



PES UNIVERSITY, Bengaluru

Department of Computer Science and Engineering

B. Tech (CSE) – 5th Semester – Aug-Dec 2024

---

UE22CS341A – Software Engineering

## **PROJECT REPORT**

**on**

## **Time Capsule Database System**

Submitted by: Team # 314\_343

PES1UG22CS314	M S Chandana
PES1UG22CS343	Meeraja K

Class of Prof. Raghu B. A.

5th Sem. F Sec.

<i>Table of Contents</i>		
<b>Sl. No.</b>	<b>Topic</b>	<b>Page No.</b>
1.	Project Statement / Synopsis	3
2.	Software Requirements Specification [SRS] with RTM (Initial ver)	3
3.	Project Plan with Gantt Chart (Baseline)	18
4.	Architecture & Design Choices and Diagrams	23
5.	Development - Code Files [Git link]	27
6.	Test Plans	28
7.	Test Cases and Test Results Matrix – including Screenshots of Inputs and Resulting Outputs after Execution of Test Cases	32
8.	Final Gantt Chart (Baseline and Final Timelines)	61
9.	Conclusions	61
10.	Appendix A: Glossary of Abbreviations and Acronyms	62
11.	Appendix B: RTM (Final version)	63
12.	Appendix C: Technology stack and References [Books, Links to web pages/portals, tools use]	66

## 1. Project Statement / Synopsis:

**Project statement:** The Time Capsule Database System (TCDS) solves the problem of preserving and securely storing digital memories, messages, and documents to be opened at a future date.

The **Time Capsule Database System** is a robust platform that enables users to create, manage, and share virtual time capsules containing text, images, or other digital content, set to be delivered on a specified future date and time. Users can register, log in, and manage their accounts, while ensuring data integrity with unique constraints and secure sharing options. The system supports capsule statuses like *scheduled*, *delivered*, or *failed*, tracks user actions through audit logs, and enforces size restrictions for content. Designed with a relational schema in **3rd Normal Form (3NF)**, the database ensures efficiency, scalability, and integrity through well-defined relationships and constraints, providing a reliable foundation for a seamless time capsule experience.

## 2. Software Requirements Specification [SRS] with RTM (Initial ver)

### 1. Introduction

#### 1.1 Purpose

The purpose of the Time Capsule Database System (TCDS) is to provide a secure and reliable platform for users to store personal or significant digital content—such as letters, photos and documents—intended for future access. The system is designed to address the need for long-term, secure storage, ensuring data integrity and confidentiality over extended periods. Users can create encrypted digital "time capsules" with a specified release date, allowing them to share content with themselves or others at a future time.

This SRS covers version 1.0 of the TCDS, including encryption, storage, and release functionalities. It outlines the full system scope, from the front-end user interface to the back-end database management, ensuring comprehensive security measures and the safe delivery of time capsules when the specified date is reached.

### *1.2 Intended Audience*

This Software Requirements Specification (SRS) for the Time Capsule Database System (TCDS) is designed for the following audiences:

- **Developers (Team Members):** The two-member development team will use this document to guide the design, implementation, and development of the TCDS, ensuring all functional and technical requirements are met.
- **Project Supervisor or Instructor:** To review the project's objectives, progress, and compliance with academic standards.
- **Testers:** Any individuals involved in testing (including peer reviewers) will use the SRS to create test cases and ensure that the system operates as intended.
- **End Users:** Students, faculty, families, administrators who may eventually use the system to store digital time capsules for personal or academic purposes.

Read the Introduction for an overview of the system. Focus on Functional and Non-Functional Requirements for detailed specifications. Design Constraints are crucial for understanding limitations.

### *1.3 Product Scope*

The Time Capsule Database Management System (TCDS) is a software platform that allows users to create, upload, encrypt, and store digital time capsules containing various types of content (e.g., letters, documents, and photos). The purpose of the system is to securely store these digital artifacts, which can be accessed or released at a future date determined by the user. This software ensures data integrity over extended periods through encryption and access control mechanisms.

The system benefits include:

- **Secure Storage:** Providing a robust mechanism for storing encrypted digital content.
- **Scheduled Release:** Enabling the release of time capsules at specific, user-defined future dates.
- **User Flexibility:** Allowing multiple user categories, such as historians, archivists, businesses, and educators, to manage time capsules.

- **Data Integrity:** Maintaining the quality and security of stored information for long durations.

The objectives of the TCDS align with corporate strategies of long-term data preservation, ensuring digital memories, important documents, and historical records are safeguarded for future generations. The system aims to serve a wide range of users, from individuals creating personal capsules to businesses archiving critical information.

### *1.4 References*

- J. Doe, "Time Capsule: A Digital Preservation and Archiving Solution," *Journal of Digital Archiving*, vol. 14, 2022.
  - Smith, "The Future of Time Capsules in Digital Preservation," *Journal of Information Storage*, 2021.
  - M. Williams, "Digital Time Capsules: Preserving Personal Memories for Future Generations," *International Journal of Digital Heritage Preservation*, 2020.
  - Surnic, Natascha, and Andreas Rauber. "Digital Preservation Time Capsule: A Showcase for Digital Preservation." Vienna University of Technology, Vienna, Austria. Accessed September 17, 2024
  - Time Capsule | Mapping the future in 3650 days. (n.d.). <https://timecapsule.co/>
- 

## 2. Overall Description

### *2.1 Product Perspective*

TCDS is a new, standalone project designed for secure digital time capsule storage. It does not build on any pre-existing systems but aims to provide a unique long-term digital storage solution. TCDS consists of a frontend built with Next.js and a backend using SQL as a relational database management system (DBMS). The system manages content encryption, user management, and scheduled data releases.

## 2.2 Product Functions

- **User Registration & Authentication:** Allow users to create accounts and securely log in.
- **Time Capsule Creation:** Users can upload content, encrypt it, and set a release date.
- **Encryption & Storage:** Securely encrypt the content and store it in the SQL-based database.
- **Scheduled Release:** Automatically release and decrypt content when the specified date is reached.
- **User Notifications:** Notify users or recipients about the release of the time capsule.

## 2.3 User Classes and Characteristics

- **General Users:** Individuals or families using the system to store personal content. Examples include a family preserving photos and letters, an individual storing personal reflections, or event organizers creating time capsules for special occasions. They may not have a technical background, so the system will be designed to be user-friendly.
- **Business Users:** Organizations utilizing the system to store critical documents. Examples include a company archiving important business records, an educational institution preserving academic achievements, or historians and archivists storing digital records and historical data. They may require enhanced security and data integrity features.
- **Administrators:** Users with technical expertise responsible for managing the system, ensuring database integrity, and handling encryption keys. Examples include IT staff managing the system's infrastructure or database administrators overseeing data security and system maintenance.

## 2.4 Operating Environment

The TCDS will operate in the following environment:

- **Frontend:** Built with Next.js, accessible via web browsers on desktop.
- **Backend:** Powered by SQL as the relational DBMS, hosted on cloud-based or on-premise servers.
- **Operating Systems:** Compatible with modern OS such as Windows, macOS, and Linux for server hosting.
- **Dependencies:** Requires web server software (e.g., Apache or NGINX) and database management systems compatible with SQL.

