

Activity 2 Document

Title of Activity

- Notes API + UI
- Short description (what the app does)

-The Notes App allows users to register with their full name, email, and password, then log in securely using a unique email and strong password. On the frontend, users can create, edit, update, and delete notes through a simple and user-friendly dashboard. On the backend API, the app handles the same CRUD operations with added security, ensuring all data is properly stored in the database.

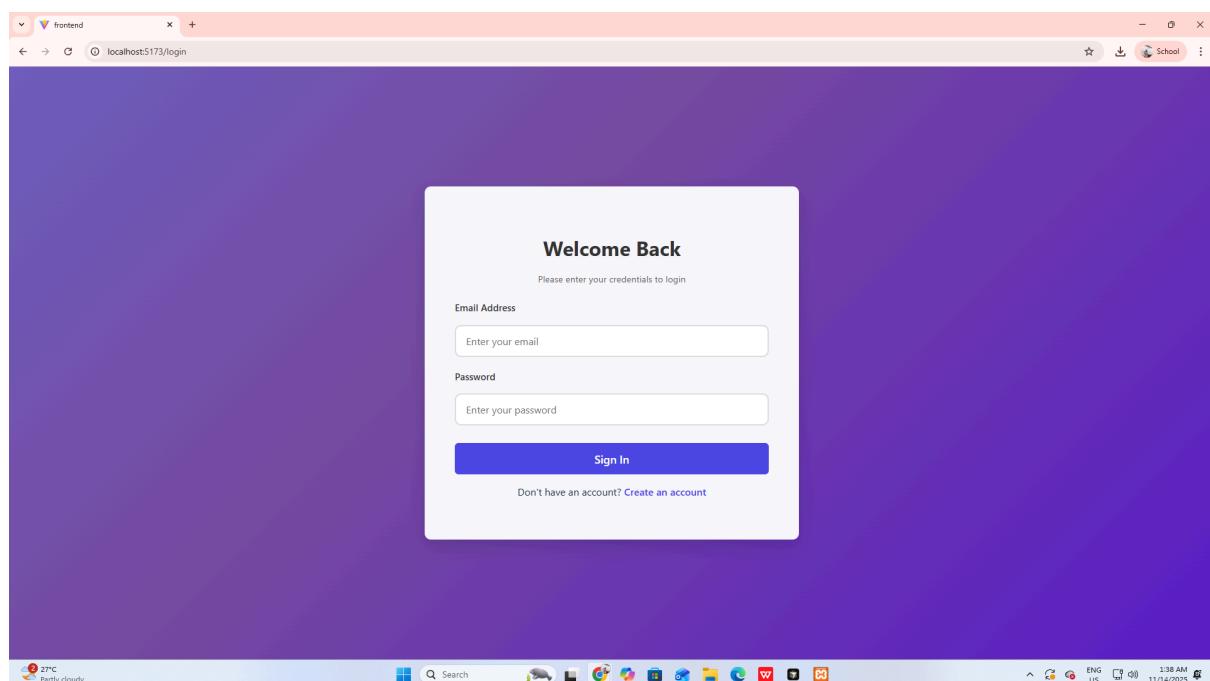
Features:

