Grup-DAW4

Creadors: Mario Gonzalez. Nacho Munné
Data:09/06/2020
Estat del Projecte: Finalitzat

Index

Manual D’usuari

La primera pàgina que veus és el mapa clickable, on pot moure't per totes les comarques de Catalunya. Tens com a definit el botó de Mapa, per a que puguis buscar la teva comarca amb informació molt detalla de la mateixa.
Si no, just al costat del botó del mapa, tens un botó que posa "Llistat". Aquest botó et porta a veure un taula amb menys informació, però en la qual pots veure de 10 en 10 totes les comarques de Catalunya en un llistat.

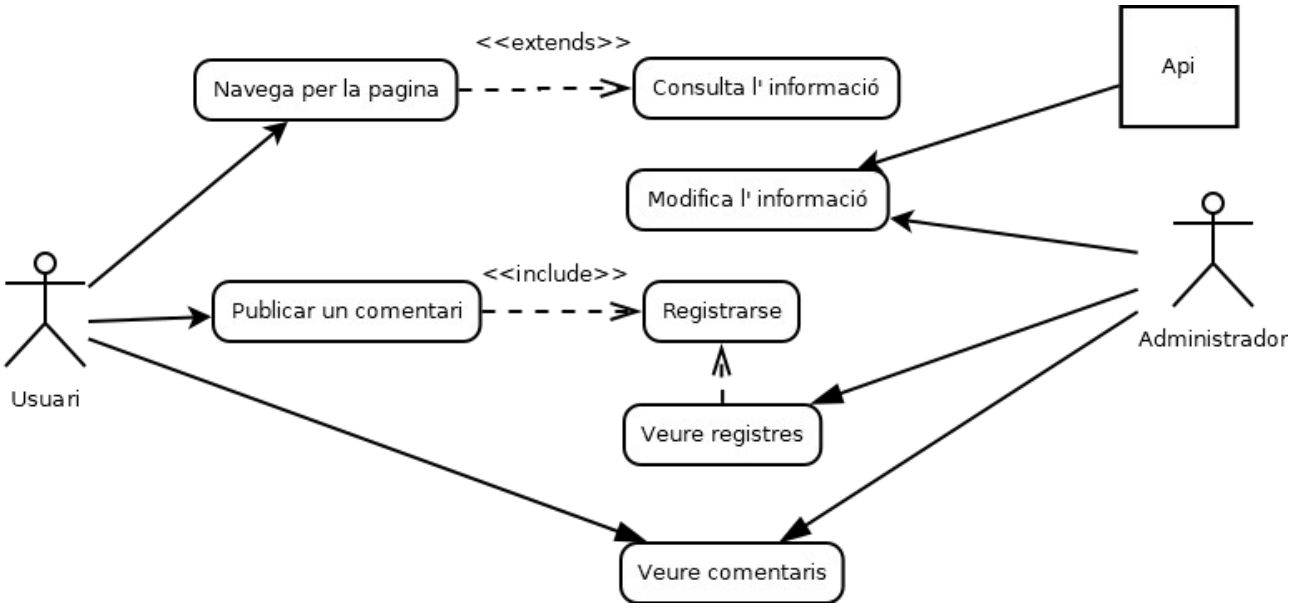
La següent pestanya ens porta al test, aquest apartat tens com a primera plana un test que si l'aplicació detecta que pots ser un possible cas de Covid—19, l'aplicació treu un missatge dient que pots ser un possible infectat i el puja a una taula on reuneix tots els possibles casos detectats per la nostra aplicació.
Aquesta es la segona part d'aquesta pestanya. Quan acabes de fer el test la pestanya et porta a una taula novament amb les 42 comarques, però ara la informació que registra són els casos que la nostra aplicació ha agafat. Amb un comptador de tots aquest casos registrats per nosaltres.

La tercera pestanya del header es la de "Informació", aquesta pestanya et porta a un munt d'informació sobre el Covid-19, possibles símptomes, tots els horaris i diferencies entre les fases, números de contacte...

I per últim, la nostra aplicació té un fòrum on pots parlar/comentar del tema amb altres usuaris.

Si no has fet un Login, no tindràs opció d’utilitzar ni el test ni el fòrum.
Si no tens compte no et preocupis, just al costat de fer Login test un botó per registrar-te i poder fer un ús complert de l’aplicació.

