# Software Requirements Specification

# PRJ566 NBB – Fall 2023

# PRJ566 – Team No: 04

# Name of Project:

Chronicle: The Ultimate Travel Companion

# Project Leader: Rotates Every 3 Weeks

**Last updated:** December 6, 2023

**Team Members:**

**1. Riccardo Moncada**

**2. Artem Pankov**

**3. Vaibhav Parmar**

**4. Juan David Rodriguez Castelblanco**

# TABLE OF CONTENTS

1. **Introduction/Overview - Document Information**
   1. **Document Authors**
   2. **Revision History**
   3. **Document Conventions**
   4. **Document Purpose**
   5. **Intended Audience**
   6. **Group Agreement**
2. **Project Overview**
   1. **Project Proposal**
   2. **Stakeholders and Users**
   3. **Functional Requirements**
   4. **Nonfunctional Requirements**
   5. **Project Scope**
   6. **System risks**
   7. **Operating Environment**
   8. **UI/UXD Interface Mockups**
3. **Process & Data Modeling**
   1. **UML Modeling: DFDs & Activity Diagrams**
   2. **Use Case Specification** 
      1. **Business Rules**
      2. **System Use Case Diagrams**
      3. **Use Case Description Tables**
4. **Domain Class Diagram**
5. **Database (Select either 5.1 or 5.2)**
   1. **RDBMS Artifacts**
      1. **Scripts to create, populate, delete tables**
      2. **Data Dictionary**
   2. **NoSQL Artifacts**
6. **Work breakdown Structure (WBS)**
7. **Milestones & Acceptance Criteria**
8. **Implementation Schedule (Agile/Waterfall)**
9. **Client / Faculty Sign-off**

# 1 - Introduction/Overview - Document Information

## 1.1 Document Authors

1. Riccardo Moncada

2. Artem Pankov

3. Vaibhav Parmar

4. Juan David Rodriguez Castelblanco

## 1.2 Revision History

|  |  |
| --- | --- |
| Week 03 | 1.1 Document Authors – Completed (2023-09-24)  1.2 Revision History - (ongoing)  1.3 Document Conventions - Completed (2023-09-24)  1.4 Document Purpose – Completed (2023-09-24)  1.5 Intended Audience – Completed (2023-09-24)  1.6 Group Agreement – Completed (2023-09-24)  2.1 Project Proposal - Completed (2023-09-24) |
| Week 04 | 2.2 Stakeholders and Users – Completed and Last Updated (2023-09-30)  2.3 Functional Requirements - Completed and Last Updated (2023-09-30)  2.4 Nonfunctional Requirements - Completed and Last Updated (2023-09-30) |
| Week 05 | 2.5 Project Scope – Completed (2023-12-06)  2.6 Systems Risks – Completed (2023-12-06)  2.7 Operating Environment - Completed (2023-12-06)  2.8 Prototype: UI/UXD - Wireframes/Mock-ups – Completed (2023-12-06) |
| Week 06 | 3.1 Data Flow Diagrams (DFDs.) – Completed (2023-12-06)  3.1 Activity Diagram – Completed (2023-12-06) |
| Week 07 | 3.2.1 Business Rules – Completed (2023-12-06)  3.2.2 System Use Case Diagrams – Completed (2023-12-06)  3.2.3 Use Case Descriptions – Completed (2023-12-06) |
| Week 08 |  |
| Week 09 | 4. Domain Class Diagrams- Completed (2023-12-06) |
| Week 10 | 5.2 NoSQL Data Models and corresponding Documents (2023-12-06) |
| Week 11 | 6. Work Breakdown Structure (WBS.) (2023-12-06)  7. Project Milestones & Acceptance Criteria. (2023-12-06) |
| Final |  |

## 1.3 Document Conventions

For example:

Any text in red indicates an exception or error.

Any text in blue is in-progress.

Any text highlighted in yellow is an important point.

Any text in green was recently added.

Any text *italicized* represents definitions.

Any text with ~~strike-through~~ is deleted.

## 1.4 Document Purpose

This Software Requirements Specification (SRS) is designed to offer an overview of the vision, objectives, and detailed features of our project, 'Chronicle: Your Ultimate Travel Companion'. This document serves several core purposes:

**Clarity & Documentation:**

* Act as a point of reference for any questions regarding the project, ensuring clarity and a common understanding among team members.
* Outline the functional and non-functional requirements of the Chronicle app.

**Advice and Direction:**

* Serve as guidance throughout the development process, ensuring that our coding efforts continue to be in accordance with the established specifications.

**Communication with Stakeholders:**

* Make our mission and purpose apparent to all parties involved, including the professor.
* Organize the gathering of user input and make sure the application complies with academic requirements.

**Quality Control:**

* Ensure that all defined requirements are met, and the system operates as intended, a foundation for testing is established.
* Guarantee a systematic development process, minimizing the chance for errors and assuring high standards of quality.

## 1.5 Intended Audience

**Development team and UI/UX Designers:**

* This is our team of software engineers and programmers who will be building and designing the software.

**Project Manager**:

* The Project Manager will plan and manage the development project efficiently.

**Production team and Quality Assurance team**

* The production team will be monitoring the running software in production and QA testers can refer to the documentation to create test cases and test scenarios.

Also indented for **Support and Maintenance team**, **Legal and Compliance team**, **Stakeholders** and **Sponsors** of the software.

## 1.6 Group Agreement

**TEAM AGREEMENT**

**Team #: 4**

**Project Title:** Chronicle: The Ultimate Travel Companion

**Project Time Frame:** 2 Terms

**Team Members:**

1. Riccardo Moncada

2. Artem Pankov

3. Vaibhav Parmar

4. Juan David Rodriguez Castelblanco

**Team Leadership:**  
The role of team leader will be shared among the project team members throughout the project lifespan. We will rotate as team leader every 3 weeks. The team leader will coordinate tasks between group members and provide support to them as well as review team member completed tasks and deliverables before submission to the professor.

**Team Functions:**

* Weekly completion of tasks to fulfil milestone requirements.
* We will share information through MS Teams, OneDrive, WhatsApp, e-mail and meetings.
* Updating the state of a team member's task completion status on GitHub is the responsibility of the team member assigned to that task.
* Tasks that are not volunteered to be worked on shall be assigned to group members by the team leader. (Refer to Note Below)

*(Note: This clause does not wholly give the leader “carte blanche” to dictate team members what to. The team leader must keep in mind the capabilities of each team member and whether they legitimately are unable to work on such a task because of external factors, including personal or course workload related.)*

**Team Meetings:**

It’s important that the team leader schedules and provides direction during team meetings. It will help if the team leader sets an agenda for the meeting before the meeting begins. Notes will be taken during team meetings, usually by the team leader but can request another team member to do so if the situation requires them to. There is a template for meeting notes in the ‘Team Meetings’ folder. Team meeting notes will be kept in the ‘Team Meetings’ folder in our team files. Team meetings will be recorded ONLY if someone was not able to attend the meetings. It is the responsibility of each team member to review the meeting notes.

**Team Problems:**

Problems within the team shall be brought to the attention of the team leader for them to address. Problems may include other team leaders not completing work, task-related disagreements between team members, harassing another team member as well as any other issues with completing tasks. The team leader is there to support and aid all team members when required. Any issues with team leadership should be brought to the attention of the professor if the team leader is unreceptive to addressing them or communicating about them. It is understandable if there may be personal problems that a team member is not willing to share with the team (or team leader) and in that situation they may directly report to the professor.

**Team Commitment**

**The undersigned members agree to work together on the project until the end of the PRJ666 next Semester. They recognize that as a team and individually they are responsible for the quality of all deliverables.**

**Name Date**

|  |  |
| --- | --- |
| Juan David Rodriguez Castelblanco | 9/21/2023 |
| Artem Pankov | 9/22/2023 |
| Riccardo Moncada | 9/23/2023 |
| Vaibhav Parmar | 9/23/2023 |

ShapeShapeShapeShapeShapeShape

# 2 - Project Overview

## 2.1 Project Proposal

**Project Background**

The project team is in the process of developing "Chronicle" as a comprehensive mobile application for travelers. Chronicle's primary goal is to become the ultimate travel companion, offering a unified platform where travelers can:

1. Chronicle their journeys using text, photos, and videos, consolidating all their travel experiences into a cohesive timeline.
2. Share their travel adventures with friends and family, facilitating collaboration and interaction with fellow travelers.
3. Provide ratings and reviews for destinations, accommodations, and dining establishments, enriching the depth of their travel documentation.
4. Engage in travel-related discussions, seek advice, and connect with a community of like-minded travelers on a dedicated message forum.
5. Organize and participate in meetups with fellow travelers at popular tourist destinations, further enhancing their travel experiences.

The project's objective is to create a specialized platform tailored exclusively to the unique needs of travelers, enabling them to comprehensively document, share, and enhance their travel experiences. By addressing the challenges associated with fragmented documentation and limited interaction on existing social media platforms, Chronicle aims to redefine how travelers document their journeys and establish connections with others who share their passion for traveling.

**Problem Statement**

|  |  |
| --- | --- |
| The Problem of: | Fragmented travel documentation and limited traveler interaction |
| Affects: | * Travelers * Travel enthusiast communities * Friends and families of travelers * The tourism industry * Travel agencies * Tour operators * Tourism marketing organizations * Project team and its developers * Project sponsor * Other social media platforms |
| The impact of which is: | * Travelers benefit from enhanced documentation and interaction. * Travel enthusiasts find a platform for sharing experiences. * Families and friends enjoy easier access to travel updates. * The tourism industry gains insights and improved marketing. * Travel agencies benefit from better customer understanding. * Travel communities thrive. * Developers and teams gain real-world experience. * Investors see project potential. * Social media platforms may face indirect competition. * Tourism authorities can boost local economies. |
| A successful solution would: | * Improve travel documentation. * Increase traveler interactions. * Improve travel planning. * Provide valuable tourism insights for businesses. * Improve connectivity with travelers’ family and friends. * Provide travel agencies, tour operators and travel marketing firms with valuable data to help tailor travel experiences and marketing for their clientele. * Provide real-world experience for the project team as application developers. * Provide an alternative platform for documenting and sharing travel experiences. |

**Product Vision**

|  |  |
| --- | --- |
| For | People who have a large interest in travel. |
| Who | Need a comprehensive, dedicated digital platform solution to document, share and enhance their travel experiences. |
| The Product Name | ***“Chronicle”*** is a digital travel companion. |
| That | Enables travelers to transform their journeys into immersive, well-connected, and unforgettable adventures, consolidating them within their travel timeline, and seamlessly integrating comprehensive travel documentation, a thriving community of like-minded explorers, and tools for organizing meetups at popular destinations. |
| Unlike | Travel-oriented social media mobile applications such as “Travel Buddy” or “Travello”, and more generic social media application platforms like Instagram, Twitter, and Facebook. |
| Our product | Offers:   1. Comprehensive Travel Documentation 2. Traveler Interaction 3. Destination Ratings 4. Meetup Organization 5. Travel timeline; a central feature of ***Chronicle***, serving as a digital journal and record of a traveler's journey. It's a dynamic and interactive component that plays a pivotal role in enhancing the travel experience. |

## 2.2 Stakeholders and Users

|  |  |
| --- | --- |
| **Stakeholder Name/Identifier** | **Category** |
| **Travelers** | **User**  The end user who will use the app and the needs of them are the epicenter of the project |
| **UI/UX Designers** | **Design and User Experience**  To create user-friendly interfaces for the app. |
| **Web Developers (Front-end, Back-end, Full Stack)** | **Development and Technical**  Front-end developers handle the user interface, while back-end and full-stack developers handle server-side logic. |
| **Database Programmers, Administrators and Database Engineers** | **Database Management and Development**  Programmer designs and implements the database structure, while Administrator maintains and manages it, while an Engineer optimizes database performance and data retrieval processes. |
| **Business Analyst** | **Business Analysis and Strategy**  To analyze business requirements, user needs, and market trends to inform development team. |
| **Cyber Security Analyst** | **Security and Compliance**  To identify and mitigate security risks and vulnerabilities, and to ensure user data protection. |
| **Quality Assurance Team** | **Testing and Quality Assurance**  To ensure that the app functions as expected by conducting rigorous testing and identifying and reporting bugs or issues. |
| **Project Managers** | **Project Management and Leadership**  To oversee the planning, execution, and delivery of the project, and to ensure that it aligns with the project's objectives and timeline. |
| **Support Managers** | **Customer Support and Service**  To manage customer support services, address user inquiries, and to ensure a positive user experience |
| **Sponsors** | **Financial Stakeholders**  The chief financial and strategic supporter for the project, to provide resources, funding and guidance. |
| **Investors** | **Financial Stakeholders**  Who are to invest in the project and looking for returns on their investment |
| **Other Competitors** | **External Stakeholders**  Rival social media and Travel-oriented social media mobile applications may be indirectly impacted by Chronicle's presence in the market. |
| **Tourism Agencies** | **Business Stakeholders**  Organizations that can utilize data from Chronicle app to better understand and cater to their customers' preferences. |
| **Marketing and Advertising Agencies** | **Promotion Stakeholders**  To promote the app and its services through marketing and advertising efforts. |
| **Market Researchers** | **Market Research Stakeholders**  To gather market data, feedback, and insights that inform product development and marketing strategies |
| **Legal and Compliance Team** | **Legal and Compliance**  To ensure that the Chronicle app complies with legal requirements and industry regulations |
| **Government** | **Government or Regulatory Stakeholders**  Regulatory bodies that may have an interest in ensuring compliance with relevant laws and regulations. |

