**MemoryBoard: A Social Journaling Platform for Sharing and Saving Memories.**

Minor Project – II Report

Submitted in partial fulfilment of the requirement for the award of the degree of

# **MASTER OF COMPUTER APPLICATION**

SUBMITTED BY

**Vasu Jain (04411804423)**

**Khumar Girdhar (04511804423)**

**Puneet Kumar (06711804423)**

**Gaurav Sharma (03711804423)**

**Yogesh Choudhary (07511804423)**

UNDER THE GUIDANCE OF

**Mr. Manoj Purohit**



## CENTRE FOR DEVELOPMENT OF ADVANCED COMPUTING, NOIDA



**GURU GOBIND SINGH INDRAPRASTHA UNIVERSITY (14PT) SECTOR 16C, DWARKA, DELHI, 110078**

**TABLE OF CONTENTS**

|  |  |  |  |
| --- | --- | --- | --- |
| **S No** | **Topic** | **Page No** | **Sign** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

### Problem Statement

### In the digital age, individuals seek meaningful ways to document and share their life experiences. Traditional journaling is personal but lacks the social interaction and convenience that modern technology provides. Existing social media platforms, while popular, do not offer a dedicated space that combines the reflective nature of journaling with the social features of modern web apps. These platforms focus on ephemeral content rather than fostering deep, lasting connections through personal stories and memories.

### Purpose

### Design and build a web application that allows users to create, share, and archive personal journal entries within a social networking environment. The platform should enable users to record their memories using text, images, and other media, share these entries with selected groups or the public, and interact with other users through comments and stories.

### Main Features

### User Registration and Profiles

### Users can register and log in using email or social media accounts.

### Profiles include personal information and customizable privacy settings.

### Creating Journal Entries

### A rich text editor for writing journal entries.

### Ability to embed images, videos, and audio files within entries.

### A tagging system for classifying and organizing content.

### Privacy and Sharing Options

### Users can save entries privately, share them with specific friends or groups, or make them public.

### Granular privacy settings for each journal entry.

### Social Interaction

### Annotation and like system for journal entries.

### Follow and follower programs to build network connections.

### Notification system for updates, likes, and followers.

### Review and Discovery

### Search functionality for finding posts by keywords, tags, or dates.

### Recommendations for discovering new entries and connections.

**2*.* UML Diagrams**

**2.1 Use-Case Diagram**

### 

### Use Case Description

The use case diagram for the "Journal Application" represents the interactions between users and the system, as well as the functionalities provided by the application. The diagram includes two primary actors: User and Admin, and various use cases that illustrate the core features and actions available within the system.**Actors:**

1. User: Represents a typical user of the journal application who can perform various actions such as creating and managing journal entries, interacting with other users, and customizing their profile.
2. Admin: Represents an administrative user with additional privileges to manage users and moderate content within the application.

**Use Cases:**

**Register/Login**: Allows users to create an account and log into the application using email or social media accounts.

**View/Edit Profile:** Enables users to view and update their personal information and configure privacy settings.

**Create Journal Entry:** Provides users with a rich text editor to create new journal entries, including the ability to embed multimedia content.

**Edit Journal Entry:** Allows users to modify existing journal entries.

**Delete Journal Entry:** Permits users to remove their journal entries from the application.

**Set Entry Privacy**: Users can configure the privacy settings for each journal entry, deciding whether to keep them private, share with specific friends or groups, or make them public.

**Share Entry:** Enables users to share their journal entries with selected friends, groups, or the public.

**Comment on Entry:** Allows users to add comments to journal entries, facilitating social interaction.

**Like Entry:** Users can like journal entries to show appreciation or agreement.

**Follow/Unfollow Users:** Users can follow or unfollow other users to build and manage their network connections.

**Receive Notifications:** Users receive notifications about updates, likes, comments, and new followers.

**Search Entries:** Provides search functionality for users to find journal entries using keywords, tags, or dates.

**View Recommendations:** Users can view recommendations for discovering new journal entries and connections based on their interests and activities.

**Manage Users:** Admins can manage user accounts, including tasks such as approving new users, updating user information, or removing users.