Diagrama D’us



Documentació Tècnica

Api Dades

La carpeta API té com a objectiu agafar les dades que ens atorga la generalitat, convertir-la al format que ens interessa i emmagatzemar-la.

api.php

Aquest document agafa i guarda totes les dades directament de la generalitat en la variable $data.

Posteriorment crea 42 (una per a cada comarca) variables de tipus "comarca", una classe pròpia que conté les variables que volem emmagatzemar per a cadascuna.

Mitjançant un foreach anem recorrent totes les dades que hi ha en $data i les anem afegint a les variables de tipus "comarca" que corresponen.

Per últim, es crea una connexió amb la nostra bdd utilitzant la classe ComarquesBDD ,que es troba a l'arxiu /API/comarques_BDD.php, i inserta les dades de les 42 comarques a la bdd.

api_actualitza.php:

Aquest document agafa i guarda totes les dades directament de la generalitat en la variable $data.
Link Generalitat: http://analisi.transparenciacatalunya.cat/resource/jj6z-iyrp.json

Posteriorment crea 42 variables (una per a cada comarca) de tipus "comarca", una classe pròpia que conté les variables que volem emmagatzemar per a cadascuna. Mitjançant un foreach anem recorrent totes les dades que hi ha en $data i en el cas de que la data sigui del dia anterior a l'actual (les últimes dades disponibles) es guarda a la informació en la respectiva variable de tipus comarca.

Per últim, es crea una connexió amb la nostra bdd utilitzant la classe ComarquesBDD, que es troba a l'arxiu /API/comarques_BDD.php, i fa un update a la bdd de les comarques que han petit algun canvi.

Aquest arxiu s'ha d'executar en el servidor cada dia per tal de que les dades estiguin actualitzades, en el nostre cas ho fem mitjançant crontab com el següent:

0 7 * * * php ProjecteFinal/API/api_actualitza.php

comarques_BDD.php

En aquesta classe es defineix tot el que respecta a la bdd i la seva gestió.

La classe té un self construct per emmagatzemar-les dades de la bdd a la que es connectarà.

La funció open_connection() estableix la connexió amb la bdd i per tant és obligatori que s'executi abans d'utilitzar qualsevol altra funció.

close_connection() tanca la connexió amb la bdd per tant s'haurà d'executar en terminar d'utilitzar-la.

execute_single_query() agafa la query SQL emmagatzemada a $query i l'executa, en cas de no donar error imprimeix "Bien" en cas contrari "Mal" + missatge d'error. Abans d'acabar tanca connexió amb la bdd.

insertComarca($comarca) separa les dades de $comarca en variables independents i reemplaça el caràcter ' per \' per a que posteriorment no hi hagi problemes a l'executar la query en SQL. Després estableix la query en $query i crida a execute_single_query() per a que l'executi.

updateComarca($comarca) fa exactament el mateix que insertComarca($comarca) però en comptes d'inserir a la bdd només actualitza.

deleteComarques() esborra tot el contingut de la taula COMARCA a la bdd.

dades_BDD.php

En aquest fitxer s'han de guardar les dades de la connexió a la bdd en les següents variables:
$db_host = "";
$db_user = "";
$db_pass = "";
$db_name = "";

Crud Api

api.php és una aplicació externa que simplifica les peticions a la bdd realitzades a través d'axios.

Link Crud Api: https://github.com/mevdschee/php-crud-api

JavaScript

La carpeta JS emmagatzema tots els scripts que s'executaran en la part de client.

appVue.js

En aquest arxiu es troben dos variables de Vue (app i app2) que controlen la informació que es mostra tant del mapa (app2) com del llistat (app).

En carregar l' script la funció getData() s'executa i emplena la variable comarques amb les dades de la taula COMARCA de la bdd. Al div amb id "llista" es mostraran les dades que hi ha a la variable comarca de 10 en 10 com estableix la variable perPag mitjançant la sentència Vue:
v-if="(pag - 1) * perPag <= i && pag * perPag > i"

Dins el div amb id "targetaMapa" es mostra la informació que hi ha a la variable comarcaActual, que és la comarca que té seleccionada l'usuari en el mapa. Quan l'usuari fa click en una comarca s'executa la funció setComarcaActual(comarca) on "comarca" es la posició de la comarca seleccionada i atribueix a comarcaActual les dades que hi ha en la variable comarques[comarca] de la variable app.

borrarComent.js

borrarComent(num) num es el propi element (que ens interessa perquè te com a id la clau primaria del comentari), llavors ens quedem només la id.