## 2.3 Functional Requirements

1. **User Profile, Registration and Authentication**
   1. Application must have an option for the user to create an account, filling it with required data.
   2. Users must be able to create accounts with a valid email address and password.
   3. Users should have the ability to customize their profiles, including personal, travel-related information, or contact details.
   4. Application should support visual customization of profiles.
   5. Users must be able to view and edit their own profile information and settings.
2. **Travel Journal Creation**
   1. Users should be able to create new travel journal entries, including adding text descriptions, map locations, photos, and videos.
   2. The system must allow users to specify the date, location, and category for each journal entry.
   3. Users should be able to use hashtags to categorize and tag their journal entries.
   4. Users must have access to a chronological travel timeline where they can view, edit, and organize their journal entries.
3. **Sharing Adventures**
   1. Users must be able to share their travel timeline with friends, family, and fellow travelers.
   2. Users should have the ability to control who can view and comment on their travel timeline.
4. **Destination Ratings**
   1. Users must be able to leave and view star ratings for various destinations including historical sites, museums, art galleries, hotels, and restaurants.
   2. Users must be able to document destination ratings within their journal entries.
   3. Ratings and reviews should be visible within the user's timeline and contribute to overall destination ratings.
5. **Message Forum**
   1. Users must be able to participate in a dedicated message forum for sharing travel tips and seeking advice.
   2. Users should be able to create and respond to message threads within the forum.
6. **Traveler Meetups**
   1. Application should provide a feature for organizing and joining meetups with other travelers.
   2. Users should be able to receive notifications about meetups at their travel destinations.
7. **Cross-Platform Compatibility**
   1. The application should function on various web browsers, including Chrome, Firefox, Safari, and Edge.
   2. The mobile application version should be compatible with both iOS and Android mobile web browsers.
8. **Data Backups and Recovery:**
   1. The system should perform automated daily backups of user data to prevent data loss.
   2. Users should have the ability to recover their data in case of accidental deletion or system failures.
9. **Data Privacy and Consent:**
   1. Users must be informed about data collection practices and provided with options to consent or opt out of data sharing.
   2. The application should ensure that personally identifiable information (PII) is handled securely and in compliance with privacy regulations.

## 2.4 Nonfunctional Requirements

1. **Performance:**
   1. The system should ensure optimal user experience by responding to user actions within a 2-second timeframe.
   2. The application must sustain satisfactory performance even during peak travel seasons, managing increased usage without a substantial decrease in functionality.
   3. The system's uptime should be maintained at a minimum of 99% to guarantee consistent and reliable access to users' travel timelines.

1. **Security:**
   1. The application must employ robust user authentication methods to safeguard against unauthorized access.
   2. To mitigate the risk of data loss, automated, routine backups of user data should be conducted.

1. **Usability:**
   1. The user interface should be intuitive and user-friendly, ensuring that users can easily document and share their travel experiences.
   2. The application must adhere to accessibility standards to ensure usability by individuals with disabilities.
   3. "Chronicle" should seamlessly operate across a variety of web browsers and mobile devices for a consistent user experience.

1. **Scalability:**
   1. The database architecture must accommodate growth in the user base, guaranteeing efficient data storage and retrieval as the platform expands.

1. **Data Privacy and Compliance:**
   1. The application should adhere to GDPR (General Data Protection Regulation, (EU)), PIPEDA (The Personal Information Protection and Electronic Documents Act, (CAN)) regulations regarding user data protection and privacy.
   2. Users must provide clear, explicit consent for data collection, usage, and sharing in accordance with relevant privacy laws.
   3. Users should be transparently informed about the utilization of their data for monetization purposes and afforded the choice to either participate or decline.

1. **Data Handling:**
   1. The application must anonymize and aggregate user-generated data, such as destination ratings and reviews, to protect user privacy while making it available for monetization purposes.
   2. Users must be provided with clear options to consent or opt out of data sharing, ensuring compliance with privacy regulations.
   3. Robust data security measures should be in place to protect both private and non-private user data from unauthorized access or breaches.

1. **Performance Testing:**
   1. The system should undergo load testing to determine its maximum capacity and identify performance bottlenecks.
   2. The application should be tested under extreme conditions to ensure it remains stable and responsive.

1. **Backup and Recovery:**
   1. In case of data loss or system failures, the application should have a robust data recovery mechanism in place.
   2. Backups should be performed daily, and historical backups should be retained for a specified period.

1. **Third-Party Integrations:**
   1. If integrating with third-party services (e.g., mapping services or weather data), the application should handle these integrations smoothly without significant delays or errors.

1. **Data Retention Policy:**
   1. The system should have a data retention policy that allows users to delete their data permanently if they choose to do so.

1. **Geographic Availability:**
   1. The application's servers should be strategically located to ensure users from different geographic regions experience reasonable response times.

## 2.5 Project Scope

**1.Objective:**

Develop a platform, Chronicle, enabling travelers to document, share, and interact, enhancing their travel experiences and fostering a global community.

**2. Deliverables:**

Web and mobile application.

User-centered UI/UX.

Secure cloud storage.

**3. Features:**

Travel Journal: Document and manage journeys.

Share Adventures: Share and collaborate on entries.

Destination Ratings: Rate and review places.

Message Forum: Community discussions.

Traveler Meetups: Organize/join travel meetups.

**4. Technical Requirements:**

Utilize JavaScript, React/Next, MongoDB or PostgreSQL.

Ensure mobile responsiveness.

**5. Milestones:**

Phase 1 (4 months): Defining requirements, Design.

Phase 2 (4 months): Development, Testing.

**6. Quality Assurance:**

Thorough testing on various devices/browsers.

**7. Communication Plan:**

Hold regular team meetings and use collaboration tools.

## 2.6 System Risks

|  |  |
| --- | --- |
| **Risk** | **Response** |
| **Performance Risk:** The system must respond to user actions within a 2-second timeframe. | The response is to frequently perform the load testing and optimize the application for speed. |
| **Performance Risks:** The user interface should be intuitive and user-friendly. | To regularly conduct usability testing and adhere to application accessibility standards. |
| **Application and System Architecture:** The database architecture should accommodate growth. | To frequently monitor and optimize the database as the user base grows. |
| **Organizational or Scope Creep:** Expanding the scope of the project beyond what was originally defined can lead to delays and can overrun time, resources and budget. | The response is to have a well-established project scope and strictly manage changes to it. |
| **Data Loss and Security Risks:** The system must ensure that automated daily backups are performed. | Response is to test the backup mechanisms to ensure their reliability. |
| **Backup and Recovery Risks:** In case of data loss or system failures, the application should have a robust data recovery mechanism. | Create a rule to always test and maintain the recovery system. |
| **Data Privacy and Compliance Risks:** Users' personal information needs to be handled securely in compliance with privacy regulations. | To implement robust data protection measures and keep up to date with privacy regulations. |
| **Data Handling and Retention Risks:** The application must encrypt and aggregate user-generated data to protect privacy. | Response is to implement strong data encryption and aggregation techniques. |
| **Data Deletion Risk:** The system should allow users to delete their data permanently. | Response it to develop a robust data deletion mechanism and clearly communicate data retention policies to users. |

## 2.7 Operating Environment

**1. Operating System:**

Chronicle should be optimized to run seamlessly within various mobile web browsers on various operating systems, including Android and iOS.

**2. Hardware:**

The application should be lightweight and designed to operate efficiently on a range of mobile devices, accounting for varying screen sizes, processing power, memory, and storage capabilities.

**3. Software Dependencies:**

Chronicle should be self-contained within the mobile web environment, minimizing dependencies on external software components that may not be readily available on mobile browsers.

**4. Networking and Connectivity:**

The application should be designed to handle variable network conditions commonly encountered on mobile devices, ensuring users can access the application features even with limited bandwidth or intermittent connectivity.

**5. Database Systems:**

Chronicle will need to utilize a database, that should be compatible with mobile browsers and designed for efficient data storage and retrieval on mobile devices.

**6. Security Measures:**

Robust security measures, including encryption, authentication, and authorization mechanisms, should be integrated into Chronicle to protect user data within the mobile web environment.

**7. User Interfaces:**

The mobile web application's user interface (UI) and user experience (UX) should be responsive and adaptable to different mobile screen sizes, touch-based input methods, and accessibility requirements.

**8. Localization and Internationalization:**

During development and prototyping, Chronicle will be designed in English and within the North American region. Future development will support localization for various languages and internationalization to adapt to cultural norms and preferences, ensuring a user-friendly experience on mobile devices globally.

**9. Scalability and Performance Requirements:**

Developers should optimize Chronicle for performance on mobile browsers, considering the limitations of mobile devices while ensuring it remains responsive even during peak usage periods.

**10. Compliance and Regulations:**

If Chronicle processes user data subject to privacy regulations like GDPR or PIPEDA, it must adhere to these regulations within the mobile web environment, ensuring data protection and user consent mechanisms are in place.

**11. Deployment Environment:**

Chronicle should be accessible as a mobile web app, eliminating the need for installation while providing flexibility in accessing it from various mobile devices and browsers.

**12. Monitoring and Management Tools:**

Implementing monitoring and management tools within the mobile web environment is essential for diagnosing issues, ensuring uptime, and maintaining the application's health and performance in the context of mobile usage.

## 2.8 UI/UXD Interface Mock-ups

|  |
| --- |
| Screen 1 |
| Screen 2 |
| Screen 3 |
| Screen 4 |
| Screen 5 |
| Screen 6 |
| Screen 7 |
| Screen 8 |
| Screen 9 |
| Screen 10 |
| Screen 11 |
| Screen 12 |
| Screen 13 |
| Screen 14 |
| Screen 15 |

# Process and Data Modeling

## **3.1 UML/DFD Modeling and Data Modeling**

## Activity Diagrams

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

Data Flow diagram

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

## **3.2 Business Rules**

|  |  |  |
| --- | --- | --- |
| Business Rule Number | Business Rule Description | Related UC |
| BR01 | New users must provide a valid username/email, and password to register for the app | Create User Account (UC01) |
| BR02 | Users can sign in only if they provide valid account credentials, including username and password. | User Sign In (UC02) |
| BR03 | Users can edit their profile information only after successfully signing into their account. | Edit Profile Info (UC03) |
| BR04 | Users can delete their profile, but this action requires confirmation to avoid accidental deletions. | Delete Profile (UC04) |
| BR05 | Users can view profiles of other users upon selecting the respective user's icon. | View Other User Profile (UC05) |
| BR06 | Users can view another user's timeline upon selecting that user's timeline. | View Other User Timeline (UC06) |
| BR07 | Users can search their timelines by entering specific keywords or criteria or hashtags. | Search User Timelines (UC07) |
| BR08 | Users can view their own timeline, which displays their timeline entries. | View Timeline (UC08) |
| BR09 | Users can add journal entries to their timeline, providing details such as the entry title, date, and text. | Add Journal Entry to Timeline (UC09) |
| BR10 | Users can view detailed journal entries in their timeline, including text, media, and location information. | View Detailed Timeline Entry (UC10) |
| BR11 | Users can search for other users' timelines by entering specific keywords or criteria or hashtags. | Search Other User Timeline (UC11) |
| BR12 | Users can view the contents of their own travel journal, and if no entries exist, a message indicates that no entries added. | View Journal (UC12) |
| BR13 | Users can view a detailed version of a specific journal entry in their travel journal. | View Journal Entry (UC13) |
| BR14 | Users can add new journal entries to their travel journal, including optional media and location information. | Add Journal Entry (UC14) |
| BR15 | Users can edit or update existing journal entries in their travel journal. | Edit Journal Entry (UC15) |
| BR16 | Users can delete specific journal entries from their travel journal, with confirmation to prevent accidental deletion. | Delete Journal Entry (UC16) |
| BR17 | Users can search for journal entries containing specific keywords within their travel journal. | Search Journal Entries (UC17) |
| BR18 | Users can view the message forum and participate in discussions. | View Forum (UC18) |
| BR19 | Users can view specific threads in the message forum, including the thread's content and replies. | View Thread (UC19) |
| BR20 | Users can create new message forum threads, including titles, text, and attachments. | Create Forum Thread (UC20) |
| BR21 | Users can search for message forum threads containing specific keywords or criteria. | Search For a Thread (UC21) |
| BR22 | Users can reply to existing message forum threads, adding comments or contributions to the discussion. | Reply to Thread (UC22) |

