Informe Inicial

Pipoyan-Paronyan, Arman

arman.pipoyan26@gmail.com
Bellatera, 2022

1 Introducció

Per aprendre a programa s'ha de programar, això és el que diuen tots, però també és important equivocar-se i aprendre dels errors que es cometen. Una bona manera de fer-ho, i més si s'està a una classe, és preguntant a algú que sàpiga molt del tema, que a les aules acostuma a ser el professor. Gran part d'aquests errors són comuns, per tant, tenir la possibilitat d'explicar la solució d'un error a tot el grup tenint un exemple real al que assenyalar resulta molt útil, ja que alguns conceptes són molt difícils de deixar clar sense pràctica. Tot i ser una cosa tan important és una tasca que es dificulta quan la quantitat d'estudiants és molt gran i es disposa d'un temps limitat.

Per aquest motiu es va crear aquesta aplicació que permet als estudiants escriure codi que pot ser revisat pel professor que està present a l'aula sense la necessitat de desplaçar-se a la seva taula [1].

2 Estat actual del projecte

Actualment el projecte permet als professors afegir assignatures i dins d'aquestes poden afegir, editar, amagar i eliminar problemes, que tenen un títol, descripció, fitxers inicials i algunes dades que no s'utilitzen, com per exemple el temps màxim d'execució i la memòria que pot utilitzar el programa mentre s'executa. Els professors també tenen la possibilitat de mirar, editar i executar l'última solució guardada dels alumnes de qualsevol problema.

Els alumnes tenen menys permisos, només poden resoldre problemes eliminant, afegit o modificant els fitxers que el professor estableix en crear el problema i a més, poden executar el codi per veure el resultat.

Pel que fa al registre de nous usuari, els alumnes es poden crear comptes nous sense cap problema, però els professors necessiten una invitació per part d'altres docents per tal de tenir permisos especials quan es registri.

Van haver objectius que no es van poder aconseguir van ser:

- Controlar que no es puguin importar llibreries que afectin el rendiment del servidor o sistema.
- Controlar els timeouts i bucles infinits fent servir la variable de temps màxim d'execució d'un problema.
- Controlar la memòria que utilitza el programa utilitzant la variable que es defineix en crear un problema.
- Controlar els inputs per teclat per millorar l'experiència de programació.
- Permetre descarregar els problemes.
- Eliminar assignatures i problemes definitivament, és a dir, esborrar les carpetes del servidor.

Per tal de poder facilitar la interacció entre estudiants i professor, millorar la qualitat docent i assolir alguns dels propòsits que no es van poder satisfer s'han decidit implementar funcionalitats noves que es descriuran a la secció posterior.

3 Objectius

Els objectius que s'intentaran assolir són:

- Refactorització del codi amb l'objectiu d'uniformitzar l'estil, arreglar errors existents al codi i fer neteja de mètodes i variables no utilitzades.
- Millorar el sistema de control d'accés a les activitats d'una assignatura per iniciar sessions interactives només si el professor vol.
- Analitzar la viabilitat d'integrar l'aplicació amb Caronte [2] utilitzant la api de Moodle [3] i realitzar la integració si és viable.
- Integrar l'aplicació amb git per tal de poder pujar i descarregar fitxers de Github [4] utilitzant la api que proporcionen [5].
- Analitzar la viabilitat d'integrar notebooks interactius de jupyter [6] a l'aplicació.

Integrar l'aplicació amb el moodle i jupyter pot ser una tasca molt llarga, per tant, es dedicarà un temps per analitzar la seva viabilitat i només es realitzaran si es té temps suficient. Si alguna d'aquestes integracions resulta molt difícil s'analitzaran noves funcionalitats per afegir a l'aplicació.

4 Metodologia

Per desenvolupar el projecte s'utilitzarà una metodologia Agile [7], utilitzant Trello [8] per poder organitzar les tasques i tenir una visió general del que s'ha fet i el que falta per

fer. La durada dels esprints serà d'una setmana, intentant durant aquesta franja de temps tenir una porció del mòdul que s'estigui implementat finalitzat, assegurant així que es treballa amb constància.

Per tal de tenir sempre una versió estable del codi es farà servir un programari de control de versions, concretament Github. Principalment, es tindran dues branques actives: la principal, on estarà la versió estable del codi i la branca de desenvolupament, on s'implementaran les noves funcionalitats. Cada vegada que un mòdul estigui acabat i testejat els canvis es pujaran a la branca principal i es continuarà treballant a la segona branca.

5 Planificació

La planificació que s'intentarà seguir és la que podem veure a la figura 1.

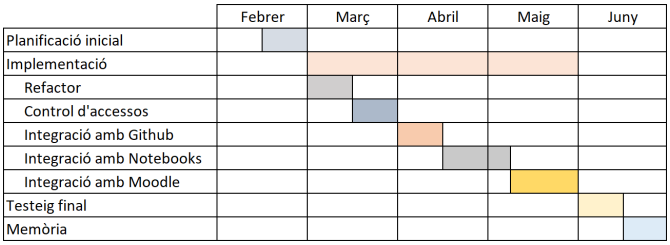

Figura 1. Planificació inicial del projecte

La primera de les tasques consistirà a refactoritzar el codi sencer per tal de poder arreglar errors ja existents i per dividir el codi en mòduls més petits per tenir més facilitat quan s'afegeixen els canvis planificats.

Encara que hi hagi una fase de testeig final planificat es testejarà tot el codi a mesura que es implementat, com s'ha dit anteriorment.

Referències

[1] Y. A. Asbahi, "Aula de programació interactiva." https://shorturl.at/jkruG, 2022.

[2] "Caronte." https://caronte.uab.cat.

[3] "Moodle rest api." https://docs.moodle.org.

[4] "Github." https://github.com.

[5] "Github rest api." https://docs.github.com/en/rest.

[6] "Jupyter." https://jupyter.org.

[7] "What is agile?." https://www.atlassian.com/agile.

[8] "Trello." https://trello.com/en.