Executem un swal.fire (plugin per a millorar les alertes) per confirmar que l'usuari vol eliminar el comentari i en cas afirmatiu fem una petició delete, si es realitza amb èxit un swal.fire avisa a l'usuari que s'ha esborrat el comentari i actualitza la pàgina per a que ja no mostri el comentari esborrat.

foro2.php - Script function comentari()

comentari() guarda en la variable comentari el que ha escrit l'usuari, la data en el format que volem el nom que tenim guardat en la variable $_SESSION de php. Mitjançant una petició post afegim el comentari a la taula "comentaris" de la bdd

login.js

login() agafa els valors introduïts per l'usuari i fa una petició post a login.php, en cas de que la petició retorni el valor desitjat "Ok" s'avisa a l'usuari de que ha iniciat sessió amb swal.fire en cas contrari, se li notifica que no.

mapaLlistat.js

Aquest script controla el que es mostra a la web a través de la interacció amb els botons.
veureMapa() mostra el mapa i la seva targeta i oculta el llistat.
veureLlistat() oculta el mapa i la seva targeta i mostra el llistat.

mapaInteractivo.js

Aquest script fa ús d'un script extern anomenat snapSVG. la variable "s" es l'objectiu on es carregara el que modifiquem, en aquest cas el node amb id "mapa". Snap.load() el que fa es carregar el svg seleccionat i executa una funció que agafa tots els nodes del svg, els fica en una variable i els carrega al node que havíem guardat en la variable "s". Després agafa cada comarca (un node de l'arxiu svg que com a id té el seu nom) i el guarda en una variable a la que li afegeix un event onclick que executa la funció setColors(g, comarca, num).

Autor mapa SVG: Joan Borràs Comes

setColors(g, comarca, num) la variable "g" es l'agrupació de tots els nodes de l'arxiu svg, "comarca" es el node concret de la comarca que s'ha clickat i num es la posició de la comarca. Aquesta funció el que fa és modificar els colors fent vermella la comarca seleccionada i posant gris la resta, també crida a la funció setComarcaActual de la variable app2 de appVue.js

registre.js

register() agafa les dades que ha introduït l'usuari i les emmagatzema, comprova que no estiguin buits i en el cas del nom i cognoms que no continguin números. Una vegada fetes les comprovacions en cas de que tot estigui correcte es fa una petició post que si es completa amb èxit informa a l'usuari que ha estat registrat. En cas contrari informa a l'usuari que les dades no son correctes.

SospitososVUE.js

Aquí està la variable app de Vue que controla el node amb id="tablaSospitososAPI". Conté les variables numCasos (on s'emmagatzemen la quantitat de casos detectats), comarcas (un objecte que conté les diferents comarques amb la respectiva informació), pag i perPag que estableixen el numero de comarques que es mostraran per pantalla a la vegada amb la següent sentencia:
v-if="(pag - 1) * perPag <= i && pag * perPag > i".

En carregar l' script la funció getData() s'executa i emplena la variable comarcas amb la informació que rep de la petició get de axios a la taula SospitososAPI.

En cas de que el formulari es completi amb èxit s'executa la funció sumaCas(comarca, sexe), que afegeix la dada rebuda a la variable "comarcas" en la comarca tingui com a "nom" la variable "comarca". També augmenta la variable numCasos.

test.js

veureCoincidencies() agafa la informació que introdueix l'usuari al formulari i en cas de que no hi hagi 3 o més preguntes afirmatives, s'informa al usuari que no sembla un cas positiu mitjançant un swal.fire. En cas contrari es crida a el mètode sumaCas de la variable app de SospitososVUE.js, s'informa a l'usuari que podria estar un cas positiu i es realitza una petició post a la taula SospitososAPI per comptabilitzar el cas. Per últim oculta el formulari i mostra els resultats.

vueForo.js

Conté una variable "app" de Vue que controla el contingut del node amb id="main". En carregar l' script s'executa la funció getData() que el que fa es una petició get a la taula comentaris i els guarda a la variable coments des de on es mostren a la pag foro2.php.