## **3.3 Use Case Specifications with corresponding interface mockups:**

**Each use case needs to have the following:**

1- **Business Rules.**

**2- System Use Case Diagrams.**

**3- Use Case Descriptions.**

**4- Corresponding Mockups**

**Below are our Use Case specifications:**

***Note: first diagrams followed by the use case descriptions, for each description, associated use case diagram and Mock-up image numbers are listed. Above mentioned business rules have related Use Case Diagram number.***

***Use Case Diagrams:***

|  |
| --- |
| Use Case Diagram 1 |

|  |
| --- |
| Use Case Diagram 2 |

|  |
| --- |
| Use Case Diagram 3 |

|  |
| --- |
| Use Case Diagram 4 |

|  |
| --- |
| Use Case Diagram 5 |

|  |
| --- |
| Use Case Diagram 6 |

|  |
| --- |
| Use Case Diagram 7 |

|  |
| --- |
| Use Case Diagram 8 |

|  |
| --- |
| Use Case Diagram 9 |

|  |
| --- |
| Use Case Diagram 10 |

|  |
| --- |
| Use Case Diagram 11 |

|  |
| --- |
| Use Case Diagram 12 |

|  |
| --- |
| Use Case Diagram 13 |

|  |
| --- |
| Use Case Diagram 14 |

|  |
| --- |
| Use Case Diagram 15 |

|  |
| --- |
| Use Case Diagram 16 |

|  |
| --- |
| Use Case Diagram 17 –Accumulation of All the Diagrams |

***Use Case Description: Create user Account.***

|  |  |
| --- | --- |
| **ID:** | **2**  **(Corresponding Mockup - Screen 1, 2)**  **(Corresponding Use Case Diagram – 1,2)** |
| **Use Case Name:** | Create User Account |
| **Brief Description:** | The user wants to create an account. |
| **Stakeholders and Interests:** | The User – wants to create an account |
| **Primary Actor:** | User |
| **Preconditions:** | The User has opened the app. |
| **Postconditions:** | The User’s account is created. |
| **Normal Flow of Events:** | 1. The user pressed the ‘create account’ button. 2. The System displays the popup form for user account creation, waits for user input. 3. User enters account information, submits the form. 4. System saves data to the database, creates a new account and redirects user to the homepage. |
| **Sub Flows:** | N/A |
| **Alternate/Exception Flows:** | N/A |
| **Frequency of Use:**  **(High/Medium/Low)** | High |
| **Status:**  **(Developed/In Development/Undeveloped** | Undeveloped |
| **Priority:**  **(High/Medium/Low)** | High |

***Use Case Description:*** ***User Sign In***

|  |  |
| --- | --- |
| **ID:** | **3**  **(Corresponding Mockup - Screen 3)**  **(Corresponding Use Case Diagram – 1,3)** |
| **Use Case Name:** | User Sign In |
| **Brief Description:** | The user wants to sign in |
| **Stakeholders and Interests:** | The User – wants to sign in |
| **Primary Actor:** | User |
| **Preconditions:** | The User has opened the app.  The user has an account. |
| **Postconditions:** | The User is signed in. |
| **Normal Flow of Events:** | 1. The user pressed the ‘sign in’ button. 2. The System displays the popup form for user sign in, waits for user input. 3. User enters account information, submits the form. 4. System authenticates the user, and redirects user to the homepage. |
| **Sub Flows:** | N/A |
| **Alternate/Exception Flows:** | N/A |
| **Frequency of Use:**  **(High/Medium/Low)** | High |
| **Status:**  **(Developed/In Development/Undeveloped** | Undeveloped |
| **Priority:**  **(High/Medium/Low)** | High |

***Use Case Description: Edit Profile Info***

|  |  |
| --- | --- |
| **ID:** | **4**  **(Corresponding Mockup - Screen 7)**  **(Corresponding Use Case Diagram - 7)** |
| **Use Case Name:** | Edit Profile Information |
| **Brief Description:** | The user wants to edit their profile information. |
| **Stakeholders and Interests:** | The User – wants to edit profile info |
| **Primary Actor:** | User |
| **Preconditions:** | The User has an account.  The User has signed into the app successfully.  The User opened ‘Edit Profile’ page. |
| **Postconditions:** | The User’s profile is edited. |
| **Normal Flow of Events:** | 1. The user pressed the ‘edit account’ button. 2. The System displays the popup form for user account editing, waits for user input. 3. User enters updated account information, submits the form. 4. System saves data to the database, and redirects user to the profile page.. |
| **Sub Flows:** | N/A |
| **Alternate/Exception Flows:** | N/A |
| **Frequency of Use:**  **(High/Medium/Low)** | High |
| **Status:**  **(Developed/In Development/Undeveloped** | Undeveloped |
| **Priority:**  **(High/Medium/Low)** | High |

***Use Case Description: Delete Profile***

|  |  |
| --- | --- |
| **ID:** | **5**  **(Corresponding Mockup - Screen 15)**  **(Corresponding Use Case Diagram - 8)** |
| **Use Case Name:** | Delete profile. |
| **Brief Description:** | The user wants to delete their profile. |
| **Stakeholders and Interests:** | The User – wants to delete their profile. |
| **Primary Actor:** | User |
| **Preconditions:** | The User has an account.  The User has signed into the app successfully.  The User opened ‘Edit Profile’ page. |
| **Postconditions:** | The User’s profile is deleted. |
| **Normal Flow of Events:** | 1. The user pressed the ‘delete account’ button. 2. The System asks user to confirm the deletion of an account.   If the User confirms:   * The S-1: Delete profile sub-flow is performed.   If the User cancels:   * Return User to ‘Edit Profile’ page. |
| **Sub Flows:** | S-1: Delete Profile   1. Delete all the user’s records from database. |
| **Alternate/Exception Flows:** | N/A |
| **Frequency of Use:**  **(High/Medium/Low)** | Low |
| **Status:**  **(Developed/In Development/Undeveloped** | Undeveloped |
| **Priority:**  **(High/Medium/Low)** | High |

***Use Case Description: View Other User Profile***

|  |  |
| --- | --- |
| **ID:** | **6**  **(Corresponding Mockup - Screen 14)**  **(Corresponding Use Case Diagram - 13)** |
| **Use Case Name:** | View other user profile. |
| **Brief Description:** | The user wants to view other user profile. |
| **Stakeholders and Interests:** | The User – wants to view their profile. |
| **Primary Actor:** | User |
| **Preconditions:** | The User has an account.  The User has signed into the app successfully. |
| **Postconditions:** | The User’s profile is displayed. |
| **Normal Flow of Events:** | 1. The user presses on the icon of another user’s profile. 2. The system retrieves the info about another user’s profile. Y=The user is redirected to the viewProfile page. |
| **Sub Flows:** | N/A |
| **Alternate/Exception Flows:** | N/A |
| **Frequency of Use:**  **(High/Medium/Low)** | Low |
| **Status:**  **(Developed/In Development/Undeveloped** | Undeveloped |
| **Priority:**  **(High/Medium/Low)** | High |

***Use Case Description: ViewOtherUserTimeline***

|  |  |
| --- | --- |
| **ID:** | **7**  **(Corresponding Mockup - Screen 4)**  **(Corresponding Use Case Diagram - 14)** |
| **Use Case Name:** | ViewOtherUserTimeline |
| **Brief Description:** | The user wants to view another user’s timeline. |
| **Stakeholders and Interests:** | The User – wants to view wants to view another user’s timeline. |
| **Primary Actor:** | User |
| **Preconditions:** | The User has an account.  The User has signed into the app successfully.  The User opened Home page. |
| **Postconditions:** | The User’s timeline is displayed to the user. |
| **Normal Flow of Events:** | 1. The user pressed the ‘view timeline’ button on another user’s post. 2. The System retrieves information about the user’s timeline and displays it. |
| **Sub Flows:** | N/A |
| **Alternate/Exception Flows:** | N/A |
| **Frequency of Use:**  **(High/Medium/Low)** | Medium |
| **Status:**  **(Developed/In Development/Undeveloped** | Undeveloped |
| **Priority:**  **(High/Medium/Low)** | Low |

***Use Case Description: Search User Timelines***

|  |  |
| --- | --- |
| **ID:** | **8**  **(Corresponding Mockup - Screen 4)**  **(Corresponding Use Case Diagram – 13, 14)** |
| **Use Case Name:** | Search User Timelines |
| **Brief Description:** | The user wants to search for another user timeline. |
| **Stakeholders and Interests:** | The User – wants to search for another user timeline. |
| **Primary Actor:** | User |
| **Preconditions:** | The User has an account.  The User has signed into the app successfully.  The User opened the profile page of another user.  The user clicked on the Search Bar. |
| **Postconditions:** | The list of timelines is displayed. |
| **Normal Flow of Events:** | 1. The user entered the name/hashtag/place to the search bar and clicked “Search” button. 2. The System retrieved all the relevant records and displayed it. |
| **Sub Flows:** | N/A |
| **Alternate/Exception Flows:** | N/A |
| **Frequency of Use:**  **(High/Medium/Low)** | Low |
| **Status:**  **(Developed/In Development/Undeveloped** | Undeveloped |
| **Priority:**  **(High/Medium/Low)** | Low |

***Use Case Description: View Timeline***

|  |  |
| --- | --- |
| **ID:** | **9**  **(Corresponding Mockup - Screen 4)**  **(Corresponding Use Case Diagram - 15)** |
| **Use Case Name:** | View Timeline |
| **Brief Description:** | The user wants to view a timeline. |
| **Stakeholders and Interests:** | The User – wants to view a timeline. |
| **Primary Actor:** | User |
| **Preconditions:** | The User has an account.  The User has signed into the app successfully.  The User opened Home page.  The user clicked on the Search Bar. |
| **Postconditions:** | The list of timelines is displayed. |
| **Normal Flow of Events:** | 1. The user clicks on the timeline. 2. The System retrieved all the records about the single timeline and displayed it. |
| **Sub Flows:** | N/A |
| **Alternate/Exception Flows:** | N/A |
| **Frequency of Use:**  **(High/Medium/Low)** | High |
| **Status:**  **(Developed/In Development/Undeveloped** | Undeveloped |
| **Priority:**  **(High/Medium/Low)** | High |

***Use Case Description: Add Journal Entry to Timeline***

|  |  |
| --- | --- |
| **ID:** | **10**  **(Corresponding Mockup - Screen 13)**  **(Corresponding Use Case Diagram - 16)** |
| **Use Case Name:** | Add Journal Entry to Timeline |
| **Brief Description:** | The user wants to add journal entry to timeline. |
| **Stakeholders and Interests:** | The User – wants to add journal entry to timeline. |
| **Primary Actor:** | User |
| **Preconditions:** | The User has an account.  The User has signed into the app successfully.  The User opened the Journal. |
| **Postconditions:** | Journal Entry is Added to TimeLine |
| **Normal Flow of Events:** | 1. The user clicks ‘Add Journal Entry’ Button 2. The System displays the popup. Waits for user input. 3. User enters info for journal entry, clicks ‘save’ button. 4. The system publishes the journal entry, adds to the TimeLine, saves info in database. |
| **Sub Flows:** | N/A |
| **Alternate/Exception Flows:** | N/A |
| **Frequency of Use:**  **(High/Medium/Low)** | High |
| **Status:**  **(Developed/In Development/Undeveloped** | Undeveloped |
| **Priority:**  **(High/Medium/Low)** | High |

***Use Case Description: View Detailed Timeline Entry***

|  |  |
| --- | --- |
| **ID:** | **11**  **(Corresponding Mockup - Screen 12)**  **(Corresponding Use Case Diagram - 17)** |
| **Use Case Name:** | View Detailed Timeline Entry |
| **Brief Description:** | The user wants to add journal entry to timeline. |
| **Stakeholders and Interests:** | The User – wants to add journal entry to timeline. |
| **Primary Actor:** | User |
| **Preconditions:** | The User has an account.  The User has signed into the app successfully.  The User opened the Journal. |
| **Postconditions:** | Journal Entry is Added to TimeLine |
| **Normal Flow of Events:** | 1. The user clicks ‘Add Journal Entry’ Button 2. The System displays the popup. Waits for user input. 3. User enters info for journal entry, clicks ‘save’ button. 4. The system publishes the journal entry, adds to the TimeLine, saves info in database. |
| **Sub Flows:** | N/A |
| **Alternate/Exception Flows:** | N/A |
| **Frequency of Use:**  **(High/Medium/Low)** | High |
| **Status:**  **(Developed/In Development/Undeveloped** | Undeveloped |
| **Priority:**  **(High/Medium/Low)** | High |

