se'n dóna el resultat final. Finalment hi ha la part de visualització de les dades. Aquesta última part és la part més diferenciadora del projecte.

### 10.2.1 Interfície gràfica

La interfícies gràfica es tractada en 5 arxius. Aquest arxius es poden diferencia en els necessaris pel funcionament de l'aplicació i els informatius. Els necessaris pel funcionament de l'aplicació són inici.html, inputs.html i graf.html. Els arxius informatius son funcionament.html i quisom.html.

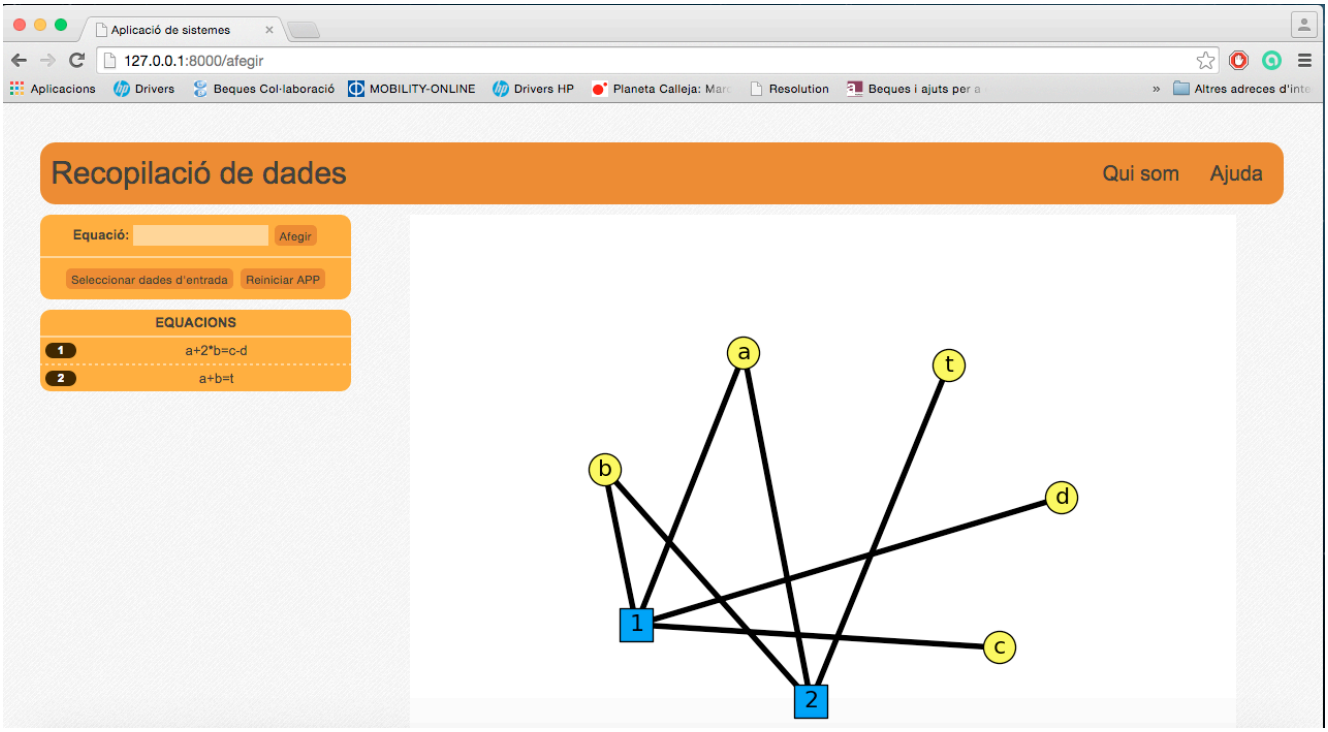
#### 10.2.1.1 Inici.html




Il·lustració 10-3 Pàgina principal

Aquest és l'arxiu que conté l'estructura inicial de l'aplicació. Com podem comprovar a l'imatge conté una capçalera per redirigir-nos a les pestanyes informatives. A més a més, aquest arxiu inclou l'entrada d'equacions per part de l'usuari. Aquestes són escrites a la part superior esquerra i afegides clicant el botó "Afegir". També podem observar un requadre que ens indica que hem d'afegir equacions. Aquest requadre és on es mostrarà el graf complert que s'anirà modificant a mesurà que anem afegint equacions.

Aquest és el primer punt de filtratge ja que no acceptar tot tipus d'equacions. Si detecta una equació amb un error no l'afegeix a la llista.




Il·lustració 10-4 Pàgina principal amb fórmules afegides i graf relacional de les equacions.

Un cop s'han afegit les equacions, tal i com podem comprovar a l'imatge, l'estructura ha canviat. A la part esquerra es van afegint les diverses equacions i a la part dreta es va creant el graf complert que inclou totes les equacions. A més a més les equacions estan numerades per tal de poder-les visualitzar millor al graf.
Aquesta pàgina també ens permet reiniciar l'aplicació posant tots els valors a zero i tornant a la captura inicial.
Finalment, també ens permet anar a la pàgina per introduir les dades, la qual si accedeix clicant al botó "Seleccionar dades d'entrada".

### 10.2.1.2 Inputs.html


Il·lustració 10-5 Pàgina d'introducció de dades.

Aquest arxiu és l'encarregat d'obtenir quines són les dades d'entrada i quines són les dades de sortida. Per tal d'obtenir les dades es mostraran a través d'una taula totes les variables detectades en les equacions introduïdes anteriorment, les quals se'ls hi podrà donar valor o indicar com a output. En cas

que l'usuari no hagi afegit cap equació se li mostrarà un text indicant que no ha afegit equacions amb l'opció de tornar a la pàgina principal.
A més a més, l'arxiu dóna l'opció de reiniciar l'aplicació o de tornar a l'usuari a la pàgina principal perquè segueixi afegint fórmules. L'arxiu conté també la capçalera superior per redireccionar-nos per l'aplicació.


Il·lustració 10-6 Pàgina d'introducció de dades amb les dades introduïdes. En aquest cas dos valors inputs i un output.

Si l'usuari ja ha introduït les dades i ha indicat quins són els outputs te l'opció "Agafar dades d'entrada" per tal de que l'aplicació comenci a resoldre el sistema.

En aquest punt passa el segon filtratge. Si no és te cap valor d'input o cap valor d'output no es podrà realitzar el càlcul.

#### 10.2.1.3 graf.html

Aquest és l'arxiu que mostra les dades finals. En aquest arxiu es mostra una taula amb totes les equacions introduïdes, així com les seves canòniques. La taula també pot mostrar algunes equacions creades per la mateixa aplicació per igualació en cas que l'aplicació ho trobi necessari per resoldre el problema.
També inclourà tots els passos a seguir per poder resoldre el problema. Aquest passos estaran ordenats i inclouran quina variable troba, quin és el seu valor, quina equació utilitza i l'operació a realitzar.
Finalment mostrarà cadascun d'aquests passos mitjançant un graf. Primerament mostrarà el graf complert, seguit de totes les imatges del graf, pas a pas, on només si apreciarà la part del graf complert utilitzada per a la resolució del sistema. Aquest arxiu és l'encarregat de mostrà les dades finals.

