WAWASAN OPEN UNIVERSITY

Integrated Application Development

Project Presentation
Course Code : DSE204/03 & MCDSE104/03-FT
Course Lead: Tok Bee Choo (Grace)

Start Date	: 25-06-2025
End Date	: 06-07-2025
Submission Date	: 06-07-2025

Student Name	: Benjamin Tan Zhi Hong
Student ID	: 041240323
Presentation Date	: 02-07-2025

Contents

1. Introduction

- **Project Title**: Know-Your-Neighborhood
- **Objective**: Enhance the existing application with API-based login
- **Scope**: To research of different APIs and identifying any potential security issue and build a login with selected API in existing website.

2. APIs and Their Role

What is an API?

- Application Programming Interface (API) enables communication between different software systems.
- Acts as an intermediary, allowing data and functionality exchange without exposing internal code.

Role and Need for APIs

- Enhances application scalability and modularity.
- Enables rapid development without building it from scratch every time.
- Enables integration of third-party services

2. APIs and Their Role

APIs used in KYN?

- **Google OAuth2 API**: Secure authentication for user login.
- **Google Weather API**: Provides real-time and forecasted weather data.

3. Relationship Between API and SDK

Relationship:

- APIs are components within SDKs.
- SDKs provide a framework to use APIs efficiently, often with language-specific libraries (e.g., JavaScript SDK for Google APIs).

Example in KYN:

- Google OAuth2 API is used directly in Login.js via HTTP requests, but Google provides an SDK for easier integration in larger applications.

4. Types of APIs and Their Uses

1. **REST APIs**:
    1. Allow different software application to communicate over the internet using HTTP methods.
    2. Use Case: Data Exchange & Scalability

2. **OAuth APIs**:
    1. Used for secure authentication and authorization.
    2. Use Case: 3rd party Authentication and Secure API Access

5. API Security Issues and Evaluation

Potential Security Issues:

- **Data Exposure**: APIs may return sensitive information (e.g., user email, name, or location) beyond what is necessary, violating the principle of least privilege.
- **Rate Limiting**: Uncontrolled API calls can lead to service abuse or cost overruns.
- **Weak Authentication:** If API keys, tokens, or credentials are not securely managed, attackers can gain unauthorized access.

5. API Security Issues and Evaluation

Evaluation of Google OAuth2 API:

Strengths:

- Ease of integration, Google OAuth2 is widely adopted standard
- Secure Token Management, provide short-lived access tokens and refresh tokens, ensuring secure authentication

Weaknesses:

- Dependency on Google policies

Security Mitigation:

- Store tokens securely on server side, not in client-side storage such as local storage.
- Limit token scope to openid, profile, email.

5. API Security Issues and Evaluation

Evaluation of Google Weather API:

Strengths:

- Reliability, Google APIs are backed by robust infrastructure, ensuring uptime and scalability
- Integration, integrated with other Google Service providing a cohesive experience

Weaknesses:

- Privacy Risks, location data is sent to the API, must be carefully to avoid exposing user location.

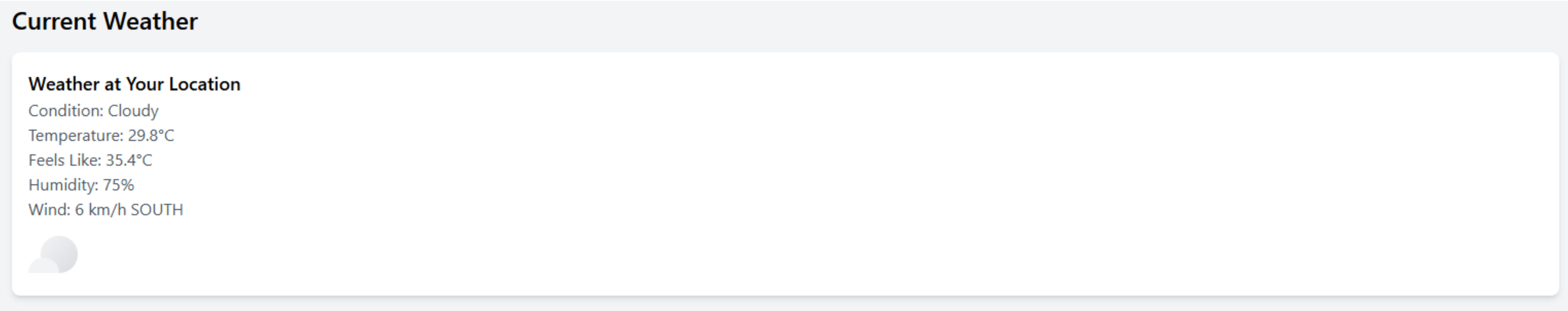
Security Mitigation:

- Store tokens securely on server side and use environment variables, not client-side code.
- Sanitize user inputs (e.g., location queries) to prevent injection attacks.