***Use Case Description: Search Other User Timeline***

|  |  |
| --- | --- |
| **ID:** | **12**  **(Corresponding Mockup - Screen 4)**  **(Corresponding Use Case Diagram - 14)** |
| **Use Case Name:** | Search Other User Timeline |
| **Brief Description:** | The user wants to search another user timeline. |
| **Stakeholders and Interests:** | The User – wants to search another user timeline |
| **Primary Actor:** | User |
| **Preconditions:** | The User has an account.  The User has signed into the app successfully.  The User has opened the home page. |
| **Postconditions:** | Another User’s timeline is displayed. |
| **Normal Flow of Events:** | 1. The user clicks on the search bar. 2. The system displays the input field, waits for the user input. 3. User enters info into the input field, clicks the ‘search’ button. 4. The system retrieves records of timelines from the database, displays the list of the records. |
| **Sub Flows:** | N/A |
| **Alternate/Exception Flows:** | N/A |
| **Frequency of Use:**  **(High/Medium/Low)** | medium |
| **Status:**  **(Developed/In Development/Undeveloped** | Undeveloped |
| **Priority:**  **(High/Medium/Low)** | medium |

***Use Case Description: View Journal***

|  |  |
| --- | --- |
| **ID:** | **13**  **(Corresponding Mockup - Screen 5)**  **(Corresponding Use Case Diagram - 5)** |
| **Use Case Name:** | View Journal |
| **Brief Description:** | The User wants to view the contents (i.e., journal entries) of their travel journal. |
| **Stakeholders and Interests:** | The User – wants to view their journal entries. |
| **Primary Actor:** | User |
| **Preconditions:** | The User has an account.  The User has signed into the app successfully. |
| **Postconditions:** | The User can view their journal contents. |
| **Normal Flow of Events:** | 1. The user selects the “Journal” tab on the dashboard controller. 2. The System redirects the User to their Journal page.   If the Journal page contains entries:   * The S-1: Display Journal Contents sub-flow is performed.   If the Journal page contains entries:   * The S-2: Display Empty Journal Message sub-flow is performed. |
| **Sub Flows:** | S-1: Display Journal Contents:   1. The System fetches all of the Users journal entries and renders them within the “ViewJournal” page. 2. The System display a chronological listing of the journal entries.   S-2: Display Empty Journal Message:   1. The System displays a message “You Have No Journal Entries” |
| **Alternate/Exception Flows:** | N/A |
| **Frequency of Use:**  **(High/Medium/Low)** | High |
| **Status:**  **(Developed/In Development/Undeveloped)** | Undeveloped |
| **Priority:**  **(High/Medium/Low)** | High |

***Use Case Description: View Journal Entry***

|  |  |
| --- | --- |
| **ID:** | **14**  **(Corresponding Mockup - Screen 5)**  **(Corresponding Use Case Diagram - 5)** |
| **Use Case Name:** | View Journal Entry |
| **Brief Description:** | The User wants to view a detailed version of a journal entry. |
| **Stakeholders and Interests:** | The User – wants to view their entire journal entry. |
| **Primary Actor:** | User |
| **Preconditions:** | The User is viewing the “ViewJournal” page of the application. |
| **Postconditions:** | The User is displayed a detailed view of a selected journal entry. |
| **Normal Flow of Events:** | 1. The User selects a Journal entry by tapping the entry header, description, or attachment of a journal entry on the “ViewJournal” page. 2. The system fetches the information of the journal entry the user selected from the Journal database within the Journal Management Subsystem. 3. The system redirects the user to the “ViewJournalEntry” page, where the journal entry is displayed in a more detailed layout. The full journal text is displayed, along with larger versions of any photos attached to the entry, videos are playable, and any destination reviews are displayed as well. |
| **Sub Flows:** | N/A |
| **Alternate/Exception Flows:** | N/A |
| **Frequency of Use:**  **(High/Medium/Low)** | High |
| **Status:**  **(Developed/In Development/Undeveloped)** | Undeveloped |
| **Priority:** | High |

***Use Case Description: Add Journal Entry***

|  |  |
| --- | --- |
| **ID:** | **15**  **(Corresponding Mockup - Screen 6)**  **(Corresponding Use Case Diagram - 6)** |
| **Use Case Name:** | Add Journal Entry |
| **Brief Description:** | The User wants to add a new journal entry to their travel journal. |
| **Stakeholders and Interests:** | The User – wants to add new entries to their journal. |
| **Primary Actor:** | User |
| **Preconditions:** | The User is viewing the “ViewJournal” page of the application. |
| **Postconditions:** | The User has added a new journal entry to their travel journal. |
| **Normal Flow of Events:** | 1. The User selects the “+” icon on the “ViewJournal” page. 2. The System redirects the user to a fillable form for adding a new journal entry. 3. The User enters in the appropriate information for creating a new journal entry on the form such as: Entry Title, Date, and Entry Text  The User also can include attachments such a picture, a video, or choose a location/destination and leave a star rating. 4. If the User selects “Add Entry” Button:  * The S-1: Post New Journal Entry sub-flow is performed.   If the User selects “Discard” Button:   * The S-2: Discard Journal Entry Form Contents sub-flow is performed. |
| **Sub Flows:** | S-1: Post New Journal Entry:   1. The System creates a new journal entry with the information from the form and adds it to the Users journal (added to the Journal Management Sub-system database) 2. The System displays a modal alert that states “New Journal Entry Added”   S-2: Discard Journal Entry:   1. The System displays a modal confirmation to the User: “Are you sure you want to discard this entry?”   If the User Selects “Yes”:   1. The System dismisses the modal confirmation and clears the form. 2. The User is redirected back to the “ViewJournal” page.   If the User Selects “No”:   1. The System dismisses the modal confirmation and returns control back to the form. |
| **Alternate/Exception Flows:** | N/A |
| **Frequency of Use:**  **(High/Medium/Low)** | High |
| **Status:**  **(Developed/In Development/Undeveloped)** | Undeveloped |
| **Priority:** | High |

***Use Case Description: Edit Journal Entry***

|  |  |
| --- | --- |
| **ID:** | **16**  **(Corresponding Mockup - Screen 5)**  **(Corresponding Use Case Diagram - 5)** |
| **Use Case Name:** | Edit Journal Entry |
| **Brief Description:** | The User wants to edit/make changes to an existing journal entry in their travel journal. |
| **Stakeholders and Interests:** | The User – update or correct a previously added journal entry. |
| **Primary Actor:** | User |
| **Preconditions:** | The User is viewing the “ViewJournal” page or the “ViewJournalEntry” page, and has selected the three dots:  (“ . . . “) above a journal entry. A modal menu is displayed with a n edit option. |
| **Postconditions:** | All changes the User has made to the post they chose to edit have been updated, and those new changes are viewable to the User. |
| **Normal Flow of Events:** | 1. The User selects the “Edit” option from the “three-dot” modal menu choices displayed to them. 2. The System redirects the User to the “EditJournalEntry” page. 3. The System fetches the journal entry details for the entry selected by the User and fills the form with the current entry information. 4. The User edits information on the form such as: Entry Title, Date, and Entry Text  The User also can add/remove attachments such a picture, a video, or location/destination star rating. 5. If the User selects “Save Changes” Button:  * The S-1: Save Changes sub-flow is performed.   If the User selects the “Discard” Button:   * The S-2: Discard Changes sub-flow is performed. |
| **Sub Flows:** | S-1: Save Changes:   1. If there are any changes to the form fields:  * Any altered field values of the form are posted as changes to the journal entry and the changes are updated to the Journal Management Subsystem database. Any new photos, videos, or ratings are updated to the database. Any media selected for removal are deleted from the journal entry and updated to the database. * The System displays a modal alert that states “Changes Saved”.   If there are no changes to the form fields:   * The System displays a modal message that states “No Changes Were Made”  1. The System redirects the User to the previous page (either “ViewJournal” page or “ViewJournalEntry” page.)   S-2: Discard Changes   1. The System displays a modal confirmation to the User: “Are you sure you want to discard all changes?”   If the User Selects “Yes”:   * The System dismisses the modal confirmation, and the contents of the form are reverted to the original journal entry details and the System returns control back to the form.   If the User Selects “No”   * The System dismisses the modal confirmation and returns control back to the form. |
| **Alternate/Exception Flows:** | N/A |
| **Frequency of Use:**  **(High/Medium/Low)** | Medium |
| **Status:**  **(Developed/In Development/Undeveloped)** | Undeveloped |
| **Priority:** | High |

***Use Case Description: Delete Journal Entry***

|  |  |
| --- | --- |
| **ID:** | **17**  **(Corresponding Mockup - Screen 5)**  **(Corresponding Use Case Diagram - 6)** |
| **Use Case Name:** | Delete Journal Entry |
| **Brief Description:** | The User wants to delete a Journal entry from their travel journal. |
| **Stakeholders and Interests:** | The User – have a journal entry deleted. |
| **Primary Actor:** | User |
| **Preconditions:** | The User is viewing the “ViewJournal” page or the “ViewJournalEntry” page, and has selected the three dots:  (“ . . . “) above a journal entry. A modal menu is displayed with a delete option. |
| **Postconditions:** | The Journal entry the user selects for deletion is removed from the Users journal and is no longer listed in the Users journal. |
| **Normal Flow of Events:** | 1. The User selects the “Delete” option from the “three-dot” modal menu choices displayed to them. 2. The System displays a model alert confirmation prompt asking: “Are you sure you want to delete this entry?” 3. If the User selects “Yes” Button:  * The S-1: Delete Journal Entry sub-flow is performed.   If the User selects the “Cancel” Button:   * The S-2: Cancel Deletion sub-flow is performed.  1. The System redirects the User to the previous page (either “ViewJournal” page or “ViewJournalEntry” page.) |
| **Sub Flows:** | S-1: Delete Journal Entry:   1. The System dismisses the modal alert confirmation prompt. 2. The journal entry is deleted from the Journal Management Subsystem database. 3. The System displays a modal confirmation message stating “The selected journal entry has been deleted!” 4. The System dismisses the modal confirmation message.   S-2: Cancel Deletion:   1. The System dismisses the modal alert confirmation message prompt. |
| **Alternate/Exception Flows:** | N/A |
| **Frequency of Use:**  **(High/Medium/Low)** | Medium |
| **Status:**  **(Developed/In Development/Undeveloped)** | Undeveloped |
| **Priority:** | High |

***Use Case Description: Search Journal Entries***

|  |  |
| --- | --- |
| **ID:** | **18**  **(Corresponding Mockup - Screen 5)**  **(Corresponding Use Case Diagram - 5)** |
| **Use Case Name:** | Search Journal Entries |
| **Brief Description:** | The User wants to search for journal entries containing one or more keywords. |
| **Stakeholders and Interests:** | The User – allow them to search their travel journal. |
| **Primary Actor:** | User |
| **Preconditions:** | The User is viewing the “ViewJournal” page. |
| **Postconditions:** | The User is displayed journal entries containing one or more keywords used to search their journal. |
| **Normal Flow of Events:** | 1. The User enters search criteria text in the search field. 2. The User taps the magnifying glass button. 3. The System fetches journal entries containing any of the keywords used as search criteria. 4. The System displays all entries containing the search criteria keywords to the user within the “ViewJournal” page, in chronological order. |
| **Sub Flows:** | N/A |
| **Alternate/Exception Flows:** | N/A |
| **Frequency of Use:**  **(High/Medium/Low)** | High |
| **Status:**  **(Developed/In Development/Undeveloped)** | Undeveloped |
| **Priority:** | High |