PHP

bdd.php

En aquesta classe es defineix tot el que respecta a la bdd i la seva gestió.

La classe té un self construct per emmagatzemar-les dades de la bdd a la que es connectarà.

La funció open_connection() estableix la connexió amb la bdd i per tant és obligatori que s'executi abans d'utilitzar qualsevol altra funció.

close_connection() tanca la connexió amb la bdd per tant s'haurà d'executar en terminar d'utilitzar-la.

execute_single_query() agafa la query SQL emmagatzemada a $query i l'executa, en cas de no donar error imprimeix "Bien" en cas contrari "Mal". Abans d'acabar tanca connexió amb la bdd.

inserirUsuari($usuari, $password, $email, $nom, $primer_cognom, $segon_cognom) estableix la query en $query per a fer un insert en la taula usuaris i crida a execute_single_query() per a que l'executi.

inserirComentari($comentari) estableix la query en $query per a fer un insert en la taula comentaris i crida a execute_single_query() per a que l'executi.

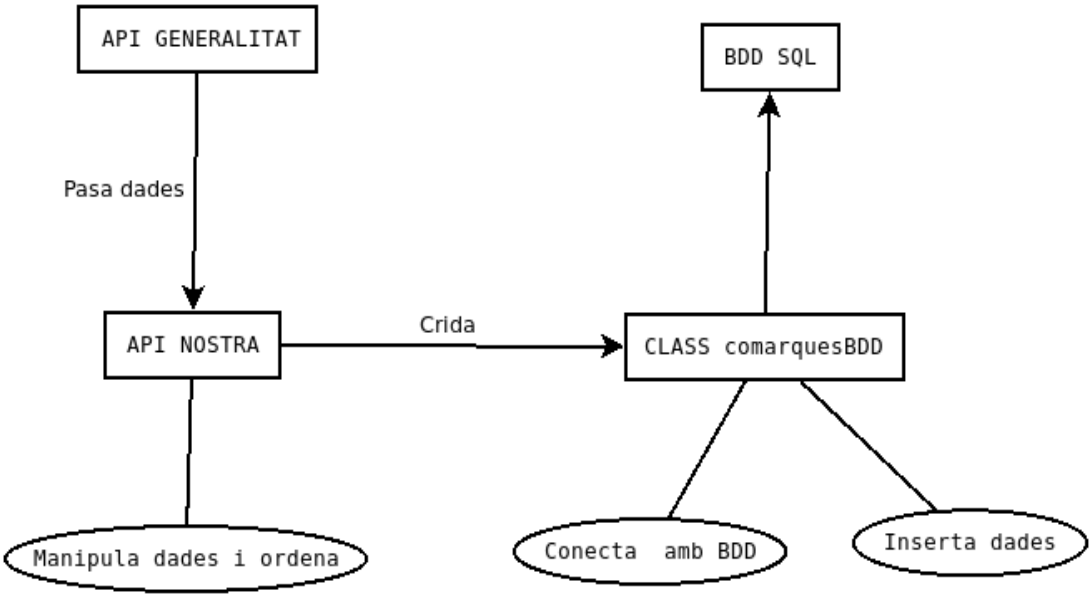
comprovaLogin($usuari, $password) estableix una query per fer un select amb les dades que contenen $usuaris i $password, s'obre connexió a la bdd i es guarden a $results els resultats de la query.
En cas de que $results contingui més d'una columna s'obre una session amb el nom de l'usuari i retorna "Ok". EN cas contrari es retorna "Error".

