|  |
| --- |
| School of Software Design and Data Science |
| **Full stack JS App** |
| WEB422 - Project |
|  |
| **Daniel Park, Angela Marie Tagudin Chua, Amany Omar**  **Submission Date** |
| **4/15/2025** |

|  |
| --- |
| This document explains how to build a frontEnd dashboard that consume backend API …………………………. |

Table of Contents

[Task 1: Pick Your Store and Product/Service Inventory 2](#_Toc174091863)

[Task 2: Project Planning, Wireframing, Layout, Dynamic Content 3](#_Toc174091864)

[Task 3: Authentication/Authorization 4](#_Toc174091865)

[Task 4: "Favourites" functionality 5](#_Toc174091866)

[Task 5: "Route Guard" functionality 6](#_Toc174091867)

[Task 6: deployment 7](#_Toc174091868)

[Bonus questions 8](#_Toc174091869)

[Project planning/sharing tasks 9](#_Toc174091870)

[Project performance check 10](#_Toc174091871)

[Summary 11](#_Toc174091872)

# Task 1: Pick Your Store and Product/Service Inventory

(Describe the major steps for identifying product/service/API)

For this project, we chose a marketplace model that offers a variety of Pokemon cards.  
  
**Store Name, Slogan, and Theme:  
Store Name:** Poké Mart

**Slogan:** “Your #1 Hub For Pokémon Cards”

**Theme:** A modern digital space for browsing and collecting Pokémon cards

The major steps for identifying our services, products, and API were:

1. **Initiation:** We brainstormed multiple different ideas and finalized on creating a  
   marketplace for Pokemon cards as it was a shared interest and we felt that it was a  
   niche product to work with.
2. **Research:** We discussed and researched through a variety of API’s and products  
   that we could use for our Pokemon marketplace but ended up finalizing and  
   choosing the Pokemon TCG API. This API provided all of the information we  
   needed along with data sets of the products (cards) we were going to list.
3. **Service Inventory:** We defined our product inventory based on the available data  
   and information from the API, making sure that our inventory covers necessary  
   details such as name, image, description, prices, and categories for each card.
4. **Task Management:** Utilized Jira to issue and track project task implementation.

# Task 2: Project Planning, Wireframing, Layout, Dynamic Content

(Describe the project planning (Gantt Chart) and major steps for designing the layout (Figma) and visualizing data using Card design)

When designing the wireframes on Figma, we made sure to consider all possible routes that the user might take such as the Favorites page, the page that contains the entire inventory, and more. We wanted a layout that was both easy to read and was easily navigational.

Each wireframe only contained a basic layout for what we wanted to do., as we could develop the visual design later on. Then, after creating each basic frame, we mapped out a basic prototype for how the user can interact with the website. We also exported each frame as a separate png file for reference later on.

# Task 3: Authentication/Authorization

(Describe the major steps on how to make JWT user authentication in the app)

When the user registers in the site, their inputted password is automatically encrypted using b-crypt by the User schema.

Afterwards, when the user tries to login to the site, it compares the inputted password with the hash version on the server. If the password is valid, the server sends a JWT object to the client, where it is stored in local storage.

In the backend, we use jwt.verify to double-check if the user is authorized to visit the route. (Unfortunately, we couldn’t figure out how to configure passport into the Next.js middleware.). For instance, if a user that does not have a valid token tries to get a list of their favorites, they will be greeted with a 401 status page. We used this on routes that involves the favorites and search history list of any user.

# Task 4: "Favourites" functionality

(Describe the major steps on how to make “favorite products” in the app)

When a user goes to a specific card’s page, if the user is logged in, it will check to see if the card is it their Favorites list or not via the user API. The user will then be shown a “Add To Favorites” button or a “Remove From Favorites” button depending on if the card is already their favorite or not.

When visiting the Favorites page, the website will get the user’s Favorites list through the API once more. The user can view 4 cards per page when the Favorites data is fully loaded.

# Task 5: "Route Guard" functionality

(Describe how to implement protected routes)

To ensure only registered users could access features ike searching and favoriting Pokemon cards, we implemented a Route Guard in our app using Next.js and Jotai to restrict the favorite page and search page to authenticated users only, by redirecting the user to login page when attempting to search or favorite.  
  
Steps:

1. Created RouteGuard.js Component inside components/ folder. It wraps around the main layout and checks user login status using loggedInAtom from jotai.
2. Used PUBLIC\_PATHS array including to only protect the /favorites and /search pages. This made the homepage and public content accessible without foricing the user to log in immediately.
3. Used useEffect to check login status
4. Reused the updateAtoms() function from our login page to re-sync user data (favorites and history) to ensure favorites didn’t disappear after refreshing the page.

# Task 6: deployment

(Describe the major steps for deployment)

Once the Poke Mart project was initialized, added and pushed to github repository. Created initial homepage with “Website coming soon!” and deployed to Vercel.

Steps:

1. Created initial homepage with initial message.
2. Pushed our code to GitHub under a private repository
3. Signed into vercel.com using GitHub authentication
4. Imported our GitHub repo to Vercel and selected the correct project directory to deploy.
5. Clicked “Deploy”
6. Vercel generated live URL that automatically updates evertime we push to main.

\*Learned how to read, understand, and debug deployment errors

# Bonus questions

(Describe the major steps for designing the bonus question)

# Project planning/sharing tasks

(Describe how did you divided the work and did project planning. Have you changed the day-1-planning/milstones/deliverables that we did at the beginning of project in the class?)