### *2.5 Design and Implementation Constraints*

- *Security and Encryption:* The system must implement robust encryption for storing and retrieving time capsule content. This requires careful design to ensure that encryption processes do not hinder system performance and that encryption keys are managed securely.
- *Database and Storage:* Using SQL for backend data management imposes constraints on handling large or unstructured data. The design must accommodate efficient querying, file size limitations, and storage optimization to manage increasing volumes of time capsules and ensure long-term data integrity.
- *Performance:* The design must ensure minimal latency during capsule creation and retrieval processes. Efficient handling of encryption, storage, and user notifications is critical to maintain system responsiveness and performance as the user base grows.
- *Technology and Compatibility:* The use of Next.js for the frontend and SQL for the backend introduces constraints related to compatibility and integration. The system design must ensure that third-party libraries for encryption and authentication are compatible and that the system scales effectively with increasing user numbers and data volume.
- *Design Constraints:* The user interface must be intuitive and accessible, supporting both technical and non-technical users. It should provide a consistent experience across devices, with clear labeling and effective error messaging. The design must integrate encryption seamlessly and offer real-time feedback for actions like capsule creation and retrieval.

### *2.6 Assumptions and Dependencies*

- **Assumptions:**

Users have access to modern web browsers that support the required functionalities of Next.js.

Encryption libraries and tools used for securing data are up-to-date and compatible with the chosen technologies.

Users will have a stable internet connection for accessing and managing their time capsules.

- Dependencies:

The project depends on third-party libraries for encryption and possibly for handling authentication.

The system relies on cloud infrastructure or server environments for hosting and managing the database.

Integration with external services or APIs for additional functionalities (e.g., email notifications for release reminders).

---

### 3. External Interface Requirements

#### *3.1 User Interfaces*

- Logical Characteristics: The user interface will be intuitive and user-friendly, designed to be accessible to users with varying levels of technical expertise. Key elements include easy navigation, clear instructions for uploading content, and straightforward controls for setting release dates.
- Screen Layout: Layouts will follow standard web design practices, with consistent use of buttons, error messages, and help functions.

#### *3.2 Software Interfaces*

- Databases: The system will interface with a SQL relational database for storing user data, time capsules, and encryption metadata.
- Libraries and Tools: Integration with encryption libraries (e.g., for AES encryption) and possibly third-party authentication tools.
- Data Exchange: Data items such as user-uploaded content and encryption keys will be exchanged between the front-end and back-end. The purpose of these exchanges includes secure content upload, storage, and retrieval.

#### *3.3 Communications Interfaces*

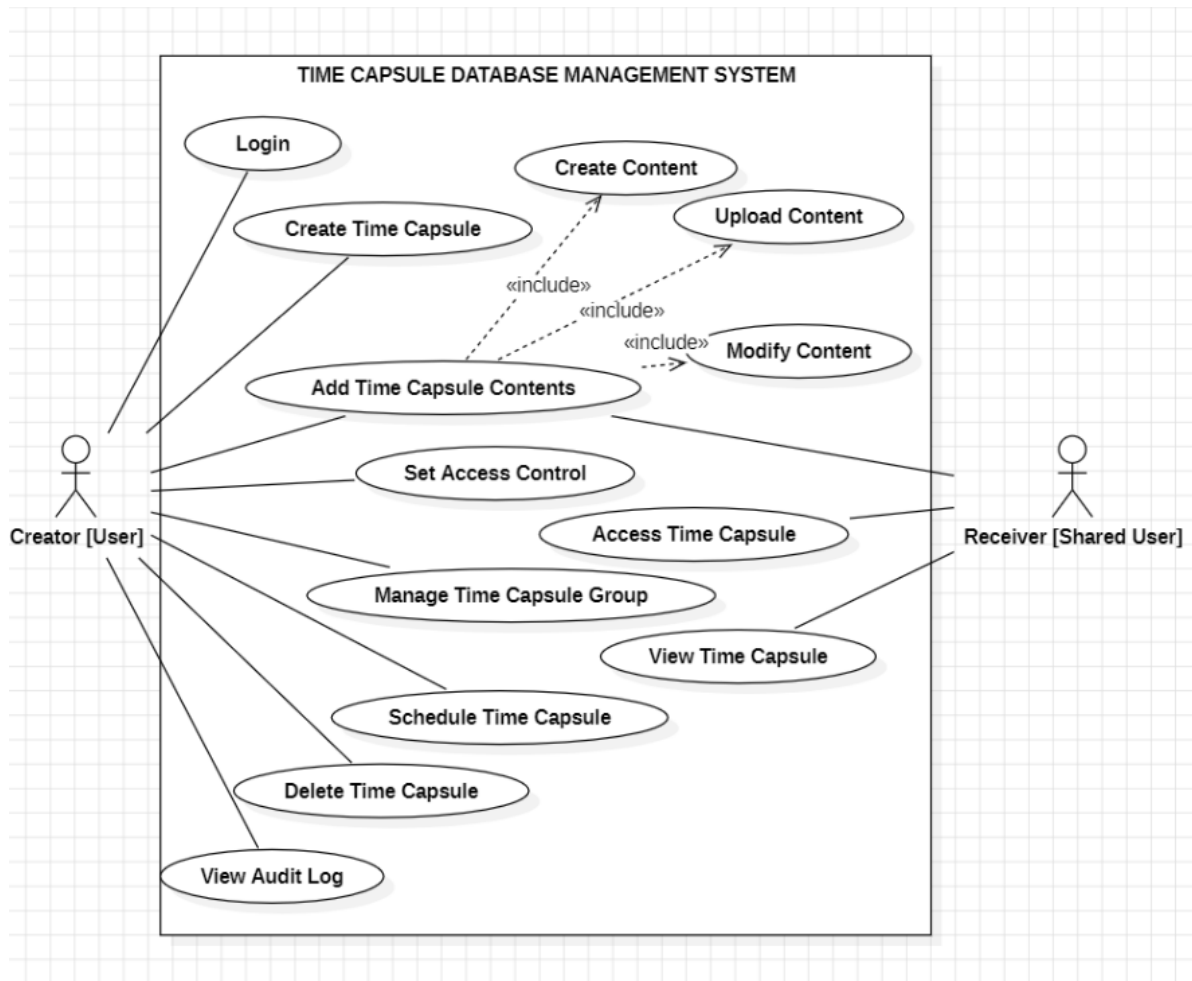
- Protocols: The system will use HTTP/HTTPS for web communications. It may also use email protocols (e.g., SMTP) for sending notifications related to time capsule releases.



