Quiosc 2.0

Manuel Lobato Funes

Resum– La comunicació entre l'alumnat i el professorat se sol fer de manera clàssica mitjançant el correu electrònic o tutories. Per tal de millorar aquest aspecte, al departament d'Enginyeria de la Informació i de les Comunicacions (dEIC) es va proposar el projecte Quiosc. L'objectiu d'aquesta eina és millorar l'experiència d'interacció entre els dos col·lectius. Per tal d'assolir aquesta fita, es fa servir la pantalla tàctil situada a l'entrada del departament per oferir una sèrie de funcionalitats com poden ser la consulta d'informació, la possibilitat de demanar tutories o conèixer el calendari d'ocupació dels laboratoris del departament. El Quiosc 2.0 és la segona versió d'aquest aplicatiu i pretén resoldre els problemes amb els quals compta la versió original. Eliminar els apartats que han quedat obsolets o no estan funcionant correctament i afegir mòduls que es requereixen actualment per tal de donar servei als diversos col·lectius que interactuen dins el departament.

Paraules clau– quiosc, tecnologies web, calendari, scraping, geolocalització, interacció, pantalla tàctil, model vista controlador

Abstract– Communication between students and teachers is usually done via email or interviews. In order to improve this aspect, the Kiosk project was proposed by the departament d'Enginyeria de la Informació i de les Comunicacions (dEIC). The goal of this tool is to improve the experience of interaction between the two groups. In order to achieve this goal, the touchscreen located at the entrance of the department is used to offer a series of functionalities such as the consultation of information, the possibility of requesting interviews or knowing the department laboratories schedule. Quiosc 2.0 is the second version of this application and wants to solve the problems that the original version has. Remove the sections that have become obsolete or are not working properly and add modules that are currently required to serve the various groups that interact within the department.

Keywords– kiosk, web technologies, calendar, scraping, geolocation, interaction, touchscreen, model view controller

1 Introducció

Quiosc 2.0 és una iniciativa del dEIC que pretén donar informació rellevant a professors i alumnes sobre diferents aspectes del departament. Entre les seves funcionalitats cal destacar la consulta d'informació del professorat, la possibilitat de demanar tutories i conèixer el calendari d'ocupació dels laboratoris del departament.

En l'actualitat es poden trobar diferents desenvolupaments per a pantalles tàctils, des d'aplicacions per a mòbils i tauletes, aplicacions per fer la comanda en cadenes de menjar ràpid o plànols de la situació de les botigues a les grans superfícies. Malgrat existir gran quantitat, la singularitat, del projecte fa que l'únic aspecte interessant a considerar d'aquestes sigui la d'User eXperience (UX). Com a concepte similar respecte a utilitzar una pantalla tàctil, però amb serveis diferents, es pot trobar el projecte Kiosko Multimedia [1] que s'encarrega de l'emissió de certificats per a estudiants. Malgrat tenir un concepte similar el llenguatge utilitzat és obsolet, ja que fa servir Adobe Flash a la banda del client el qual ha entrat en desús.

Actualment existeix una versió del Quiosc funcionant a la pantalla vertical situada a l'entrada del departament, però la majoria de les seves funcionalitats han quedat obsoletes a causa de diversos factors. La utilització de recursos externs ha provocat que algunes de les pantalles han deixat d'estar operatives i algunes llibreries no estan actualitzades. A més, algunes funcions en l'actualitat s'han deixat de requerir. Per resoldre aquests problemes es proposa realitzar un treball d'actualització de la plataforma per tal de donar ús a aquest servei.

L'objectiu d'aquest TFG és crear una nova versió del Quiosc des de zero considerant que el projecte ha de ser funcional i perdurar en el temps. Per tal d'aconseguir aquesta fita és vol minimitzar l'ús de llibreries i recursos de tercers així com l'ús de frameworks que amb el pas del temps queden obsolets. El sistema es farà servir des de la pantalla vertical situada a l'entrada del dEIC i ha de comptar amb

• E-mail de contacte: manuel.lobato@e-campus.uab.cat
• Menció realitzada: Tecnologies de la Informació
• Treball tutoritzat per: Sergi Robles Martínez (dEIC)
• Curs 2019/20

funcionalitats per l'equip docent i l'alumnat. Els usuaris han d'interactuar amb el programari sense fer ús dels inputs habituals com poden ser el teclat i el ratolí, això ha de comportar una sèrie de consideracions que cal tenir en compte a l'hora de realitzar la interficie d'usuari.

Ambdós col·lectius poden fer servir totes les característiques del Quiosc 2.0 indistintament, però el sistema ha de comptar amb funcionalitats que satisfan les necessitats de tots ells. L'aplicatiu ha de ser gestionat per un *back office* on usuaris administradors podran controlar els mòduls del software, les traduccions a diferents idiomes i les característiques generals del Quiosc.

Per assolir amb èxit la realització del projecte cal considerar un conjunt de subobjectius a l'hora de realitzar el seu desenvolupament. A continuació es mostren aquests ordenats per importància.

1. Utilitzar la menor quantitat possible d'elements externs.
2. Considerar l'experiència d'usuari a través d'una pantalla tàctil.
3. Fer l'aplicatiu a partir de mòduls que permetin la durabilitat i l'escalabilitat.
4. Utilitzar una metodologia de treball que permeti assolir la finalització del projecte.
5. Realitzar proves per tal d'assegurar el correcte funcionament del software.

