

Project Development Tracking

Technologies to apply

Once I've read carefully the whole assignment, I'm considering resolving it using React JS because I've realized that I won't need to develop any databases or overly complex backend implementations, which would take more time. This is because all the data is provided by the APIs.

Another alternative would be to develop the challenge using HTML, CSS and JavaScript. However, I want to take a step forward and apply React JS, which offers many benefits, such as:

- **Component-Based Architecture:** This means it's made up of reusable and modifiable components, facilitating construction, scalability and maintenance.
- **Virtual DOM:** it achieves high performance by rendering only the changed parts rather than the entire page.
- **JSX:** it's a JavaScript extension that allows writing HTML code within JavaScript, improving readability.

On the other hand, I will use Chakra UI to facilitate the building of components. Chakra UI is a library for UI design that provides building blocks for React applications.

UI/UX Design

For this stage, I used Figma to simulate the initial idea of what I wanted my page to look like.

First of all, I defined my color palette:



It's based on the four colors of Hogwarts houses.

Then, I designed the frames that will contain my app:

Welcome to the Wizarding World Fan Club!

Step into the magical world of Harry Potter and join fellow fans in celebrating J.K. Rowling's famous series. Explore Hogwarts houses, meet wizards, discover potions, ingredients, and spells, and share your thoughts in our feedback section

Your journey into the magical world awaits!



GRYFFINDOR

[Visit house](#)

HUFFLEPUFF

[Visit house](#)

RAVENCLAW

[Visit house](#)

SLYTHERIN

[Visit house](#)



Also, I choose this two fonts to work along with:

- The link to the project UI/UX design is:

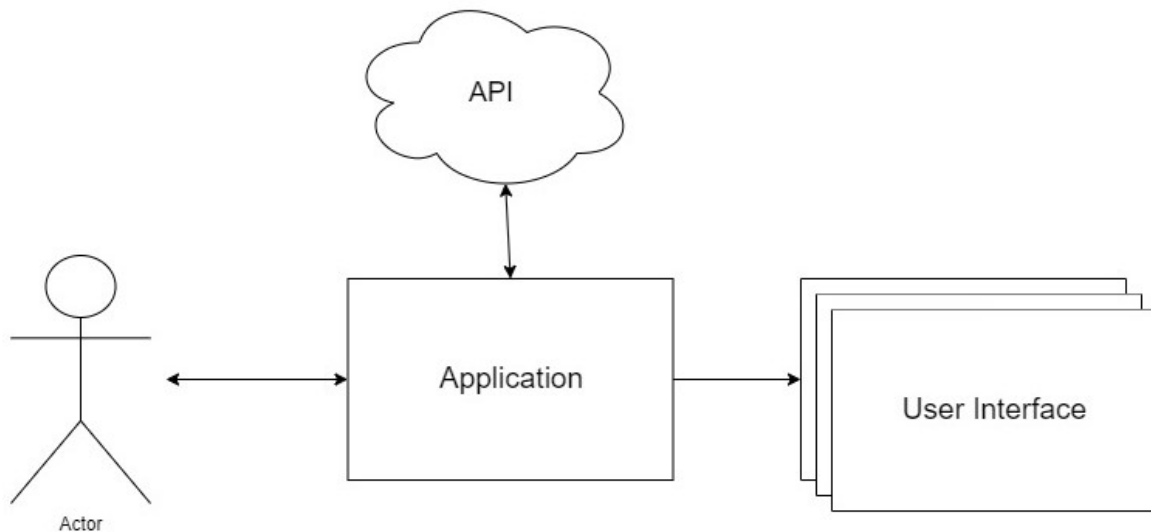
Project Development Tracking

Tool Development

In the initial phase of designing the system architecture, I developed a series of diagrams to visually represent both the high-level and low-level aspects of the architecture.

High Level Diagram

The high-level diagrams provide an overview of the system's structure, focusing on the major components and their interactions:



Actor: Represents the user interacting with the application via a browser

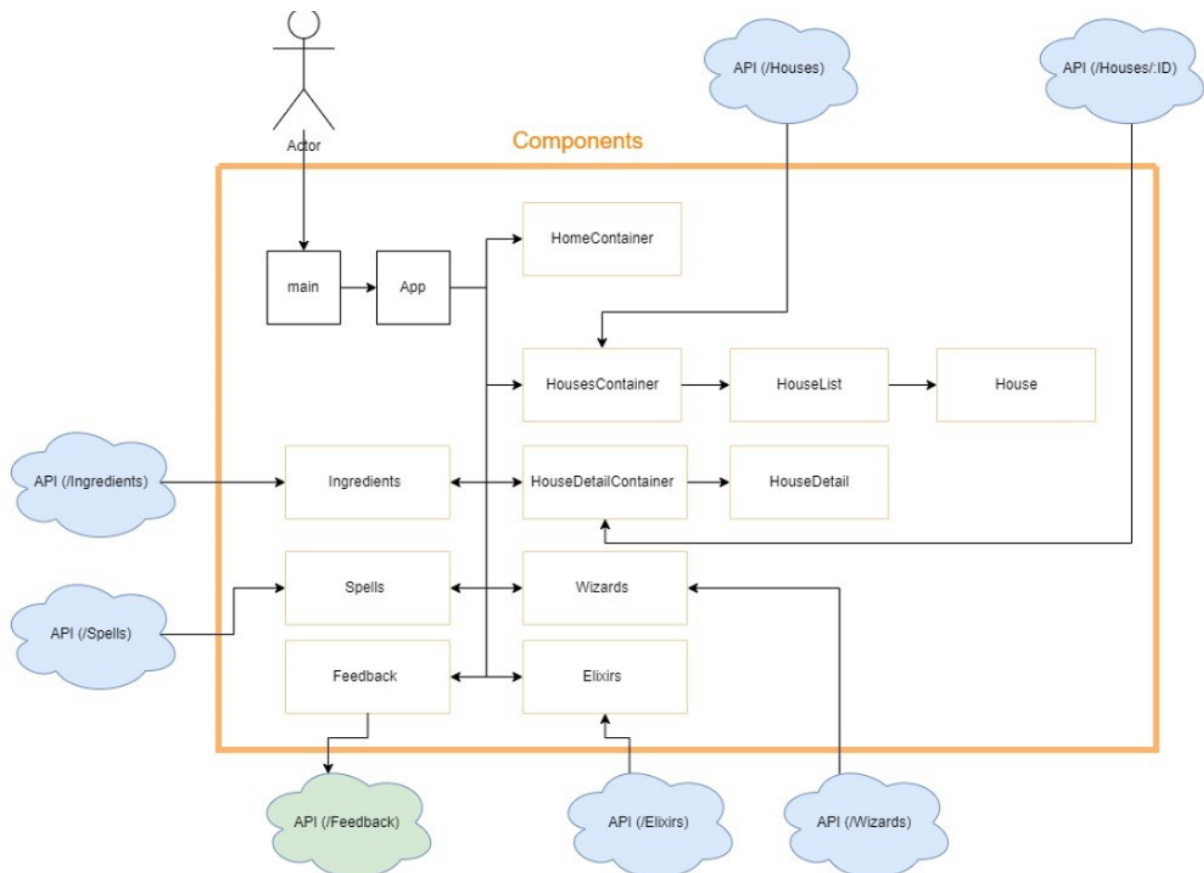
Application: Represents the main React application. It handles routing between different parts of the UI and manages state

API: Represents the external APIs that provide the data to the application

User Interface: Different parts of the application represented by React components. Each component might make specific API calls to fetch the data it needs.

Low Level Diagram

The low-level diagrams delve into the finer details of the system, showing specific parts, how they connect, and the data flow between them.



The blue clouds represent the API GET connections, while the green ones represent the POST interactions. Inside the orange box are all the system's React components necessary to display the UI.

Tool Implementation

Navbar

This component will always be shown. I started implementing this part first. I downloaded the font "Jim Nightshade" from <https://fonts.google.com/specimen/Jim+Nightshade?query=jim> for the page title.

Houses

This is the next step, in order to prioritize the mandatory tasks. In this case I decided to bring only the names and IDs of each house from the API <https://wizard-world-api.herokuapp.com/Houses>. With this information I created a page where cards will be exposed with the name, shield and visit buttons of each house. Since colors and images weren't specified, I decided to add them to a dictionary:

```
const default_color_house = "#B06E3F"
const default_img_house = "/img/default.png"
const dicc_color_house = {
  "Gryffindor" : "#990000",
  "Ravenclaw" : "#2376AD",
  "Slytherin" : "#1F8A70",
```

```

    "Hufflepuff" : "#DAA000"
  }
  const dicc_img_house = {
    "Gryffindor" : "/img/gryffindor.png",
    "Ravenclaw" : "/img/ravenclaw.png",
    "Slytherin" : "/img/slytherin.png",
    "Hufflepuff" : "/img/hufflepuff.png"
  }

```

In case the house isn't in the json or a new house is added in the future, the program won't break, default values were considered.

House Detail

When the button of "Visit house" is clicked, the website will redirect you to the house details page. This information is retrieved from the API <https://wizard-world-api.herokuapp.com/Houses/:id> where the ID is obtained from the URL params (the button sets the url with the house ID)

Feedback

This page is a dedicated feedback space for users. Users select a feedback type and provide their comments in a textarea field. Upon form submission, the system utilizes the API at <https://wizard-world-api.herokuapp.com/Feedback>. The API returns a response indicating the success or failure of the POST action.

Amplitude Implementation

After reading Amplitude's documentation, I set up my own organization on their app and created a project called 'Harry Potter Fan Club'. Then, I began developing a tracking plan. Along with using some pre-made tracking options, I also created my own custom events:

[continue ...]


Application Repository

<https://github.com/valenpontaut/Harry-Potter-fan-club>

Hosting server

I chose to host my app on Netlify. I connected my GitHub repository to enable automatic deployment every time I make changes to my code.

This is the link:

HP fan club
 <https://hpfanclub.netlify.app/>