En aquesta pàgina és on se sabrà si s'ha resolt o no el sistema d'equacions. En cas que l'aplicació no hagi resolt el sistema d'equacions o només una part del sistema se li indicarà a l'usuari per pantalla que el sistema no s'ha resolt completament. Això pot succeir per dos opcions, o que l'usuari no ha introduït totes les dades o que el número de variables desconegudes és superior al número d'equacions introduïdes, i per tant, el sistema és un sistema d impossible .

En el capítol "estructura de dades" que trobem més endavant s'explica amb més detall cadascuna de les parts de la pàgina.

#### 10.2.1.4 quisom.html

Aquest arxiu és només informatiu. Dóna informació del projecte i qui la realitzat. No és necessari per l'ús de l'aplicació. S'hi accedeix a través del *header*.

#### 10.2.1.5 funcionament.html

Cinquè i definitiu arxiu. Aquest explica com funciona l'aplicació mitjançant explicacions escrites i captures de pantalla. Les explicacions estan explicades per passos i ajudades de diferents imatges que ajuden a entendre el funcionament de l'aplicació. S'hi accedeix a través del *header*. És el que es conèixer com *Help*(Ajuda).

### 10.2.2 Estructura de dades

L'estructura de dades és la part que tracta la lògica de l'aplicació web. En aquest punt explicarem els diferents arxius python del projecte, així com el seu funcionament i les diferents classes i funcions destacades.

#### 10.2.2.1 Setting.py

És l'arxiu que conté tota la configuració de Django. En ell guardarem la direcció del Host, les rutes on guardarem arxius i imatges, la ruta dels templates, el llenguatge de l'aplicació, la *timezone* i fins i tot la ruta de la base de dades. És imprescindible per l'execució del *framework*.

#### 10.2.2.2 Init.py

Arxiu buit que informa a python que és un paquet de python. Només té aquesta funció.

#### 10.2.2.3 Manage.py

És l'arxiu que ens permet interactuar amb el projecte Django.

#### 10.2.2.4 Urls.py

Arxiu que conté totes les rutes de l'aplicació.

```
urlpatterns = patterns('',
    url(r'^$', inici),
    url(r'^afegir$', afegir),
    url(r'^afegirInputs$', afegirInputs),
    url(r'^afegirFormula$', afegirFormula),
    url(r'^reinicia$', reinicia),
    url(r'^funcionament$', funcionament),
    url(r'^inici$', inici),
    url(r'^graf$', graf),
    url(r'^quisom$', quisom),
)
```

Ilustració 10-7 Codi principal arxiu urls.py

És l'encarregat de redireccionar la crida del navegador a una funció de l'arxiu views.py.

- $: Pàgina d'inici
- afegir$: Crida la primera funció per tal d'afegir una equació
- afegirInputs$: Crida la funció que mostra la pantalla d'obtenció dels valors tan de inputs com de outputs.

- afegirFormula$: Envia l'usuari a la pantalla d'inici perquè pugui afegir més fórmules.
- reinicia$: Crida la funció que posar tots els valors globals inicials i envia l'usuari a la pantalla principal.
- funcionament$: Crida la funció perquè mostri la pantalla informativa de com funciona l'aplicació.
- inici$: Pàgina d'inici.
- Graf$: Pàgina que mostra tota l'informació final. Mostra el resultat i el graf.

#### 10.2.2.5 Models.py

Arxiu on s'hi declaren les classes necessàries pel funcionament de l'aplicació. En el nostre cas tenim cinc classes.

```
class resultat():#Classe resultat
    """docstring for ClassName"""
    def __init__(self, var, valor, operacio,pas,equacio,numFor):
        self.var=var#variable
        self.valor=valor#valor de la variable
        self.operacio=operacio#operacio realitzada
        self.pas=pas#Numero de operacio feta
        self.equacio=equacio+"=0"#Equacio
        self.numFor=numFor#Numero de formula de la equqacio completa

class equacioCompleta():#Classe equacio
    """docstring for ClassName"""
    def __init__(self, equacio, canonica, num, tipus):
        self.equacio=equacio#Equacio
        self.canonica=canonica#Canonica
        self.num=num#Identificador
        self.tipus=tipus#Tipus d'equacio

class image():#Classe imatge del graf
    """docstring for ClassName"""
    def __init__(self, path, num, nom):
        self.path=path#Ruta
        self.num=num#Pas
        self.nom=nom#Nom del graf

class igualacio():#Classe imatge del graf
    """docstring for ClassName"""
    def __init__(self, equacio,numEquacio, equacio1, numEqua1, equacio2,numEqua2):
        self.equacio=equacio#Ruta
        self.numEquacio=numEquacio#Número d'equació
        self.equacio1=equacio1#Pas
        self.numEqua1= numEqua1#Número d'equació 1
        self.equacio2=equacio2#Nom del graf
        self.numEqua2= numEqua2#Número d'equació 2

class relacio():#Classe imatge del graf
    """docstring for ClassName"""
    def __init__(self, valor, canonica):
        self.valor=valor#Ruta
        self.canonica=canonica#Pas
```

Il·lustració 10-8 Classes utilitzades per l'aplicació

- Classe resultat. Obté com a paràmetres d'entrada la variable, el valor de la variable, l'operació realitzada i el número d'operació. També obté l'equació canònica i quin número de fórmula és. Aqueta classe ha estat creada per mostrar a la pàgina final les dades del resultat obtingut i així poder entendre millor el graf.
- Classe equacioCompleta. Classe amb la qual guardem l'equació, la seva canònica, el número que l'identifica i quin tipus és, si es una equació inicial o ha estat creada per igualació.
- Classe imatge. Classe que guarda la ruta on es troba el graf, quin número de graf és i quin nom li és donat. El nom ens indicarà si és el graf complert o quin pas és. El graf complert sempre es

troba a la posició 0 de l'array i anirem col·locant les diferents imatges obtingudes de l'array segons el pas que correspongui de resolució del projecte.

- Classe igualació: És la classe on es guarden els atributs que composen una igualació, és a dir, la nova equació creada per igualació i les dos equacions que ens permeten crear l'igualació.
- Classe relació: És una classe creada per ajudar en realitzar l'igualació. En ella té dos atributs, l'equació en si, i la part de l'equació que es pot igualar o la que no es pot igualar.

#### 10.2.2.6 Views.py

Arxiu on es troba tota la lògica del projecte. En aquest arxiu hi han totes les funcions necessàries per l'execució de l'aplicació web. Seguidament explicarem cadascuna de les funcions.

- funcionament(): Mostra la pàgina web informativa del funcionament de l'aplicació

- quisom(): Mostra la pàgina web informativa de la creació del projecte.