Aquest article està distribuït de la següent forma. Primer es parlarà dels requisits amb els quals ha de comptar el projecte. Seguidament es tracta la metodologia utilitzada per assolir de forma eficient els requisits imposats pel departament. A continuació es veurà l'arquitectura del software. A la secció cinc es poden trobar els mòduls que compondran l'aplicatiu amb la descripció de les seves funcionalitats. El següent apartat parla sobre la implementació, on s'explica com s'han desenvolupat els diferents aspectes de l'aplicatiu. A la secció set es pot trobar informació sobre les proves realitzades així com el control de les versions. A l'apartat vuit es discuteixen els resultats obtinguts del projecte i en la següent s'expliquen els treballs a realitzar de cara al futur al Quiosc 2.0. Finalment es presenten les conclusions del treball.

2 REQUISITS

En aquesta secció es recullen els requisits derivats de les històries d'usuari recollides durant la reunió inicial amb els responsables del projecte al departament. Es divideix en dues seccions, els requisits funcionals que són aquelles funcionalitats amb les que ha de comptar el programari i els no funcionals que són aquelles característiques i limitacions que ha de complir el sistema. La descripció d'ells diu de forma inequívoca quin actor ha de fer certa acció dins del subsistema corresponent del projecte.

2.1 Funcionals

- El sistema Quiosc 2.0 ha de permetre als usuaris accedir als mòduls del sistema.
- El sistema Quiosc 2.0 ha de permetre als usuaris consultar el directori de professors [2] del departament dEIC a través del mòdul Directori de Professors.
- El sistema Quiosc 2.0 ha de permetre als usuaris localitzar el despatx dels professors del dEIC mostrant per pantalla informació relacionada amb el professor i un mapa de localització.
- El sistema Quiosc 2.0 ha de permetre als usuaris demanar tutories als professors del dEIC a través d'un formulari que envia un email a un professor concret.
- El sistema Quiosc 2.0 ha de permetre als usuaris consultar informació de la pàgina web del dEIC.
- El sistema Quiosc 2.0 ha de permetre als usuaris consultar el Top d'Enginyeria [3].
- El sistema Quiosc 2.0 ha de permetre als usuaris consultar el calendari d'ocupació dels tres laboratoris del dEIC així com conèixer quina assignatura s'imparteix a l'aula. Cal mostrar la informació emmagatzemada al calendari Microsoft Calendar 365 del grup del professorat.
- El sistema Quiosc 2.0 ha de permetre als usuaris veure notícies importants relacionades amb la UAB i el dEIC.
- El sistema Quiosc 2.0 ha de permetre als usuaris interactuar amb ell a través d'un teclat lògic.
- El sistema Administració ha d'estar protegit per usuari i contrasenya.
- El sistema Administració ha de permetre a l'usuari administrador habilitar i deshabilitar mòduls del sistema Quiosc 2.0.
- El sistema Administració ha de permetre a l'usuari administrador veure, crear, modificar i eliminar notícies relacionades amb el departament i la universitat.
- El sistema Administració ha de permetre a l'usuari administrador gestionar els mòduls del quiosc.
- El sistema Administració ha de permetre a l'usuari administrador posar el quiosc en mode *Debug* per tal de detectar possibles errades que s'estiguin produint i recollir les traces.

2.2 No funcionals

- El Sistema Quiosc 2.0 s'ha d'utilitzar en una pantalla tàctil vertical situada al passadís del dEIC.
- El sistema Quiosc 2.0 ha de ser multiidioma permetent com a mínim el català i l'anglès.
- El sistema Quiosc 2.0 ha de perdurar en el temps, en conseqüència ha de ser robust, fiable i consistent.
- El sistema Quiosc 2.0 ha de ser el més automatitzat possible evitant la interacció de l'administrador així com afectacions causades per llibreries i continguts de tercers.

3 METODOLOGIA DE DESENVOLUPAMENT

Els objectius a l'hora de realitzar l'aplicació estan clars, però el seu desenvolupament en alguns casos és una incògnita i l'aproximació que cal portar de cada secció és incerta. Per aquest motiu la millor opció és fer servir una metodologia de desenvolupament de software àgil [4] que permet adaptar el programa als canvis. Els canvis són un aspecte clau en el manteniment del software. Això és a causa del fet que el client, per petit que sigui el programa, segur que vol introduir canvis i millores a l'aplicatiu. Aquest és un aspecte que se sol oblidar amb freqüència a l'hora de triar una metodologia, per això s'ha volgut considerar com un aspecte clau a l'hora de triar una que permeti assolir els objectius definits.

La metodologia triada és una adaptació d'*eXtreme Programming* (XP) [5], ja que s'ha volgut prioritzar el codi per sobre dels altres aspectes del desenvolupament de software. Aquesta es basa en la simplicitat, la comunicació, el feedback, el valor i el respecte. El desenvolupament està guiat per les històries d'usuari, que són les funcionalitats que el client vol pel sistema. Es fan iteracions curtes de dues setmanes amb entregues que han de ser completament funcionals, estar provades i per elles mateixes han de donar valor a l'aplicatiu. La part de test és important, ja que s'ha de realitzar *refactoring* [6] del codi durant les diferents iteracions i seran una part fonamental per comprovar que tot està funcionant correctament.