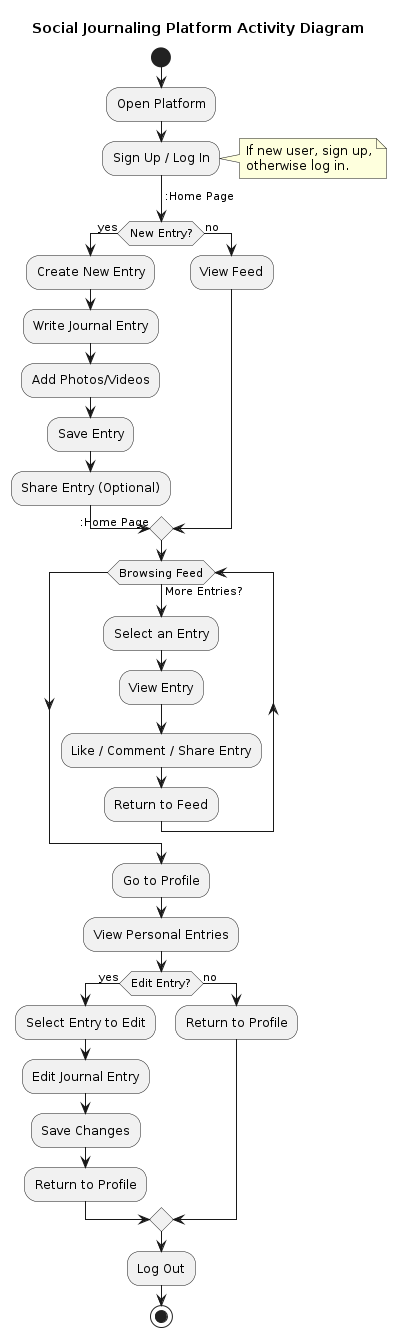
**Moderate Content:** Admins are responsible for moderating the content within the application to ensure it adheres to community guidelines and standards.

Relationships:

* **User to Register/Login**: Users interact with the registration and login functionality using email or social media accounts.
* **User to Create Journal Entry**: Users utilize the rich text editor to create journal entries.
* **User to Set Entry Privacy:** Users can set privacy options for each journal entry.
* **User to Share Entry:** Users can share their journal entries with friends, groups, or the public.

This use case diagram provides a high-level overview of the main interactions and functionalities of the journal application, ensuring that all essential features are identified and well-documented for further development and design.

**2.2 Activity Diagram**



Explanation of the Activity Diagram

1. **Start**: The user opens the social journaling platform.
2. **Log in or Register**: The user must either log in if they have an account or register for a new account.
3. **Main Interface**: Upon logging in, the user views their timeline.
4. **Parallel Activities**:

* **Post a New Memory**: The user can create a new post by writing a description, attaching media, setting privacy options, and then saving or publishing it.
* **View Friend's Memories**: The user can view memories shared by their friends by navigating to a friend's profile.
* **Search for Memories**: The user can search for specific memories using keywords and filters.
* **Edit or Delete a Memory**: The user can edit or delete their own memories.

1. **Log Out**: Finally, the user can log out of the platform.This activity diagram provides a visual representation of the various actions a user can

perform on the social journaling platform, highlighting the flow and interaction with different features of the system.

**3. Software Requirements Specification (SRS)**

A Social Journaling Platform for Sharing and Saving Memories

1. Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to provide a detailed description of the requirements for a Social Journaling Platform. This platform aims to enable users to document and share their life experiences through personal journal entries, combining the reflective nature of journaling with the interactive features of modern social networking.1.2 Scope

The Social Journaling Platform will allow users to create, share, and archive personal journal entries using text, images, and other media. Users will have the ability to interact with others through comments, likes, and a follower system. The platform will offer various privacy settings for journal entries and include features for search and content discovery.1.3 Definitions, Acronyms, and Abbreviations

* SRS: Software Requirements Specification
* UML: Unified Modeling Language
* Admin: Administrator
* User: Registered user of the platform
  1. References
* Unified Modeling Language (UML) Documentation
* IEEE SRS Standards
* Web Application Development Guidelines
  1. Overview

This SRS document outlines the overall description, specific requirements, and supporting information for the Social Journaling Platform. It is intended to be used by the development team, project managers, and stakeholders.