- afegir(): Funció que inicia mitjançant la lectura del post, l'obtenció de l'equació introduïda per l'usuari a l'aplicació, així com les seves variables, la canònica i el dibuix del graf, que pot ser l'inicial o el reestructurat en cas que ja existís un primer graf complert.

- afegirInputs(): Mostra la pàgina perquè l'usuari pugui introduir els valors de les variables que coneix i pugui indica quina variable busca.

- inici(): Mostra la pàgina d'inici de l'aplicació.

- afegirFormula(): Mostra la pàgina d'inici. Aquesta funció és cridada quan és clica el botó "Afegir fórmula" de la pàgina inputs.html.

- reinicia(): Funció que dóna els valors inicials a les variables globals de l'aplicació.

- Error404(): Funció que mostra una pantalla d'error en cas d'estar penjada la web a un servidor en mode no debug.

- graf(): Reinicia a els valors inicials les variables globals que necessita per tornar a dibuixar el graf. Borra en cas que existeixin imatges antigues del servidor del l'usuari concret. Agafa les dades de les variables i obté quines són outputs. Inicia la cerca del resultat del sistema d'equacions i en torna a dibuixar el graf complert. Ens envia a la pàgina graf.html per tal de poder visualitzar els resultats.

- dibuixarGrafInicial(): Mètode que dibuixa el graf amb les equacions introduïdes per l'usuari. Únicament mostra el graf en relació a les equacions introduïdes.

- dibuixarGrafInicialValors(): Mètode que dibuixa el graf amb les equacions introduïdes per l'usuari i els valors que ha introduït l'usuari. És el graf inicial amb els colors i formes corresponents.

- dibuixarGrafInicialIgualacio(): Funció que dibuixa el graf en cas que utilitzi el mètode d'igualació, adjuntant les variables desconegudes que s'utilitzaran per l'igualació en un única variable.

- dibuixarGrafInicialIgualacioValors(): Funció que dibuixa el graf anterior però donant color als nodes segons el seu valor. Si és un node variable conegut i el pinta de color verd, si no és un node variable conegut i el volem cercar de color vermell i si és un node variable desconegut però no el volem cercar de color groc. Els nodes equació els pinta de color blau.

- dibuixarGrafComplert(): Funció que mitjançant la llibreria NetworkX i Matplotlib dibuixa el graf, indicant els colors dels nodes i dels enllaços. En cas que ja hi hagi un graf complert dibuixat, n'elimina el graf anterior.

- dibuixarGrafComplertValors(): Funció que dibuixa el mateix graf que l'anterior però a més, els pinta de color segons el seu valor.

- dibuixarGrafIntermitg(): Realitza la mateixa funció que l'anterior però en aquest cas va dibuixant el graf segons els diferents passos que l'usuari haurà de realitzar. També és l'encarregat de dibuixar el graf final ja que dibuixa tots els passos i per tant, l'últim pas és el graf final.

- dibuixarGrafUnica(): Dibuixa un graf on es veu l'equació que s'agafarà per el pròxim pas marcant els enllaços discontinus.

- dibuixarGrafUnica2(): Mètode que dibuixa el graf final.

- buscaCheckOuts(): És el mètode que inicia la cerca dels resultats.

- reiniciarValors(): Mètode que encapsula l'opció de dóna els valors inicials a les variables globals. És cridat per la funció reinicia().

- agafarValors(): Mètode que agafar els valors donats per l'usuari a les variables així com quines són les variables a buscar. També estableix les variables que no són ni variables conegudes, ni les volem cercar.

- calcularVariable(): Mètode que inicia la cerca de la solució d'una variable.

- calcularCami(): Mètode que inicia la cerca de tots els possibles camins per trobar la solució.

- mirarNivell(): Mètode que calcula si l'equació és pot resoldre. En cas que es pugui resoldre crida el mètode resoldreEquacio() perquè ho resolgui. En cas que no pugui retorna un False.

- miremFormulesComu(): Mètode que inicia la cerca de la solució mitjançant igualació.

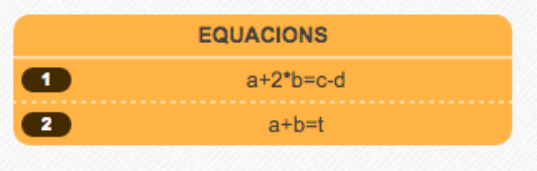
- reiniciarValorsPasAPas(): Mètode que reinicia les llistes necessàries per dibuixar el graf que mostrar el pas a realitzar.

- resoltEquacio(): Mètode que resolt l'equació mitjançant la llibreria Sympy. També és l'encarregat de crear el graf per tal de que després el puguem dibuixar. Per aquí passen tots els passos a seguir, és a dir, tan el final com els passos previs al final. És l'encarregat de cridar la funció per dibuixar el graf, ja sigui de passos previs o el final.

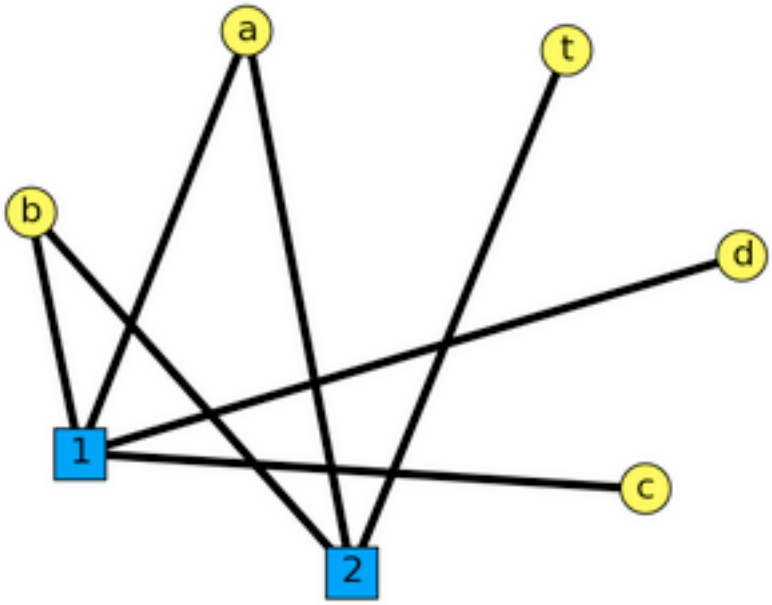
- reanudarGraf(): Mètode que torna a crear el Graf degut a que quan es canvia de pàgina el graf es borra. Aquest mètode es cridat en la funció graf().

