Project documentation:
Develop your Project with PHP

Authors: Guillermo, Jaime, Sayeed & Yulia

Table of content

Requirements

For this project, it is mandatory to comply with the following specifications:

- Frontend layer:
  - You must use HTML, CSS and JS
  - Use NPM to manage our frontend dependencies (use at least one dependency) -> in our case JQUERY and BOOTSTRAP
- Backend layer:
  - You must use PHP
  - Use Composer to manage our backend dependencies (use at least one dependency) -> in our case PHPUnit
  - Consume at least one third-party API with PHP -> itunes: *https://itunes.apple.com/search?*
  - Use PHPUnit to test at least three functions of your project
  - You must create your own database using local files like JSON, TXT, CSV, YAML or similar -> *userlogin.json*
- General requirements:
  - Use Git and Composer
  - Both the code and the comments must be written in English
  - Divide the tasks in sub-tasks so it is possible to associate each particular step of the construction with a specific commit.
  - Delete unused files or files not necessary to evaluate the project

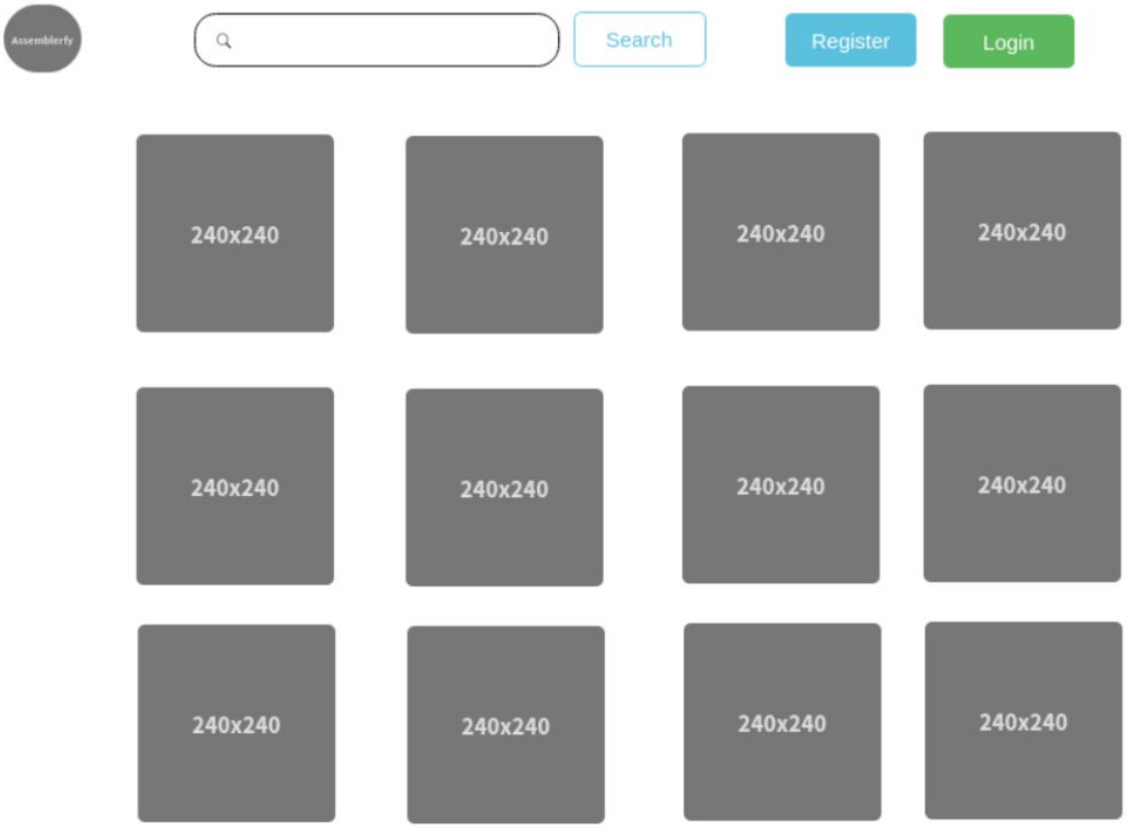
Wireframes

Here below is included a couple of the pages that we planned to show, firstly the main page *index.html* which shows the logo, the search bar and the login and register buttons; below this the results of the search, but it will show per default the search of the *top hits* songs.

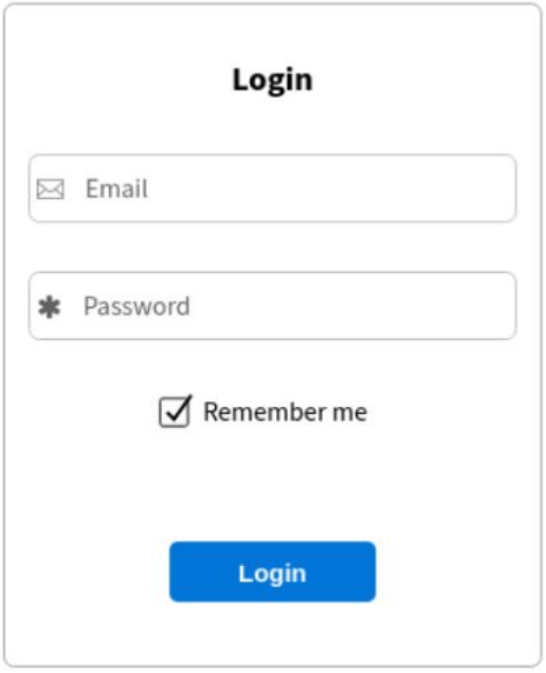
Secondly you can see the panel/form used as a page to log the user in or to register a new user

Thirdly the admin panel which will just show the information of the content played by each user.