***Use Case Description: View Forum***

|  |  |
| --- | --- |
| **ID:** | **19**  **(Corresponding Mockup - Screen 8)**  **(Corresponding Use Case Diagram - 9)** |
| **Use Case Name:** | View Forum |
| **Brief Description:** | The User wants to view the message forum. |
| **Stakeholders and Interests:** | The User – ability to use the message forum |
| **Primary Actor:** | User |
| **Preconditions:** | The User has an account.  The User has signed into the app successfully. |
| **Postconditions:** | The User is displayed a listing of the message forum threads, in chronological order. |
| **Normal Flow of Events:** | 1. The User selects the “Forum” tab on the dashboard controller. 2. The System fetches message forum threads and renders them within the “ViewForum” page. 3. The System redirects the User to the “ViewForum” and displays all forum threads. |
| **Sub Flows:** | N/A |
| **Alternate/Exception Flows:** | N/A |
| **Frequency of Use:**  **(High/Medium/Low)** | High |
| **Status:**  **(Developed/In Development/Undeveloped)** | Undeveloped |
| **Priority:** | High |

***Use Case Description: View Thread***

|  |  |
| --- | --- |
| **ID:** | **20**  **(Corresponding Mockup - Screen 10)**  **(Corresponding Use Case Diagram - 9)** |
| **Use Case Name:** | View Thread |
| **Brief Description:** | The User wants to view a thread selected from the “ViewForum” page. |
| **Stakeholders and Interests:** | The User – ability to access threads. |
| **Primary Actor:** | User |
| **Preconditions:** | The User is viewing the “ViewForum” page. |
| **Postconditions:** | The User is displayed the selected thread and its replies. |
| **Normal Flow of Events:** | 1. The User selects a thread by tapping on its title or thread description. 2. The System fetches the thread (including the replies) from the Message Forum Subsystem database. 3. The System renders the thread on the “ViewForum” page and redirects the User to the page. |
| **Sub Flows:** | N/A |
| **Alternate/Exception Flows:** | N/A |
| **Frequency of Use:**  **(High/Medium/Low)** | High |
| **Status:**  **(Developed/In Development/Undeveloped)** | Undeveloped |
| **Priority:** | High |

***Use Case Description: Create Forum Thread***

|  |  |
| --- | --- |
| **ID:** | **21**  **(Corresponding Mockup - Screen 8, 9)**  **(Corresponding Use Case Diagram - 11)** |
| **Use Case Name:** | Create Forum Thread |
| **Brief Description:** | The User wants to create a new message forum thread. |
| **Stakeholders and Interests:** | The User – ability to start a new message forum thread. |
| **Primary Actor:** | User |
| **Preconditions:** | The User is viewing the “ViewForum” page. |
| **Postconditions:** | The User has created a new message forum thread, |
| **Normal Flow of Events:** | 1. The User has selected the “+” icon on the “ViewForum” page. 2. The System redirects the User to a “CreateForumThread” page that contains a fillable form with fields for thread title, text, and attachments. 3. The User fills in the form fields. 4. If the User selects the “Add Thread” Button:  * The S-1: Post New Forum Thread sub-flow is performed.   If the User the “Discard” Button:   * The S-2: Discard Thread Creation sub-flow is performed. |
| **Sub Flows:** | S-1: Post New Forum Thread:   1. The system creates a new forum thread with the information from the form and adds it to the Message Forum Subsystem database. 2. The System displays a modal alert that states: “New Forum Thread Created”.   S-2: Discard Thread Creation:   1. The System displays a modal confirmation to the User: “Are you sure you want to discard this thread?”   If the User selects “Yes”:   1. The System dismisses the modal confirmation and clears the form. 2. The System redirects the User back to the “ViewForum” page.   If the User selects “No”:   1. This System dismisses the modal confirmation and returns control back to the form. |
| **Alternate/Exception Flows:** | N/A |
| **Frequency of Use:**  **(High/Medium/Low)** | High |
| **Status:**  **(Developed/In Development/Undeveloped)** | Undeveloped |
| **Priority:** | High |

***Use Case Description: Search For a Thread***

|  |  |
| --- | --- |
| **ID:** | **22**  **(Corresponding Mockup - Screen 8)**  **(Corresponding Use Case Diagram - 10)** |
| **Use Case Name:** | Search For a Thread |
| **Brief Description:** | The User wants to search for journal entries containing one or more keywords. |
| **Stakeholders and Interests:** | The User – allow them to search the message forum. |
| **Primary Actor:** | User |
| **Preconditions:** | The User is viewing the “ViewForum” page. |
| **Postconditions:** | The User is displayed journal entries containing one or more keywords used to search the message forum threads. |
| **Normal Flow of Events:** | 1. The User enters search criteria text in the search field. 2. The User taps the magnifying glass button. 3. The System fetches message threads containing any of the keywords used as search criteria. 4. The System displays all entries containing the search criteria keywords to the user within the “ViewForum” page, in chronological order. |
| **Sub Flows:** | N/A |
| **Alternate/Exception Flows:** | N/A |
| **Frequency of Use:**  **(High/Medium/Low)** | High |
| **Status:**  **(Developed/In Development/Undeveloped)** | Undeveloped |
| **Priority:** | High |

***Use Case Description: Reply to Thread***

|  |  |
| --- | --- |
| **ID:** | **23**  **(Corresponding Mockup - Screen 11)**  **(Corresponding Use Case Diagram - 12)** |
| **Use Case Name:** | Reply to Thread |
| **Brief Description:** | The User wants to create a reply to a thread. |
| **Stakeholders and Interests:** | The User – the ability to reply to a message forum thread. |
| **Primary Actor:** | User |
| **Preconditions:** | The User is viewing a thread on the “ViewTread” page. |
| **Postconditions:** | The User has replied to a thread. |
| **Normal Flow of Events:** | 1. The User has selected the “Reply” icon on a thread on the “ViewThread” page. 2. The System redirects the User to a “ReplyToTread” page that contains a fillable form with fields for reply text and attachments. 3. The User fills in the form fields. 4. If the User selects the “Reply” Button:  * The S-1: Post Thread Reply sub-flow is performed.   If the User the “Discard” Button:   * The S-2: Discard Reply sub-flow is performed. |
| **Sub Flows:** | S-1: Post Thread Reply:   1. The system creates a reply with the information from the form and adds it to the Message Forum Subsystem database. 2. The System displays a modal alert that states: “Reply Posted”.   S-2: Discard Thread Creation:   1. The System displays a modal confirmation to the User: “Are you sure you want to discard this reply?”   If the User selects “Yes”:   1. The System dismisses the modal confirmation and clears the form. 2. The System redirects the User back to the “ViewThread” page.   If the User selects “No”:   1. This System dismisses the modal confirmation and returns control back to the form. |
| **Alternate/Exception Flows:** | N/A |
| **Status:**  **(Developed/In Development/Undeveloped)** | Undeveloped |
| **Priority:** | High |

***The Week 8 Video Prototype Demonstration Video Link:***

[***https://seneca.sharepoint.com/:v:/s/2023-09-06PRJ666ZAA-Team04/ESixbWJ32JJDvafKPp99guAB4qvn8cKp8IYAlayC\_wDPqQ?e=N4W7hP***](https://seneca.sharepoint.com/:v:/s/2023-09-06PRJ666ZAA-Team04/ESixbWJ32JJDvafKPp99guAB4qvn8cKp8IYAlayC_wDPqQ?e=N4W7hP)

***The Chronicle App Figma Project link:***

[***https://www.figma.com/team\_invite/redeem/IfzYfy4BObApGZJWPy2of4***](https://www.figma.com/team_invite/redeem/IfzYfy4BObApGZJWPy2of4)