**2. Overall Description**

**2.1 Product Perspective**

The Social Journaling Platform is a standalone web application designed to offer a dedicated space for users to document and share their life experiences. It will integrate with social media accounts for registration and login purposes and provide features that enhance user interaction and content management.**2.2 Product Functions**

* User Registration and Login
* Profile Management
* Creation and Management of Journal Entries
* Privacy and Sharing Options
* Social Interaction Features (Comments, Likes, Follows)
* Notifications
* Search and Discovery of Content

Administrative Tools for User and Content Management

**2.3 User Classes and Characteristics**

* General Users: Individuals who will use the platform to create and share journal entries.
* Admin Users: Individuals responsible for managing users and moderating content to ensure compliance with community guidelines.

**2.4 Operating Environment**

* Client Side: Modern web browsers (Chrome, Firefox, Safari, Edge)
* Server Side: Web server hosting the platform, database server for data storage

**2.5 Design and Implementation Constraints**

* Compliance with web accessibility standards
* Secure handling of user data
* Scalability to handle a growing number of users and content

**2.6 Assumptions and Dependencies**

* Users have access to the internet and a modern web browser.
* The platform will integrate with third-party social media services for registration and login.

**3. Specific Requirements**

**3.1 Functional Requirements**

**3.1.1 User Registration and Profiles**

* Users can register and log in using email or social media accounts.
* Users can view and edit their profiles, including personal information and privacy settings.

**3.1.2 Creating Journal Entries**

* Users can create journal entries using a rich text editor.
* Users can embed images, videos, and audio files within their entries.
* A tagging system allows users to classify and organize content.

**3.1.3 Privacy and Sharing Options**

* Users can save entries privately, share with specific friends or groups, or make them public.
* Each journal entry will have customizable privacy settings.

**3.1.4 Social Interaction**

* Users can comment on and like journal entries.
* Users can follow and unfollow other users to build network connections.
* A notification system alerts users to updates, likes, comments, and new followers.

**3.1.5 Review and Discovery**

* Users can search for journal entries by keywords, tags, or dates.
* The platform will recommend entries and connections based on user interests and activities.

**3.1.6 Administrative Functions**

* Admins can manage user accounts, including approving, updating, or removing users.
* Admins can moderate content to ensure compliance with community guidelines.

**3.2 Non-Functional Requirements**

**3.2.1 Performance Requirements**

* The platform must support concurrent access by multiple users without significant performance degradation.
* Journal entries should load within a reasonable time frame (e.g., under 3 seconds).

**3.2.2 Security Requirements**

* User data must be securely stored and transmitted.
* Implement authentication and authorization mechanisms to prevent unauthorized access.

**3.2.3 Usability Requirements**

* The user interface should be intuitive and easy to navigate.
* Provide help and documentation for users to understand the platform's features.

**3.2.4 Reliability Requirements**

* The platform should have an uptime of 99.9% to ensure availability.
* Implement backup and recovery procedures to prevent data loss.

**3.2.5 Scalability Requirements**

* The platform should be able to scale to accommodate an increasing number of users and content without significant changes to the architecture.

**3.3 Interface Requirements**

**3.3.1 User Interfaces**

* Web-based user interface compatible with modern web browsers.
* Responsive design to support various screen sizes and devices.

**3.3.2 Hardware Interfaces**

* Not applicable as the platform is web-based.

**3.3.3 Software Interfaces**

* Integration with third-party social media services for registration and login.
* Use of web APIs for functionalities such as notifications and content recommendations.

**3.4 System Requirements**

* Web server with support for modern web application frameworks.
* Database server for storing user data, journal entries, and metadata.

**4. Supporting Information**

**4.1 Appendices**

* UML Diagrams (Use Case, Class Diagrams)
* Sample User Interfaces
* Glossary of Terms

**4.2 Index**

* Not applicable for this document.