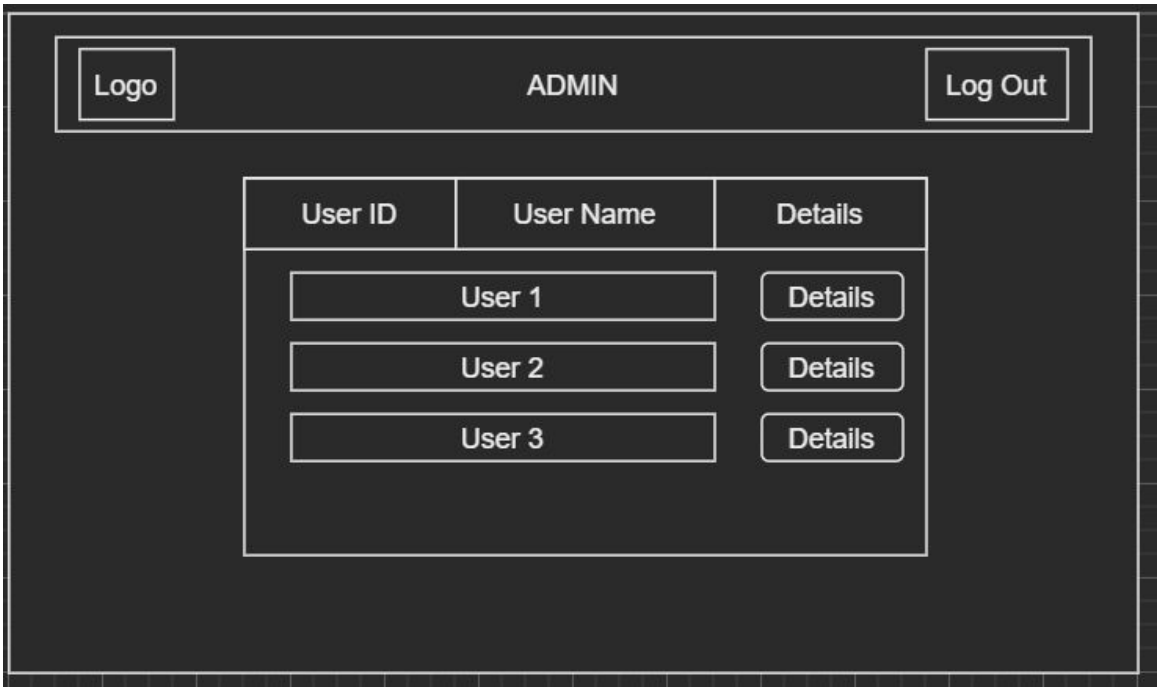
General overview



Login and Register page



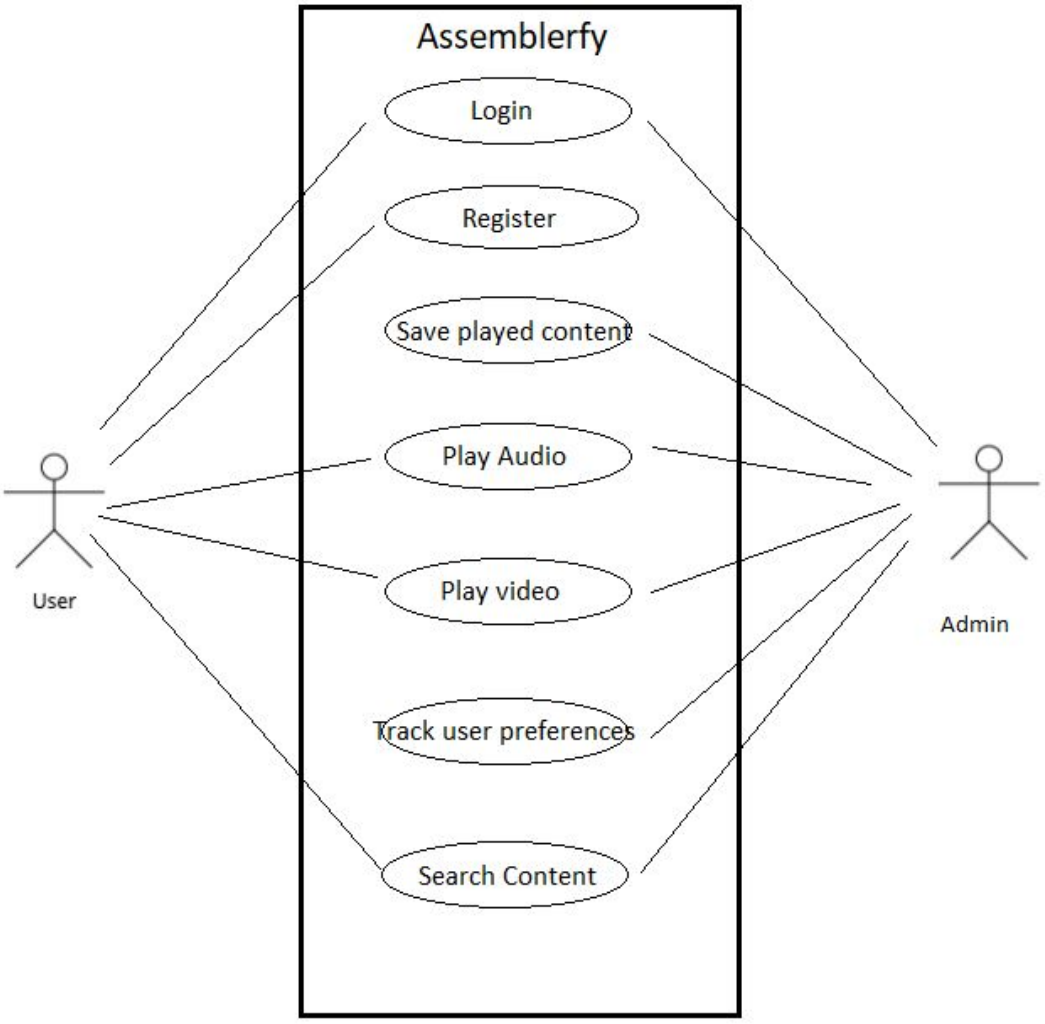
Preview of the admin panel



Use case diagram

The use case diagram should represent in a graphic, simplified manner the interaction possibilities of the different actors using a given platform (like a webpage) or program.

In this project, we could identify the following:



Incidents record

#1	Search with PHP
Description of the incident: JS sends get request to PHP, PHP returns data received with curl. Error when parsing the response from php in js due to incorrect format	
How was the issue resolved? Solved by adding curl_setopt($ch, CURLOPT_RETURNTRANSFER, true) to get response from API as string.	
What lessons could we learn from the incident? Use curl to send requests to APIs with PHP	

#2	Sending requests with curl
Description of the incident: Itunes API doesn't accept whitespaces in URL.	
How was the issue resolved? Solved by replacing them manually in PHP with string functions	
What lessons could we learn from the incident? Use the Itunes API properly with curl	

#3	Visual implementation of the multimedia content
Description of the incident: Different aspects of layout created problems with bootstrap, width of cards vary depending of the image, button of cards displaced depending on the title length, resolution of image	
How was the issue resolved? Itunes procured us with different img sources with different resolution, besides jquery lets us specify the resolution of images. Plus defining fixed height and width of internal elements inside card	
What lessons could we learn from the incident? How to defeat bootstrap (at least a little)	

#4	Problems with code assembler repo or VS Code
Description of the incident: When we did a pull sometimes the content of the local code was not being updated. Updated code only appeared when we pressed “Save” telling us that we will overwrite content (that wasn’t visible)	