El compromís més important que no es fa servir de XP és la programació per parelles, ja que el treball es realitza de forma individual. A causa d'aquest fet s'indica que s'adapta la metodologia però la resta d'aspectes rellevants que la componen si s'aprofiten.

4 ARQUITECTURA

L'arquitectura que es fa servir per desenvolupar el Quiosc 2.0 és la més comuna en el desenvolupament web, Model Vista Controlador (MVC). Aquesta permet separar la lògica de negoci de la de client, és a dir, les pantalles amb les quals interactua l'usuari (vistes) mitjançant una sèrie de classes intermèdies (controladors) que s'encarreguen d'interactuar amb les entitats (models). Fer servir una arquitectura ens ajuda a crear i mantenir el codi amb més facilitat, ja que ens permet preservar una estructura més simple on cada part només s'ha d'encarregar de fer les tasques adients.

Els models representen la lògica de negoci i són els encarregats de realitzar la tasca que proposen els requisits funcionals del sistema. Tanmateix tenen la missió de controlar la capa de dades del sistema.

Les vistes són les parts del codi amb les que ha d'interactuar l'usuari. En Php [7] solen funcionar a través d'esdeveniments controlats a la banda del client amb Javascript que envien peticions a la banda del servidor on Php s'ha d'encarregar de recollir, netejar i processar les dades rebudes per part del client.

Els controladors són els intermediaris entre els usuaris i la lògica de negoci. Quan reben una petició s'han d'encarregar d'instanciar els models i objectes necessaris per dur a terme l'acció requerida per part de l'usuari. En finalitzar l'execució de les accions cal notificar al client retornant una vista amb el resultat de l'acció.

L'aplicatiu web se sol dividir en dues parts ben diferenciades, el *backend* i el *frontend*. El *backend* és la part que s'executa a la banda del servidor i inclou els models i els controladors. El *frontend* és la part amb la qual interactua l'usuari i inclou les vistes.

A continuació es detallen les tecnologies principals utilitzades per desenvolupar el projecte. Pel que fa a la part del *backend* es farà servir PHP. El *frontend* farà servir principalment HTML5, Javascript, CSS3 i algunes llibreries necessàries per dotar de certa funcionalitat al navegador, com pot ser mostrar un teclat per pantalla, crear un editor de text per generar notícies o fer que la pàgina s'adapti a la mida de la finestra de manera *responsive*.

El projecte es divideix en carpetes que representen els components que emmagatzemen. A més dels que han de contenir el codi de l'arquitectura, hi ha altres elements que són imprescindibles pel funcionament de l'aplicatiu.

- Config - Aquests components contenen la configuració de l'aplicatiu. Es pot trobar els paths dels elements principals del quiosc, la configuració d'accés a la base de dades i el control de debug que pot activar o desactivar l'administrador.

- Core - A core es pot trobar les classes principals de l'aplicació. Aquestes solen ser classes de comunicació amb components externs del quiosc i classes base de les quals s'ha d'heretar o incloure per part dels components de l'arquitectura MVC.

- Routing - La funcionalitat d'encaminament web és una eina que s'ha d'utilitzar a l'hora de fer servir l'arquitectura MVC. El *Routing* s'ha d'encarregar de dirigir les peticions dels URL al controlador i funció corresponent per tal de dur a terme l'acció desitjada per part de l'usuari. Hi ha tres possibilitats per dur a terme aquesta tasca. Es pot fer a través de la configuració d'Apache, per codi de la banda del servidor o a través dels fitxers .htaccess. L'opció més versàtil sol ser la darrera què ha de ser habilitada des de la configuració d'Apache per a un directori determinat.

- Images - Carpeta per emmagatzemar tots els assets d'imatges necessàries per visualitzar de forma correcta el quiosc.

- Css - En aquest apartat s'ha de guardar tot el contingut relacionat amb els estils dels diferents mòduls de l'aplicació. Cal tenir en compte que alguns mòduls tenen la seva pròpia fulla d'estils.

- Js - En aquest component s'ha de guardar tot el codi de la banda del client. Mitjançant scripts javascript es permet controlar certes interaccions que fan els usuaris amb el quiosc.

5 MODULS

Aquest apartat està destinat a explicar en què consisteix cadascun dels mòduls que ha de contenir el Quiosc 2.0 segons les especificacions recollides en la reunió amb el dEIC.

5.1 Llançadora

Aquest mòdul és la pantalla principal del Quiosc i serà el contenidor per a llençar les diferents funcionalitats que tindrà l'aplicatiu. Aquest consta d'una grid d'enllaços que permetrà als usuaris accedir al mòdul corresponent.

5.2 Directori

Aquesta secció ha de contenir informació relacionada amb els professors. Partint d'un llistat on s'han de poder filtrar els professors per diferents característiques, cal poder accedir a la informació rellevant de cadascun d'ells. Aquest mòdul ha d'agafar la informació de directori de la pàgina del dEIC amb informació del personal.

5.3 Localització

Una situació habitual entre l'alumnat és la de fer tutories als despatxos dels professors. Per tal de facilitar la localització del despatx de cada professor del dEIC es vol mostrar un mapa amb una marca aprofitant la funcionalitat *floor* de Google Maps que permet obtenir mapes d'interiors. A més de geolocalitzar el despatx es mostra informació rellevant del professor així com instruccions que permetin a l'alumne arribar al seu destí.