This SRS document serves as a comprehensive guide for the development of the Social Journaling Platform, ensuring that all functional and non-functional requirements are well-defined and understood by all stakeholders.

1. **Implementation**

The implementation of the Social Journaling Platform for Sharing and Saving Memories is divided into four main modules, each responsible for a specific set of functionalities. The implementation details of each module are outlined below, along with the corresponding test cases to ensure each module operates as expected.**4.1 Module-1: User Registration and Profiles**

Description:

This module handles user registration, authentication, and profile management. Users can register using email or social media accounts, log in, and manage their profile information and privacy settings.Implementation Details:

* Registration: Integration with email and social media OAuth services for user registration.
* Authentication: Secure login system with password encryption.
* Profile Management: User interface for viewing and editing profile information, including personal details and privacy settings.

Key Components:

* Registration Form
* Login Form
* Profile Management Interface
* Integration with OAuth Providers

Test Cases:

* TC1.1: Verify that users can register using an email address.
* TC1.2: Verify that users can register using social media accounts.
* TC1.3: Verify that registered users can log in using their credentials.
* TC1.4: Verify that users can view and edit their profile information.
* TC1.5: Verify that privacy settings can be updated and saved.

**4.2 Module-2: Creating and Managing Journal Entries**

Description:

This module provides the functionality for users to create, edit, and delete journal entries. Users can use a rich text editor to write entries and embed multimedia content.Implementation Details:

* Rich Text Editor: Implementation of a text editor with support for text formatting, and embedding images, videos, and audio.
* Entry Management: Interfaces for creating, editing, and deleting journal entries.
* Tagging System: Allows users to tag and categorize their entries for easy organization and search.

Key Components:

* Rich Text Editor
* Entry Management Interface
* Multimedia Embedding
* Tagging System

Test Cases:

* TC2.1: Verify that users can create a journal entry using the rich text editor.
* TC2.2: Verify that users can embed images, videos, and audio files in their entries.
* TC2.3: Verify that users can edit existing journal entries.
* TC2.4: Verify that users can delete journal entries.
* TC2.5: Verify that users can add and manage tags for their entries.

**4.3 Module-3: Privacy and Sharing Options**

Description:

This module handles the privacy settings and sharing options for journal entries. Users can choose to keep entries private, share with specific friends or groups, or make them public.Implementation Details:

* Privacy Settings: Implementation of granular privacy controls for each journal entry.
* Sharing Functionality: Allows users to share entries with friends, groups, or the public.
* Customizable Access: Interfaces for setting and updating privacy preferences.

Key Components:

* Privacy Settings Interface
* Sharing Options Interface

Test Cases:

* TC3.1: Verify that users can set journal entries to private.
* TC3.2: Verify that users can share entries with specific friends.
* TC3.3: Verify that users can share entries with groups.
* TC3.4: Verify that users can make entries public.
* TC3.5: Verify that privacy settings are respected when accessing entries.

**4.4 Module-4: Social Interaction Features**

Description:

This module includes features that enable social interactions such as commenting on entries, liking entries, following users, and receiving notifications.Implementation Details:

* Commenting System: Allows users to comment on journal entries.
* Like System: Enables users to like journal entries.
* Follow System: Users can follow and unfollow other users.
* Notifications: Real-time notifications for updates, likes, comments, and new followers.

Key Components:

* Commenting Interface
* Like Button
* Follow/Unfollow System
* Notification System

Test Cases:

* TC4.1: Verify that users can comment on journal entries.
* TC4.2: Verify that users can like journal entries.
* TC4.3: Verify that users can follow other users.
* TC4.4: Verify that users can unfollow other users.
* TC4.5: Verify that notifications are sent for updates, likes, comments, and new followers.

**4.5 Test Cases**

General Test Cases:

* TC5.1: Verify the system handles concurrent user logins without performance degradation.
* TC5.2: Verify the responsiveness of the web interface across different devices and screen Sizes.
* TC5.3: Verify the security of data transmission and storage.
* TC5.4: Verify the scalability of the system with increasing user load and content.
* TC5.5: Verify the backup and recovery processes to prevent data loss.