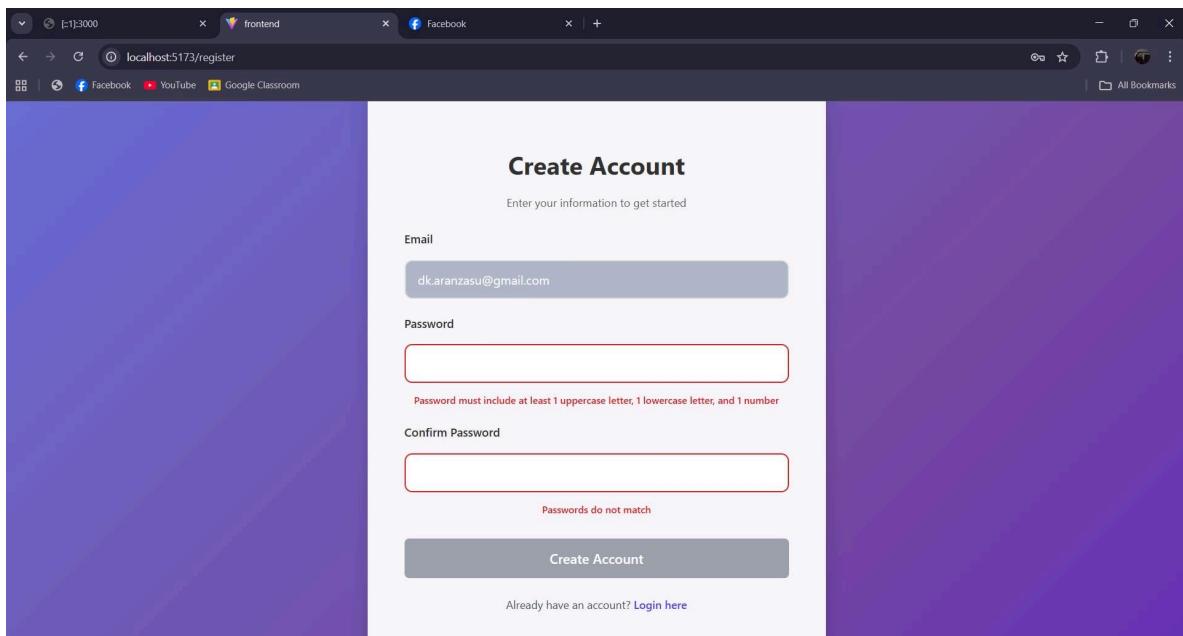
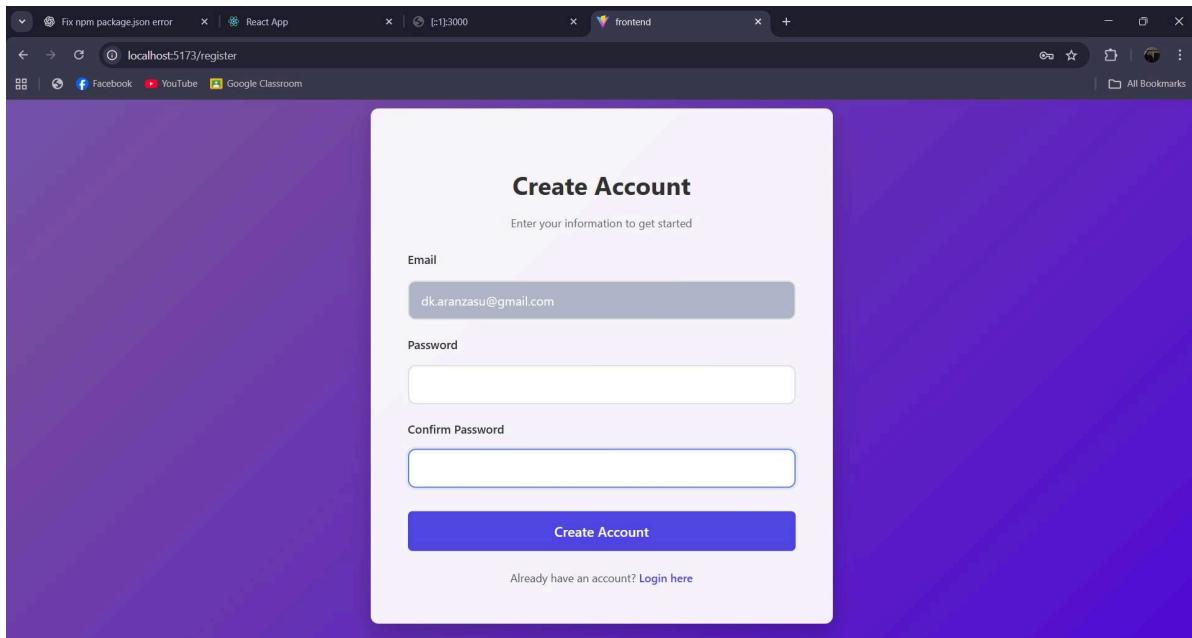
- Login that you can add your email and password
 - Register that you can add your email ,password and confirm password
 - Dashboard that you can add,edit,update and delete your notes and save directly to database and also have a CRUD API, that can you also test in swagger docs user cursor ai
- Screenshot(s) of working system (UI + API example)

Working System UI

Frontend /Backend

- **Front End**





The screenshot shows a 'Create Account' form on a purple-themed web page. The form includes fields for Email (dk.aranzasu@gmail.com), Password, and Confirm Password. A red error message 'Passwords do not match' is displayed below the Confirm Password field. A 'Create Account' button is at the bottom, and a link to 'Login here' is below it.