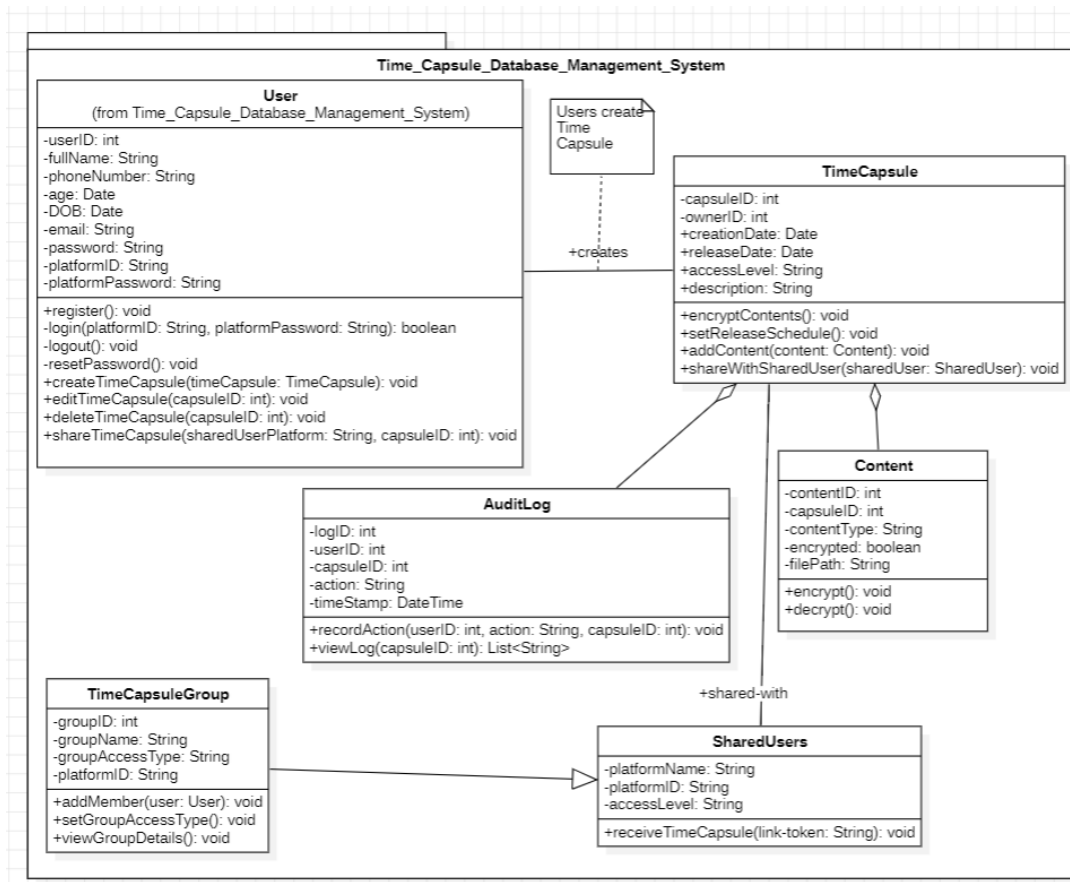
- **Security and Encryption:** Communications between the client and server will be encrypted using HTTPS. Any email communications will also follow security best practices to ensure data confidentiality.
- **Data Transfer Rates:** The system should be optimized for efficient data transfer and synchronization, ensuring minimal latency during content uploads and retrieval.

#### 4. Analysis Models

Use Case Diagram:



## Class Diagram:



## 5. System Features

### 5.1 User Authentication and Registration

#### 5.1.1 Description and Priority

This feature allows users to create an account, log in, and manage their time capsules. It is of High priority as it ensures secure access to the system.

#### 5.1.2 Stimulus/Response Sequences

**User Action:** The user registers or logs in.

**System Response:** The system verifies the credentials and provides access to the user's dashboard.

### *5.1.3 Functional Requirements*

- REQ-1: Users must register with an email and password.
- REQ-2: The system must support two-factor authentication for secure login.

## 5.2 Time Capsule Creation and Encryption

### *5.2.1 Description and Priority*

This feature allows users to create a time capsule by uploading content (letters, photos, documents) and setting a release date. The system ensures that all uploaded content is encrypted before storage. It is of High priority as it forms the core functionality of the system.

### *5.2.2 Stimulus/Response Sequences*

- User Action: A user selects "Create New Time Capsule," uploads content, sets a release date, and submits.
- System Response: The system encrypts the content, stores it in the database, and confirms successful creation with a unique identifier.

### *5.2.3 Functional Requirements*

- REQ-1: The system must allow users to upload multiple file types
- REQ-2: The system must encrypt all uploaded content before storing it in the database.
- REQ-3: The system must allow users to set a future release date.
- REQ-4: The system should validate content for size limits (e.g., max file size of 100MB).
- REQ-5: The system must send a confirmation message upon successful capsule creation.

## 5.3 Time Capsule Retrieval

### *5.3.1 Description and Priority*

This feature allows users to retrieve and open their time capsule once the release date is reached. It is of High priority as it is a fundamental operation of the system.

### *5.3.2 Stimulus/Response Sequences*

- User Action: The user requests to open a time capsule once the release date is reached.
- System Response: The system decrypts the content and provides access for the user to view or download.

### *5.3.3 Functional Requirements*

- REQ-1: The system must track the release date of each time capsule.
  - REQ-2: The system must notify users when their time capsule is ready to be opened.
  - REQ-3: The system must decrypt the content upon the release date.
  - REQ-4: The system must prevent access to the time capsule before the set date.
- 

## 6. Other Nonfunctional Requirements

### *6.1 Performance Requirements*

- The system must handle simultaneous access from multiple users efficiently, with minimal latency during upload or retrieval processes.
- The system must respond to user actions (e.g., capsule creation or retrieval) within 3 seconds under normal operating conditions.
- The system should scale to accommodate future growth in the number of time capsules and users.

### *6.2 Safety Requirements*

- The system must safeguard against data loss during content upload by using reliable storage and backup mechanisms.
- The system must have a mechanism to ensure that decryption keys are not compromised, preventing unauthorized access to sensitive data.

### *6.3 Security Requirements*

- All user interactions, including login and time capsule management, must be over secure HTTPS connections.
- The system must implement strong encryption (e.g., AES-256) for all stored content.
- The system must authenticate users before allowing access to any time capsule or sensitive functionality.
- Access control mechanisms must ensure only authorized users can manage or open specific time capsules.

### *6.4 Software Quality Attributes*

- **Adaptability:** The system should be adaptable to future enhancements, such as allowing users to add audio or video files.
- **Availability:** The system must be available 99.9% of the time, ensuring users can access their time capsules at critical moments.
- **Correctness:** All time capsules must be securely encrypted and retrievable only at the specified release date.
- **Usability:** The system will have an intuitive interface to ensure non-technical users can easily create and manage time capsules.
- **Maintainability:** The system must be easy to maintain, with clean, well-documented code to support future updates.
- **Security:** Data security is a high priority, with encryption and secure access control measures implemented.

### *6.5 Business Rules*

- Only the creator of a time capsule can modify or delete the capsule before the release date.
- Administrators must have the ability to monitor system health and manage encryption protocols but cannot access user content.

- Businesses may require contracts for custom storage solutions for large volumes of time capsules.

---

## 7. Other Requirements:

- Database Requirements: The SQL database should efficiently handle the storage of encrypted content, user information, and future release schedules.
- Internationalization: The system should support multiple languages for global users, especially for notifications and the user interface.
- Legal Requirements: The system must comply with data protection laws like GDPR for users in the EU, ensuring privacy and security of stored data.

---

## Appendix A: Field Layouts

### Field Layouts for Time Capsule Creation

Field Name	Length	Data Type	Description	Is Mandatory
User ID	10	Alphanumeric	Unique identifier for the user	Y
User Name	60	String	Full name of the user	Y
Email Address	100	String	User's email address for contact	Y
Email Password	20	Alphanumeric	User's password for authentication.	Y
Time Capsule ID	12	Alphanumeric	Unique identifier for the time capsule	Y
Creation Date	8	Date	Date the time capsule is created	Y

Release Date	8	Date	Date when the time capsule will be opened	Y
Content Description	255	String	Short description of the time capsule content	N
Content Size	10	Numeric	Size of the content in MB	Y
Encryption Key	32	Alphanumeric	Unique encryption key for the capsule	Y
Decryption Key	32	Alphanumeric	Decryption key used upon release	Y
Status	20	Alphanumeric	Status of the time capsule (e.g., Active, Released)	Y
Access Level	10	String	Access control level (e.g., Private, Public).	Y
Shared Users' Name	60	String	Names of the shared user.	N
Platform Name	50	String	Name of the platform associated with the time capsule.	Y
Platform Info	100	Alphanumeric	Credentials of the platform	Y

### Sample Report Requirements

#### Registration Report

Field	Description
User ID	Unique identifier for the user.
Full Name	The full name of the user.
Email Address	User's email address.

