Sri Lanka Institute of Information Technology

Programming Applications and Frameworks (IT3030)

Continuous Assignment – 2025, Semester 2 Initial Document

GROUP ID: Y3S2-WD-34

Group Details:

Name	IT Number
Thilakarathna H.M.S.D	IT 22 62 6110
Prabhashwara G.A.D	IT 22 19 3704
Lakshan V.G.S	IT 22 56 2210
Damhari P.A.M.T	IT 22 57 0208

Table of Contents

Project Description

1.1 Introduction

The **SkillHive Platform** is a web-based application where users can share and learn various skills ranging from coding and cooking to photography and DIY crafts. It fosters a collaborative community of users who post content, provide feedback, and track learning progress. This project aims to develop a robust system using **Java Spring Boot** for the backend and **React** for the frontend.

1.2 Objectives

1. **Enable Skill Sharing:** Provide a platform where users can post tutorials, short videos, and images to share knowledge.
2. **Support Learning Plans:** Allow users to create, update, and track structured learning plans.
3. **Encourage Interaction:** Implement features such as liking, commenting, and following to enhance community engagement.
4. **Provide Secure Access:** Ensure users can safely log in using OAuth 2.0 authentication and manage their data securely.

1.3 Scope

- The application will focus exclusively on **web-based functionalities** (no mobile application scope at this stage).
- Users can upload up to **three photos** or **short videos (max 30 seconds)** for each post.
- The solution covers everything from user registration, authentication, and post creation to content moderation and notifications.

Third-party integrations, such as external social media APIs, are only for OAuth 2.0 login.

1 Functional Requirements

1.1 Functional requirements for the Client Web Application

1. **User Interface for Authentication:**
   - Integrate OAuth 2.0 login and registration forms.
   - Provide user profile pages with personalization options.

2. **Skill-Sharing Posts & Learning Progress:**
   - Let users upload multiple photos or short videos along with descriptions.
   - Offer progress update templates for tracking learning milestones.

3. **Engagement & Community:**
   - Provide interfaces for commenting on and liking posts.
   - Display follower/following lists with the ability to follow other users.

4. **Notifications & Alerts:**
   - Show real-time or periodic notifications for likes and comments.

5. **User Profiles & Settings:**
   - Profile editing (name, bio, profile picture).
   - Privacy settings to manage visibility and notifications.

1.2 Functional requirements for the REST API

1. **User Authentication & Authorization:**
    - Allow users to sign up and log in using OAuth 2.0 (e.g., Google or Facebook).
    - Secure endpoints with role-based access control (where applicable).
2. **Resource Management (CRUD):**
    - **Posts:** Create, read, update, and delete posts.
    - **Comments:** Add, edit, or delete comments on posts.
    - **Likes:** Like or unlike posts.
    - **Learning Plans:** Create, update, and delete structured learning plans.
3. **Notifications:**
    - Notify users when their posts receive comments or likes.
4. **Rate Limiting & Caching:**
    - Implement rate limiting to safeguard against misuse.
    - Utilize caching headers or mechanisms to improve performance.

2 Non Functional Requirements

2.1 Non-Functional Requirements for the Client Web Application

1. **Usability:**
    - Provide a simple, intuitive UI design that even non-technical users can navigate.
2. **Compatibility:**
    - Ensure cross-browser compatibility (Chrome, Firefox, Safari, Edge).
    - Responsive design for varied screen sizes.
3. **Performance:**
    - Minimize page load times and use efficient resource loading (e.g., lazy loading images).

4. **Reliability:**

    - Maintain high availability with minimal downtime.
    - Graceful error handling and fallback messages for interrupted services.

2.2 Non-Functional Requirements for the REST API

5. **Performance:**

    - Aim for minimal response time and quick data retrieval, even under high load.

6. **Scalability:**

    - Design to support a growing number of users and interactions without sacrificing performance.

7. **Security:**

    - Use SSL/TLS for data transmission.
    - Protect endpoints with Spring Security and employ secure coding practices.

8. **Maintainability:**

    - Implement clear separation of concerns (controllers, services, repositories).
    - Write clean, modular code with comprehensive documentation.

3 Work Distribution

3.1 User Management

Users can create an account by registering with **social media logins (OAuth 2.0)**, such as **Google or Facebook**, or by using their **email and password**. To enhance security, email-based registrations require a **one-time password (OTP) verification**, ensuring that only valid email addresses can be used. Upon successful registration, a **user profile is automatically generated**, including default settings like **name, profile picture, and bio**.

Users can **view and manage their profiles**, update personal information, and search for other users by **name or interests**. They can also explore profiles of users they follow or those who share relevant content.

Profiles are **fully customizable**, allowing users to update their **profile picture, bio, skills list, and privacy settings**.

Additionally, users can personalize their experience by modifying **notification preferences and communication settings**.