5.4 Demanar Tutoria

Per agilitzar el contacte entre professors i alumnes aquest mòdul ha de donar suport als alumnes a l'hora de demanar una tutoria amb els professors. Un formulari, que inclourà informació de l'alumne, ha de permetre enviar una petició de tutoria al professor seleccionat a través de l'enviament d'un correu electrònic.

5.5 Informació web dEIC

Aquest mòdul ha de permetre afegir certa informació de la web del DEIC. Per fer-ho es faran servir tècniques de *web scraping* [8] per tal d'extreure la informació que es vol considerar a incloure dins el quiosc.

5.6 Top Enginyeria

S'ha de poder visitar el Top d'Enginyeria des del Quiosc 2.0. L'objectiu d'aquest mòdul és permetre l'accés total al top, permetent en qualsevol moment tornar al quiosc.

5.7 Calendari Laboratoris

El dEIC disposa de tres laboratoris als quals s'imparteixen classes de la menció Tecnologies de la Informació. Per tal de conèixer la seva disponibilitat es vol consultar el calendari de Microsoft Calendar 365 on està guardada aquesta informació. Aquest mòdul ha de presentar una pantalla on poder seleccionar els laboratoris a mostrar, així com un calendari setmanal interactiu que permeti el desplaçament entre les setmanes del curs actual.

5.8 Noticies Importants

Hi ha certs esdeveniments a la UAB o al departament que cal poder mostrar al Quiosc 2.0. La informació s'ha de donar d'alta des de la part d'administració i aparèixer en forma de notícia amb un carrusel al Quiosc 2.0. També cal disposar d'una secció on apareguin les notícies llistades per data de publicació, ja que el carrusel només contindrà les notícies més rellevants.

5.9 Lector tarjeta estudiant

A alguns mòduls del Quiosc seria interessant autenticar a l'usuari per tal de donar valor afegit a la seva funcionalitat. Per aconseguir identificar d'una forma segura i assegurar la integritat dels usuaris es vol llegir la identificació de la seva targeta d'estudiant mitjançant un lector. Recollir aquesta informació pot ajudar a crear estadístiques, omplir certes parts dels formularis o validar l'alumne al Top d'Enginyeria.

6 IMPLEMENTACIÓ

En aquest apartat es discutiran els aspectes més rellevants de la implementació de l'aplicatiu a través de l'arquitectura i el disseny definits. També inclou les tecnologies utilitzades així com informació rellevant de les APIS utilitzades.

6.1 Core

El projecte està construït al voltant d'un nucli *backend* realitzat amb Php. Aquest està programat des de zero i s'encarrega de gestionar la interacció de l'usuari amb els mòduls a través de les vistes, consultar al servidor la informació necessària a través dels controladors i instanciar els models que conten la lògica de l'acció que es vol realitzar. Existeixen una sèrie de classes base que proveeixen de funcionalitat a la resta de components del projecte. La vista és una classe única que s'encarrega de carregar els Html pertinents. La classe Model gestiona les accions genèriques dels models com els *getters*, *setters* i conversions entre vectors-objectes (acció molt utilitzada a Php). Finalment la classe controlador gestiona les rutes i les peticions de l'usuari. Pel que fa a la base de dades hi ha una classe implementada com a patró de software *Facade* [9] de PDO que s'encarrega de gestionar les peticiones a la base de dades. A partir d'aquí es generen els models, les vistes i els controladors necessaris per a cada mòdul per tal de dotar a l'aplicatiu de les funcionalitats requerides.

6.2 Microsoft API i MSAL

Per implementar aquest mòdul cal seguir un procés de tres parts. La primera consisteix a registrar a Microsoft Azure una aplicació la qual permet accedir a les APIs de Microsoft. La segona té com a objectiu autenticar l'aplicació contra la llibreria d'autenticació de Microsoft i la darrera és extreure la informació adient del calendari o calendaris d'un usuari concret.

Per tal d'autenticar l'aplicació es fa servir la llibreria *MicroSoft Authentication Library* (MSAL) [10] que permet diversos tipus d'autenticació tant de la banda del client com del servidor. A la banda client no es pot realitzar una autenticació on no intervingui l'usuari per tant s'ha d'optar

per autenticar el quiosc a partir de codi situat a la banda del servidor.

Per autenticar des del servidor es pot fer de dues formes diferents. La primera possibilitat és fer servir un certificat digital i l'altre és utilitzar un secret que permetrà a Microsoft autenticar l'aplicació amb la qual es comunica.

Per fer proves es pretén utilitzar l'usuari de Microsoft proporcionat pel departament, que pot gestionar calendaris i es podrà subscriure als calendaris del dEIC. Per fer l'autenticació cal realitzar certes accions dins de Microsoft Azure amb aquest compte, el qual no té permisos d'accés a certes àrees. Per solucionar aquest problema s'ha de tramitar una sol·licitud de canvis cap als encarregats de gestionar els comptes de Microsoft de la Universitat Autònoma. S'ha triat utilitzar la tècnica del secret, ja que ha de resultar més senzill fer aquesta petició i caldrà gestionar menys canvis entre el dEIC i l'entitat que controla els comptes de correu Microsoft a la UAB.