# Domain Class Diagram

A screenshot of a computer flowchart

Description automatically generated

# Database

***Chronicle\_JSON\_Schema.json***

|  |
| --- |
| [Chronicle\_JSON\_Schema.json](https://seneca.sharepoint.com/:u:/s/2023-09-06PRJ666ZAA-Team04/EUHnNIM9CnFKtNu6wfkqSEsBL7jnk60EuSA8hbERzOGpZQ?e=E5cMTQ) |
| {  "$schema": "http://json-schema.org/draft-07/schema#",  "title": "Generated schema for Root",  "type": "array",  "items": {  "type": "object",  "properties": {  "User": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "FIRST\_NAME": {  "type": "string"  },  "LAST\_NAME": {  "type": "string"  },  "USER\_ID": {  "type": "number"  },  "USER\_NAME": {  "type": "string"  },  "USER\_PASS": {  "type": "string"  },  "DATE\_CREATED": {  "type": "string"  },  "IS\_ACTIVE": {  "type": "boolean"  },  "EMAIL\_ADDRESS": {  "type": "string"  },  "USER\_JOURNAL": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "OWNER\_USER\_ID": {  "type": "number"  },  "DATE\_CREATED": {  "type": "string"  },  "JOURNAL\_ID": {  "type": "number"  },  "JOURNAL\_ENTRY\_SET": {  "type": "array",  "items": {  "type": "object",  "properties": {  "journalEntry1": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "JOURNAL\_ENTRY\_ID": {  "type": "number"  },  "JOURNAL\_ENTRY\_OWNER\_ID": {  "type": "number"  },  "DATE\_CREATED": {  "type": "string"  },  "JOURNAL\_ENTRY\_BODY": {  "type": "string"  },  "JOURNAL\_ENTRY\_SUBJECT": {  "type": "string"  },  "ATTACHMENT\_SET": {  "type": "array",  "items": {  "type": "object",  "properties": {  "att1": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "ATTACHMENT\_ID": {  "type": "number"  },  "ATTACHMENT\_TYPE": {  "type": "string"  },  "DATE\_ADDED": {  "type": "string"  },  "ATTACHMENT\_DESC": {  "type": "string"  }  },  "required": [  "\_comment",  "ATTACHMENT\_ID",  "ATTACHMENT\_TYPE",  "DATE\_ADDED",  "ATTACHMENT\_DESC"  ]  },  "att2": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "ATTACHMENT\_ID": {  "type": "number"  },  "ATTACHMENT\_TYPE": {  "type": "string"  },  "DATE\_ADDED": {  "type": "string"  },  "ATTACHMENT\_DESC": {  "type": "string"  }  },  "required": [  "\_comment",  "ATTACHMENT\_ID",  "ATTACHMENT\_TYPE",  "DATE\_ADDED",  "ATTACHMENT\_DESC"  ]  },  "att3": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "ATTACHMENT\_ID": {  "type": "number"  },  "ATTACHMENT\_TYPE": {  "type": "string"  },  "DATE\_ADDED": {  "type": "string"  },  "ATTACHMENT\_DESC": {  "type": "string"  }  },  "required": [  "\_comment",  "ATTACHMENT\_ID",  "ATTACHMENT\_TYPE",  "DATE\_ADDED",  "ATTACHMENT\_DESC"  ]  }  },  "required": []  }  }  },  "required": [  "\_comment",  "JOURNAL\_ENTRY\_ID",  "JOURNAL\_ENTRY\_OWNER\_ID",  "DATE\_CREATED",  "JOURNAL\_ENTRY\_BODY",  "JOURNAL\_ENTRY\_SUBJECT",  "ATTACHMENT\_SET"  ]  },  "journalEntry2": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "JOURNAL\_ENTRY\_ID": {  "type": "number"  },  "JOURNAL\_ENTRY\_OWNER\_ID": {  "type": "number"  },  "DATE\_CREATED": {  "type": "string"  },  "JOURNAL\_ENTRY\_BODY": {  "type": "string"  },  "JOURNAL\_ENTRY\_SUBJECT": {  "type": "string"  },  "ATTACHMENT\_SET": {  "type": "array",  "items": {  "type": "object",  "properties": {  "att1": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "ATTACHMENT\_ID": {  "type": "number"  },  "ATTACHMENT\_TYPE": {  "type": "string"  },  "DATE\_ADDED": {  "type": "string"  },  "ATTACHMENT\_DESC": {  "type": "string"  }  },  "required": [  "\_comment",  "ATTACHMENT\_ID",  "ATTACHMENT\_TYPE",  "DATE\_ADDED",  "ATTACHMENT\_DESC"  ]  },  "att2": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "ATTACHMENT\_ID": {  "type": "number"  },  "ATTACHMENT\_TYPE": {  "type": "string"  },  "DATE\_ADDED": {  "type": "string"  },  "ATTACHMENT\_DESC": {  "type": "string"  }  },  "required": [  "\_comment",  "ATTACHMENT\_ID",  "ATTACHMENT\_TYPE",  "DATE\_ADDED",  "ATTACHMENT\_DESC"  ]  },  "att3": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "ATTACHMENT\_ID": {  "type": "number"  },  "ATTACHMENT\_TYPE": {  "type": "string"  },  "DATE\_ADDED": {  "type": "string"  },  "ATTACHMENT\_DESC": {  "type": "string"  }  },  "required": [  "\_comment",  "ATTACHMENT\_ID",  "ATTACHMENT\_TYPE",  "DATE\_ADDED",  "ATTACHMENT\_DESC"  ]  }  },  "required": []  }  }  },  "required": [  "\_comment",  "JOURNAL\_ENTRY\_ID",  "JOURNAL\_ENTRY\_OWNER\_ID",  "DATE\_CREATED",  "JOURNAL\_ENTRY\_BODY",  "JOURNAL\_ENTRY\_SUBJECT",  "ATTACHMENT\_SET"  ]  }  },  "required": []  }  }  },  "required": [  "\_comment",  "OWNER\_USER\_ID",  "DATE\_CREATED",  "JOURNAL\_ID",  "JOURNAL\_ENTRY\_SET"  ]  },  "USER\_TIMELINE": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "OWNER\_USER\_ID": {  "type": "number"  },  "DATE\_CREATED": {  "type": "string"  },  "TIMELINE\_ID": {  "type": "number"  },  "TIMELINE\_ENTRY\_SET": {  "type": "array",  "items": {  "type": "object",  "properties": {  "timeLineEntry1": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "TIMELINE\_ENTRY\_ID": {  "type": "number"  },  "DATE\_ADDED": {  "type": "string"  },  "TIMELINE\_ENTRY\_OWNER\_ID": {  "type": "number"  },  "TIMELINE\_ENTRY\_BODY": {  "type": "string"  },  "TIMELINE\_ENTRY\_SUBJECT": {  "type": "string"  },  "ATTACHMENT\_SET": {  "type": "array",  "items": {  "type": "object",  "properties": {  "att1": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "ATTACHMENT\_ID": {  "type": "number"  },  "ATTACHMENT\_TYPE": {  "type": "string"  },  "DATE\_ADDED": {  "type": "string"  },  "ATTACHMENT\_DESC": {  "type": "string"  }  },  "required": [  "\_comment",  "ATTACHMENT\_ID",  "ATTACHMENT\_TYPE",  "DATE\_ADDED",  "ATTACHMENT\_DESC"  ]  },  "att2": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "ATTACHMENT\_ID": {  "type": "number"  },  "ATTACHMENT\_TYPE": {  "type": "string"  },  "DATE\_ADDED": {  "type": "string"  },  "ATTACHMENT\_DESC": {  "type": "string"  }  },  "required": [  "\_comment",  "ATTACHMENT\_ID",  "ATTACHMENT\_TYPE",  "DATE\_ADDED",  "ATTACHMENT\_DESC"  ]  },  "att3": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "ATTACHMENT\_ID": {  "type": "number"  },  "ATTACHMENT\_TYPE": {  "type": "string"  },  "DATE\_ADDED": {  "type": "string"  },  "ATTACHMENT\_DESC": {  "type": "string"  }  },  "required": [  "\_comment",  "ATTACHMENT\_ID",  "ATTACHMENT\_TYPE",  "DATE\_ADDED",  "ATTACHMENT\_DESC"  ]  }  },  "required": []  }  },  "LIKED\_BY\_SET": {  "type": "array",  "items": {  "type": "object",  "properties": {  "userFriend1": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "FRIEND\_USER\_ID": {  "type": "number"  },  "DATE\_ADDED": {  "type": "string"  }  },  "required": [  "\_comment",  "FRIEND\_USER\_ID",  "DATE\_ADDED"  ]  },  "userFriend2": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "FRIEND\_USER\_ID": {  "type": "number"  },  "DATE\_ADDED": {  "type": "string"  }  },  "required": [  "\_comment",  "FRIEND\_USER\_ID",  "DATE\_ADDED"  ]  }  },  "required": []  }  }  },  "required": [  "\_comment",  "TIMELINE\_ENTRY\_ID",  "DATE\_ADDED",  "TIMELINE\_ENTRY\_OWNER\_ID",  "TIMELINE\_ENTRY\_BODY",  "TIMELINE\_ENTRY\_SUBJECT",  "ATTACHMENT\_SET",  "LIKED\_BY\_SET"  ]  }  },  "required": [  "timeLineEntry1"  ]  }  }  },  "required": [  "\_comment",  "OWNER\_USER\_ID",  "DATE\_CREATED",  "TIMELINE\_ID",  "TIMELINE\_ENTRY\_SET"  ]  },  "USER\_FRIEND\_SET": {  "type": "array",  "items": {  "type": "object",  "properties": {  "userFriend1": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "FRIEND\_USER\_ID": {  "type": "number"  },  "DATE\_ADDED": {  "type": "string"  }  },  "required": [  "\_comment",  "FRIEND\_USER\_ID",  "DATE\_ADDED"  ]  },  "userFriend2": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "FRIEND\_USER\_ID": {  "type": "number"  },  "DATE\_ADDED": {  "type": "string"  }  },  "required": [  "\_comment",  "FRIEND\_USER\_ID",  "DATE\_ADDED"  ]  }  },  "required": [  "userFriend1",  "userFriend2"  ]  }  }  },  "required": [  "\_comment",  "FIRST\_NAME",  "LAST\_NAME",  "USER\_ID",  "USER\_NAME",  "USER\_PASS",  "DATE\_CREATED",  "IS\_ACTIVE",  "EMAIL\_ADDRESS",  "USER\_JOURNAL",  "USER\_TIMELINE",  "USER\_FRIEND\_SET"  ]  },  "\_comment": {  "type": "string"  },  "VIDEO\_TYPE": {  "type": "string"  },  "PICTURE\_TYPE": {  "type": "string"  },  "RATING\_TYPE": {  "type": "string"  },  "FORUM\_ID": {  "type": "number"  },  "thread\_set": {  "type": "array",  "items": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "THREAD\_ID": {  "type": "number"  },  "DATE\_TIME\_CREATED": {  "type": "string"  },  "THREAD\_TITLE": {  "type": "string"  },  "THREAD\_OWNER\_ID": {  "type": "number"  },  "THREAD\_BODY": {  "type": "string"  },  "THREAD\_ATTACHMENT\_SET": {  "type": "array",  "items": {  "type": "object",  "properties": {  "attach1": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "ATTACHMENT\_ID": {  "type": "number"  },  "ATTACHMENT\_TYPE": {  "type": "string"  },  "DATE\_ADDED": {  "type": "string"  },  "ATTACHMENT\_DESC": {  "type": "string"  }  },  "required": [  "\_comment",  "ATTACHMENT\_ID",  "ATTACHMENT\_TYPE",  "DATE\_ADDED",  "ATTACHMENT\_DESC"  ]  },  "attach2": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "ATTACHMENT\_ID": {  "type": "number"  },  "ATTACHMENT\_TYPE": {  "type": "string"  },  "DATE\_ADDED": {  "type": "string"  },  "ATTACHMENT\_DESC": {  "type": "string"  }  },  "required": [  "\_comment",  "ATTACHMENT\_ID",  "ATTACHMENT\_TYPE",  "DATE\_ADDED",  "ATTACHMENT\_DESC"  ]  },  "attach3": {  "type": "object",  "properties": {  "\_comment": {  "type": "string"  },  "ATTACHMENT\_ID": {  "type": "number"  },  "ATTACHMENT\_TYPE": {  "type": "string"  },  "DATE\_ADDED": {  "type": "string"  },  "ATTACHMENT\_DESC": {  "type": "string"  }  },  "required": [  "\_comment",  "ATTACHMENT\_ID",  "ATTACHMENT\_TYPE",  "DATE\_ADDED",  "ATTACHMENT\_DESC"  ]  }  },  "required": []  }  },  "THREAD\_REPLY\_SET": {  "type": "array",  "items": {  "type": "object",  "properties": {  "threadReply1": {  "type": "object",  "properties": {  "REPLY\_ID": {  "type": "number"  },  "REPLY\_BODY": {  "type": "string"  },  "REPLY\_OWNER\_ID": {  "type": "number"  },  "REPLY\_ATTACHMENT\_SET": {  "type": "array",  "items": {  "type": "object",  "properties": {  "ATTACHMENT\_ID": {  "type": "number"  },  "ATTACHMENT\_TYPE": {  "type": "string"  },  "DATE\_ADDED": {  "type": "string"  },  "ATTACHMENT\_DESC": {  "type": "string"  }  },  "required": [  "ATTACHMENT\_ID",  "ATTACHMENT\_TYPE",  "DATE\_ADDED",  "ATTACHMENT\_DESC"  ]  }  }  },  "required": [  "REPLY\_ID",  "REPLY\_BODY",  "REPLY\_OWNER\_ID",  "REPLY\_ATTACHMENT\_SET"  ]  },  "threadReply2": {  "type": "object",  "properties": {  "REPLY\_ID": {  "type": "number"  },  "DATE\_CREATED": {  "type": "string"  },  "REPLY\_BODY": {  "type": "string"  },  "REPLY\_OWNER\_ID": {  "type": "number"  },  "REPLY\_ATTACHMENT\_SET": {  "type": "array",  "items": {  "type": "object",  "properties": {  "ATTACHMENT\_ID": {  "type": "number"  },  "ATTACHMENT\_TYPE": {  "type": "string"  },  "DATE\_ADDED": {  "type": "string"  },  "ATTACHMENT\_DESC": {  "type": "string"  }  },  "required": [  "ATTACHMENT\_ID",  "ATTACHMENT\_TYPE",  "DATE\_ADDED",  "ATTACHMENT\_DESC"  ]  }  }  },  "required": [  "REPLY\_ID",  "DATE\_CREATED",  "REPLY\_BODY",  "REPLY\_OWNER\_ID",  "REPLY\_ATTACHMENT\_SET"  ]  }  },  "required": [  "threadReply1",  "threadReply2"  ]  }  }  },  "required": [  "\_comment",  "THREAD\_ID",  "DATE\_TIME\_CREATED",  "THREAD\_TITLE",  "THREAD\_OWNER\_ID",  "THREAD\_BODY",  "THREAD\_ATTACHMENT\_SET",  "THREAD\_REPLY\_SET"  ]  }  }  },  "required": []  }  } |