login.php

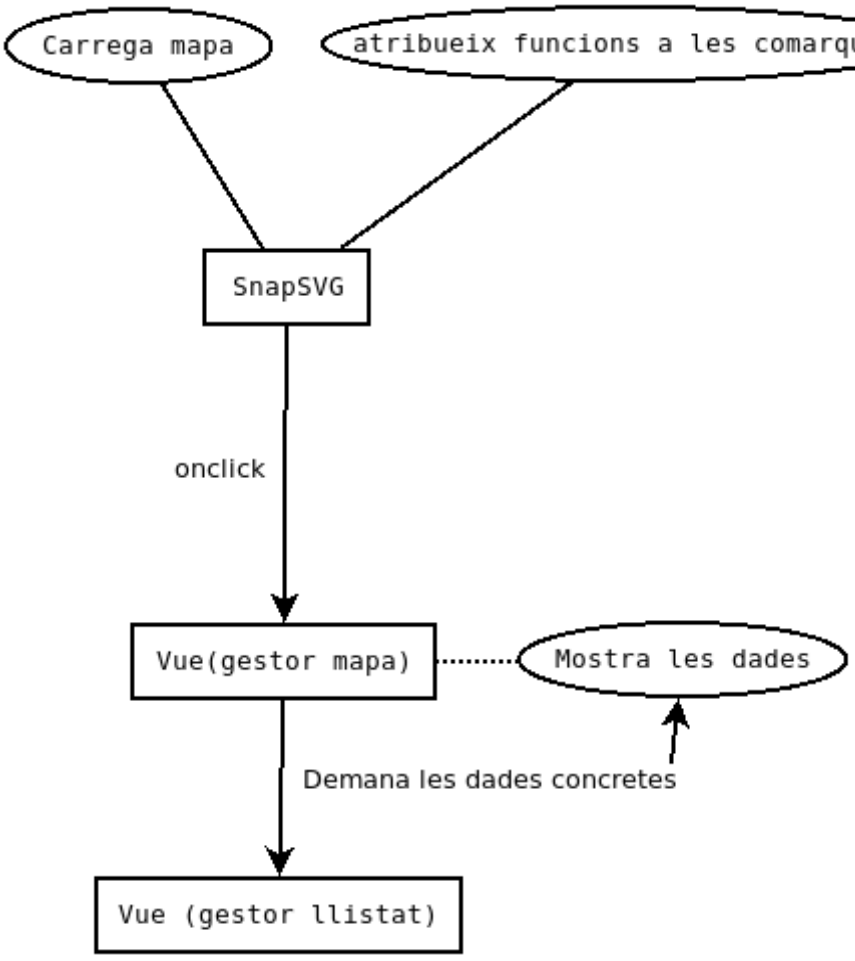
Rep per $_POST els valors que se li passen a través d'axios des de js, comprova que no estiguin buits i estableix connexió amb la bdd. Crida a la funció comprovaLogin i imprimeix el resultat.