Create Account

Enter your information to get started

Email

dk.aranzasu@gmail.com

Password

Confirm Password

Passwords do not match

Create Account

Already have an account? [Login here](#)

The screenshot shows a 'Create New Note' dialog box. It has a title 'Laboratory Activities' and a content area containing 'Activity 2: Notes API + UI'. There are 'Cancel' and 'Create Note' buttons at the bottom.

>Create New Note

Title

Laboratory Activities

Content

Activity 2: Notes API + UI

Cancel

Create Note

The screenshot shows an 'Edit Note' dialog box. It displays the same note as the previous screenshot ('Laboratory Activities' title and 'Activity 2: Notes API + UI' content). There are 'Cancel' and 'Update Note' buttons at the bottom.

Edit Note

Title

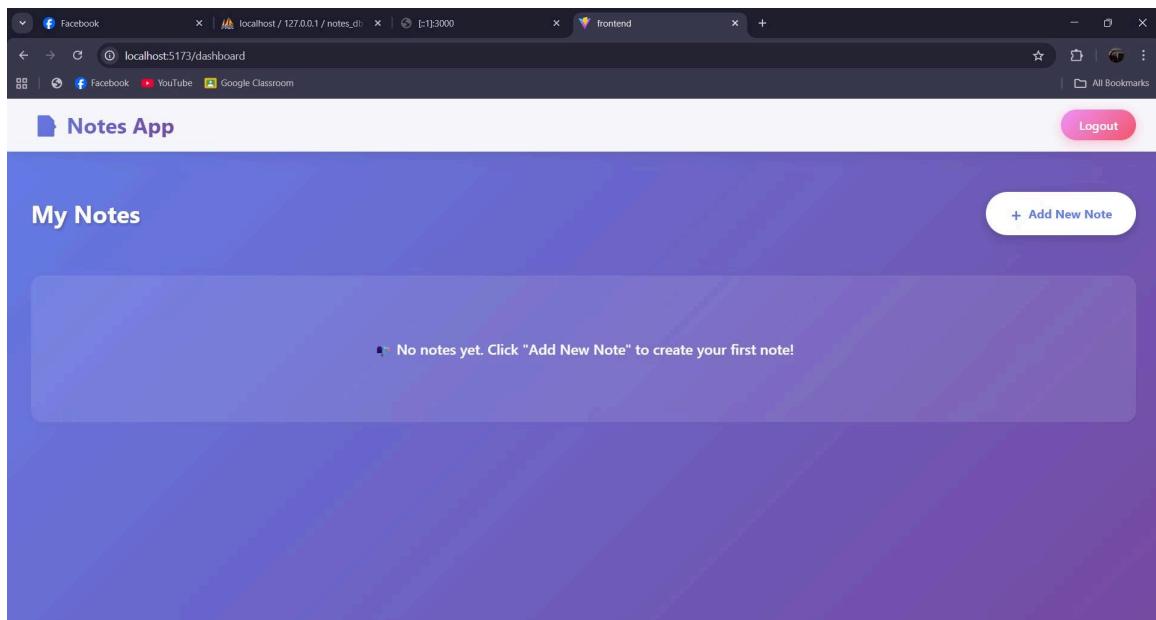
Laboratory Activities 1

Content

Activity 2: Notes API + UI

Cancel

Update Note



Backend



API

The image displays three screenshots of the Swagger UI interface, showing the API documentation and execution for a 'notes' endpoint.

Screenshot 1: API Documentation Overview

This screenshot shows the main API documentation page for 'Group-4A API'. It includes the title 'Group-4A API 1.0 OAS 3.0', a brief description 'API documentation for the Group-4A project', and an 'Authorize' button. Below this, there's a section for the 'App' endpoint, which contains a 'GET /' operation. Under the 'notes' section, there are three operations: 'POST /notes' (Create a new note), 'GET /notes' (Get all notes for the current user), and 'GET /notes/{id}' (Get a note by ID). Each operation has a lock icon and a dropdown menu.