If a user chooses to leave the platform, they have the option to **delete their account permanently**, which will remove all associated data, including **posts, comments, and followers**.

3.2 Achievements Management – Thilakarathna H.M.S.D

Users can create and upload achievements in various formats, including titles, descriptions, and media (such as certificates or images), to showcase their milestones and personal accomplishments. Each achievement can include a brief description, the date of achievement, and any supporting media to add context.

Users can view all of their achievements in a dedicated section on their profile, where they can see details like the achievement title, description, date, and media. They can interact with the achievements by updating, editing, or removing them as needed.

Additionally, users have full control over their achievements, including the ability to edit the title or description, upload new media, or delete any achievement they no longer wish to display on their profile.

3.3 Content Management – Prabhashwara G.A.D

Users can create and upload **posts** in various formats, including **text, images, or short videos (up to 30 seconds)**, to showcase their **skills or learning progress**. Each post can include **descriptions, tags, and categories** (such as **coding, cooking, or photography**) to enhance **searchability** and help users find relevant content.

Users can **browse and explore** all available posts, searching by **topics, skills, or usernames**. Posts feature **interactive elements**, allowing users to **like, comment, and share** content to engage with the community.

Additionally, users have **full control** over their posts, with the ability to **edit or update descriptions, replace media, or delete posts** if they no longer wish to share them with the platform.

3.4 Social Engagement Management – Lakshan V.G.S

Users can engage with content by **liking** or **commenting** on posts, fostering interaction within the community. They can **follow** other users to stay updated on their latest posts and activities, with a **personalized activity feed** displaying content from those they follow.

Users can **view and manage interactions** on their own posts, including **likes** and **comments**, while also reading and responding to comments on posts they are interested in.

Additionally, users have control over their **comments**—they can **edit**, **delete**, or **adjust visibility settings** (**public** or **private**). **Post owners** can moderate discussions by **deleting unwanted comments** on their posts.

If users wish to **curate their feed**, they can **unfollow** others at any time to stop seeing their updates.

3.5 Learning Plan Management – Damhari P.A.M.T

Users can create **structured learning plans** for any skill they wish to master, incorporating **topics, resources** (articles, videos, etc.), and a **timeline for completion**.

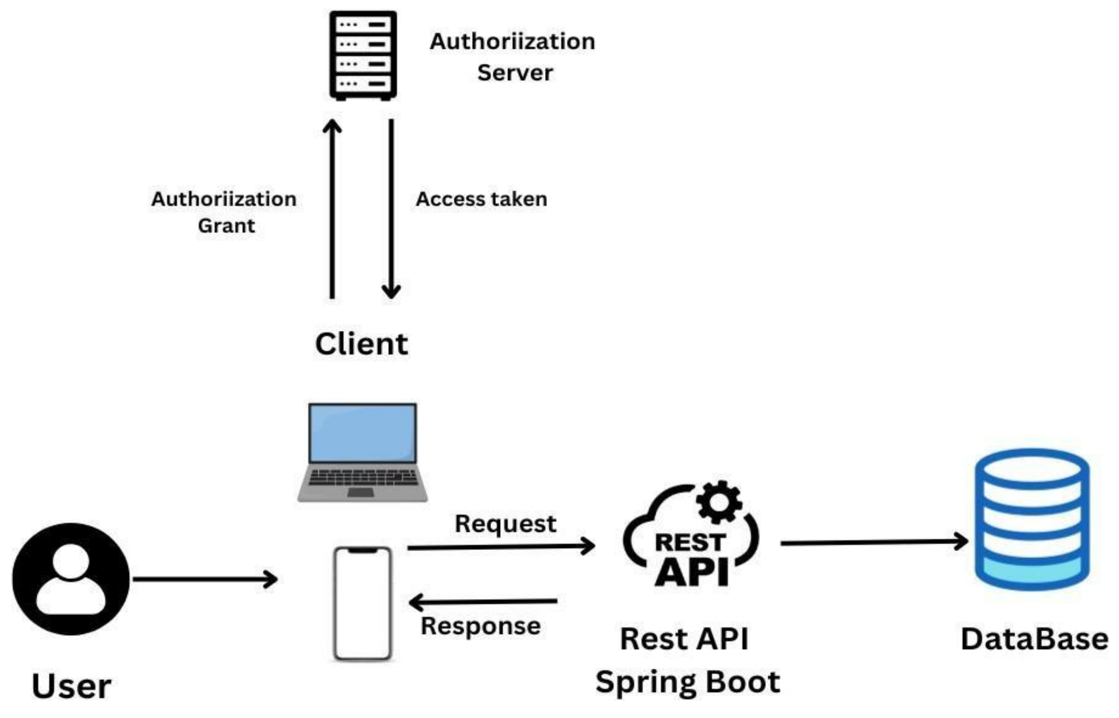
To stay on track, they can set **milestones** within their plans and **monitor their progress** over time. Users have the ability to **view and manage their own learning plans**, ensuring they follow their **intended schedules**.