- separacio(): Separa la part dreta de l'esquerra d'una equació.
- canonica(): Mètode que crea l'equació canònica
- canonicaEdge(): Mètode que crea l'equació canònica d'un enllaç que és una fórmula.
- Invertida(): Mètode que inverteix una funció canònica i la retorna.
- canviarSigne(): Mètode que retorna el signe contrari.
- creemNovesFormules(): Mètode que crear noves formules per igualació. En cas que trobi noves fórmules, és l'encarregat d'afegir-les al graf i crear-lo de nou.
- modificarVariablesIgualacio(): Mètode que crea la variable nova funció que englobarà les variables desconegudes
- modificarGraf(): Mètode que elimina del graf les variables que han estat utilitzades per fer la igualació
- variablesIndividualsNoves(): Mètode que crea variables segons les variables que desconeixem. Introdueix totes les variables que desconeixem a l'esquerra o a la dreta i d'aquesta manera pot crear noves fórmules per igualació.
- multiplicacions(): Mètode que controla que si l'usuari no posar el símbol *, la fórmula sigui modificada amb el símbol * dins la fórmula. Això es degut a que la llibreria Simpy no ho detecta com una multiplicació i així evitem errors.
- multiplicacioInvertida(): Té la mateixa funció que l'anterior però en aquest cas el número es troba després de la variable.
- variablesIndividuals(): Mètode que extreu les variables de les fórmules introduïdes per l'usuari.
- Classe equació(): Conté totes les llistes, *Strings*, booleans i Grafs necessaris per a poder dur a terme l'aplicació.

Un cop explicats tots els mètodes es necessari entendre el recorregut que realitzarà per dins l'aplicació per poder obtenir el resultat final. Partint de que l'usuari introdueix les equacions a la pàgina inicial, en aquest punt l'usuari ha d'introduir totes les equacions que te al seu abast, és a dir, totes les equacions que puguin ajudar a obtenir un resulta final. Pot donar-se el cas que algunes equacions introduïdes no s'utilitzin, es per això que es sol·licita a l'usuari que les introdueixi totes, ja que d'aqueta manera podem cerca tots els possibles camins i a més, informar de quines equacions no són necessàries per la resolució del sistema.

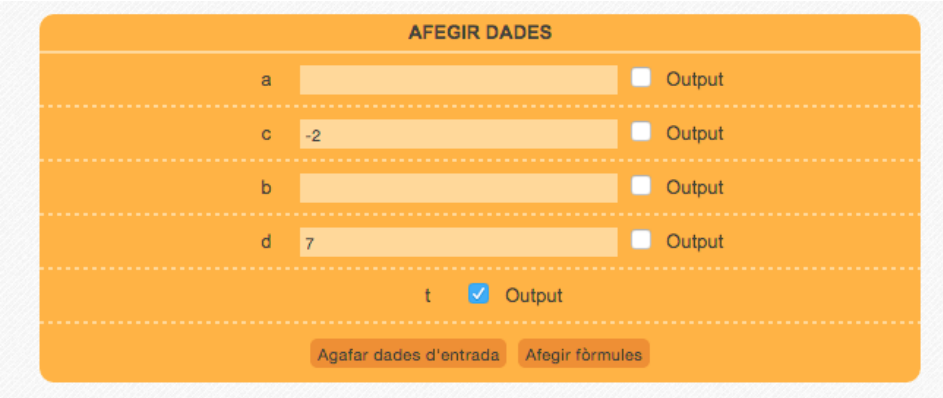



Il·lustració 10-9 Equacions introduïdes per l'usuari




Il·lustració 10-10 Graf que mostra les relacions de totes les equacions introduïdes per l'usuari. Graf inicial sense valors.

Per cada equació introduïda se n'obtindrà les diferents variables que composen l'equació, creant un graf amb les diferents variables i les diferents equacions. Cada variable pot trobar-se enllaçada amb diferents equacions, tal i com es pot observar en l'imatge, ja que les variables mantindran enllaç amb totes les equacions que les continguin. Cada equació només estarà enllaçada amb les variables que la composen.




Il·lustració 10-11 Variables detectades segons les equacions introduïdes per l'usuari. A més l'usuari ja ha marcat quines variables vol cercar.

Un cop s'ha obtingut totes les variables i totes les equacions, l'usuari haurà d'introduir mitjançant l'aplicació els valors de cada variable que coneix, tal i com es mostra a l'imatge superior. Això es produirà a la pàgina web inputs.html on es mostraran totes les variables detectades. Per tal de conèixer quines de les variables detectades són les que volem trobar, l'usuari haurà de seleccionar un *checkbox*, que ens indicarà que és output.

Ara ja tindríem totes les dades necessàries per intentar resoldre el sistema. Per cada variable que volem trobar valor realitzarem el següent camí.

- Primerament és mirarà de solucionar mitjançant una única equació, és a dir, una incògnita en una equació. Es mirarà totes les equacions enllaçades amb la variable que busquem i comprovarem per cada equació, si tenim totes les altres variables de l'equació menys la variable que busquem. En cas que tinguem totes les altres variable de l'equació es solucionarà l'equació donant valor a la variable que busquem.
- En cas que no haguem pogut donar valor mitjançant aquest camí, mirarem dins les equacions si ens trobem amb equacions de nivell 2. És a dir, mirarem si en elles únicament hi ha una variable que desconeixem. Si és el cas intentarem donar valor a aquesta variable que desconeixem segons els enllaços que te i mitjançant el mètode inicial de resoldre l'equació amb tots els valors menys una incògnita. Buscarem en els seus enllaços si té alguna equació on únicament la variable que desconeixem és l'incògnita. En cas que si es resoldrà i tornarem a mirar de resoldre el sistema ja que ara si la variable que busquem inicialment tindrà una equació on només ella serà l'incògnita. En cas que no es pugui resoldre mitjançant substitució es passarà al següent pas.
- S'intentarà crear noves equacions. Aquestes equacions seran creades mitjançant igualació. El procediment a seguir es comprovar totes les possibles equacions, separar-ne la part que coneixem de la part que desconeixem(segons els valors inicials donats per l'usuari) i mirar si podem realitzar una nova equació. En cas que si es tornarà a intentar resoldre el sistema d'equacions des del principi.

Cada vegada que que es troba una variable, ja sigui una variable buscada o una variable que ens ajuda a resoldre el sistema d'equacions es dibuixarà un graf. Això es realitza perquè d'aquesta manera podem tenir visualment cadascun dels passos realitzats per a poder resoldre el problema. Aquests passos es mostraran per ordre a la pantalla de mostreig de dades.
També cal comentar que cada vegada que es crea una fórmula nova, aquesta es afegida al graf. A la pantalla de mostreig de dades s'indicaran totes les fórmules noves creades i quines han estat utilitzades.
Finalment en cas de que hi hagi més d'una variable a busca es realitzarà el mateix procediment.

Ens podem trobar el cas que busquem més d'una variable. Si es així pot ser que el fet de trobar-ne una ens permeti solucionar un altre sistema d'equacions. És per això que hi ha un control lògic de que cada vegada que es troba una variable s'hagin d'intentar calcular totes les altres variables a buscar. D'aquesta manera mirem d'abastir tots els possibles camins per resoldre el sistema d'equacions.

Podem visualitzar pas a pas els moviments lògics al diagrama de flux lògic que trobem més endavant.