Module-Specific Test Cases:

* Module-1: User Registration and Profiles TC1.1 to TC1.5
* Module-2: Creating and Managing Journal Entries

TC2.1 to TC2.5

* Module-3: Privacy and Sharing Options

TC3.1 to TC3.5

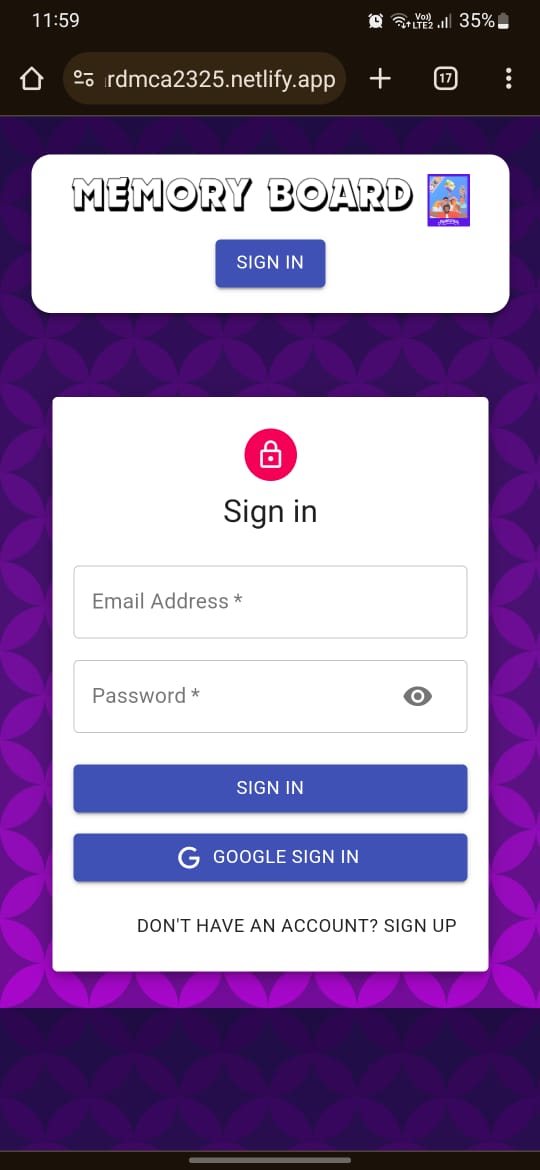
* Module-4: Social Interaction Features TC4.1 to TC4.5

This implementation provides a structured approach to developing and testing the Social Journaling Platform, ensuring that each module is thoroughly implemented and verified for functionality and performance.

**5.Code Snippets and Outputs**

**5.1 User Registration and Login**Code Snippet: User Registration (Node.js/Express)import jwt from "jsonwebtoken";const secret = 'test';const auth = async (req, res, next) => { try { const token = req.headers.authorization.split(" ")[1]; const isCustomAuth = token.length < 500; let decodedData; if (token && isCustomAuth) { decodedData = jwt.verify(token, secret); req.userId = decodedData?.id; } else { decodedData = jwt.decode(token); req.userId = decodedData?.sub; } next(); } catch (error) { console.log(error); }};export default auth;

**output :**



* 1. **Creating Journal Entries**Code Snippet: Creating Journal Entry (React)

import React, { useState } from 'react';

import axios from 'axios';

const CreateJournalEntry = () => {

const [title, setTitle] = useState('');

const [content, setContent] = useState('');

const [media, setMedia] = useState(null);

const handleSubmit = async (e) => {

e.preventDefault();

const formData = new FormData();

formData.append('title', title);

formData.append('content', content);

if (media) formData.append('media', media);

try {

const response = await axios.post('/api/journal-entries', formData, {

headers: {

'Content-Type': 'multipart/form-data'

}

});

console.log('Entry created:', response.data);

} catch (error) {

console.error('Error creating entry:', error);

}

};

return (

<form onSubmit={handleSubmit}>

<div>

<label>Title</label>

<input type="text" value={title} onChange={(e) => setTitle(e.target.value)} required />