Screenshot 2: Create Note Request

This screenshot shows the 'POST /notes' request details. It includes a 'Parameters' section (No parameters) and a 'Request body' section (required, application/json). The request body is defined as:

```
{  
  "title": "Meeting Notes",  
  "content": "Discussed project timeline and assigned tasks."  
}
```

An 'Execute' button is present at the bottom.

Screenshot 3: Response Details

This screenshot shows the response details for the 'POST /notes' request. It includes a 'Curl' command, a 'Request URL' (http://localhost:3000/notes), and a 'Server response' section. The response code is 401 (Error: Unauthorized). The response body is:

```
{  
  "message": "Unauthorized",  
  "statusCode": 401  
}
```

The response headers include:

```
access-control-allow-credentials: true  
access-control-allow-origin: http://localhost:5173  
connection: keep-alive  
content-length: 43  
content-type: application/json; charset=utf-8  
date: Wed,22 Oct 2025 01:24:06 GMT  
etag: W/\"e302d032fb7c4vnAjXxD99RzD0ptng\"  
keep-alive: timeout=5  
vary: Origin  
x-powered-by: Express
```

Below the response, there are 'Responses' and 'Links' sections.

The screenshot shows the Swagger UI interface for a POST request to the endpoint `/notes`. The request body is defined as follows:

```
{ "title": "Hello World", "content": "Hello World." }
```

The screenshot shows the Swagger UI interface for a POST request to the endpoint `/notes`. The curl command for the request is:

```
curl -X 'POST' \ 'http://localhost:3000/notes' \ -H 'accept: application/json' \ -H 'Content-Type: application/json' \ -d '{ "title": "Hello World", "content": "Hello World." }'
```

The response details show a 401 Unauthorized error with the following message:

```
{ "message": "Unauthorized", "statusCode": 401 }
```

The response headers include:

```
access-control-allow-credentials: true
access-control-allow-origin: http://localhost:5173
connection: keep-alive
content-length: 26
content-type: application/json; charset=utf-8
date: Wed, 22 Oct 2025 01:17:08 GMT
etag: W/2b-dGnJt6gv1n3jX6D39RztDWptng
keep-alive: timeout=5
vary: Origin
x-powered-by: Express
```

Instructions on how to run the project

Activity 2 Requirements

- Node.js (version 16 or higher)
- React (TypeScript & JavaScripts)
- npm (Node Package Manager)
- Git (for cloning the repository)

Step-by-Step Instructions

1. Clone the Repository

- Open Visual Studio Code.
- Open the terminal
- Run the following command to clone the GitHub repository:

```
git clone https://github.com/CVSU-IMUS-BSIT-4A/Group-4A-Repository.git
```

- Navigate to the project folder:

```
cd C:\Users\demro\OneDrive\Desktop\Group-4A-Repository-1
```

2. Install Backend Dependencies

- Go to the backend directory:
- cd C:\Users\demro\OneDrive\Desktop\Group-4A-Repository-1\ACTIVITY-2\backend

Install all required dependencies:

```
npm install
```

3. Set Up the Database

Make sure XAMPP is running (Apache and MySQL).

Open phpMyAdmin at:
<http://localhost/phpmyadmin>

Create a new database (e.g., notes_db).

Once configured properly, the backend will automatically create the necessary tables (for email, password, etc.) in your database.

4. Start the Backend Server

If your package.json includes a start script (e.g., start:dev), run the backend server using

```
npm run start:dev
```

After it starts successfully, the backend should be running at:

<http://localhost:3000>

You can check the API documentation or routes by visiting:

<http://localhost:3000/api>

5. Install Frontend Dependencies

Open a new terminal in Visual Studio Code.

Navigate to the frontend directory:

```
cd C:\Users\demro\OneDrive\Desktop\Group-4A-Repository-1\ACTIVITY-2\frontend
```

Install the required dependencies:

```
npm install
```

6. Start the Frontend Server

In the same terminal, run:

```
npm run dev
```

After it starts, open your browser and go to:

<http://localhost:5173>

7. Access the Application

Frontend Application: <http://localhost:5173>

Backend API: <http://localhost:3000>

Swagger or API Docs (if available): <http://localhost:3000/api>