### 10.2.3 Visualització de dades

La visualització de les dades és la part més diferenciable del projecte respecte els seus competidors. Podem diferencia la visualització en dos parts. La visualització en taules i la visualització en imatges.

**Visualització en taules.**

En el mostreig de dades finals s'hi pot observar dos taules.

En la primera taula podem observar totes les equacions, tan les afegides inicialment per l'usuari com les creades mitjançant igualació. En elles s'hi mostra l'equació, el seu identificador, la equació canònica de l'equació i quin tipus és, és a dir, si és una equació inicial introduïda per l'usuari o una equació creada per igualació.

| EQUACIONS OBTINGUDES | | | |
|---|---|---|---|
| Identificador equació | Equació | Canònica | Tipus |
| 1 | a+2*b=c-d | c-d-a-2*b=0 | Equació inicial |
| 2 | a+b=t | t-a-b=0 | Equació inicial |
| 3 | c-d-2=t | t-c+d+2=0 | Igualació |
| 4 | -t=-c+d+2 | -c+d+2-+t=0 | Igualació |

Il·lustració 10-12 Taula de totes les equacions obtingudes.

En la segona taula s'hi mostren els passos que ha de realitzar l'usuari per a poder solucionar el sistema d'equacions ordenats per l'ordre d'execució del sistema. A la taula s'hi pot observar el número de pas que és, quina variable ha trobat, quin és el resultat i quina equació ha fet servir, mostrant l'identificador de la fórmula que utilitza.

| SOLUCIONS FINALS | | | |
|---|---|---|---|
| Pas | Variable | Resultat | Identificador equació |
| 1 | t | (-11) | 3 |

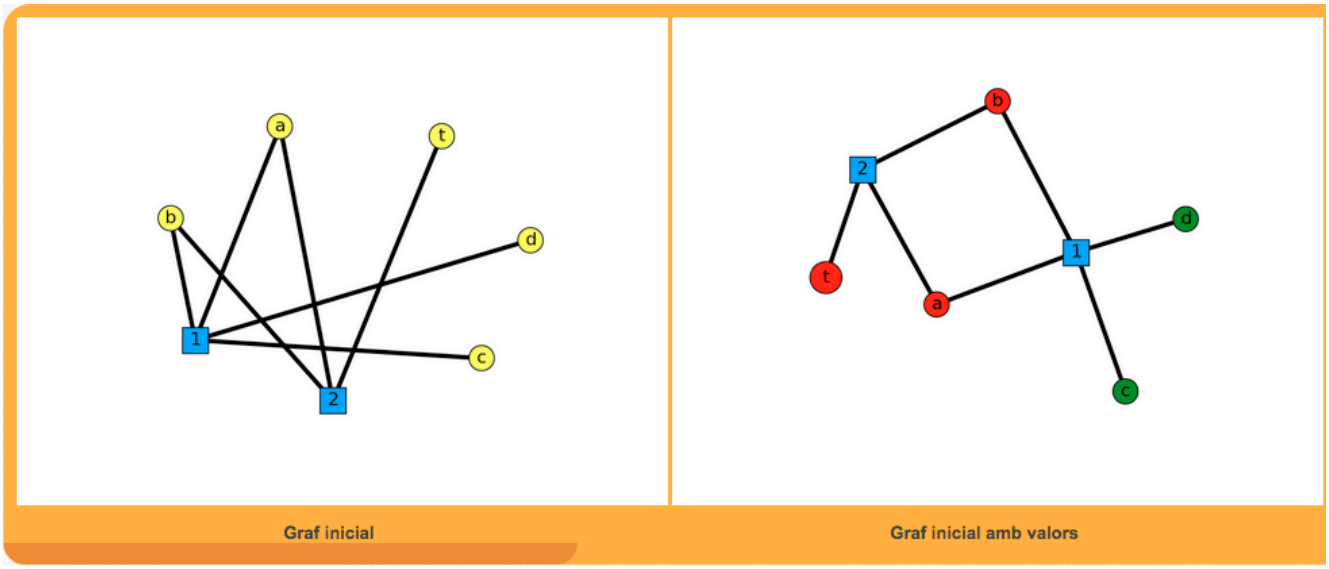
Il·lustració 10-13 Taula que mostra els resultats obtinguts.

**Visualització mitjançant imatges.**

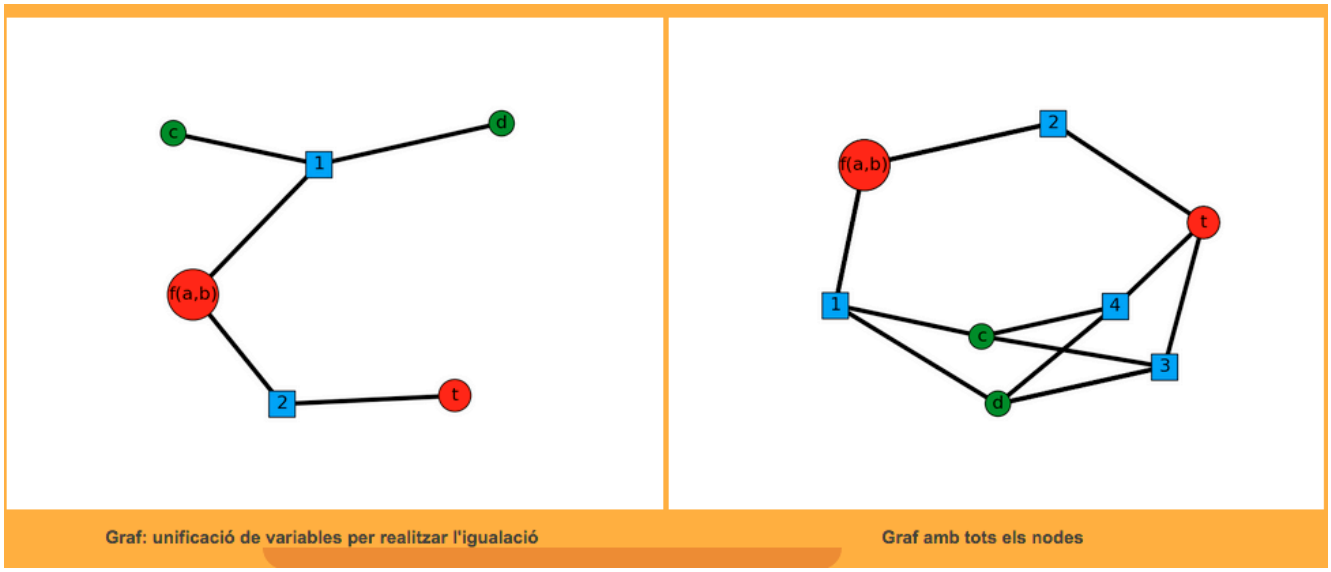
Tal i com s'ha comentat a l'apartat d'enfoc, per tal de que l'usuari pugui entendre les imatges s'ha realitzat una llegenda utilitzant els colors com a base per facilitar l'aprenentatge de com solucionar el sistema. La llegenda utilitzada és la mostrada en l'imatge inferior.