</div>

<div>

<label>Content</label>

<textarea value={content} onChange={(e) => setContent(e.target.value)} required></textarea>

</div>

<div>

<label>Media</label>

<input type="file" onChange={(e) => setMedia(e.target.files[0])} />

</div>

<button type="submit">Create Entry</button>

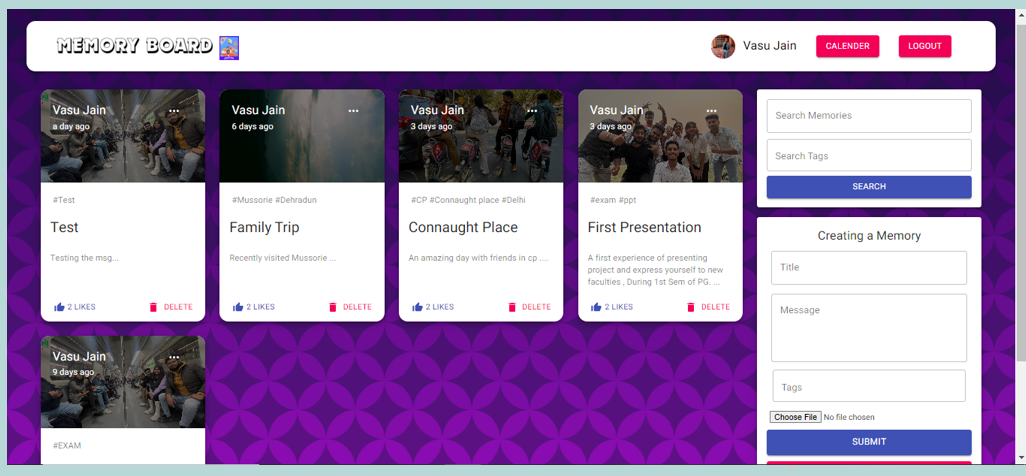
</form>

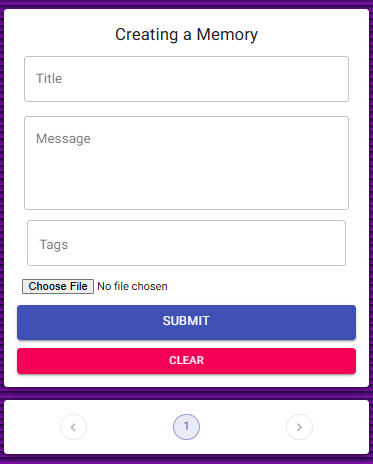
);

};

export default CreateJournalEntry;

**Output :**





* 1. **Privacy and Sharing Options** Code Snippet: Setting Privacy Options (MongoDB Schema)