Field	Description
Phone Number	User's contact number.
Date of Birth	User's date of birth (format: YYYY-MM-DD).
Account Status	Current status of the user's account (e.g., Active, Inactive).
Registration Date	Date when the user registered (format: YYYY-MM-DD).

### Time Capsule Report

Field	Description
Capsule ID	Unique identifier for the time capsule.
Title	Title of the time capsule.
Description	Detailed description of the time capsule's content.
Created Date	Date when the time capsule was created (format: YYYY-MM-DD).
Encryption Key	Key used to encrypt the time capsule's content.
Release Date	Date when the time capsule is scheduled to be released (format: YYYY-MM-DD).
Access Level	Access control level (e.g., Private, Public).
Shared User Name	Name of the user with whom the capsule is shared.
Share Date	Date when the time capsule was shared (format: YYYY-MM-DD).
Platform Name	Name of the platform where the time capsule is stored.



## Appendix B : Requirement Traceability Matrix [RTM]

Sl. No	Requirement ID	Brief Description of Requirement	Architecture Reference	Design Reference	Code File Reference	Test Case ID	System Test Case ID
01	REQ-1	The system must allow users to upload multiple file types	TCDS-Architecture-V1.0	TimeCapsule	CreateCapsule.js	TC-01	STC-01
02	REQ-2	The system must encrypt all uploaded content	TCDS-Encryption-V1.2	EncryptionModule	EncryptCapsule.js	TC-02	STC-02
03	REQ-3	The system must allow users to set a future release date	TCDS-Database-V1.0	ReleaseDateManager	SetReleaseDate.js	TC-03	STC-03
04	REQ-4	The system should validate content for size limits	TCDS-Architecture-V1.0	FileSizeValidator	ValidateSize.js	TC-04	STC-04
05	REQ-5	The system must notify users when their capsule is ready	TCDS-Notification-V1.3	NotificationModule	NotifyUser.js	TC-05	STC-05

### 3. Project Plan with Gantt Chart (Baseline)

#### *Life-cycle followed*

The lifecycle model followed for the Time Capsule Database System project is the *Incremental Model*, chosen for its phased, structured approach.

#### *Justification for Choosing the Incremental Model:*

1. **Phased Development:** The project is divided into distinct stages, with each phase building on the previous ones.
2. **Feedback Incorporation:** Feedback is gathered at each stage, allowing necessary adjustments before proceeding to the next phase.
3. **Manageable Complexity:** Breaking the project into increments helps manage complexity by focusing on smaller, well-defined tasks.
4. **Early System Development:** Initial phases, like design and frontend work, provides a foundation for later stages of development.
5. **Parallel Development:** Frontend and backend could be developed concurrently, improving efficiency.

---

#### *Tools Used for this Project*

The following tools are identified for use throughout the lifecycle of the Time Capsule Database System (TCDS) project:

##### *1. Planning Tools:*

Trello: For task management and tracking project progress.

##### *2. Design Tools:*

Lucidchart: For creating Gantt charts and other visual diagrams.

Figma/Canva: For designing UI/UX mockups and visual elements.

StarUML: For creating ER diagrams, class diagrams, and use case diagrams.

##### *3. Version Control:*

Git: For source code management and version control, allowing collaboration and tracking

changes.

GitHub: For hosting the repository and facilitating collaboration among team members.

#### 4. *Development Tools:*

Visual Studio Code: As the primary code editor.

Next.js: For building the frontend of the application.

Node.js: For server-side development and API creation.

SQL Database (MySQL): For managing the relational database.

#### 5. *Bug Tracking:*

GitHub Issues: For tracking bugs, issues, and feature requests throughout the development process.

#### 6. *Testing Tools:*

Postman: For API testing to ensure backend functionality.

Selenium: For automated end-to-end testing of the application

---

### *Deliverables classified as reuse/build components*

#### Reuse Components:

- **Frontend UI Libraries:** Reusing Next.js features (which are based on React) along with reusable components from libraries such as Material UI or React Bootstrap.
- **Authentication Module:** Reuse an existing user authentication package like OAuth or JWT for managing user logins and session handling within the Next.js framework.
- **Styling/CSS Frameworks:** Utilize CSS frameworks (e.g., Tailwind CSS, Bootstrap) for consistent, responsive styling across the application.

*Justification:* Leveraging reusable tools like Next.js, OAuth/JWT for authentication, and Tailwind CSS/Bootstrap for styling ensures efficient, reliable development. These components are widely adopted, well-tested, and adaptable, helping to reduce development time while maintaining high standards.

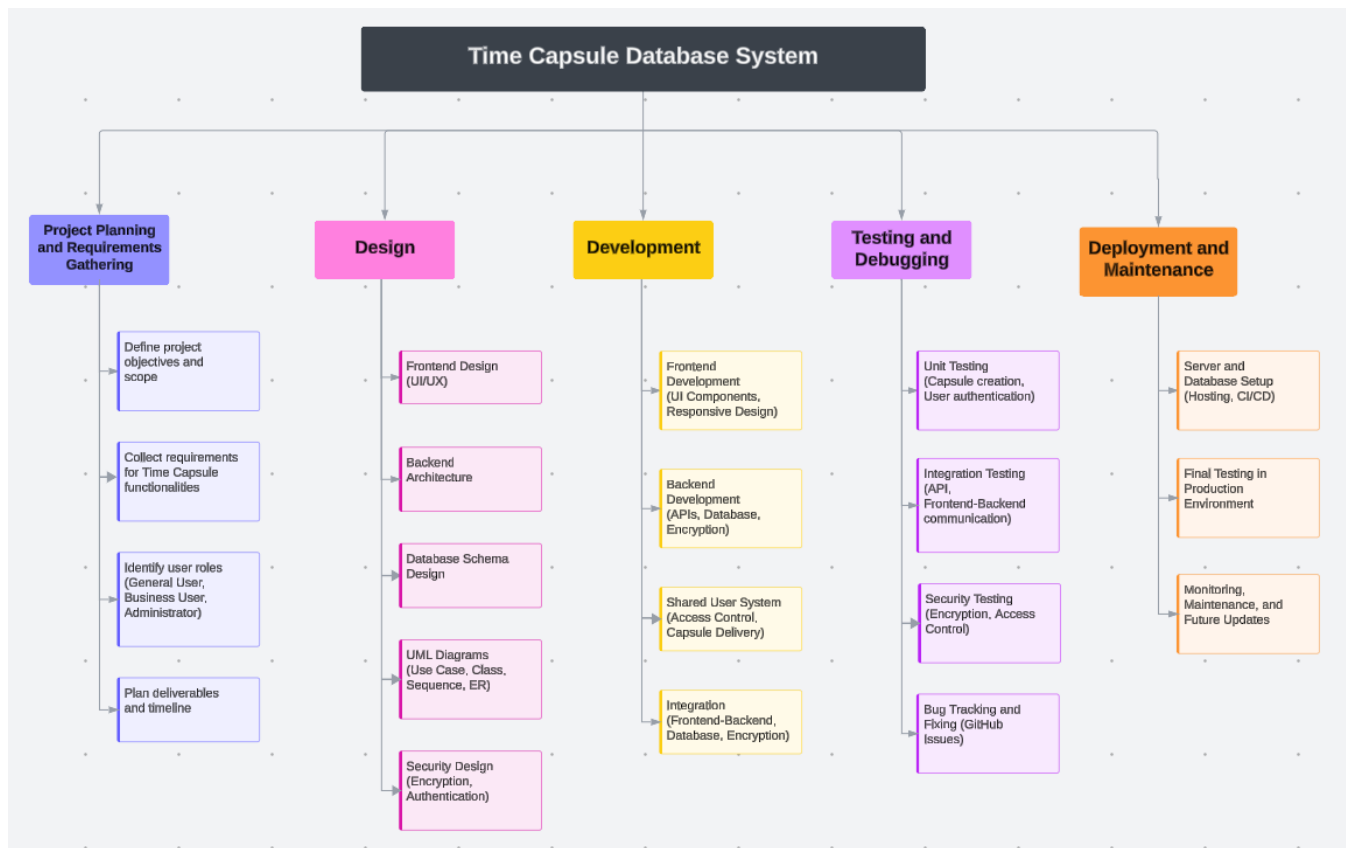
### Build Components:

