Weekly Report

10/06/2024 – 17/06/2024

Group ID: **3**
Project Name: Gogle
Prepared by: **Lê Duy Anh**

Team members:
22127012 – **Le Duy Anh**
22127083 – **Cao Huu Khuong Duy**
22127219 – **Huynh Cao Tuan Kiet**
22127255 – **Ly Dinh Minh Man**
22127360 – **Vo Nguyen Phuong Quynh**

1. Achievements since last week:

STT	Description	Due Date	Responsibility	%Complete
1	Reviewing Project Plan/Vision/ Iterations	14/06/2024	All members	100%
2	Defining database schema	14/06/2024	Le Duy Anh Huynh Cao Tuan Kiet	100%
3	Token-based access control middlware	13/06/2024	Le Duy Anh	100%
4	Backend GetPlace API	13/06/2024	Huynh Cao Tuan Kiet	100%
5	Login and Registeration Forms with APIs Communication	10/06/2024	Ly Dinh Minh Man	100%
6	Implementing Header/Footer Component	10/06/2024	Vo Nguyen Phuong Quynh	100%
7	Reviewing and Refactoring code	12/06/2024	Cao Huu Khuong Duy	100%
8	Search Algorithm Proposing	14/06/2024	Cao Huu Khuong Duy	100%

1. Defining Database Schema

This task involves creating a blueprint for how data will be stored and organized within the project's database. It will define:

- Tables: The structures that will hold specific types of data.
- Data Types: The format of the data stored in each record (e.g., text, numbers, dates).

2. Token-based Access Control Middleware

This task involves implementing a system that manages user access to the application's functionalities. It will use tokens (unique identifiers) to:

- Authenticate users: Validate user login credentials.
- Authorize actions: Determine which actions a user is permitted to perform based on their role or permissions.

3. **Backend GetPlace API**

This task involves building an API endpoint that retrieves place data from the backend server. This endpoint will be accessible by the frontend application to:

- Fetch place information: Retrieve details about specific places.
- Potentially filter or search for places based on certain criteria.

4. **Login and Registration Forms with API Communication**

This task involves developing user login and registration forms on the frontend that interact with the backend APIs:

- Login Form: Will send user credentials to an API for authentication and provide access to the application upon successful login.
- Registration Form: Will send user data to an API for creating a new user account.

5. **Implementing Header/Footer Component**

This task involves building reusable components for the application's header and footer sections. These components will likely:

- Provide consistent visual elements across all application pages.
- Potentially include elements like navigation menus, branding logos, or copyright information.

6. **Reviewing and Refactoring Code**

This task involves reviewing existing code for potential improvements. The goal is to:

- Enhance readability: Make the code easier to understand for yourself and other developers.
- Improve maintainability: Ensure the code is easier to modify and update in the future.
- Optimize performance: Identify areas where code can be made more efficient.
- Refactoring might involve techniques like:
    - Renaming variables and functions for clarity.
    - Breaking down complex logic into smaller, more manageable functions.
    - Eliminating code duplication.

7. **Search Algorithm Proposing**

This task involves designing a method to search for information within the application. This algorithm will determine how the application finds and retrieves relevant data based on user input. The proposal will likely consider factors such as:

- Type of data being searched (text, numbers, dates).

- Desired search functionality (exact matches, partial matches, filtering).
- Performance and efficiency of the search process.

2. Issues and impacts:

1. Completing Project Plan/Vision/Iterations

- **Issue:** An incomplete or unclear project plan can lead to development inefficiencies, missed deadlines, and confusion among team members. There might be a lack of clarity on the project's goals, specific deliverables for each phase, and how iterations will be handled.
- **Impact:** Delays in project progress, wasted resources, and potential scope creep (adding unplanned features) could occur due to a lack of a solid roadmap. Team members might be unsure of priorities, leading to wasted effort on non-essential tasks.

2. Defining Database Schema

- **Issue:** An poorly designed database schema can lead to data redundancy, difficulty in managing and querying data, and potential security vulnerabilities.
- **Impact:** The application might experience performance issues as data retrieval becomes complex. Maintaining and scaling the application becomes more challenging as the data structure becomes cumbersome. Additionally, security risks might arise if sensitive information isn't stored securely due to a flawed schema.

