Ajuntament de Barcelona

Read before project :)

(PHP Edition)

Marc 2022

Barcelona Activa

Unió Europea
Fons Europeu
de Desenvolupament Regional

Índex

Introducció

Fins ara hem treballat amb:

- Llenguatges de presentació de continguts: HTML i CSS.
- Llenguatges de programació orientat a objectes: PHP
- Bases de dades:
  - SQL
  - NoSQL(MongoDB)

Introducció

Ho hem treballat de forma separada, sovint amb fitxers independents.

Ara és l'hora de combinar-ho tot per tal de crear projectes d'un abast més gran.

I per això, és necessari familiaritzar-nos amb sistemes de fitxers que ens permetin organitzar tots els recursos necessaris per dur a terme aquestes fites.

Quan utilitzem el *framework* Laravel, ens adonem que ens proposa una estructura de carpetes concreta. A l'exercici "*Developers Team"* simulem una estructura de carpetes similar per anar acostumant-nos a aquesta idea.

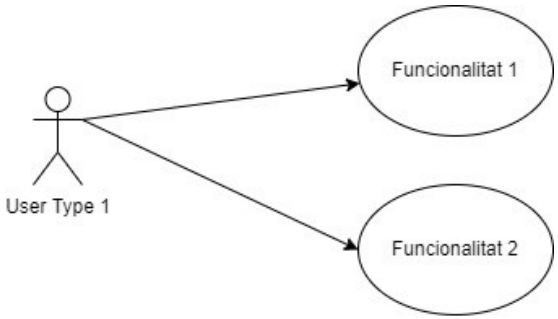
Què fa la nostra aplicació?

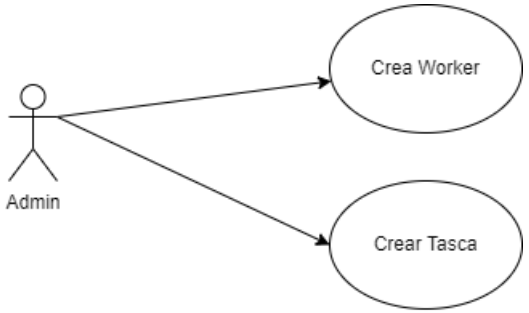
Abans de res

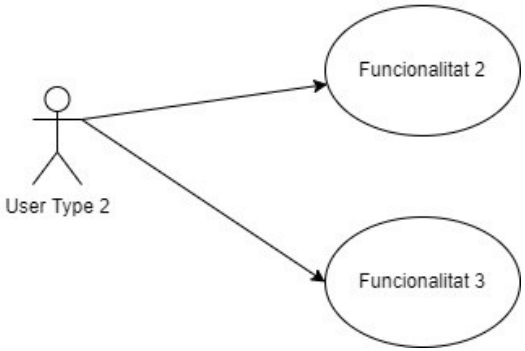
Una manera molt útil d'aclarir les funcionalitats de la nostra aplicació és mitjançant diagrames de casos d'ús.

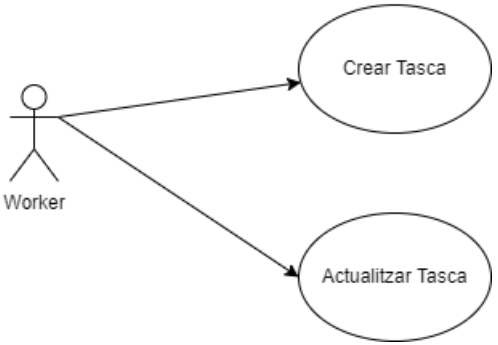
Ens permeten representar de manera senzilla usuaris i funcionalitats.

Abans de res


Funcionalitat 1
User Type 1
Funcionalitat 2


Crea Worker
Admin
Crear Tasca


Funcionalitat 2
User Type 2
Funcionalitat 3


Crear Tasca
Worker
Actualitzar Tasca

Abans de...Front-end

De cara a imaginar com volem que les dades de la nostra aplicació en entrin pels ulls, és bo fer un esbós. Una bona manera sería mitjançant la creació de wireframes.

Una llista de llocs on poder trobar recursos per crear-los seria:

- Balsamiq
- Lucidchart
- Moqups
- Draw.io

Abans de...Back-end

Per altra banda, també hem d'imaginar que necessitarem per emmagatzemar les dades del problema que tenim davant.

Per això haurem de recórrer a **eines de modelatge**(SQL i NoSQL) que coneixem bé de l'Sprint de base de dades.