6. Screen Capture of Developed Application

Home Page With Weather:

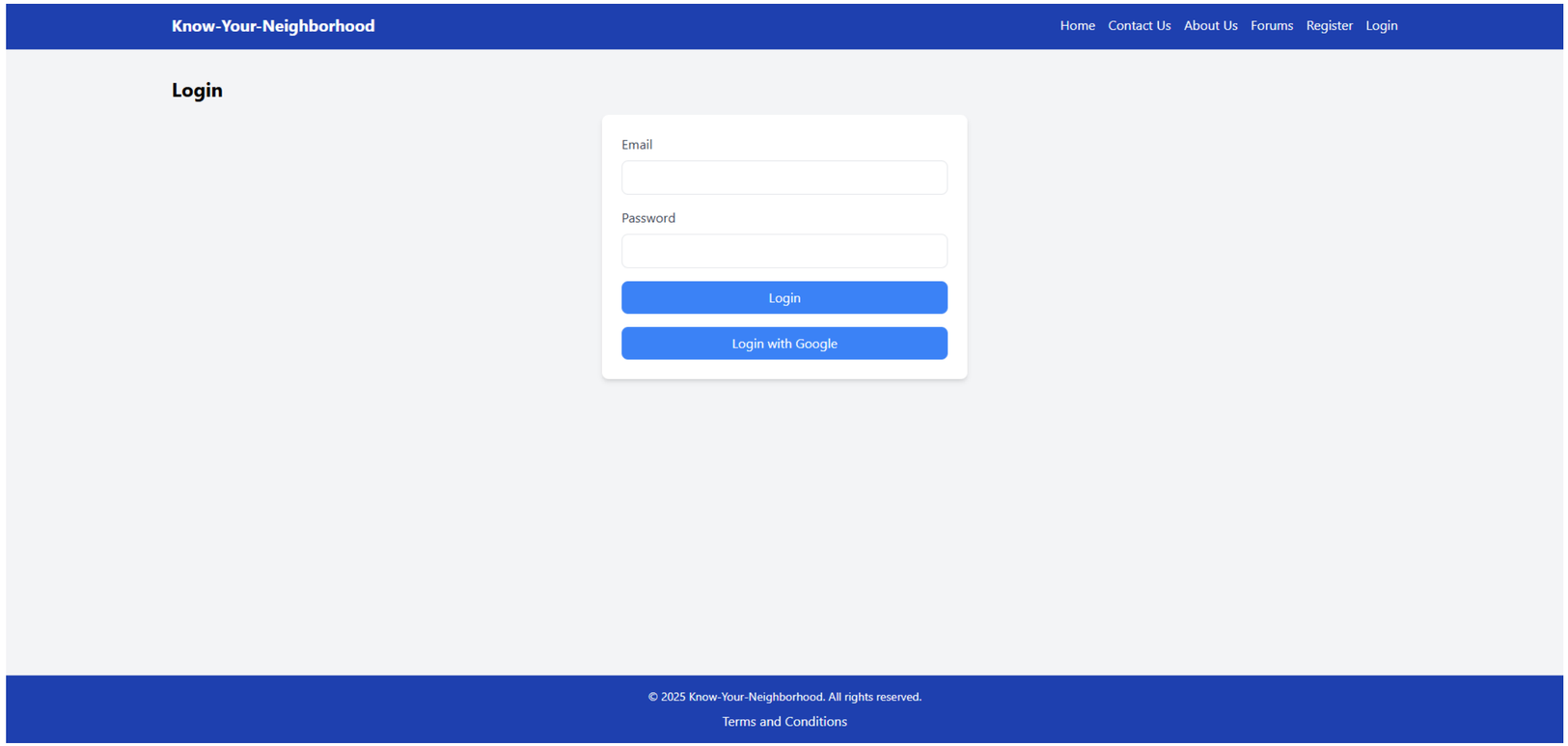
Displays user’s location-based weather using Google Weather API.