Additionally, **public learning plans** from other users can be browsed for **inspiration**, allowing users to 3.5.1.1 **follow others' progress** and adopt **effective strategies**.

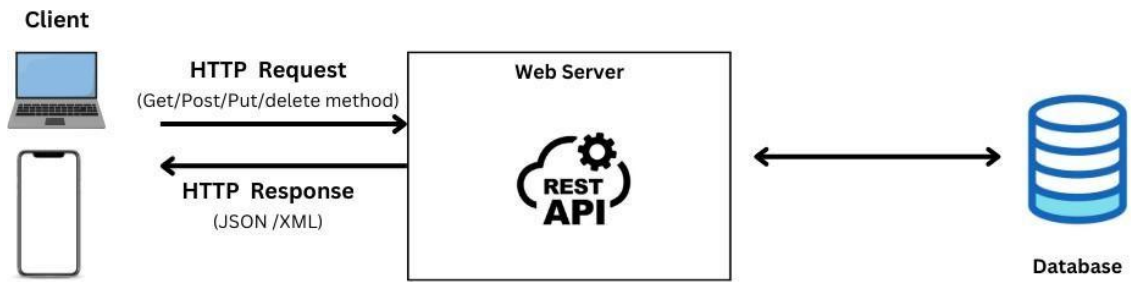
Plans are **fully customizable**, enabling users to **add new resources, adjust timelines, or refine milestones** as needed.

If a user no longer wishes to **track or share** their learning journey, they can **delete their learning plan**, removing it from the platform.

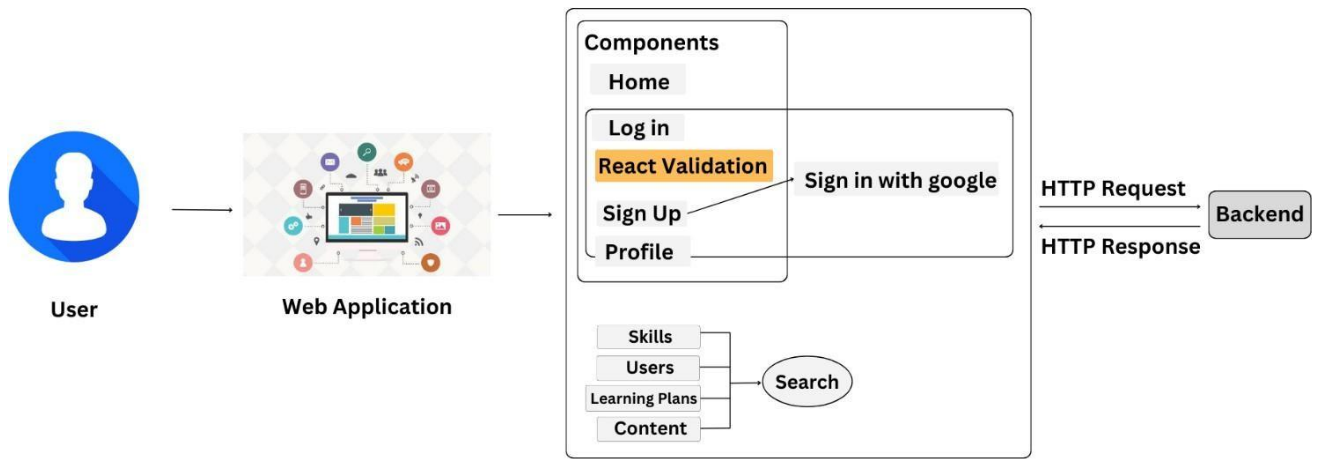
4 Overall Architecture Diagram



5 REST API Architecture Diagram



6 Client Web Application Architecture Diagram



7 Gantt chat

Task	Timeline (March – May) 2025													
	Week 1 (11-17th)		Week 2		Week 3		Week 4		Week 5		Week 6		Week 7	
Project planning.	🟦													
Work Distribution	🟦													
Initial documentation		🟦												
Adjusting project according to feedbacks			🟦											
System Architecture & Design				🟦	🟦									
Backend Development						🟦	🟦							
Frontend Development								🟦	🟦					
Integration & Testing										🟦				
Deployment & Version Control											🟦	🟦		
Documentation & Reviews													🟦	
Final Submission														🟦

8 References

- **Spring Boot & Spring Security Documentation**
  https://spring.io/projects/spring-boot
  https://spring.io/projects/spring-security

- **React Official Documentation**
  https://react.dev/

- **OAuth 2.0 Guide**
  https://oauth.net/2/

- **Database & SQL Basics**
  https://www.mysql.com/

- **Architecture Diagram Best Practices**
  https://nulab.com/learn/software-development/architectural-diagrams-what-to-know-and-how-to-draw-one