Presentació Tècnica

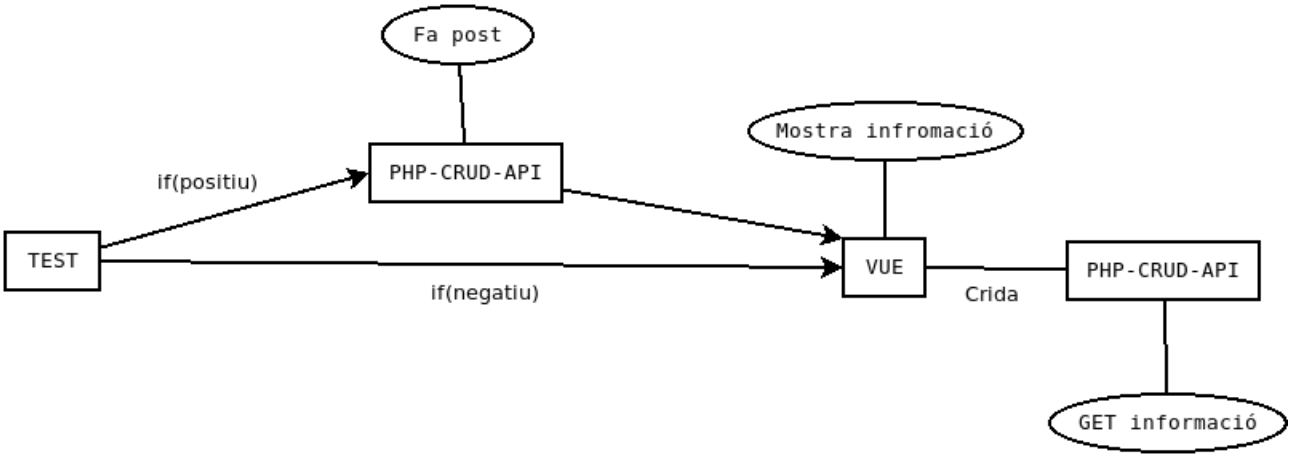
Api



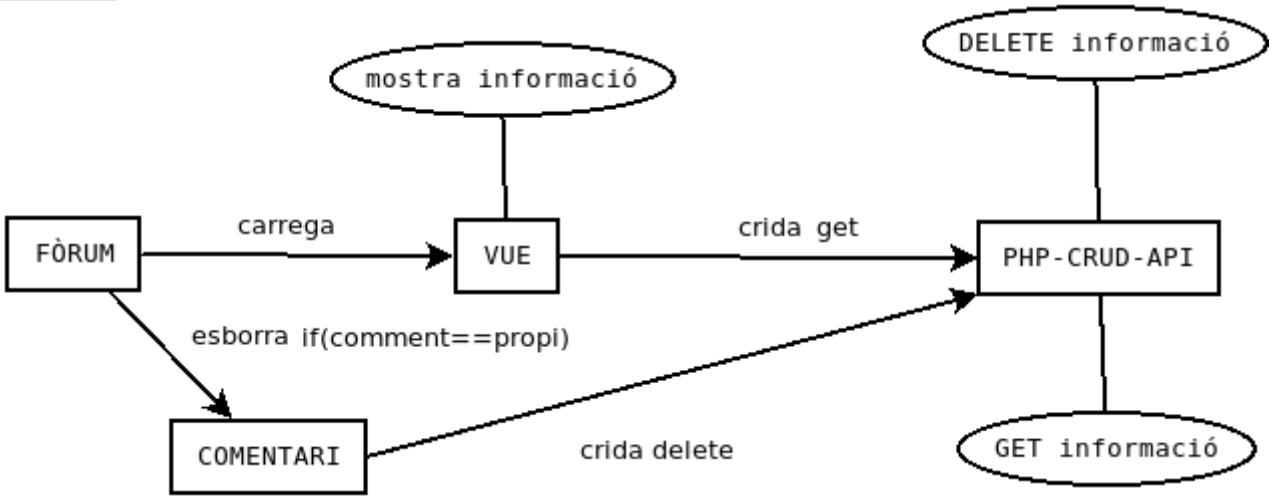
Mapa



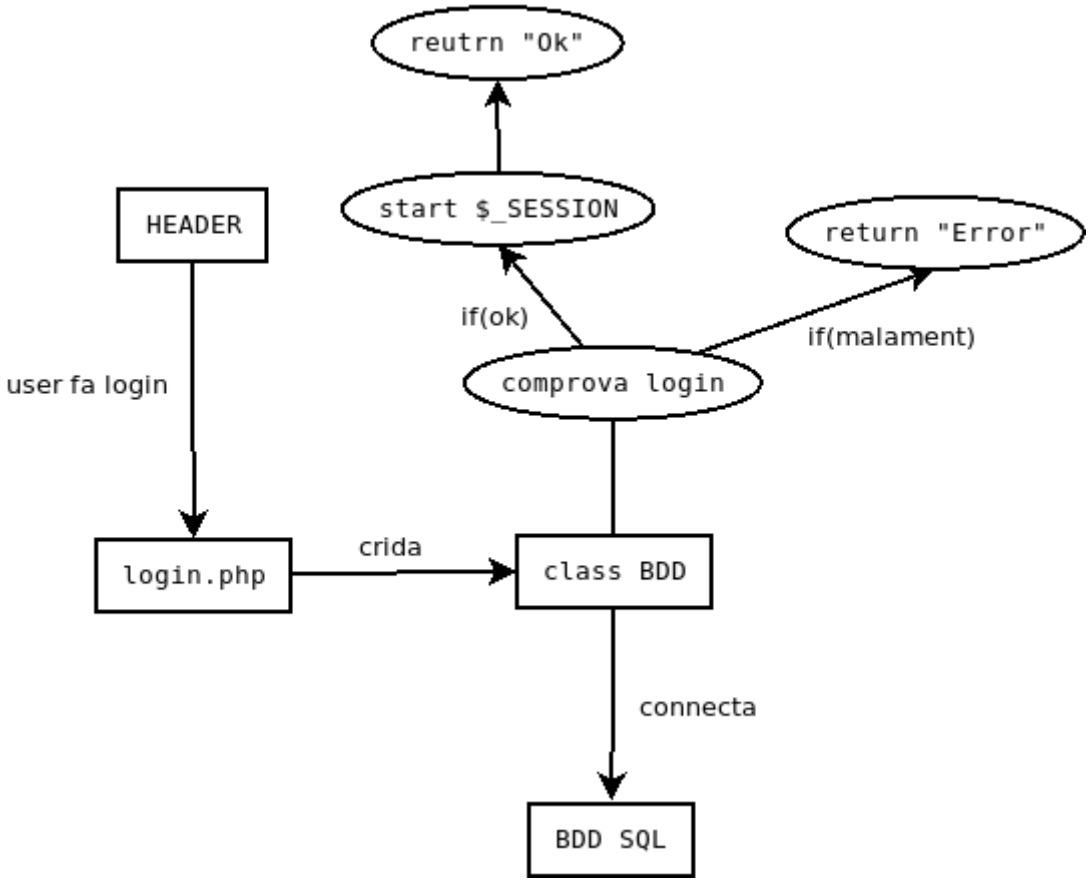
Test


Fa post
PHP-CRUD-API
if(positiu)
TEST
if(negatiu)
Mostra infromació
VUE
Crida
PHP-CRUD-API
GET informació

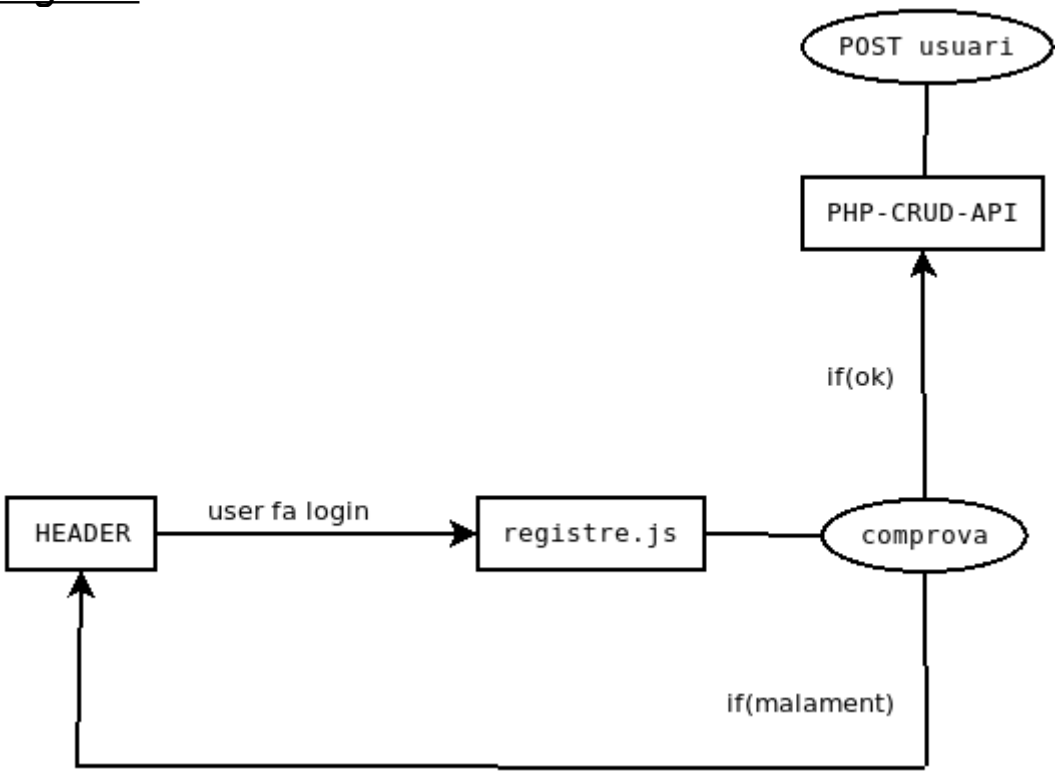
Fòrum


mostra informació
DELETE informació
FÒRUM
carrega
VUE
crida get
PHP-CRUD-API
esborra if(comment==propi)
COMENTARI
crida delete
GET informació

Login


reutrn "Ok"
start $_SESSION
return "Error"
HEADER
if(ok)
if(malament)
user fa login
comprova login
crida
login.php
class BDD
connecta
BDD SQL

Register


POST usuari
PHP-CRUD-API
if(ok)
HEADER
user fa login
registre.js
comprova
if(malament)