***Chronicle\_DB\_raw\_data.json***

|  |
| --- |
| [Chronicle\_DB\_raw\_data.json](https://seneca.sharepoint.com/:u:/s/2023-09-06PRJ666ZAA-Team04/EV4sZFj5YiRMorZAcwH5GwAB0Zpa45Q2EkBNq0Y4daFvVw?e=XR82jg) |
| [    {  "User":  {  "\_comment": "User",  "FIRST\_NAME": "Juan",  "LAST\_NAME": "David",  "USER\_ID": 1,  "USER\_NAME": "jdrcastel",  "USER\_PASS": "password456",  "DATE\_CREATED": "2023-01-02T00:00:00Z",  "IS\_ACTIVE": true,  "EMAIL\_ADDRESS": "jdrcastel@example.com",  "USER\_JOURNAL": {  "\_comment": "UserJournal",  "OWNER\_USER\_ID": 1,  "DATE\_CREATED": "2023-01-05T12:30:00Z",  "JOURNAL\_ID": 101,  "JOURNAL\_ENTRY\_SET": [  {  "journalEntry1":  {  "\_comment": "UserJournalEntry",  "JOURNAL\_ENTRY\_ID": 123456,  "JOURNAL\_ENTRY\_OWNER\_ID": 999999,  "DATE\_CREATED": "17-6-2022",  "JOURNAL\_ENTRY\_BODY": "Today was a beautiful day in Paris. I saw the Eiffel Tower!",  "JOURNAL\_ENTRY\_SUBJECT": "My First Journal Entry in Paris",  "ATTACHMENT\_SET": [  {  "att1":  {  "\_comment": "Attachment",  "ATTACHMENT\_ID": 123,  "ATTACHMENT\_TYPE": "Picture",  "DATE\_ADDED": "17-6-2022",  "ATTACHMENT\_DESC": "Picture\_Data"  }  },  {  "att2":  {  "\_comment": "Attachment",  "ATTACHMENT\_ID": 456,  "ATTACHMENT\_TYPE": "Video",  "DATE\_ADDED": "17-6-2022",  "ATTACHMENT\_DESC": "Video\_Data"  }  },  {  "att3":  { "\_comment": "Attachment",  "ATTACHMENT\_ID": 789,  "ATTACHMENT\_TYPE": "Rating",  "DATE\_ADDED": "17-6-2022",  "ATTACHMENT\_DESC": "Location=Eiffel\_Tower\*\*\*\*\*"  }  }  ]  }  },  {  "journalEntry2":  {  "\_comment": "UserJournalEntry",  "JOURNAL\_ENTRY\_ID": 7891011,  "JOURNAL\_ENTRY\_OWNER\_ID": 999999,  "DATE\_CREATED": "17-6-2022",  "JOURNAL\_ENTRY\_BODY": "The Louvre is so beautiful",  "JOURNAL\_ENTRY\_SUBJECT": "My second Journal Entry in Paris",  "ATTACHMENT\_SET": [  {  "att1":  {  "\_comment": "Attachment",  "ATTACHMENT\_ID": 123,  "ATTACHMENT\_TYPE": "Picture",  "DATE\_ADDED": "17-6-2022",  "ATTACHMENT\_DESC": "Picture\_Data"  }  },  {  "att2":  {  "\_comment": "Attachment",  "ATTACHMENT\_ID": 456,  "ATTACHMENT\_TYPE": "Video",  "DATE\_ADDED": "17-6-2022",  "ATTACHMENT\_DESC": "Video\_Data"  }  },  {  "att3":  { "\_comment": "Attachment",  "ATTACHMENT\_ID": 789,  "ATTACHMENT\_TYPE": "Rating",  "DATE\_ADDED": "17-6-2022",  "ATTACHMENT\_DESC": "Location=Eiffel\_Tower\*\*\*\*\*"  }  }  ]  }  }  ]  },  "USER\_TIMELINE":  {  "\_comment": "UserTimeline",  "OWNER\_USER\_ID": 1,  "DATE\_CREATED": "2023-01-05T12:30:00Z",  "TIMELINE\_ID": 1,  "TIMELINE\_ENTRY\_SET":[  {  "timeLineEntry1":  {  "\_comment": "UserTimelineEntry",  "TIMELINE\_ENTRY\_ID": 1,  "DATE\_ADDED": "2023-01-05T12:30:00Z",  "TIMELINE\_ENTRY\_OWNER\_ID": 1,  "TIMELINE\_ENTRY\_BODY": "This is a journal entry.",  "TIMELINE\_ENTRY\_SUBJECT": "I love paris",  "ATTACHMENT\_SET":[  {  "att1":  {  "\_comment": "Attachment",  "ATTACHMENT\_ID": 123,  "ATTACHMENT\_TYPE": "Picture",  "DATE\_ADDED": "17-6-2022",  "ATTACHMENT\_DESC": "Picture\_Data"  }  },  {  "att2":  {  "\_comment": "Attachment",  "ATTACHMENT\_ID": 456,  "ATTACHMENT\_TYPE": "Video",  "DATE\_ADDED": "17-6-2022",  "ATTACHMENT\_DESC": "Video\_Data"  }  },  {  "att3":  { "\_comment": "Attachment",  "ATTACHMENT\_ID": 789,  "ATTACHMENT\_TYPE": "Rating",  "DATE\_ADDED": "17-6-2022",  "ATTACHMENT\_DESC": "Location=Eiffel\_Tower\*\*\*\*\*"  }  }  ],  "LIKED\_BY\_SET": [  {  "userFriend1":  {  "\_comment": "UserFriend",  "FRIEND\_USER\_ID": 1,  "DATE\_ADDED": "2023-01-05T12:30:00Z"  }  },  {  "userFriend2":  {  "\_comment": "UserFriend",  "FRIEND\_USER\_ID": 2,  "DATE\_ADDED": "2023-01-05T12:30:00Z"  }  }  ]  }  }  ]  },  "USER\_FRIEND\_SET": [  {  "userFriend1":  {  "\_comment": "UserFriend",  "FRIEND\_USER\_ID": 2,  "DATE\_ADDED": "2023-01-05T12:30:00Z"  },  "userFriend2":  {  "\_comment": "UserFriend",  "FRIEND\_USER\_ID": 2,  "DATE\_ADDED": "2023-01-05T12:30:00Z"  }  }  ]  }  },  {  "\_comment": "<<enumeration>>AttachmentType",  "VIDEO\_TYPE": "Video",  "PICTURE\_TYPE": "Picture",  "RATING\_TYPE": "Rating"  },  {  "\_comment": "Forum",  "FORUM\_ID": 1,  "thread\_set": [  {  "\_comment": "ForumThread",  "THREAD\_ID": 123,  "DATE\_TIME\_CREATED": "2023-01-05T12:30:00Z",  "THREAD\_TITLE": "Title1",  "THREAD\_OWNER\_ID": 456,  "THREAD\_BODY": "Text text text text",  "THREAD\_ATTACHMENT\_SET": [  {  "attach1":  {  "\_comment": "Attachment",  "ATTACHMENT\_ID": 123,  "ATTACHMENT\_TYPE": "Picture",  "DATE\_ADDED": "2023-01-05T12:30:00Z",  "ATTACHMENT\_DESC": "Picture\_Data"  }  },  {  "attach2":  {  "\_comment": "Attachment",  "ATTACHMENT\_ID": 456,  "ATTACHMENT\_TYPE": "Video",  "DATE\_ADDED": "2023-01-05T12:30:00Z",  "ATTACHMENT\_DESC": "Video\_Data"  }  },  {  "attach3":  {  "\_comment": "Attachment",  "ATTACHMENT\_ID": 789,  "ATTACHMENT\_TYPE": "Rating",  "DATE\_ADDED": "2023-01-05T12:30:00Z",  "ATTACHMENT\_DESC": "Location=Eiffel\_Tower\_\*\*\*\*\*"  }  }  ],  "THREAD\_REPLY\_SET": [  {  "threadReply1":  {  "REPLY\_ID": 123,  "REPLY\_BODY": "Reply text",  "REPLY\_OWNER\_ID": 456,  "REPLY\_ATTACHMENT\_SET": [  {  "ATTACHMENT\_ID": 123,  "ATTACHMENT\_TYPE": "Picture",  "DATE\_ADDED": "2023-01-05T12:30:00Z",  "ATTACHMENT\_DESC": "Picture\_Data"  }  ]  },  "threadReply2":  {  "REPLY\_ID": 456,  "DATE\_CREATED": "27-05-2021",  "REPLY\_BODY": "Reply text",  "REPLY\_OWNER\_ID": 456,  "REPLY\_ATTACHMENT\_SET": [  {  "ATTACHMENT\_ID": 123,  "ATTACHMENT\_TYPE": "Picture",  "DATE\_ADDED": "17-6-2022",  "ATTACHMENT\_DESC": "Picture\_Data"  }  ]  }  }    ]  }  ]  }  ] |

# Work Breakdown Structure (WBS)

|  |
| --- |
| **Chronicle Mobile Web Application – Work Breakdown Structure:**   1. User Management System:    1. Front End Development:       1. Develop frontend UI for new user email verification.       2. Create User Account:          1. Design and create UI components for “Create Account” page and popup form.          2. Implement form validation and submission functionality.          3. Integrate frontend with backend API for account creation.          4. Implement redirection to the homepage after account creation.       3. User Sign-In:          1. Develop UI components for the “Sign-In” page and popup form.          2. Implement for validation and submission functionality.          3. Integrate frontend and backend API for user authentication.          4. Implement redirection to the homepage after successful sign-in.       4. Edit Profile Info:          1. Design UI components for the “Edit Profile” page and popup form.          2. Implement form validation and submission functionality for profile editing.          3. Connect frontend to backend API for updating user profile data.          4. Implement redirection to the profile page after editing.       5. Delete Profile:          1. Create UI elements for the “Delete Account” page and confirmation dialog.          2. Implement confirmation logic and handling for account deletion.          3. Connect frontend to backend API for deleting user accounts.          4. Handle redirection to the “Edit Profile” page or other relevant pages after deletion confirmation.       6. View Other User Profile:          1. Design UI elements for Displaying other user profiles.          2. Implement UI components for viewing other user profiles.          3. Integrate frontend with backend API to retrieve and display other user profiles.          4. Implement redirection to the “View User Profile Page” when viewing other user profiles.    2. Backend Development:       1. User Authentication and Verification:          1. Develop backend logic for verifying email for new user accounts.          2. Implement authentication mechanisms for user account creation and sign-in.       2. User Account Management:          1. Create backend endpoints to handle account creation, editing, and deletion.          2. Implement data validation and storage for user account information.       3. Profile Information Retrieval          1. Develop backend endpoints for retrieving other user profile information.          2. Implement logic to fetch and serve user profile data securely.    3. Database Schema Design:       1. Design database schema for storing user account information.       2. Define database tables for user profiles, authentication, and related data.    4. Data Storage Implementation:       1. Setup database structures for storing user accounts, profile information, and authentication data.       2. Implement CRUD (Create, Read, Update, Delete) operations for user-related data. 2. Journal Management System:    1. Front End Development:       1. Develop frontend UI for the ViewJounal Screen.       2. View Journal:          1. Design and create UI components for “View Journal” page, including the floating “+” icon to create and add a new Journal and Edit and Delete Journal options.          2. Integrate frontend with backend API to get data and showcase it on Journal View Page.          3. Implement redirection to Journal View Page after viewing one Journal.          4. Implement navigation to Add Journal Entry Form Page when clicking on “+” icon.       3. Add New Journal:          1. Develop UI components for the “Add-Journal” page which is a Add Journal Entry form page.          2. Implement text input and attachments in the form, implement form validation, and Discard and Add Entry functionality.          3. Integrate frontend and backend API for posting newly added journal into the database.          4. Implement redirection to the Journal View Page after successfully adding a Journal to showcase newly added Journal.       4. Edit Journal Entry:          1. Develop UI components for the “Edit-Journal” option which is a Edit Journal Entry form page.          2. Implement text input and attachments in the form, implement form validation, and Discard and Edit Entry functionality.          3. Integrate frontend and backend API for updating journal into the database.          4. Implement redirection to the Journal View Page after successfully editing a Journal to showcase updated Journal.       5. Delete Journal Entry:          1. Create UI elements for the “Delete Entry” option and confirmation dialog.          2. Implement confirmation logic and handling for Journal deletion.          3. Connect frontend to backend API for deleting user journals.          4. Handle redirection to the “Journal View” page to showcase the entry being deleted, if selected yes, and when selected no, should get rid of the confirmation pop-up, and redirect back to Journal View Page.    2. Backend Development:       1. Journal Management:          1. Create backend endpoints to handle Journal creation, editing, and deletion.          2. Implement Journal Entry data validation and storage for user’s Journal Entry information such as entry text, date, and attachments.       2. Journal Information Retrieval          1. Develop backend endpoints for retrieving user’s Journal Information such as date, title, attachments.          2. Implement logic to fetch and serve user journal data securely.    3. Database Schema Design:       * 1. Design database schema for storing user’s Journal Entry information.         2. Define database tables for user Journal Entry’s title, date, entry text and attachments and related data, if there are any such as User Id, etc.    4. Data Storage Implementation:       * 1. Setup database structures for storing user’s journal information, and authentication data.         2. Implement CRUD (Create, Read, Update, Delete) operations for user’s Journal Entry data. 3. User Timeline System:    1. Back End Development:       1. User Authentication and Registration:          1. Develop backend endpoints for creating and deleting travel journal entries in the Timeline.       2. Database Schema Design:          1. Design database schema for storing user’s Timeline information.          2. Define database tables for user Timeline and the access to the Journal Table       3. Data Storage Implementation:          1. Implement Create and delete operations for Journal Entry into the Timeline data.          2. Optimize database queries and operations for performance.          3. Design a scalable database architecture to handle user growth.          4. Conduct performance testing to identify and address bottlenecks.    2. Frontend Process for UserTimeline Feature:       1. Develop frontend UI for the UserTimeline Screen.       2. View Timeline:          1. Design and create UI components for "View Timeline" page, including the floating "+" icon to add a new Journal to Timeline          2. Integrate frontend with backend API to get data and post new journal to timeline.          3. Implement navigation to Journal Entries Page when clicking on "+" icon.          4. Implement redirection to Timeline with the new Journal entry in the timeline       3. Add New Journal Entry to Timeline:          1. Develop UI components for the “Add-JournalToTimeline” page.          2. Implement redirection to Journal Entry to Timeline View Page.          3. Using API POST add the selected journal entry and then redirect again to Timeline View Page       4. Delete Journal Entry:          1. Create UI elements for the "Delete Journal in Timeline" option and confirmation dialog.          2. Implement confirmation logic and handling for Journal Entry deletion.          3. Connect frontend to backend API for deleting Journal entry in Timeline.          4. Handle redirection to the "Timeline View" page to showcase the entry being deleted, if selected yes, and when selected no, should get rid of the confirmation pop-up, and redirect back to Timeline View Page. 4. Forum Feature Development    1. Front End Development       1. Forum Main Page UI          1. Develop UI for the main page of the forum.       2. Thread Creation UI          1. Design and implement UI for creating new message threads.       3. Thread Response UI          1. Develop UI components for responding to threads.       4. Forum Navigation          1. Implement navigation within the forum (e.g., search).    2. Back End Development       1. Thread Management Logic          1. Develop backend logic for creating and managing threads.       2. Response Management Logic          1. Implement logic for posting and managing responses.    3. User Participation Features       1. User Profile Integration          1. Display User Profiles in Forum          2. Implement feature to display user profiles in forum interactions.       2. Interaction Features          1. Develop functionality for users to like and react to forum posts.    4. Database and Infrastructures       1. Database Schema for Forum Data          1. Design Database for Threads and Responses          2. Create database schema for storing threads and responses.    5. Data Storage and Retrieval       1. Setup CRUD Operations for Forum Data          1. Implement Create, Read, Update, Delete operations for forum data.       2. Ensure Data Security and Privacy          1. Implement measures for data security and user privacy. |
|  |

# Milestones and Acceptance Criteria

* 1. **Milestone One: User Management Functionality**

**Definition:**

The User Management System milestone encompasses the complete development of user-related functionalities within the Chronicle Mobile Web Application. It includes frontend UI/UX development, backend logic for user authentication and account management, database schema design, and implementation of data storage structures.