- **Time Capsule Management:** Custom-built features for creating, uploading, encrypting content, and scheduling the release of time capsules.
- **User Profile Management:** Custom functionality for managing user profiles, preferences, and account settings.
- **Audit Log System:** Custom tracking system to log time capsule actions (e.g., creation, access, modifications).
- **Shared User System:** Build the system to manage shared users, access permissions, and platform-based delivery of time capsules.
- **Content Encryption Module:** Custom implementation for encrypting and securing time capsule contents.
- **Feedback and Rating System:** Build functionality to allow users to rate and provide feedback on time capsule interactions.

*Justification:* Custom-built features like Time Capsule Management, Audit Logs, and User Profile systems are essential for meeting the unique functional and security requirements of the project. These components need to be tailored specifically for time capsule creation, encryption, and user interaction, making them necessary to build from scratch.

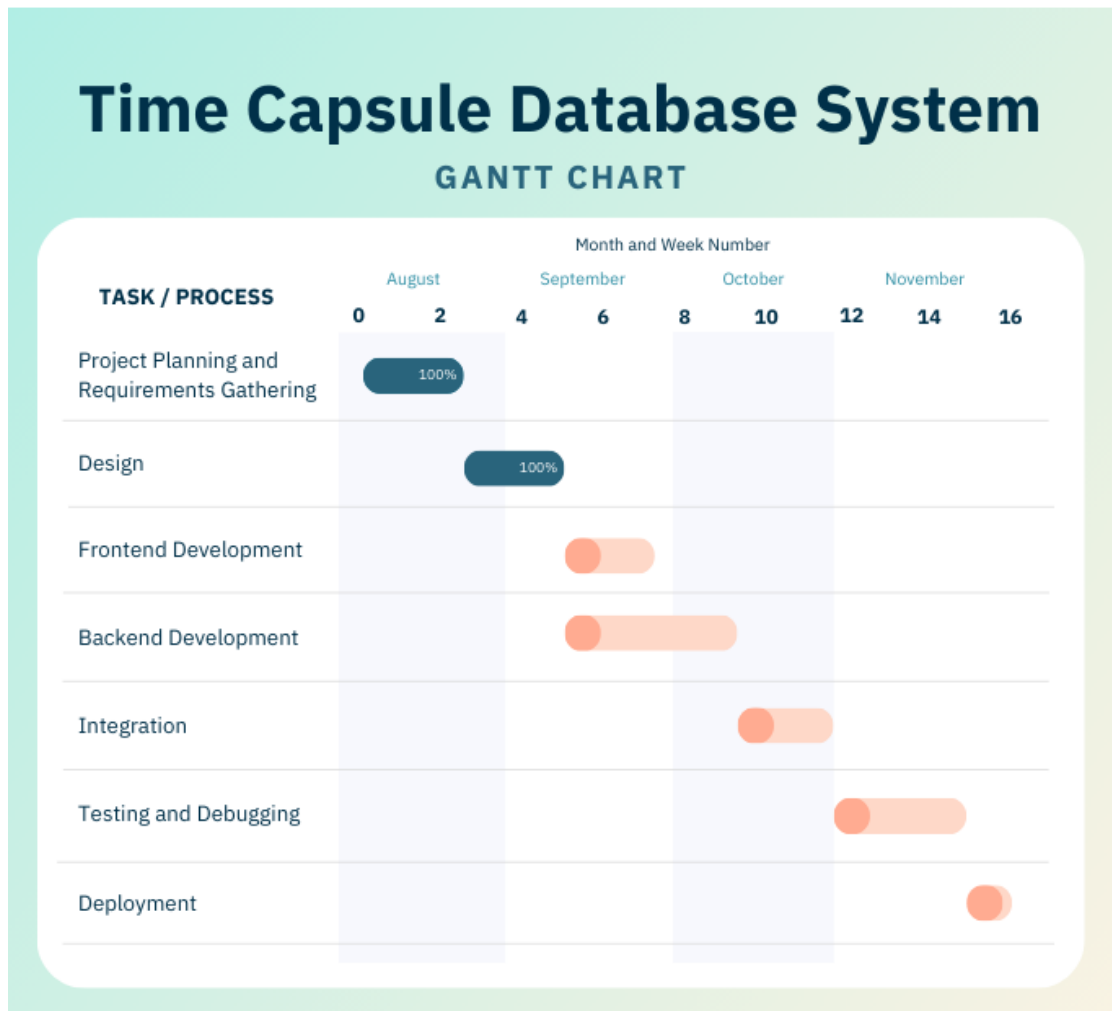
---

## Work Breakdown Structure



### Effort Estimation (in person-months)

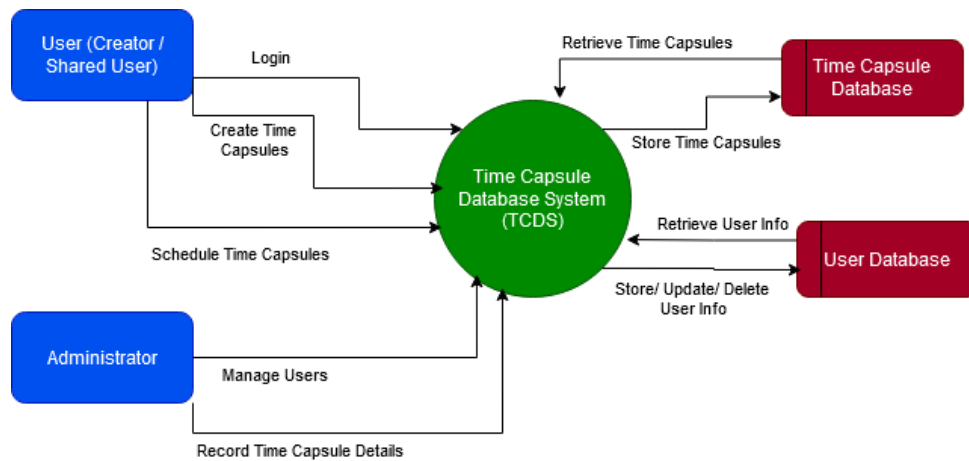
- Project Planning and Requirements Gathering: Estimated Effort: 0.692 person-months (15 days)
- Design: Estimated Effort: 0.692 person-months (15 days)
- Frontend Development: Estimated Effort: 0.323 person-months (7 days)
- Backend Development: Estimated Effort: 0.923 person-months (20 days)
- Integration: Estimated Effort: 0.323 person-months (7 days)
- Testing and Debugging: Estimated Effort: 0.692 person-months (15 days)
- Deployment: Estimated Effort: 0.230 person-months (5 days)
- Total Estimated Effort: 3.875 person-months (84 days)

