3r Sprint ASW

Canvis de l'API i descripció de la visualització

Grup 12A - ASW

Iván Risueño Martín
David Herrero Faura
David Pérez Barroso
Andreu Orensanz Bargalló

Índex

1. Canvis realitzats a l'API

En aquest apartat s'expliquen els errors identificats durant la presentació del Sprint 2 i del *feedback* del professor i, a continuació, les seves solucions.

1.1. Problemes del Sprint 2

El primer que vam fer després de crear el projecte amb el *framework* escollit (VueJs) va ser anotar tots els problemes detectats durant la presentació de l'anterior sprint i plantejar què es podia fer arreglar-los:

- Crida *microposts* ordenats per data
- Crida *microposts ask*
- Crida un *micropost* per *id*
- Retornar *id* en GET *comment*
- Controlar creació de comentaris *parent_id* i *micropost_id*
- Corregir resposta de *vote micropost*
- No s'ha de poder desvotar *comment* o *micropost* que no s'ha votat

1.2. Solucions als problemes del Sprint 2

Molts dels problemes detectats del Sprint 2 estaven relacionats en la documentació de la nostra API a través del fitxer *api.yaml*. Vam afegir a la crida GET */microposts.json* dues *queries* més per tal d'obtenir els *microposts* ordenats per data i els *microposts* de tipus *ask*. També vam fer una nova crida *GET* per tal de retornar un *micropost* específic a través del seu *id*.
Pel que fa al *refactoring* del codi, vam canviar l'estructura del *comment* que es rep en format JSON a través de les diferents crides per tal que retornés el seu *id*. Per fer això vam canviar el fitxer *_comment.json.jbuilder* i vam afegir l'atribut *id* per a què es mostrés com a resultat de les crides referents a comentari. També ens vam adonar durant la presentació del Sprint 2 que la resposta de la crida per a votar un *micropost* era tota la llista de *microposts* existents. Vam corregir el controlador de *microposts* per tal que retornés només la informació del *micropost* votat. Un altre error que es controla ara, però que no es possible que passi des del *frontend*, és el poder desvotar un *comment* que no s'ha votat. Des del *backend* ja es controla que això no sigui possible i retorni un error però, a la mateixa vegada aquest error des del *frontend* no es produirà mai, ja que el botó de *unvote* només es mostra als *microposts* o *comments* ja votats per l'usuari "loguejat".

2. Descripció de les funcionalitats

A continuació es descriuen les funcionalitats corresponents a les accions de visualitzar la informació d'un usuari "*loguejat*" i canviar l'*about* de l'usuari i fer clic a "*update*" fent servir un diagrama de seqüència per a cada funcionalitat.

2.1. Visualitzar la informació de l'usuari "loguejat"

Per tal de veure la informació de l'usuari "*loguejat*" seleccionat, cal que el sistema tingui guardada l'*api key* de l'usuari que cal demanar-la només obrir el nostre *web* i a l'hora de canviar de perfil.

Quan s'intenta fer un link cap a */profile*, el router delega el treball a la vista '*Profile*' del nostre sistema. Aquesta vista, la qual conté un desplegable amb tres usuaris a escollir, fa un *GET* a l'API del backend amb la url */users/{:id}.json*, on *{:id}* és l'identificador de l'usuari escollit mitjançant un desplegable (de manera predeterminada, el primer). L'API retorna, entre d'altres, els paràmetres necessaris per a mostrar la pàgina (*email*, *about*, *created_at* i l'*api_key*, la qual la genera el *backend*). La variable global que guarda el *loggedUser* s'actualitza i a continuació es renderitza tota la informació d'aquest usuari.