**Acceptance Criteria:**

* **Frontend Development:**
  + New User Email Verification: Successfully develop and integrate a frontend UI for verifying new user email addresses.
  + Create User Account: Design and implement UI components and functionalities for account creation, validation, and integration with backend API. Ensure redirection to the homepage after account creation.
  + User Sign-In: Develop user-friendly UI components and functionalities for user sign-in, including proper integration with backend API and redirection upon successful sign-in.
  + Edit Profile Info: Create and integrate UI elements for profile editing, validate inputs, connect with backend API for updating user profile data, and ensure redirection to the profile page after editing.
  + Delete Profile: Develop UI elements and logic for account deletion, implement confirmation handling, connect with backend API for deletion, and manage appropriate redirection post-confirmation.
  + View Other User Profile: Design UI elements and integrate components for viewing other user profiles, ensuring proper data retrieval from the backend and seamless redirection to the view profile page.
* **Backend Development:**
  + User Authentication and Verification: Successfully implement backend logic for new user email verification, account creation, and sign-in authentication mechanisms.
  + User Account Management: Create and test backend endpoints for account creation, editing, and deletion, ensuring proper data validation and secure storage.
  + Profile Information Retrieval: Develop functional backend endpoints for retrieving other user profile information securely.
* **Database Schema Design and Data Storage:**
  + Database Schema Design: Define and finalize database schema for storing user account information, profiles, and authentication data.
  + Data Storage Implementation: Setup database structures, including tables for user accounts and profiles, and implement CRUD operations for user-related data.
  1. **Milestone Two: Developing Journal View Feature**

**Definition**: This milestone focuses on implementing the core functionality of the Journal View screen, including the ability to view, add, edit, and delete journal entries. Both frontend and backend components will be developed to ensure seamless interaction.

**Acceptance Criteria**:

* **View Journal Screen**:
  + UI elements for displaying existing journals.
  + Integration with Backend API to fetch and display Journal.
  + From floating “+” icon, navigation to the “Add Journal Entry” form.
  + Redirection to the View Journal Page after viewing one Journal.
* **Add New Journal:**
  + UI elements for the "Add-Journal" page - Add Journal Entry form.
  + Form Validation for text input and attachments
  + Integration with backend API for posting newly added journal.
  + Redirection to the Journal View Page after adding a Journal, showcasing newly added journal.
* **Edit Journal Entry:**
  + UI elements for the "Edit-Journal" option - Edit Journal Entry form.
  + Form validation for text input and attachments.
  + Integration with backend API for updating journal entries.
  + Redirection to the Journal View Page after editing a Journal, showcasing the updated data.
* **Delete Journal Entry:**
  + UI elements for the "Delete Entry" option and confirmation dialog.
  + Delete confirmation pop-up and handle the yes and no options.
  + Connection with backend API for deleting user journals.
  + Redirection to the Journal View Page after deleting an entry.
* **Backend Journal Management:**
  + Create backend endpoints to handle journal creation, editing, and deletion and Journal Data retrieval.
* **Database Implementation:**
  + Design database structures for storing user’s journal information and authentication data.
  + Implement Create, Read, Update, Delete operations for user’s Journal Entry data.
  1. **Milestone Three: Developing TimeLine Feature**

**Definitions:** This milestone focuses on implementing the View Timeline screen functionality, with View Timeline screen features and add journal to timeline entries, and also adding features like deleting journal and like.

**Acceptance Criteria**:

* **View Timeline Screen**:
  + UI elements for displaying existing timeline Journal entries.
  + Integration with Backend API to fetch and display Timeline.
  + From floating “+” icon, navigation to the “Add Journal to Timeline” form.
  + Redirection to the View Timeline Page after viewing one Journal.
* **Add New Journal to Timeline:**
  + UI elements for the "Add-JournalToTimeline" page – Add Entry to Timeline form.
  + Integration with backend API for posting new journals to timeline.
  + Redirection to the Timeline View Page after adding a Journal To timeline.
* **Delete Journal Entry:**
  + UI elements for the "Delete Entry" option and confirmation dialog.
  + Delete confirmation pop-up and handle the yes and no options.
  + Connection with backend API for deleting user journals in the timeline.
  + Redirection to the Timeline View Page after deleting an entry.
* **Backend Journal Management:**
  + Create backend endpoints to handle journal creation, and deletion and Journal Entries.
* **Database Implementation:**
  + Design database structures for storing Journal entries into the timeline
  + Implement Create, Read, Delete operations for user’s Journal Entry into the timeline data.
  1. **Milestone Four: Developing Forum Feature**

**Definitions:** This milestone focuses on implementing Forum functionality, with Forum screen features and creating threads, managing threads, and responding to threads.

**Acceptance Criteria**:

* **Forum Screen**:
  + UI elements for displaying existing forum threads entries.
  + Integration with Backend API to fetch and display Forum.
  + From floating “Create Thread” button, navigation to the “Create New Thread” form.
  + Redirection to the Thread Screen after clicking on thread.
* **Thread Screen**:
  + UI elements for displaying thread details.
  + Integration with Backend API to fetch and display thread data.
  + From "Reply” button, navigation to the “Reply to Thread” form.
* **Create Thread**:
  + UI elements for the "Create Thread” page.
  + Integration with backend API for posting new threads to forum.
  + Redirection to the View Thread Page after adding a Thread to Forum.
* **Delete Journal Entry:**
  + UI elements for the "Delete Thread" option and confirmation dialog.
  + Delete confirmation pop-up and handle the yes and no options.
  + Connection with backend API for deleting threads.
  + Redirection to the Forum Page after deleting an entry.
* **Backend Journal Management:**
  + Create backend endpoints to handle thread creation, deletion and managing.
* **Database Implementation:**
  + Design database structures for storing Threads and Replies.
  + Implement Create, Read, Delete operations for user’s Journal Entry into the timeline data.

# Implementation Schedule

**Product Backlog (Agile-Scrum)** - **User Stories** **Backlog**

**GitHub Link:** <https://github.com/Vaibhav-G-Parmar/PRJ566-Team_4/issues>

**Sprint 1:**

|  |
| --- |
| **User Story: As a user, I want to create a new account for myself so I can login and access the application's features.**  **Description:** The user needs the ability to register for a new account within the application. This registration process should collect necessary information, such as username, email address, and password, to create a unique account for the user. Upon successful account creation, the user should be able to access the full range of features offered by the application. |
| **User Story:  As a user, I want to login to the application and access the application's features.**  **Description:** As a user of the Chronicle application, I aim to securely log in and access the features provided within the app. This login functionality will serve as a gateway to the platform, ensuring authenticated access to the travel journal, timeline, engagement in community discussions, ratings of destinations, and facilitation of meetups with fellow travelers. |
| **User Story: As a user, I want to edit my profile so I can update my personal information.**  **Description:** As a user of the Chronicle application, I aim to have the capability to edit and update my profile information as needed. This functionality will allow me to modify personal details, including but not limited to my name, contact information (email), profile picture, and any other relevant details to ensure my profile is accurate and up to date. |
| **User Story: As a user, I want to delete my profile so I can be removed from the application.**  **Description:** As a user of the Chronicle application, I require the ability to permanently delete my profile from the platform. This feature will facilitate the complete removal of my account and associated data, ensuring my complete disassociation from the application. |
| **User Story: As a user, I want to view another user’s profile so I can public information about that user.**  **Description:** As a user of the Chronicle application, I desire the capability to access public information about other users by viewing their profiles. This functionality will allow me to explore and access information such as the user's public bio, profile picture, username, and any other information shared publicly within their profile. |

**Sprint 2:**

|  |
| --- |
| **User Story:**  **As a user, I want to create a new journal for myself so I can document my travel memories into a journal and attach images or videos.**  **Description:**  The user needs the ability to add a journal entry after they have successfully registered to the application. The add journal entry form should collect all the necessary information such as Title of the Journal Entry, Date of their memory, the Journal Entry Text to write their experience and the attachment facility so that they can attach images or videos of their travel memory. Upon successful creation of the Journal Entry, it should store the data in the backend server using APIs. Once entry is added, user should be able to see their entry in the Journal View tab. User should also be able to cancel from their Add Journal Entry form to go back to the Journal View Tab. Hitting cancel should ask for cancel confirmation pop-up, hitting no should persist on the same screen while hitting yes should discard the entry and show the Journal View Tab. |
| **User Story:**   **As a user, I want to edit my Journal Entry so I can update the journal entry content.**  **Description:**  As a user of the Chronicle application, they aim to have the capability to edit and update their already added Journal Entry information as needed. This functionality will allow the user to modify Journal Title, Journal date, Journal Enty text and allow to remove existing attachments or add new attachments if they want to. This will allow users to keep their memories up-to-date and keep adding more things to it as they would recall from their experience. |
| **User Story:**  **As a user, I want to view a Journal Entry so that I can reexperience my travel memories.**  **Description:**  The user needs to view Journal Entries that they have added. This would be possible when the user clicks on the Journal View Tab from the Navigation board. When viewing first time, if there are no Journal Entries are added, a relevant message is shown on the screen. Otherwise, the Screen should show a table view of all the Journal Entries that User have added show far, with Title, Date, and some initial text of the Journal Entry. Also clicking on a Journal View tab will allow user to view the detailed Journal Entry, with full Journal Entry details such as Journal Tile, date, the full entry text and all the attachments. |
| **User Story:**   **As a user, I want to delete my Journal Entry so it can be removed from the application.**  **Description:**  As a user of the Chronicle application, they aim to have the capability to delete their already added Journal Entry as needed. This functionality will allow the user to delete complete Journal Entry including Journal Title, Journal date, Journal Enty text and attachments if they want to. This will allow users to delete their memories of which they want to. |

**Sprint 3:**

|  |
| --- |
| **User Stories:**  As a user, I want to post my journal to the timeline so I can add and make my journal entry public.  **Description:** The user needs the ability to add the journal entry to the timeline to be able to make it public. For this, information from the journal database such as the title, description, and attached documents must be collected and then made a copy in the timeline table. This way all users will be able to see the journal that the user created. |
| **User Stories:**  As a user, I want to remove my journal from the timeline screen so I can remove my journal entry public.  **Description:**  As a Chronicle app user, I need to be able to delete my journal on the timeline screen. This feature will make it easier for the public to delete my journal so that no other user can see it. |
| **User Stories:**  As a user, I want to see other users' public journal entries in the timeline screen so I can react to them.  **Description:**  As a user of the Chronicle application, I want to be able to see the journals of all the users who made it public. This functionality will allow me to access information from other people's journals and be able to react to them. |

**Sprint 4:**

|  |
| --- |
| **User Stories:**  As a user of Chronicle, I want to create a new thread.  **Description:**  This involves presenting the user with a form to enter the title and content of the new thread and then submitting this data. |
| **User Stories:**  As a user of Chronicle, I want to reply to an existing thread.  **Description:**  This includes displaying a 'Create Reply' form within the context of a thread and submitting the user's input. |
| **User Stories:**  As a user of Chronicle, I want to view the list of threads.  **Description:**  This involves navigating to the Forum tab and displaying a list of threads with titles, authors, and the number of replies. |
| **User Stories:**  As a user of Chronicle, I want to view the details of a thread.    **Description:**  This would involve selecting a thread from the list and displaying its content along with all the replies. |
| **User Stories:**  As a moderator of Chronicle, I want to delete a thread.  **Description:**  This would provide moderators with the option to remove a thread and its associated replies from the forum. |
| **User Stories:**  As a moderator of Chronicle, I want to delete individual reply.  **Description:**  This would allow moderators to remove specific replies from a thread without deleting the entire thread. |
| **User Stories:**  As a user of Chronicle, I want to search for threads by keyword.  **Description:**  This would include a search bar in the forum tab that filters the thread list based on the user's query. |

# Client / Faculty Sign-off

**Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

X .

Name of Client/Rep/Professor