

Seguiment del projecte

A continuació s'indiquen les tasques de seguiment setmanal del projecte.


S01: El projecte ha d'estar gestionat des del primer dia a gitlab de l'institut.

Els següents usuaris estan com a **maintainers** del projecte.

Jordi Bosom	jordi.bosom@copernic.cat
Miguel Ángel Luján Prieto	miguel.a.lujan@copernic.cat
Jordi Lordan	jordi.lordan@copernic.cat
Josep Queralt	josep.queralt@copernic.cat

S02: [S] El projecte a GIT té un readme.adoc amb les dades el projecte

Al readme principal del projecte git hi ha d'haver:

- El nom del projecte
- El nom dels membres del grup
- Els links a totes les fonts externes del projecte: Trello, figma, etc...
-  Afegiu els membres del grup a la descripció del projecte

R0:

Implementació d'una API Rest en Laravel

R1: API de gestió de productes

Aquesta API ha de fer un CRUD bàsic de productes.

Aquests productes, com a mínim han de tenir els següents camps:

- Nom
- Descripció
- Preu
- Imatges

Per alliberar el servidor de la càrrega que suposa la gestió i l'emmagatzematge de les imatges hem optat per crear una API en un nou servidor PHP (fora del servidor actual de Laravel) que permeti emmagatzemar imatges i oferir-les sota petició d'un client.

Aprofitem també per modificar la relació de les imatges amb els productes.

Amb les noves modificacions:

1. Un producte ha de poder tenir més d'una imatge
2. Cal limitar el nombre d'imatges que pot tenir un producte amb una regla de negoci. El nombre màxim d'imatges per producte el podeu determinar vosaltres però ha d'existir i ser diferent de 1. (Recordeu documentar-ho a la memòria)
3. Una i només una de les imatges de cada producte té el rol d'imatge principal, és la imatge que apareixerà com a resultat de les cerques a les pàgines de cerca.
4. Tot producte ha de tenir com a mínim la "imatge principal".
5. L'API d'imatges ha de ser una API Rest, en particular tingueu en compte que amb aquest tipus d'arquitectura:
 - a. Cada recurs (en aquest cas cada imatge) ha de tenir un identificador únic (normalment és la pròpia url)
 - b. L'accés a cada mètode es realitza amb el verb HTTP adient (GET, POST, PUT/UPDATE, DELETE)
 - c. Totes les respostes dels mètodes de la API contenen un codi d'estat (per exemple, 200 OK, 404 imatge no trobada)
6. L'API ha d'oferir els següents mètodes:
 - a. Donar d'alta un producte
 - b. Modificar un producte
 - c. Eliminar un producte
 - d. Llistar productes
 - e. Cercar producte/s
 - f. Obtenir una imatge d'un producte en concret
 - g. Obtenir totes les imatges d'un producte
 - h. Eliminar una imatge d'un producte en concret
 - i. Eliminar totes les imatges d'un producte
 - j. Pujar una imatge en concret
 - i. Cal processar les imatges pujades perquè compleixin uns criteris d'eficiència establerts.

Ens interessa especialment tenir en compte el pes de les imatges i les relacions amplada/alçada de les imatges per tal que es vegin correctament a les vistes.
7. Cada vegada que un client refresqui la pàgina no volem obtenir les imatges de nou, assegureu-vos de mantenir les imatges baixades a la *cache* del client com a mínim dos mesos. (google recomana 6 mesos)
8. En essència, l'API permet fer tres coses, pujar imatges, baixar imatges i eliminar imatges. Heu de decidir quines restriccions de seguretat apliqueu a cadascuna d'aquestes accions, **(comenteu-ho a la memòria)** a mode d'exemple:

- a. Podem determinar que l'accés a les imatges és lliure i per tant qualsevol client pot accedir-hi.
- b. Podem determinar que les imatges poden ser pujades i eliminades exclusivament per l'aplicació *marketplace*. **Cap altra aplicació ha de poder pujar ni eliminar imatges.**

R2: Aplicació de productes

Per tal de poder accedir a l'API necessitem una aplicació bàsica de productes que sigui una SPA (*Single Page Application*). Per tal de fer-ho s'ha d'implementar una aplicació web utilitzant JavaScript Asíncron.