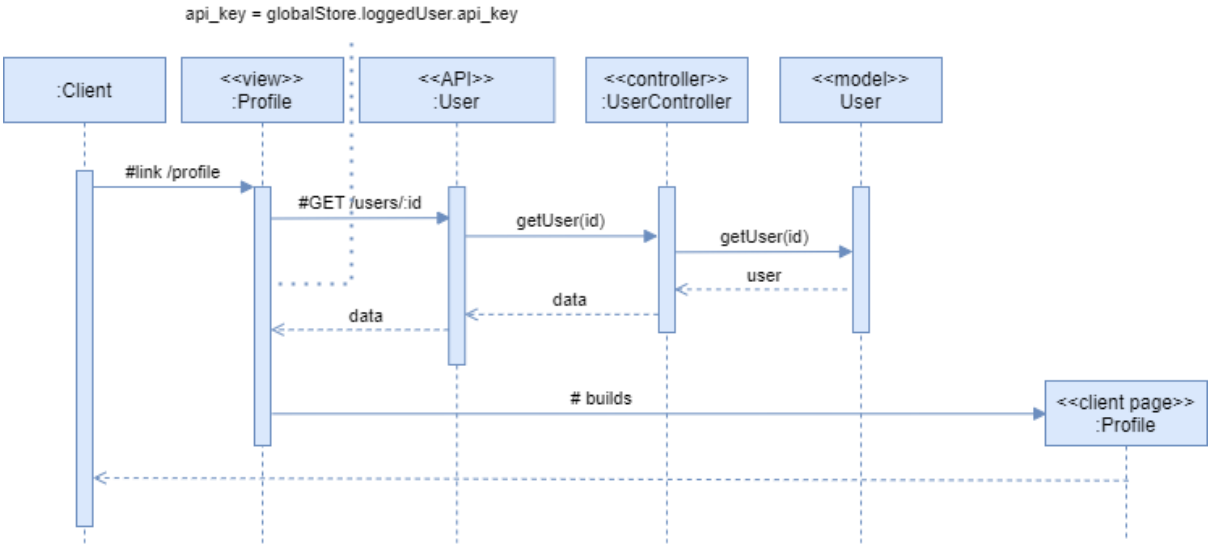

fig. 1: Diagrama mostrant la funcionalitat de mostra informació de l'usuari loguejat

2.2. Canviar l'about de l'usuari i clicar "update"

En aquest diagrama que mostrem a continuació, partim del diagrama anterior on ja tenim la informació de l'usuari "*loguejat*" amb tots els camps per a editar. A partir d'aquí es porta a terme la funcionalitat d'actualitzar el camp de l'*about*.

Quan es desitja canviar l'*about*, s'ha d'omplir el *textarea* que surt a la pàgina del client i prémer el botó '*update*'. Un cop fet, la vista '*Profile*' invoca un mètode que fa una petició de tipus *PUT* a l'URL */users/{:id}.json*, omplint un arxiu *.json* que només conté l'*about* modificat, agafat directament del *textarea*. A la petició s'afegeix com a *header* l'*api_key* de l'usuari loguejat, que agafem de la variable global que se'l guarda. Un cop processada la petició, el *UserController* es guarda aquest nou *about* fent un *update* al model de *User*.

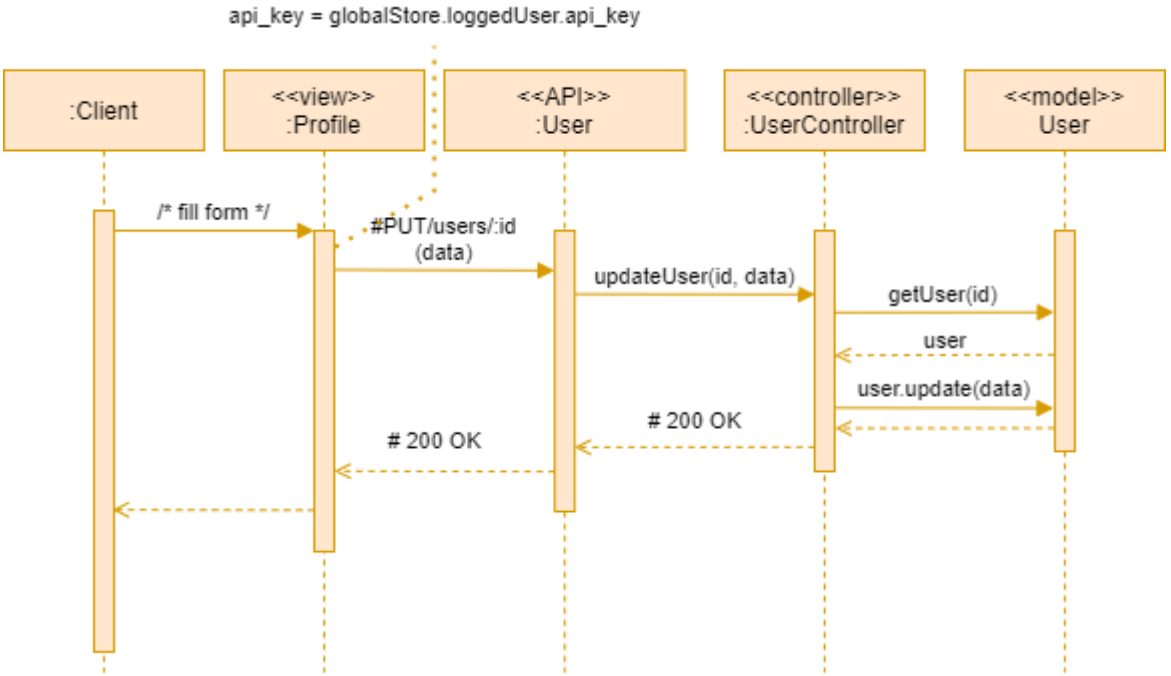

fig. 2: Diagrama mostrant la funcionalitat de actualitzar l'about de l'usuari loguejat