Els canvis a demanar són:

- Crear una aplicació a Microsoft Azure per tal de permetre la connexió al calendari.
- Donar accés a l'aplicació d'Azure a terceres aplicacions.
- Creació d'un secret per tal d'autenticar el Quiosc.

Un cop feta l'autenticació a través de MSAL cal configurar quina és l'API que es vol fer servir, quins permisos de connexió s'han d'utilitzar per extreure la informació i finalment quina és la informació que es vol obtenir del calendari.

Un cop obtinguda la informació cal integrar el conjunt de dades rebut dins el calendari setmanal. En aquest punt es fa servir la llibreria moment.js, ja que la integració horària de Javascript respecte les zones horàries no és trivial, en fer servir la llibreria s'estalvien molts problemes d'incompatibilitat entre navegadors i de formats de dates.

6.3 Interficie d'usuari

Pel que fa a l'interfície d'usuari s'ha optat pel sistema de menú superior amb divisió per grid a la part de la pantalla tàctil i per un sistema clàssic de Create Read Update Delete (CRUD) a les pantalles d'administració. Per simplificar la tasca de maquetació s'ha fet servir Bootsrap. Aquesta és una llibreria molt extesa en el món web que ens proveeix de diverses característiques. En el quiosc 2.0 s'ha fet servir principalment per maquetar i estilitzar l'aplicatiu de forma rapida i no gastar gaire temps en aquesta faceta. Com que les vistes estan ben separades de la lògica no ha de resultar gaire difícil de canviar per una implementació pròpia o una altra llibreria.

6.4 Editor WYSIWYG

Per tal d'atorgar versatilitat al mòdul de notícies s'ha fet servir un editor WYSIWYG. Aquest permet tenir un petit editor de documents a l'Html per tal que l'usuari administrador pugui fer les notícies al seu gust. Amb ell es poden introduir fàcilment diversos estils a l'hora d'escriure, inserir imatges des de l'ordinador de l'administrador o incloure vídeos.

6.5 Teclat per pantalla

La interacció amb la pantalla tàctil és bastant gran al llarg de l'aplicatiu i és un aspecte de disseny que cal considerar a l'hora del desenvolupament. Per tal d'obtenir una bona UX s'ha decidit utilitzar un teclat per pantalla per a introduir els diferents inpunts per part dels usuaris del quiosc. S'ha fet servir una llibreria en jQuery per tal de permetre la personalització del teclat, la internacionalització de tecles i reduir el temps d'implementació d'aquesta funcionalitat. En clicar en un input on es requereix un teclat es fa aparèixer el teclat flotant que no només pot ser del tipus qwerty sinó que també es pot fer servir un teclat numèric, de símbols o personalitzar les tecles segons les necessitats d'aquell camp concret.

6.6 Google Maps

Per tal de geolocalitzar la situació dels despatxos dels professors del departament s'ha volgut utilitzar l'API Google Maps Indoor de Google. Google no dóna suport, i sembla que no donarà de cara al futur, a aquesta funcionalitat per l'API de Javascript per tant s'han hagut de cercar alternatives. Malgrat no estar implementat en l'API, des de l'aplicatiu web de Google Maps o des de dispositius mòbils sí que es pot accedir a aquesta informació, per tant s'han agafat imatges amb indicadors de la posició dels despatxos pels diferents professors i s'han relacionat amb ells mitjançant la base de dades i el *backend*.

6.7 Multiidioma

La pantalla es vol fer arribar al màxim nombre de perfils possibles, per tant, s'ha decidit mantenir la funcionalitat de multiidioma del quiosc. Per tal d'implementar aquesta s'ha volgut que sigui el més escalable possible. La implementació ha consistit a introduir un fitxer javascript per cada vista que es vulgui traduir, aquest fitxer conte les traduccions pels diferents idiomes de la pantalla. Per tal de facilitar el manteniment d'aquestes traduccions s'ha creat un mòdul a l'apartat d'administració que permet gestionar tant els idiomes com les traduccions d'aquests fitxers de forma gràfica, permetent que l'administrador els pugui modificar.

6.8 Web Scrapping

Algunes de les funcionalitats requerides pel quiosc necessiten obtenir informació d'altres pàgines. Aquesta informació de forma genèrica se sol consultar a través d'*APIS Restfull* que permeten obtenir informació del sistema de fitxers o bases de dades de l'aplicatiu de tercers en format JSON. En certes ocasions aquestes APIS no estan disponibles i per obtenir la informació cal recórrer a la tècnica de *web scraping*. Aquesta consisteix a descarregar els Html d'un lloc concret i parsejar aquests fitxers per tal d'obtenir la informació requerida. Com que aquesta tècnica es fa servir per obtenir informació de diverses pàgines s'ha creat un petit mòdul a la part d'administració que permet triar la url, decidir quin tros de l'Html recollir i realitzar cerques dins d'aquest per tal d'obtenir les dades desitjades. La descàrrega es realitza a través de les llibreries cURL implementades a Php de forma nativa. La tria d'implementació amb tècniques de *scraping* a més facilita el manteniment a l'administrador quan

s'obté la informació de pàgines del departament, ja que no ha d'implementar dos cops els canvis a la pàgina objectiu i al quiosc.