3. Token-based Access Control Middleware

- **Issue:** Implementing a weak or insecure access control system can lead to unauthorized access to sensitive user data and application functionalities.
- **Impact:** Breaches or leaks of user data could occur if the token-based system is compromised. Malicious users might gain access to unauthorized features, potentially causing disruption or damage to the application.

4. Backend GetPlace API

- **Issue:** An poorly designed API can lead to data inconsistency, security vulnerabilities, and difficulty for the frontend application to interact with it effectively.
- **Impact:** Issues like data corruption or conflicts might arise if the API doesn't handle data exchange correctly. Security concerns could emerge if the API is not properly authenticated or exposes sensitive information unintentionally. Additionally, the frontend application might struggle to retrieve place data efficiently, impacting user experience.

5. Login and Registration Forms with API Communication

- **Issue:** Buggy or insecure login/registration forms can lead to failed login attempts, difficulties in user account management, and potential security vulnerabilities.
- **Impact:** Users might experience frustration due to login issues. The application could be susceptible to attacks like brute-force attempts or credential stuffing if authentication isn't robust. Additionally, user data might be compromised if the communication with backend APIs isn't secure.

6. Implementing Header/Footer Component

- **Issue:** Inconsistent or poorly designed header/footer components can create a disjointed user experience and make maintaining the application's layout more difficult.
- **Impact:** Users might find the application confusing to navigate if the header or footer elements lack consistency across pages. Updating the application's layout becomes more time-consuming if these components aren't reusable.

7. Reviewing and Refactoring Code

- **Issue:** Unreadable, poorly maintained, or inefficient code can lead to difficulty in troubleshooting bugs, adding new features, and overall project maintenance.
- **Impact:** Debugging issues becomes time-consuming due to the complexity of the code. Adding new functionalities might be challenging if the codebase is not well-structured. The project might become stagnant or require significant resources to maintain a low-quality codebase.

8. Search Algorithm Proposing

- **Issue:** An inefficient or poorly designed search algorithm can lead to inaccurate or incomplete search results, slow search performance, and a frustrating user experience.
- **Impact:** Users might find it difficult to locate the information they need within the application. Slow search times can lead to user frustration and decreased engagement. Additionally, inaccurate search results could provide misleading information.

3. Next week's goals:

This coming week, our primary objective is to finalize all project documentation, ensuring all plans and details are captured and consolidated.

Here's a breakdown of key tasks:

1. **Project Plan, Vision, and Iteration Review:**
   - We'll kick off the week by conducting a thorough review of the project plan, vision document, and planned iterations.
   - This review aims to identify any inconsistencies or missing information across these documents.
   - The goal is to achieve a cohesive picture of the project's overall direction and key milestones.
2. **Finalizing the Use-Case Model:**
   - Following the review, we'll focus on finalizing the use-case model.
   - This includes making sure all functionalities and user interactions are clearly defined.
   - Ensure the model is comprehensive and accurately reflects the project's intended actions.
3. **Updating Project Plan and Vision Document:**
   - Based on the review findings, we'll update the project plan and vision document.
   - This may involve incorporating missing details, clarifying specific aspects, or aligning them with the finalized use-case model.
   - These documents will become the central reference point for project direction and progress tracking.
4. **Developing Use-Case Specifications:**
   - With the finalized use-case model in hand, we'll then delve into developing detailed use-case specifications.
   - These specifications will outline specific steps, actors involved, and expected behavior for each use case.
   - This level of detail will be crucial for implementation and testing phases.
5. **Completing Weekly Report #5:**
   - This report will highlight the progress made on documentation finalization and capture any key decisions or discussions during the week.

By successfully completing these tasks, we'll establish a strong foundation for project execution moving forward. This comprehensive documentation will ensure clarity, alignment, and efficient development progress.

STT	Description	Due Date	Responsibility
1	Reviewing Project Plan/Vision/ Iterations	20/06/2024	All members
2	Use-case model	24/06/2024	Huynh Cao Tuan Kiet
3	Update vision document + project plan	24/06/2024	Cao Huu Khuong Duy
4	Use-case specification	24/06/2024	Ly Dinh Minh Man Vo Nguyen Phuong Quynh
5	Weekly Report 5	20/06/2024	Le Duy Anh