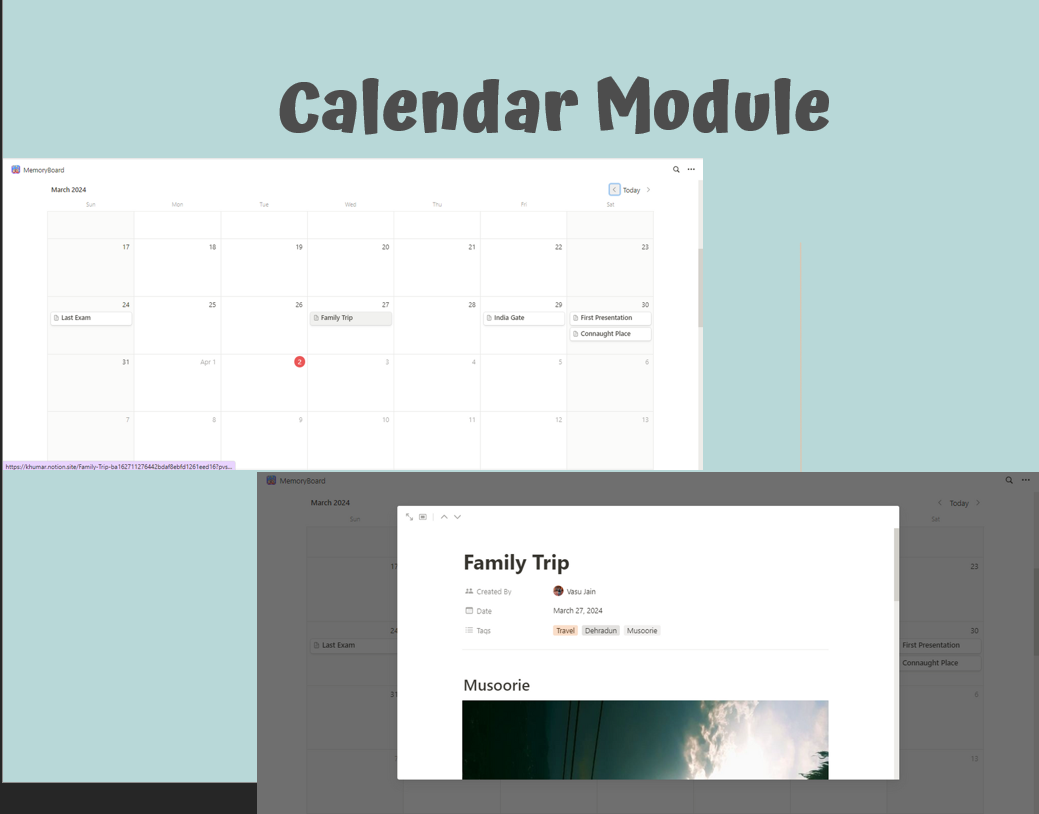
const mongoose = require('mongoose');const JournalEntrySchema = new mongoose.Schema({ title: { type: String, required: true }, content: { type: String, required: true }, media: { type: String }, privacy: { type: String, enum: ['private', 'friends', 'public'], default: 'private' }, owner: { type: mongoose.Schema.Types.ObjectId, ref: 'User', required: true }, createdAt: { type: Date, default: Date.now }});module.exports = mongoose.model('JournalEntry', JournalEntrySchema);

Expected Output:Example document in MongoDB:json

{ "\_id": "60c72b1f9b1d4c3d8f8b4567", "title": "My Private Entry", "content": "This is a private journal entry.", "media": "path/to/media.jpg", "privacy": "private", "owner": "60c72b1f9b1d4c3d8f8b1234", "createdAt": "2024-05-20T10:00:00.000Z"}

5.4 Social Interaction FeaturesCode Snippet: Commenting on Journal Entry (Node.js/Express)javascriptCopy codeconst express = require('express');const JournalEntry = require('./models/JournalEntry'); // Assuming JournalEntry model is definedconst Comment = require('./models/Comment'); // Assuming Comment model is definedconst router = express.Router();// Add a Comment Endpointrouter.post('/journal-entries/:id/comments', async (req, res) => { try { const { content } = req.body; const { id } = req.params; const comment = new Comment({ content, journalEntry: id, user: req.user.\_id }); await comment.save(); await JournalEntry.findByIdAndUpdate(id, { $push: { comments: comment.\_id } }); res.status(201).json({ message: 'Comment added successfully' }); } catch (error) { res.status(500).json({ error: 'Error adding comment' }); }});module.exports = router;

**Output :**



These code snippets and expected outputs provide a basic framework for implementing and testing the core functionalities of the Social Journaling Platform. The snippets cover user registration, journal entry creation, setting privacy options, and adding comments, ensuring a robust and interactive user experience.

1. **References:**
2. MongoDB Docs: https://www.mongodb.com/docs/atlas/tutorial/deploy-free-tier-cluster/ (accessed on 11 March, 2024)
3. Python Docs: https://docs.python.org/3/ (accessed on 2 March, 2024)
4. React Docs: https://react.dev/learn/describing-the-ui , https://react.dev/learn/your-first- component , https://react.dev/learn/importing-and-exporting-components (accessed on 24 Feb, 2024)
5. NodeJS Docs: https://nodejs.org/docs/latest/api/cluster.html (accessed on 17 Feb, 2024)
6. Firebase Docs: https://firebase.google.com/docs/auth (accessed on 10 March, 2024)
7. Google Cloud Docs: https://cloud.google.com/run/docs (accessed on 28 January, 2024)