How was the issue resolved? Not resolved,apparently it worked again by itself.	
What lessons could we learn from the incident? To be aware that unexpected and inexplicable problems can occur.	

#5	iTunes API
Description of the incident: When searching for song we send a GET request on a “keydown” event, but the API allow only 20 requests per minute per user, so if we type 20 characters within a minute, we had to wait	
How was the issue resolved? Changing the request from “keydown” (any key) to just sending the request after pressing ENTER	
What lessons could we learn from the incident? Work with external API restrictions	

#6	Authentication failed with CodeAssembler
Description of the incident: Sayeed was not able to do a Pull because “authentication failed” despite being logged in assembler with correct credentials, and with full access to the repo	
How was the issue resolved? Reset credentials of Code Assembler and reassign permissions of the repo	
What lessons could we learn from the incident? NS/NC	

#7	Backticks `` do not work well with *header(`Location: ../index.php`)*
Description of the incident: SWhen using the backticks to specify the header, PHP server does not interpret it well and does not redirect the page.	
How was the issue resolved? To pass a variable in the *header()* it was better to declare a variable before like $location and fill it with the parameter we want, and then do *header($location)*	
What lessons could we learn from the incident? How to avoid using backticks in PHP	

Lessons learnt

- Using an HTML structure as simple as possible, especially with a framework like Bootstrap.
- Breaking a complex problem into small pieces and implementing each one of them in an independent function (modularization of code).
- Sending API requests with PHP using CURL
- Working with SESSIONS
- Handle errors in forms
- Display multimedia content obtained from an API
- How to pass information between JS, HTML and PHP
- How to transform resolution of images
- Work with iTunes API