Il·lustració 10-14 Llegenda aplicada als grafs.

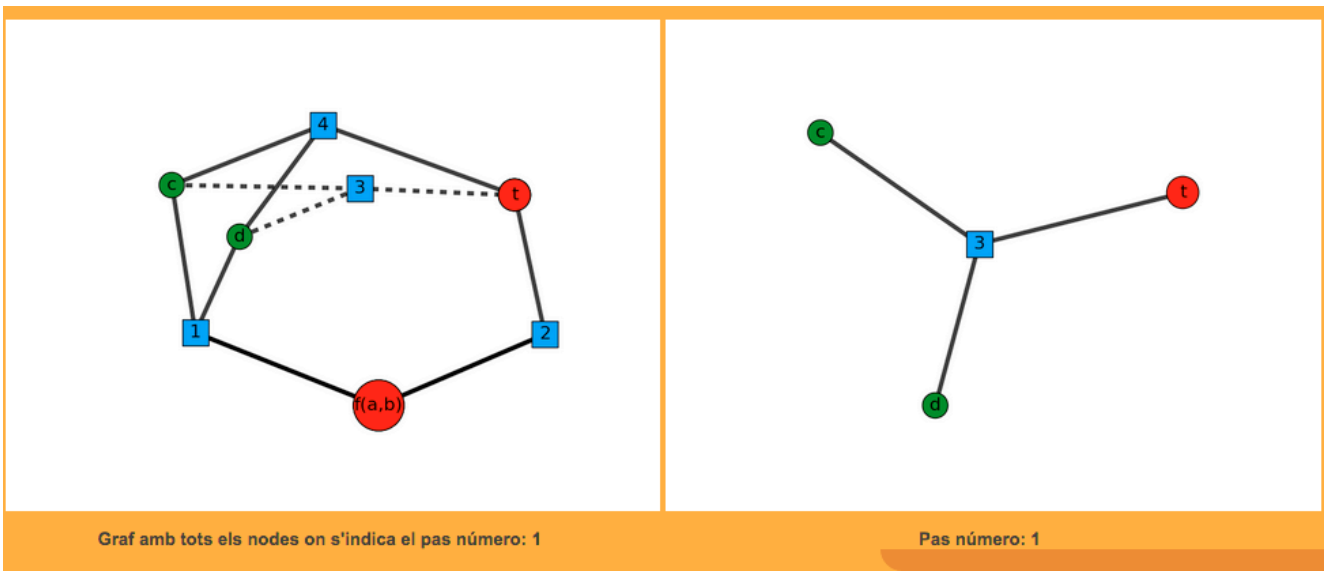
En el mostreig de dades finals s'hi pot observar un *scroll* horitzontal (imatges inferiors) on s'hi aprecia totes les imatges pas a pas. En ell podem observar un títol de cada imatge. El títol ens indica si és el graf inicial, en quin pas ens trobem de la resolució del sistema d'equacions, etc. Cada una de les imatges que mostren el pas es troben relacionades amb el pas indicat a la taula superior. A continuació es mostren totes les imatges per l'exemple mostrat m'entres es realitzava l'explicació.




Il·lustració 10-15 En aquesta imatge es mostra el graf inicial i el graf inicial amb els nodes indicant que significa, segons la llegenda mostrada anteriorment, cada node segons el color i forma.




Il·lustració 10-16 En aquesta imatge es pot observar el graf amb la variable nova(f(a,b)) creada de dos variables que desconeixem i els valors de les diferents variables. En la següent imatge es mostra el graf complert amb tots els nodes possibles.




Il·lustració 10-17. En l'imatge de la esquerra s'indica quina equació ha triat l'aplicació per resoldre el sistema marcant els enllaços amb discontinua i en la de la dreta únicament l'equació escollida amb els seus valors, indicant clarament quines variables coneix i quines no. A més podem comprovar que t és un output degut a que el node és lleugerament més gran.

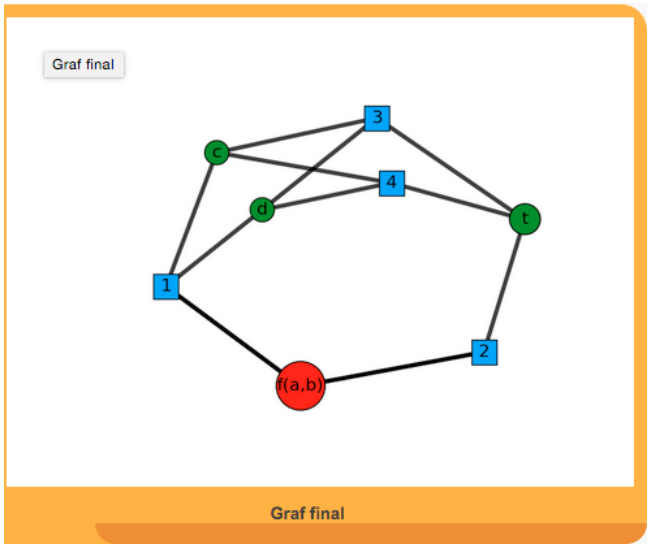
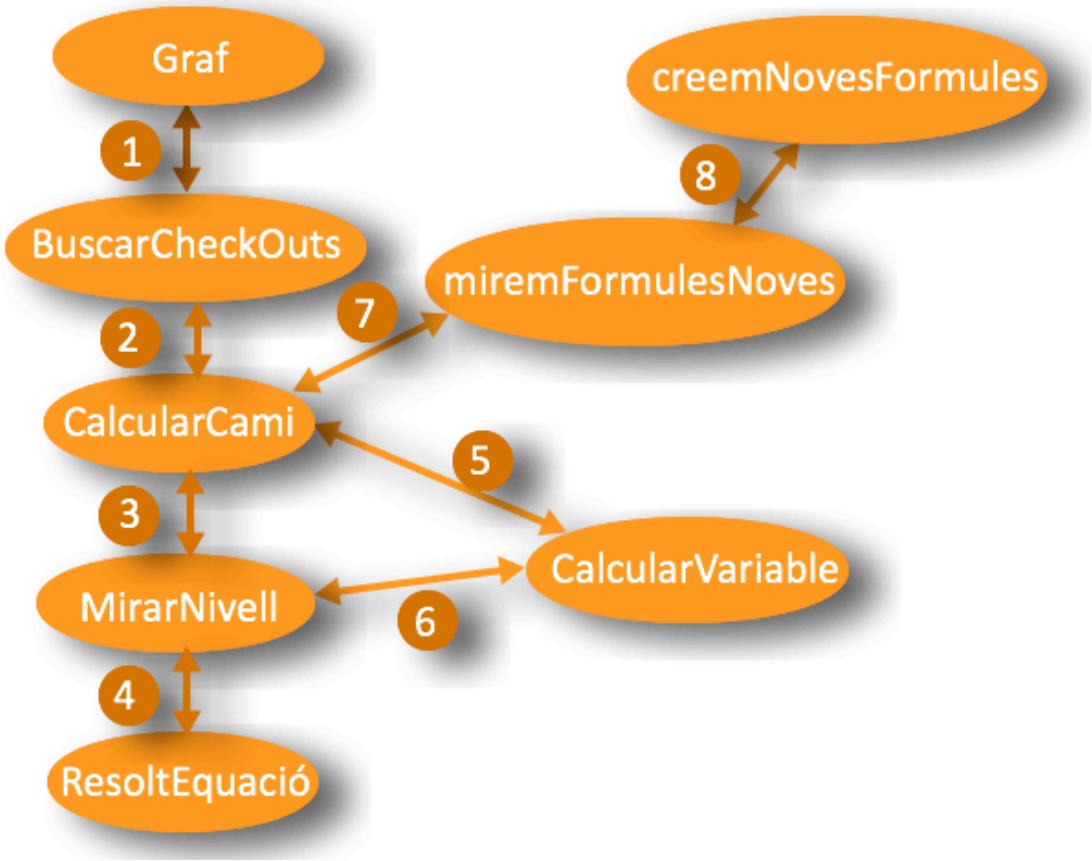