7 PROVES

La refactorització de codi és un aspecte essencial per atorgar qualitat i proveir de millores al software. Aquesta consisteix a millorar el codi, fent que les funcions siguin més optimes, la distribució de responsabilitats més correcta o la interconnexió entre mòduls més estable. Per poder introduir canvis sense por a trencar el codi o haver de provar totes les funcionalitats d'espres d'introduir qualsevol canvi és essencial comptar amb un gestor de versions de codi. Per seguir les tendències de la comunitat es fa servir Git per aquesta tasca. Un cop es compta amb un gestor de versions no cal tenir por d'introduir errors al programa, ja que es pot recuperar una versió anterior d'una forma relativament senzilla. L'altre aspecte possiblement tant o més important són les proves. De proves existeixen de diversos tipus de caixa negra, caixa blanca, de regressió, d'integració, automàtiques... Com a mínim cal comptar amb proves unitàries que ens asseguren que les classes desenvolupades fan el que han de fer, i ho fan bé. Com que la refactorització és un aspecte intrínsec de la metodologia triada s'ha seguit aquesta a l'hora de realitzar proves unitàries, per aquesta fita s'ha utilitzat PhpUnit, un entorn que segueix l'arquitectura xUnit per tal de donarnos les eines necessàries per fer els tests. En definitiva, el que ens proporciona són un conjunt de classes i interfícies que ens permeten utilitzar assercions per comprovar que es compleixen les especificacions a l'hora d'utilitzar una classe.

Per tal de construir les proves s'han fet servir les tècniques de particions equivalents, que són els subconjunts d'elements d'una entrada que produeixen el mateix valor de sortida. Els valors límit i frontera que són aquells elements dels subconjunts que es troben a les voreres d'aquests i és on se solen concentrar els problemes i finalment s'han assegurat algunes classes més critiques amb proves de caixa blanca buscant proves que exercitin el *Subject Under Test* (SUT) de forma que es cobreixin tots els camins del codi.

8 RESULTATS

En aquest apartat es pretén fer un recull dels resultats obtinguts en el projecte. Per tal de deixar constància dels resultats s'explicarà per cada mòdul si s'han assolit o no els objectius així com les causes d'haver arribat a un cert estat d'implementació. Al final de la secció es fa un recull dels aspectes positius i negatius assolits al llarg del projecte.

8.1 Llançadora

Aquest mòdul s'ha finalitzat completament en la part d'administració i usuaris. Des de la part d'administració es permet el control sobre els mòduls que es vol que apareguin, les descripcions i la seva posició. Els usuaris per la seva banda poden accedir a les seccions que necessitin visitar.

8.2 Directori

El mòdul Directori s'ha implementat correctament. A més amb la finalitat d'aconseguir l'objectiu de perdurabilitat en el temps s'ha realitzat un mòdul a l'apartat d'administració que facilita a l'administrador gestionar els diferents aspectes del mòdul.

8.3 Localització

Des del punt de vista pràctic aquest mòdul compleix amb els requisits proposats per part del departament. El problema que contempla la implementació triada és que no desacobla completament la responsabilitat de l'administrador del fet que un professor canviï de despatx. Si succeeix aquest fet cal tornar a realitzar la fotografia amb la geolocalització. S'ha preferit deixar de forma més laxa aquest requisit i aconseguir una implementació amb mapa, ja que s'ha considerat que l'aportació que feia aquest al mòdul era més interessant que la quantitat de feina a realitzar en arribar un nou professor o que un altre canviï de despatx.

8.4 Demanar Tutoria

Aquest mòdul s'ha completat al cent per cent. Malgrat això a mesura que s'anava desenvolupant s'ha vist que aquest pot tenir un potencial molt més gran que els requisits demanats per a ell en primera instància. Per això es proposen treballs futurs en la secció corresponent de cara a millorar aquest mòdul.

8.5 Informació web dEIC

Aquesta característica queda coberta amb el mòdul de *scraping* de pàgines implementat a la part d'administració. Certament en aquesta part caldria reunir-se amb el client i veure quines són les informacions que vol obtenir de la web del departament. En tenir accés a la base de dades del dEIC pot semblar més correcte accedir directament a aquesta informació mitjançant el *backend* però al realitzar aquesta tasca en forma d'*scraping* permet a l'administrador introduir els canvis en un únic punt i que aquests es propaguin als mòduls pertinents sense que hagi de realitzar més accions, complint amb l'objectiu de poca necessitat de manteniment per part de l'administrador.

8.6 Top Enginyeria

La visualització del top d'enginyeria s'ha assolit. En aquest apartat cal revisar com poder fer l'autenticació de l'usuari si es volgués veure els apartats relacionats amb les seves pròpies dades, ja que en ser una pàgina externa al Quiosc no es pot utilitzar la funcionalitat del teclat en pantalla. Una alternativa seria recollir mitjançant el mòdul d'*scraping* dades concretes del top fet que estaria relacionat amb l'apartat anterior.

8.7 Calendari Laboratoris

Aquest és un dels mòduls amb més pes dins del Quiosc i el que aporta una funcionalitat més interessant de cara als dos col·lectius que fan servir la pantalla tàctil. S'ha aconseguit

amb èxit connectar, recollir i carregar la informació dels calendaris de Microsoft del departament dins d'un calendari Javascript, per tant l'objectiu d'aquesta fita s'ha complert correctament. Pensant en què qualsevol punt pot ser millorable, Microsoft ha publicat un calendari Javascript amb les característiques del seu servei d'ofimàtica web, per tant, es podria realitzar un canvi en el calendari principal per tal que la visualització sigui millor i ajudi a les persones que estan familiaritzades amb la interfície visual de Calendar 365. Malgrat això aquest canvi seria purament estètic.