6. Screen Capture of Developed Application

Login Page with Google OAuth2:

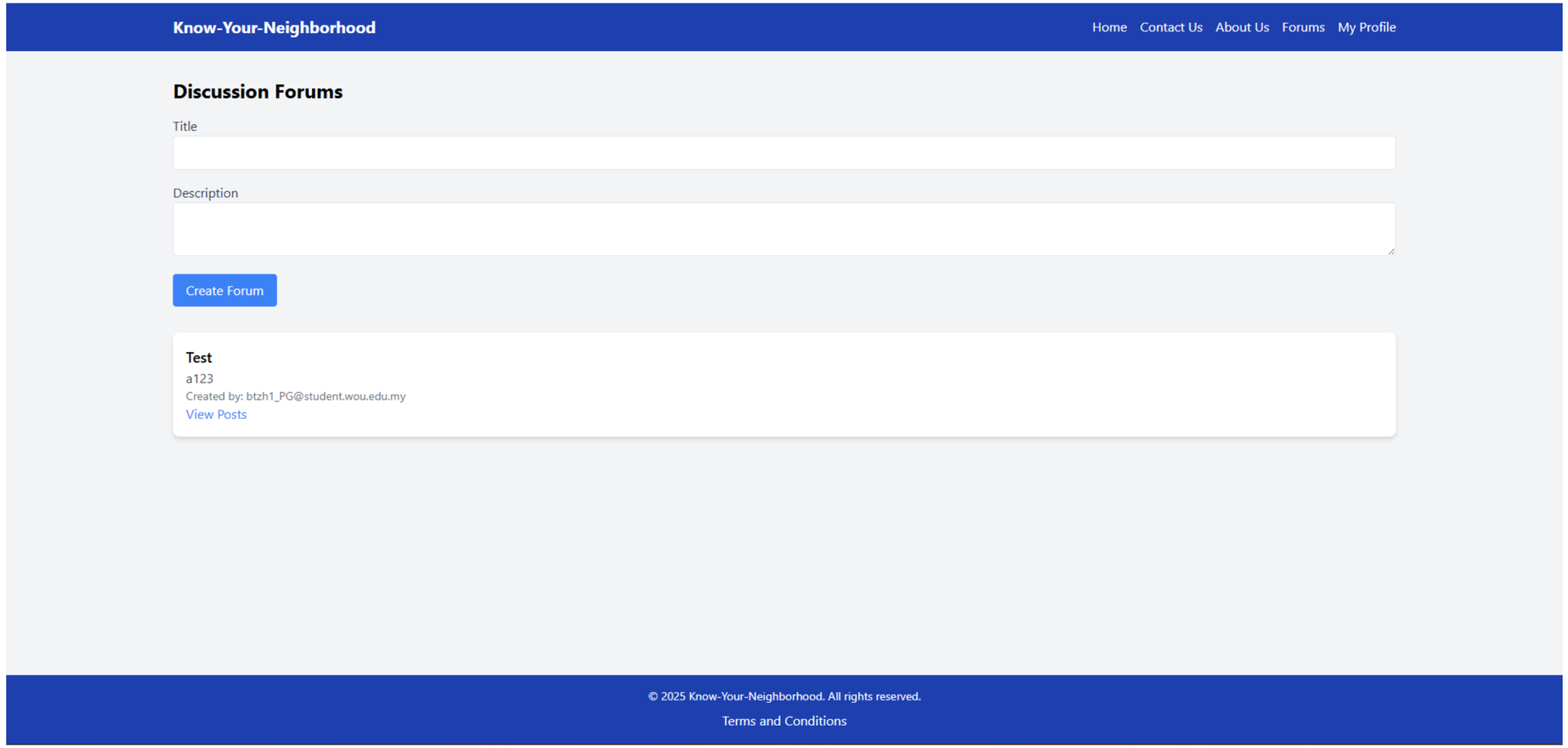
- Google OAuth2 login button integrated in Login.js



6. Screen Capture of Developed Application

Forum Page:

- Displays forums created via REST API (/api/forums)



6. Screen Capture of Developed Application

User Profile Page:

- Shows user details fetched via /api/user/info after OAuth2 authentication