Ilustración 10-18 Graf final. Mostreig del graf amb els valors, indicant les variables conegudes i desconegudes, on si s'ha resolt, les variables que buscaven estan de color verd(conegudes).

## 10.3 Flux habitual

Per tal de poder entendre millor el funcionament lògic s'ha realitzat un diagrama de flux, el qual explicarem pas a pas. Aquest diagrama engloba els mètodes utilitzats per a la resolució del sistema d'equacions.




Il·lustració 10-19 Diagrama que mostra el recorregut lògic del programa realitzat.

1. El mètode graf() agafar les dades introduïdes per l'usuari mitjançant el POST del request i realitza la crida de busca les solucions del sistema mitjançant el mètode BuscarChechOuts(), passant com a paràmetre el número de variable a buscar. En ell li passa la llargada
   És simplement un encapsulament dels mètodes per tenir més escalabilitat. Un cop BuscarChechOuts finalitzi és l'encarregat de tornar a dibuixar el graf complert.

2. El mètode BuscarChechOuts() crida al mètode CalcularCami(), en el que li passa la variable a cercar, i per tant, es crida per cada variable que ha de resoldre. És l'encarregat de busca tots els possibles camins de les variables que volem trobar m'entres no haguem trobat la solució.

3. És el pas en que el Mètode CalcularCami() cerca els possibles camins d'una variable en concret. Inicialment comprova mitjançant MirarNivell(), on si passa l'equació que ha de ser el camí i la variable, si la variable es pot resoldre en una única equació. En cas que si, passa al pas 4, en cas que no passa al pas 5.

4. Si ens trobem en aquest punt és que ha succeït el cas més fàcil. Es quan la variable que busquem té una equació enllaçada on només la variable desconeguda és la variable que

busquem i per tant, es pot trobar valor per la variable que busquem. En aquest cas, es crida al mètode resoltEquacio() perquè resolgui l'equació, passant per paràmetre la variable a resoldre i en quina equació es pot resoldre. En el mètode resoltEquacio() es crea l'imatge del pas corresponent i es dona valor a la variable.

5. En aquest cas ens trobem que a la variable no li podem donar valor únicament amb una equació. És el moment de passar a l'opció de mirar tots els possibles camins de la variable que cerquem i observar si en elles s'hi troba una equació on hi hagi dos incògnites, la variable que busquem i una altra variable. En cas que sigui així ens trobem amb una equació on s'hi resolem la variable que desconeixem podem trobar la solució, i per tant, iniciarem la cerca per donar valor a la variable. Això es realitza passant per paràmetre la variable que busquem i la variable que ens pot servir de camí. En aquest moment es passa al pas 6. En cas que això no es pugui realitzar es passa al pas 7.

6. Com hem trobat una equació on si donem valor a la variable que desconeixem, podem resoldre el sistema, es realitza el mateix procediment que si es tractes d'una variable inicial a buscar. És com tornar al pas 3 però ara intentant donar valor a una variable que inicialment no necessitàvem donar valor. Si podem donar valor a la variable resoldrem l'equació indicant primerament el pas de trobar la variable que desconeixíem i segon indicant que hem donat valor a la variable que volíem buscar. En aquest cas obtindrem dos imatges que ens indicaran cadascun dels passos.

7. Si ens trobem en aquest pas vol dir que no hem pogut donar valor a la variable que trobem ni per una equació simple on només hi hagi l'equació com incògnita ni per el sistema mencionat anteriorment. És per això que en aquest pas s'intentarà donar valor mitjançant el mètode d'igualació. Per tal de poder fer l'igualació mirarem quines equacions són les candidates per igualar. Si es així passarem al pas 8. En cas que no hi hagi candidates indicarem que no hem pogut crear noves equacions i la variable no tindrà solució.
   Per passar al pas 8, passarem de paràmetres les variables que desconeixem i la variable que cerquem per tal de mirar quines equacions tenen en comú i així poder crear l'igualació.

8. En aquest punt rebem un llistat amb les equacions on surten variables desconegudes en comú, les variables desconegudes i la variable que cerquem. Un cop tinguem les equacions candidates per realitzar l'igualació, passarem equació per equació totes les incògnites que desconeixem a l'esquerra i les que coneixem a la dreta. S'hi ens trobem que dos parts esquerres de diferents equacions són iguals voldrà dir que podem igualar les seves respectives dretes. En aquest punt retornarem que hem pogut realitzar noves equacions i es tornarà a iniciar el procés.

# 11. Test

Dins el projecte realitzat s'ha tingut que realitzar diferents proves segons l'objectiu a seguir, proves per el funcionament lògic i proves pel funcionament del disseny.

## 11.1 Proves lògiques

Dins de les proves lògiques s'ha realitzat un seguit de proves per el funcionament de l'aplicació. Les podem separar en dos tipus de proves. Les proves pel funcionament de mètode i les proves pel funcionament de l'aplicació.

### 11.1.1 Proves funcionament del mètode

Per cada mètode realitzat s'ha realitzat diferents testos per comprovar que el mètode funciona perfectament i un cop el mètode funcionava perfectament es procedia a realitzar la prova de l'aplicació.