8.8 Noticies Importants

Aquesta secció s'ha millorat respecte a la proposta proposada pel client. En ella es va detallar poder incloure imatges o PDFs amb notícies relacionades amb el departament i la universitat. Per permetre la lliure creació de contingut l'editor WYSIWYG atorga autonomia i senzillesa perquè un usuari administrador pugui introduir els canvis necessaris. La informació introduïda a l'input de la notícia es guarda en forma de bytes a la base de dades mitjançant un camp de tipus Blob. Aquesta característica també pot permetre tenir un segon nivell de permisos on un usuari redactor pugui entrar a aquesta secció de l'administració i introduir notícies d'una forma senzilla, ja que es compta amb un petit editor de texts on un usuari amb nivell d'informàtica ofimàtic el pot fer servir.

8.9 Lector tarjeta estudiant

Aquest punt no s'ha assolit. Es va estudiar la possibilitat de realització d'aquest mòdul i inclús es va revisar el protocol per dur a terme la implementació, CCID, per tal de veure si es podia realitzar la lectura del carnet. Es va decidir que implementar aquesta funcionalitat faria moure la planificació del projecte d'una forma inassumible endarrerint o imposibilitant la creació de mòduls essencials del projecte i es va decidir no implementar aquesta funcionalitat dins l'àmbit del TFG actual.

8.10 Resultats Positius

Els objectius principals del projecte actualment s'han assolit amb alguna desviació respecte a la planificació inicial. Els mòduls més importants com són el de calendari i els que proveeixen informació del professorat estan en funcionament i el panell d'administració permet tenir control sobre els mòduls del quiosc. El projecte compta ara amb una arquitectura solida, que ha de permetre afegir nous mòduls en cas de ser necessari, d'una forma relativament senzilla, tant en la seva implementació com en la incorporació d'aquests al Quiosc.

8.11 Resultats Negatius

Hi ha una sèrie de fites que no s'han complert al llarg del treball. Aquests es poden dividir en dos grups els de les funcionalitats que no s'han implementat i els objectius que no s'han assolit total o parcialment. Respecte a la funcionalitat, el mòdul no implementat ha estat la lectura de les dades de la targeta d'estudiant. Pel que fa als objectius la dependència amb llibreries o APIS de tercers no s'ha minimitzat tant com caldria esperar. En alguns casos pel fet intrínsec de voler utilitzar una característica de funcionalitats de tercers i en altres per simplificar i accelerar el desenvolupament del projecte. També m'agradaria destacar els diversos motius que considero que han portat al fracàs en aquests aspectes. Per una banda el treball no està focalitzat en una única tasca, això ha comportat gastar temps en mòduls que al final no s'han aconseguit implementar. Un altre dels problemes ha sigut intentar automatitzar massa totes les funcionalitats, aquest fet ajuda a complir un dels objectius del projecte però cal recordar que no és l'únic. Finalment un altre error ha estat ajuntar les hores de la beca de col·laboració amb la feina a introduir en el TFG. Hauria d'haver tingut la capacitat de discernir que introduir i que no dins el TFG, finalitzar la part corresponent al treball i un cop acabat, continuar introduint mòduls, modificacions i millores dins de l'àmbit de la beca de col·laboració.

En aquest punt vull fer un incís en la característica de certes hores. De cara a un projecte d'empresa on cal justificar les hores davant d'un client per tal d'obtenir una retribució econòmica, hi ha una sèrie d'hores emprades en el projecte que són difícils de justificar. Per exemple el mòdul del lector de targetes d'estudiant i l'estudi de l'API de Google Maps van suposar certes hores de treball que no han donat el resultat esperat i per tant son difícils d'imputar a un possible client. Cal considerar aquests aspectes a l'hora de realitzar la planificació, en especial en aquells camps que són incerts segons el nostre coneixement, capacitats o que no han estat abordats en projectes anteriors.

9 Treball Futur

En aquest apartat es vol indicar quin és el treball que es pot realitzar de cara a finalitzar i millorar el producte.

9.1 Entorn de producció

En primer lloc cal parlar de la posada en marxa del programa. Tot i haver comptat amb un entorn de desenvolupament igual al de producció caldria veure com es comporta cada mòdul amb la pantalla tàctil real. Aquesta feina no s'ha pogut assolir a causa de la situació d'excepció que es viu actualment a escala mundial però es reprendrà ja fora de l'àmbit del TFG amb hores de la beca de col·laboració. Un projecte no es pot donar per finalitzat un cop el software està acabat, cal donar la importància que mereix la posada en marxa i el manteniment. Aquest últim punt em sembla rellevant i de deontologia professional, un bon programador mai s'hauria de negar a mantenir el seu propi codi.

9.2 API Google Maps

La geolocalització amb l'API de Google Maps de Javascript no es possible per a interiors. Aquí es poden prendre diversos camins per tal de fer una millor implementació d'aquest modul. El primer seria esperar a que Google decidís incloure aquesta funcionalitat dins la seva API, cosa que sembla poc probable ja que la funcionalitat porta bastant temps en marxa i aquesta opció només ha estat disponible per a l'API de movils. El segon es muntar un sistema amb un emulador d'Android on mitjançant una llibreria que fes d'Adapter [9] realitzar les consultes a l'API de Google per a movils,

que si dispossa d'aquesta funcionalitat, i passar les dades al sistema del quiosc.