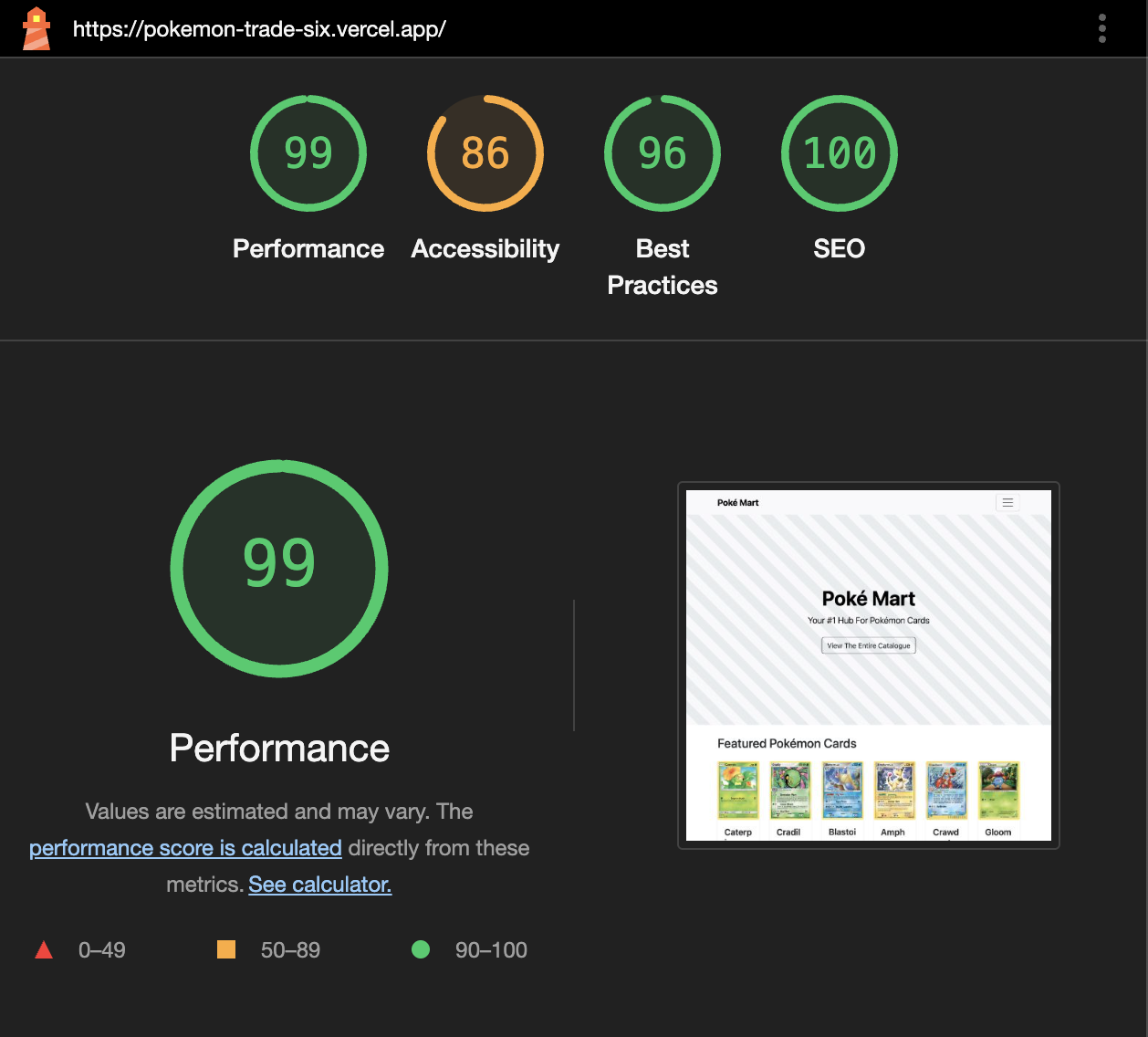
We started the project by drafting initial planning and distribution of some tasks. We changed after that to using Jira, and it was easier to manage our tasks.

Steps of using Jira:

1. Used already created team account
2. Added team members as collaborators and the professor
3. Issued Main tasks based on the required deliverables and tasks instructed by the professor, and allowed each team member to assign tasks to their name to implement
4. Each task could be divided into subtasks to break down into smaller manageable tasks and each main task was associated with a parent (Phase 3, phase 4), which helped in managing the tasks.
5. Used the tracking feature in jira, for each task (To Do, In progress, Done)

# Project performance check

(check your project performance using google lighthouse and add the screenshot of your current app status here )



# Summary

(Describe your experience in this project, share your feedback on how this project was align with the course objective and how do you think the project can be changed to provide a better experience, …)

Angela: At least for me, it was cool to combine what I learned from both Web322 and here into this new site. I felt that there was a lot of cool things I learned, like how to really combine data with the power of React. As I was in charge of the user backend and the Favorites functionality, I felt that I could’ve used localStorage more to reduce the number of API calls like in Assignment 2. Still, doing this project was a lot of fun, and I’m eager to dive more into React on my own time.

Amany:  
  
It was a special experience that I learned from:

* It was my first time to use lighthouse tool to evaluate the performance of my built website.
* Learned more about understanding the errors on Vercel and fixing them
* Experienced applying what we learned in the WEB422 class such as state management using jotai, atoms, and how to choose between useSWR and useEffect, using RouteGuard feature, examining API, and testing user API using Thunder Client.

It was valuable experience and aligned very well with the course objectives.

* One of the most difficulties I had is resolving conflicts when pulling and pushing to GitHub, but it taught me a lot about version control system.