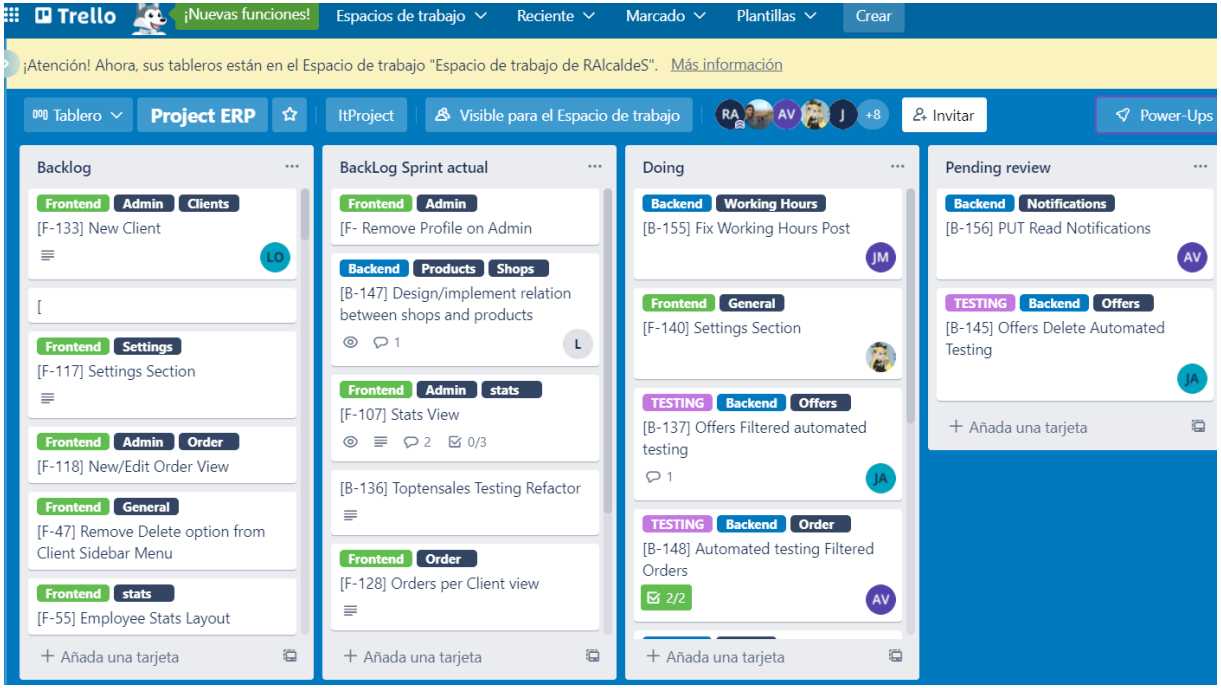
Modelarem, llavors, abans de començar.

Consells d'organització

Dividir és vèncer. Per això, com més detallem les tasques que hem de fer, millor podrem **organitzar**, **dividir** i **repartir** la feina de desenvolupament entre els **diferents membres de l'equip tècnic.**

Consells d'organització

Un tauler de tasques ens pot ajudar a representar aquesta repartició de responsabilitats i tasques. Un bon exemple és Trello.



Ens permet detallar:
- Tasques
- Autor/a(s)
- Estat de tasques

.
.
.
(etc)

Recoinexement del terreny

Normalment, quan ens incorporem a un projecte, el codi font no comença de des de zero. A més, si fem servir un *framework* aquest imposa una estructura de fitxers concreta.

Dedicar un temps a **estudiar aquesta estructura** és clau, tret que ja hàgim treballat abans amb projectes similars.

Els projectes haurien de tenir un document **README** amb una explicació bàsica de com entendre aquesta estructura.

Recoinexement del terreny

També és util jugar amb l'execució de del codi font en l'estructura donada per entendre quin camí segueix el **fluxe d'execució** del programa.

En PHP tenim:

- Xdebug.
- *Debugging* manual: Una manera menys el.legant però efectiva de *debuggar* forçant l'aturada de l'execució del programa seria mitjançant la funció: `exit();`

  No dubteu en **arrencar i curiosejar** el codi font donat per entendre millor el seu funcionament.

Recoinexement del terreny

Un dels canvis que costa més d'entendre és el fet de passar d'executar fitxers directament a l'explorador web a **fer servir rutes.**

Si us fixeu, normalment no posem fitxers en les nostres cerques web, posem rutes, ara farem el mateix en aquest exercici.

Al cap i a la fi, **una ruta està relacionada amb algun mètode d'alguna classe** del vostre codi font. No deixar de ser com un àlies.

A "Developers Team" aquestes rutes estan especificades a l'arxiu **routes.php.**

Recoinexement del terreny

Finalment, recordeu que en aquest projecte treballem amb un patró arquitectònic **MVC(Model-Vista-Controlador)** que marca en gran manera el sentit de l'estructura de fitxers del projecte.

- Les classes **model** serien les encarregades d'interactuar directament amb la base de dades.
- Les classes **vista** són les encarregades de mostrar la informació sol.licitada.
- Les classes **controlador** s'encarreguen de coordinar les crides a models i vistes segons la ruta que s'indiqui.

Conclusions

- La **fase de disseny** ens ajuda a **no perdre'ns més del necessari** en la fase de desenvolupament.
- El que dissenyem **és modificable** en la fase de desenvolupament. No ens hem de centrar doncs,a tenir el disseny perfecte abans de desenvolupar, sino en tenir un pla inicial gràcies a aquest disseny.
- Per a molts/es és la **primera vegada** en un projecte d'aquest abast. Per tant, **benvingut l'error**, marcarà el camí de millora com a desenvolupadors/es de *software*.
- **No us deixeu vèncer** per un ús generalitzat i estés de fonaments de PHP que ja domineu. Al cap i a la fi, tot són classes,variables, mètodes, repeticions i condicionals. **Esteu preparats/es** per entendre-ho.

barcelona.cat/barcelonactiva