9.3 Lector tarjeta

El mòdul del lector de targetes considero que pot tenir el volum de feina d'un TFG complet. Òbviament caldria definir més especificacions i pot arribar a tenir una càrrega de hardware bastant gran. La idea és substituir els lectors de carnets d'estudiar pels quals el departament ha de pagar una gran quantitat de diners per un hardware i software propietari que permeti controlar l'accés a les diferents aules amb pas restringit del departament.

9.4 Demanar tutoria

Aquest mòdul compta amb un gran potencial per afegir noves funcionalitats que donin suport tant a l'alumnat com al professorat. El mòdul de demanar tutories podria comptar amb noves característiques com diversos motius de contacte preestablers, selecció d'horaris de tutoria segons la disponibilitat del professor o un històric de missatges si es compta amb autenticació de l'usuari.

9.5 Proves automàtiques

Aquest tipus de prova tenen un cost molt elevat en la seva creació i cal fer-les de les seccions que es consideren crítiques del sistema. Pot ser interessant comptar amb una bona bateria de proves automàtiques per tal de saber si s'ha trencat alguna funcionalitat d'algun mòdul. En especial és important tenir controlats aquells que depenen de llibreries externes, ja que com que no depenen del Quiosc pot arribar un punt en què deixin de funcionar tot i no introduir canvis en el codi.

9.6 Altres

La refactorització de codi sempre és un bon aliat a l'hora de crear codi de qualitat, escalable i mantenible [6] per tant és una tasca que sempre cal realitzar en utilitzar metodologies àgils de desenvolupament. Per fer passar l'estona als alumnes que estan esperant a una tutoria, es pot implementar un quiz de preguntes relacionades amb temes que tinguin a veure amb les assignatures de la menció. Finalment es poden realitzar enquestes dirigides als col·lectius implicats per obtenir la seva opinió i extreure noves funcionalitats i *feedback* en general.

10 Conclusions

La realització de software és una tasca complexa on intervenen diversos factors. Una planificació correcta i la utilització d'una bona metodologia és essencial per tal d'assolir els objectius i aconseguir finalitzar amb èxit un projecte. Els objectius principals del treball s'han complert, malgrat això hi ha hagut desviacions en la planificació, objectius que no s'han assolit i mòduls que no s'han realitzat. La importància d'una bona gestió de projectes és essencial i he de considerar aquest punt com el causant principal que ha fet que el projecte no s'arribi a finalitzar completament. La planificació, moltes vegades, bé fortament guiada per l'experiència, cal saber triar quina feina realitzar i quina no, s'ha de saber dir que no i comprendre les limitacions dels recursos de què disposes per tal d'assolir les metes incloses dins d'un projecte. Certament és millor partir les entregues en diverses fases a voler fer més feina de la que pots abastar, això només porta a desviacions en totes les fases de la creació del software.

Per altra banda no cal ser només negatiu i el projecte ha aconseguit un bon nivell de desenvolupament. Les funcionalitats principals estan operatives, amb una bona cobertura de proves darrere així com una arquitectura solida que permet introduir canvis sense por a trencar el programa. D'altra banda s'ha aconseguit tenir en funcionament la part pels usuaris i l'administrador que gràcies a l'automatització de diversos aspectes permet tenir un software autocontingut, gestionable i funcional.

Agraïments

En primer lloc m'agradaria agrair al dEIC per donar-me l'oportunitat de realitzar la BECA de col·laboració amb el departament. En especial al meu tutor Sergi Robles Martínez i a Adrià Sánchez Carmona que m'han guiat durant la realització del TFG i la col·laboració. Finalment m'agradaria agrair a tots els professors de la UAB que m'han donat classe al llarg de la carrera. Gràcies a tots ells he aconseguit els coneixements que m'han permès assolir amb èxit la consecució del grau en enginyeria informàtica, millorar com a professional i obtenir la confiança, disciplina i constància necessàries per avançar.

Referències

[1] Molina Toledo, Nataly Monserrath, "Kiosco multimedia para consulta y emisión de certificados académicos de la Universidad Politécnica Salesiana.", Cuenca, Ecuador, 2010.

[2] https://deic-web.uab.cat/personal.php

[3] http://top.uab.cat/

[4] https://www.agilealliance.org/glossary/xp/

[5] Beck, Kent & Fowler, Martin, Planning Extreme Programming, Addison Wesley, 2000. ISBN: 0-201-71091-9. pp. 3-14, 27-48

[6] C. Martin, Robert. Clean Code, A Handbook of Agile Software Craftsmanship. Prentice Hall, 2009. ISBN 978-01-323-5088-4

[7] Lockhart, Josh, Modern PHP, New Features and good practices. O'Reilly, 2015. ISBN 978-1-491-90501-2 pp. 5-11, 75-124, 165-176.

[8] https://en.wikipedia.org/wiki/Web_scraping

[9] Gamma, Erich. Helm, Richard. Johnson, Ralph. vlissides, JohnC. Design Patterns Elements of Reusable Object-Oriented Software, Addison-Wesley, 1995. ISBN 0-201-63361-2

[10] https://docs.microsoft.com/en-us/azure/active-directory/develop/msal-overview