### 11.1.2 Proves pel funcionament de l'aplicació

Per les proves del funcionament de l'aplicació s'han utilitzat diferents sistemes d'equacions que van augmentant la seva dificultat. Els sistemes utilitzats són:

| **Prova 1- Equació simple** |
|---|
| **Equacions**: $a + b = t$ |
| **Variables amb valor**: $a$ i $b$ |
| **Variables que busquem**: t |
| **Altres variables**: **----** |
| **Explicació**: Una equació de primer grau i una incògnita |

| **Prova 2- Equació doble** |
|---|
| **Equacions**: $a + b = t$, $a+b=c-d$ |
| **Variables amb valor**: $b$, $c$ i $d$ |
| **Variables que busquem**: $t$ |
| **Altres variables**: $a$ |
| **Explicació**: Dos equacions de primer grau i una incògnita on primer s'ha de trobar la variable a per tal de poder trobar la variable t ja que la variable que busquem(t) es troba en una equació amb dos incògnites. |

| **Prova 3- Equació triple amb doble output** |
|---|
| **Equacions**: $a + b = t$, $a+b=c-d$, $r+a=t$ |
| **Variables amb valor**: $b$, $c$ i $d$ |
| **Variables que busquem**: $t$ i $r$ |
| **Altres variables**: $a$ |
| **Explicació**: Tres equacions de primer grau i una incògnita on primer s'ha de trobar la variable a per tal de poder trobar la variable t. Un cop trobada la variable a i t podrem trobar la variable r. |

| **Prova 4- Equació doble per igualació** |
|---|
| **Equacions**: *a + b = t, a+b=c-d* |
| **Variables amb valor**: *c* i *d* |
| **Variables que busquem**: *t* |
| **Altres variables**: *a* i *b* |
| **Explicació**: Dos equacions de primer grau que es poden resoldre pel mètode d'igualació. |

| **Prova 5- Equació triple amb doble output(1 de resoluble)** |
|---|
| **Equacions**: *a - b = t, a+b=c-d, a+b-r=c+d* |
| **Variables amb valor**: *c* i *d* |
| **Variables que busquem**: *t* i *r* |
| **Altres variables**: *a* i *b* |
| **Explicació**: Tres equacions de primer grau i dos incògnites. Tot i així, només es pot resoldre r per igualació. Útil per comprovar el sistema d'igualació i la no resolució d'una variable. |

| **Prova 6- Equació triple amb doble output(2 de resoluble) i una equació no utilitzada** |
|---|
| **Equacions**: *a - b = t, a+b=c-d, a+b-r=c+d,r-d=t* |
| **Variables amb valor**: *c* i *d* |
| **Variables que busquem**: *t* i *r* |
| **Altres variables**: *a* i *b* |
| **Explicació**: Similar a la prova 5 però en aquesta gràcies a l'última equació resolem tot el sistema. És la prova més complerta. |

# 12. Conclusions

Primerament voldria iniciar explicant que els objectius inicials s'han assolir. L'aplicació web resolt sistemes d'equacions de primer grau t'han pel mètode de substitució com pel mètode d'igualació mostrant en detall cadascun dels passos a seguir per poder resoldre el sistema mitjançant imatges creades amb el graf.
Per poder assolir els objectius s'ha necessitat un mínim de base tan de treball lògic com de treball de disseny que s'ha anat realitzant en diferent iteracions setmanals o bisetmanals, on es marcaven els objectius.
Dins dels nostres objectius també hi havia realitzar una aplicació web robusta, escalada i simple. Les tres característiques es compleixen dins el projecte realitzar. A més, els tests creats per tal de poder considerar que s'han assolit els objectius han passat satisfactòriament. Durant la realització del projecte, a més, he pogut assolir una de les meves motivacions, l'aprenentatge del funcionament d'un framework web, en el nostre cas, Django, molt utilitzat en diverses empreses del sector.

En el transcurs de les diferents iteracions han aparegut idees noves per millorar l'aplicació com per exemple la creació de dos llegendes segons la forma i el color o la creació més especifica de cadascun dels passos a seguir per resoldre el sistema d'equacions introduït. Aquestes idees noves han enriquit t'han el codi com el disseny, proporcionant una aplicació final amb més qualitat.

Finalment, com a conclusió personal comentar que crec en l'encert de realitzar una aplicació web enlloc d'una aplicació per dispositiu mòbil, ja que l'aplicació web ens proporciona millors característiques de treball que ens faciliten la tasca, com són per exemple les llibreries esmentades en el treball. A més, com l'objectiu principal és l'aprenentatge es considerà que l'accés principal serà un ordenador personal.
Tot i així, encara que no estaven dins dels nostres objectius, l'aplicació conté unes limitacions no assolides explicades en el següent capítol (treball futur).

# 13. Treball futur i millores

Som conscients que l'aplicació encara es pot millorar, és per això que ara mostrarem un seguit de punt que considerem que són les pròximes millores a realitzar a l'aplicació web.

- Augmentar els mètode per trobar la solució. Seria fantàstic poder realitzar la part lògica per poder resoldre un sistema d'equacions mitjançant el mètode de Gauss o el mètode de reducció.

- Donar l'opció a l'usuari d'afegir equacions que continguin parèntesi, poden realitzar la lògica perquè l'aplicació resolgui sense cap problema el sistema d'equacions.

- Graf fix. Que el mostreig de dades les diferents variables sempre es trobin en la mateixa posició de l'imatge, fet que facilitaria encara més a l'usuari poder entendre com s'ha resolt el sistema d'equacions.

- Poder arribar a resoldre sistemes d'equacions de segon grau.

- Tractament de decimals. L'aplicació no esta preparada per tractar equacions amb decimals. És per això que considerem aquesta millora com una de les primordials per a continuar el projecte.

- Millora de la interfície gràfica en la pantalla "Ajuda".

## 14. Agraïments

A la Mariella Dimiccoli, com a tutora del Treball Final de Grau. Sobretot per les hores invertides en mi i el meu treball. En l'esforç mostrat perquè quedes un projecte fet i dret i donar-me coneixements per millorar constantment.

Als meus millors amics, tan els que coneixes des de petits com aquells grans amics que es fan a la universitat(UBosses) pel suport donat, sobretot en aquells moments que sembla que mai es podran solucionant i que amb paciència i esforç s'acaben realitzant.

Als meus familiars per ajudar-me a poder arribar fins aquí.

I no voldria acabar aquest projecte sense agrair als que han fet possible que es puguin entregar beques a aquells estudiants que les necessiten i així poder realitzar els seus estudis. Un país no va endavant s'hi no hi ha educació.

## 15. Biblioteca

[1] Nom: numberempire, 2015
Direcció web: http://es.numberempire.com/equationsolver.php

[2] Nom: SolveMyMath, 2015-2011
Direcció web: http://es.solvemymath.com/calculadoras/algebra/resolver_ecuaciones.php

[3] Nom: Quickmath
Direcció web: http://www.quickmath.com/

[4] Nom: Wolfram Alpha LLC—A Wolfram Research COmpany, 2015
Direcció web: http://www.wolframalpha.com

[5] Nom: Mathway, 2015
Direcció web: http://www.mathway.com

[6]Nom: Networkx tutorial de Aric Hagberg, Dan Schult, Pieter Swart, 2012
Direcció: https://networkx.github.io/documentation/latest/_downloads/networkx_tutorial.pdf

[7]Nom: Networkx Developers, 2014
Direcció web: https://networkx.github.io/documentation/latest/index.html

[8] Nom: Matplotlib development team, 2015
Direcció: http://matplotlib.org/

[9] Nom: Sympy
Direcció: http://www.sympy.org/