*Gantt Chart*

## 4. Architecture & Design Choices and Diagrams

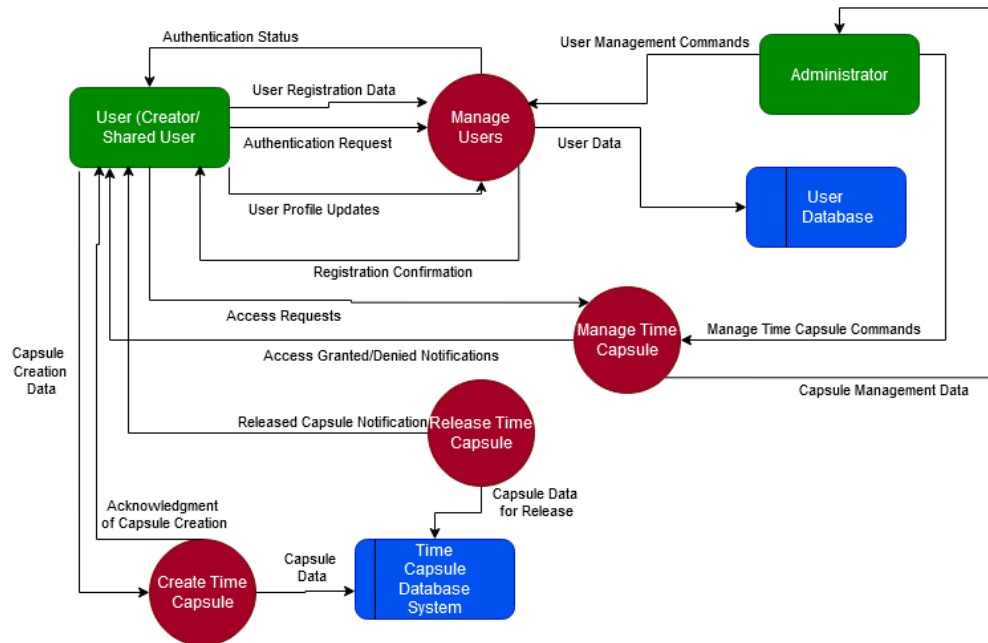
### *Design Diagrams:*

Level 0:



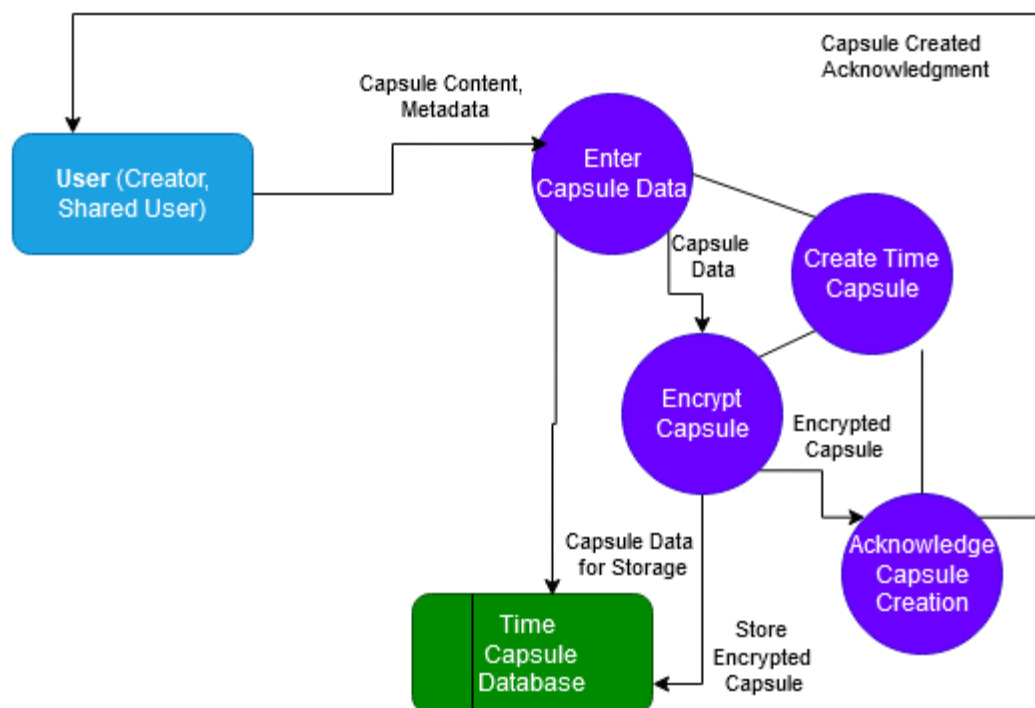
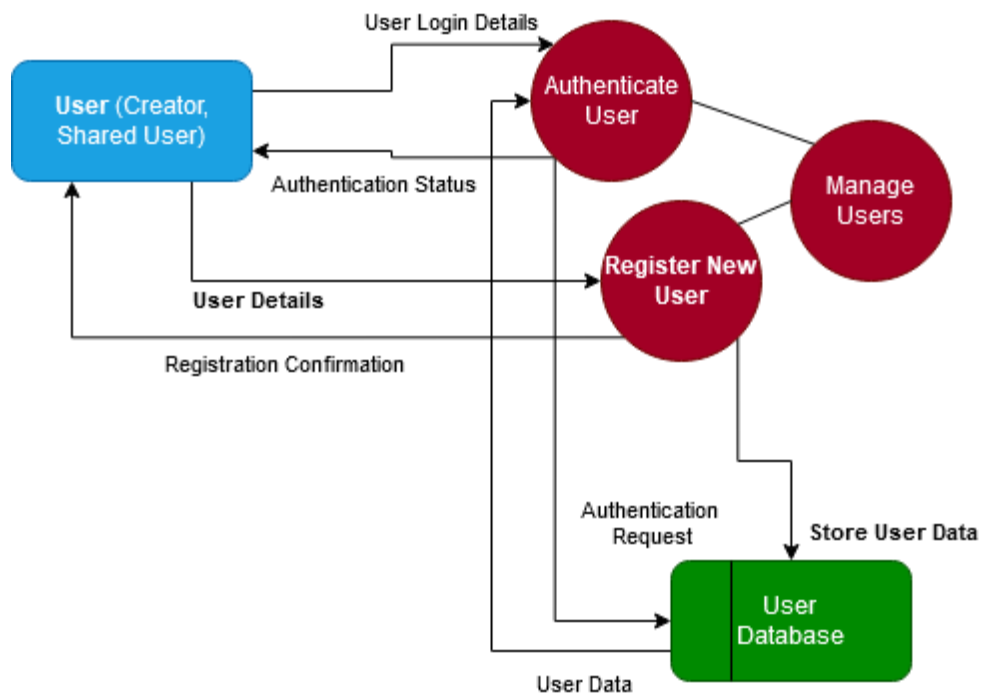
**Level 0 DFD**

Level 1:

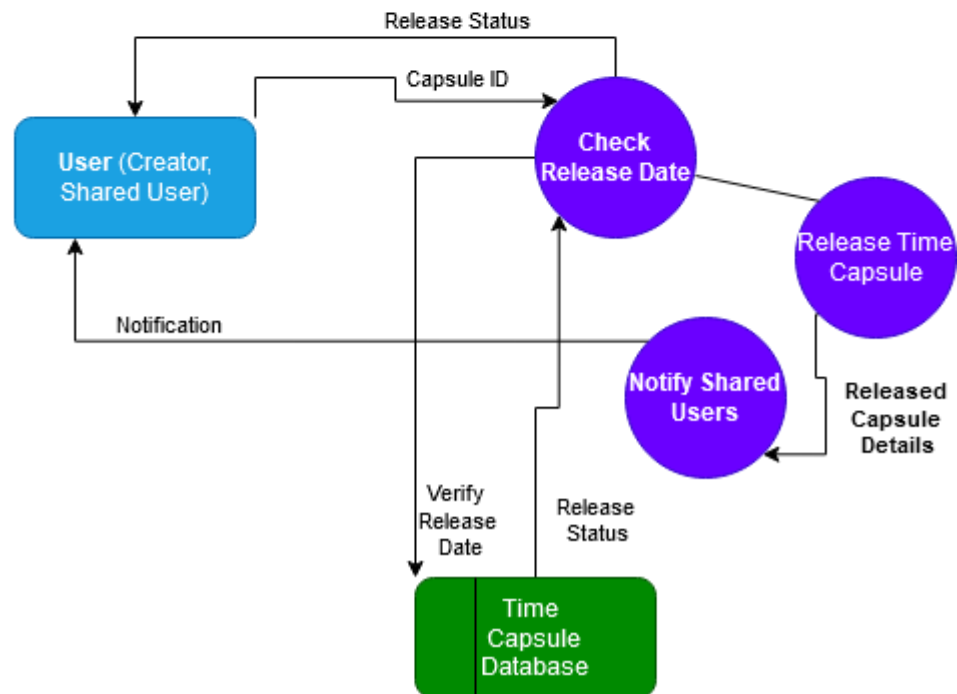
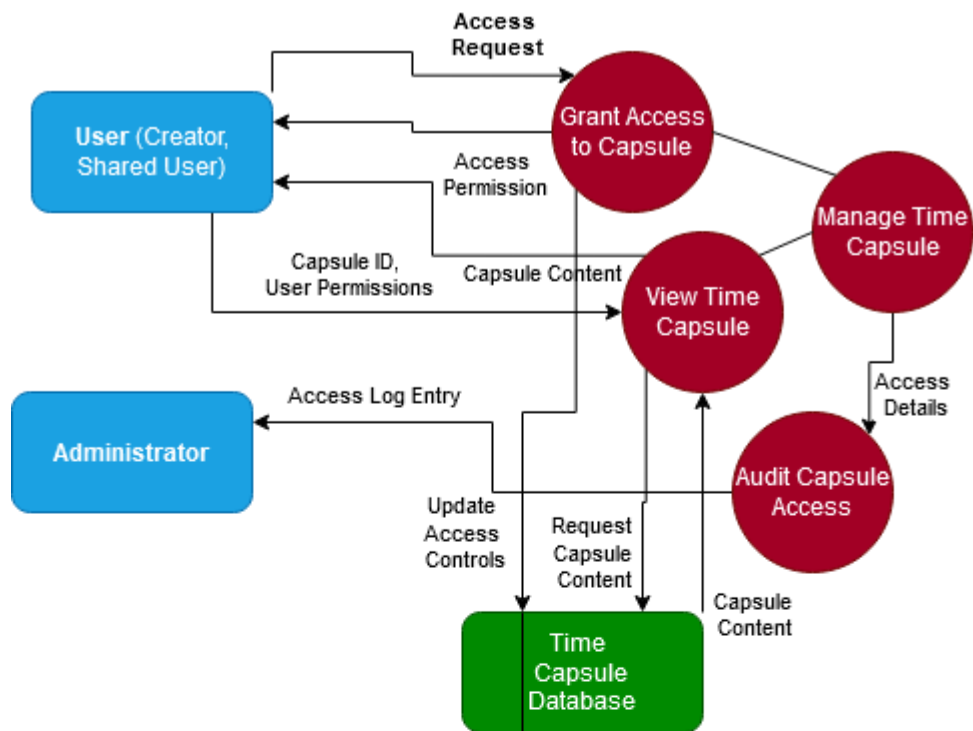


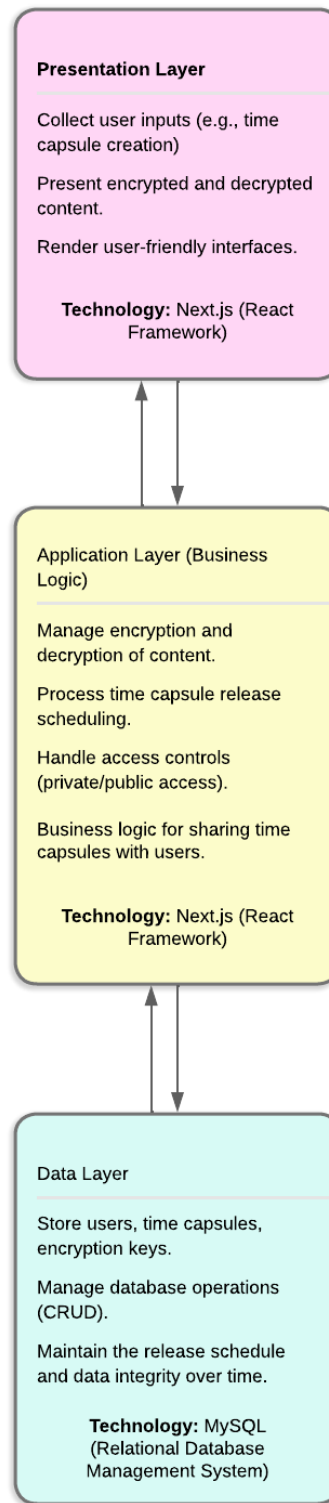
**Level 1 DFD**

Level 2:



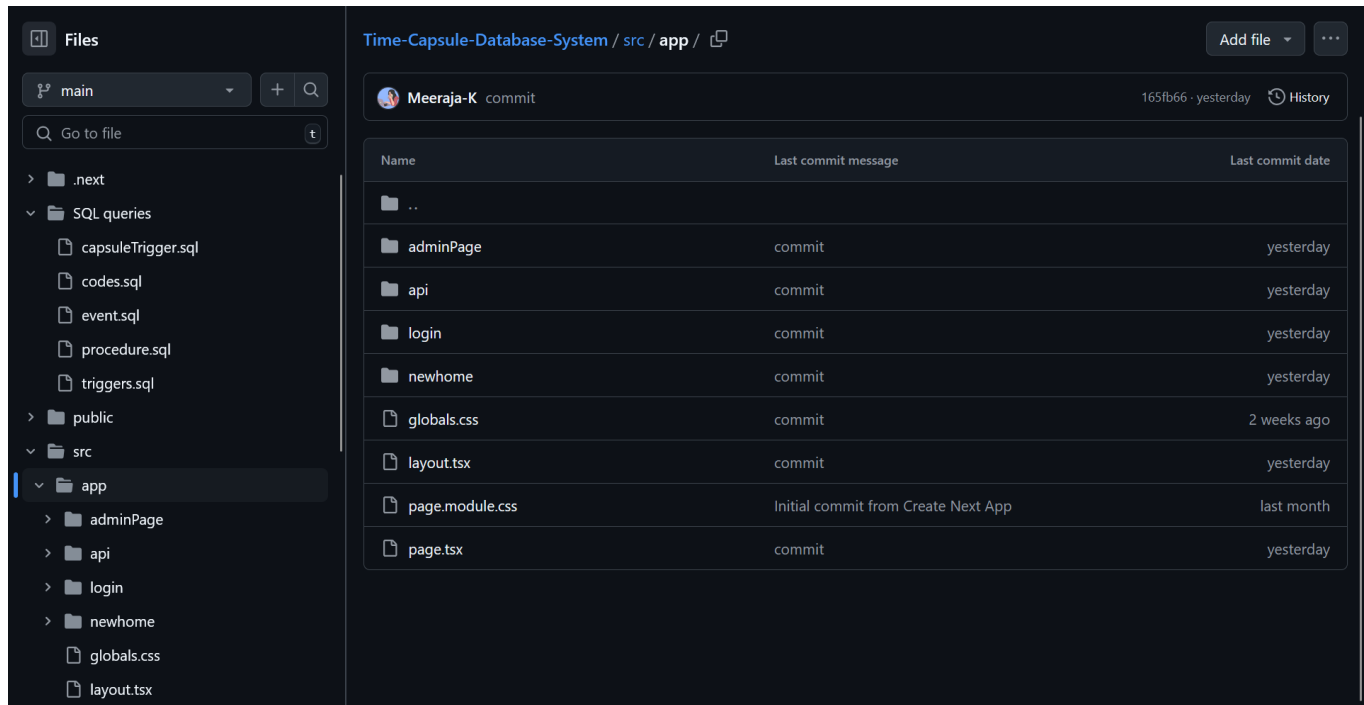




*Architectural Design: Three-Tier Architecture*

## 5. Development - Code Files [Git link]

<https://github.com/Meeraja-K/Time-Capsule-Database-System>



## 6. Test Plans

### 1. Introduction

#### 1.1 Scope

This Test Plan outlines the testing activities for the Time Capsule Database System (TCDS), focusing on functionalities such as user authentication, time capsule creation, encryption, shared user management, release date tracking, and content accessibility. The goal is to ensure a secure, user-friendly platform for preserving and retrieving digital content.

In Scope:

- User Authentication: Login and authorization for users.

- Time Capsule Creation: Uploading content, encryption, and setting release dates.
- Shared User Management: Adding shared users with controlled access.
- Time Capsule Retrieval: Capsule access for authorized users after the release date.
- Data Security: Ensuring encryption of content and secure access.
- User Interface: Testing usability, accessibility, and responsiveness across devices.

#### Out of Scope:

- Integration with third-party platforms or APIs for notifications.
- Advanced multimedia handling (e.g., videos or audio).

### 1.2 Quality Objectives

- Ensure all functionality operates as intended with minimal defects.
  - Maintain encryption and security for all user data and content.
  - Provide responsive and accessible interfaces for all user classes.
  - Meet 99.9% availability for system reliability.
- 

## 2. Test Methodology

### 2.1 Overview

The testing methodology includes manual and automated strategies. Manual testing will validate user scenarios, while automated tools will handle regression and performance testing.

### 2.2 Test Levels

1. Unit Testing: Validate individual components like login, time capsule creation, and encryption.
2. Integration Testing: Ensure seamless interaction between modules such as user authentication, database operations, and shared user functionality.
3. System Testing: Verify overall system behavior, including workflows from login to time capsule release.

4. User Acceptance Testing (UAT): Validate the system against user requirements before deployment.

## 2.3 Bug Triage

- Critical: Data loss, unauthorized access to capsules, or encryption failure.
- High: Failure in time capsule creation, retrieval, or shared user management.
- Medium: Minor usability issues with workarounds.
- Low: Cosmetic UI inconsistencies.

## 2.4 Suspension and Resumption Criteria

- Suspension: Testing will be paused if:
  - Login or encryption modules fail consistently.
  - A security breach compromises sensitive data.
- Resumption: Testing will resume once:
  - Major defects are fixed and retested.
  - System stability and security are restored.

## 2.5 Test Completion

Testing is complete when:

- All critical and high-priority defects are resolved.
- All planned test cases are executed with acceptable pass rates.
- Performance and security benchmarks are met.

---

## 3. Test Deliverables

- Test Plan Document: This document outlining the testing strategy.
- Test Cases: Detailed scenarios for all functionalities.
- Test Scripts: Automated scripts for regression testing.
- Bug Reports: Logs of all identified defects categorized by severity and priority.
- Test Summary Report: Consolidated results of all test phases.

#### *4. Resource and Environment Needs*

##### 4.1 Testing Tools

- Postman: API testing.
- Selenium: Automated UI testing.
- Jest: Unit testing.
- MySQL Workbench: Database validation.

##### 4.2 Test Environment

- Hardware:
    - Development and testing machines (Windows/Linux/macOS).
    - Cloud-hosted server for database and backend services.
  - Software:
    - Next.js-based frontend with a responsive UI.
    - Node.js and Express.js backend services.
    - MySQL relational database.
  - Network:
    - Simulated environment to validate secure HTTPS communication.
- 

#### *5. Test Scenarios*

##### 5.1 Functional Testing

- Authentication:
  - Validate login with valid and invalid credentials.
  - Ensure access is restricted to authorized users.
- Time Capsule Creation:
  - Verify content upload for PNG, JPG, and GIF files.
  - Ensure future release dates are correctly set and validated.
  - Confirm encryption before storing content.

- Shared User Management:
  - Test adding up to 3 shared users.
  - Validate access for shared users after the release date.
- Capsule Retrieval:
  - Ensure capsules remain sealed until the release date.
  - Confirm only authorized users can access delivered capsules.

## 5.2 Performance Testing

- Test system response times for:
  - Capsule creation and retrieval.
  - Concurrent access by multiple users (up to 50 at once).
- Validate database queries for efficiency and scalability.

## 5.3 Security Testing

- Ensure all capsule content is encrypted using AES-256.
- Test unauthorized access attempts and validate system defense.
- Verify secure HTTPS communication for all data transfer.

## 5.4 Usability Testing

- Evaluate the UI for clarity and ease of use.
- Test responsiveness across desktop, tablet, and mobile devices.

---

## 6. Additional Considerations

- Performance Benchmarks: Ensure capsule creation and retrieval take no longer than 3 seconds under normal load.
- Security Compliance: Validate adherence to GDPR and other applicable data privacy laws.
- User Feedback: Incorporate feedback from UAT to refine the system.

## 7. Test Cases and Test Results Matrix – including Screenshots of Inputs and Resulting Outputs after Execution of Test Cases

Test Case ID	Name of the Module	Test Case Description	Pre-conditions	Test Steps	Test Data	Expected Results	Actual Results	Test Result [Pass/Fail]
UT_01	User Authentication module	Verify successful registration with valid data	Access to any browser and website's registration page	1. Navigate to register page 2. Enter First Name, Last Name, valid email, password and username 3. Submit registration form	First Name: Daisy Last Name: Ford  Email: daisy@gmail.com Password: Daisy@23  Username: Daisy1	User registered and is redirected to Profile page.	User registered successfully message is displayed and then redirected to Profile page.	Pass
UT_02	User Authentication module	Verify error when registering with existing email	Email already exists in database	1. Navigate to registration page 2.	First Name: Daisine Last Name: Fordy	Error message: "Email or username already exists"	Error message: "Email or username already exists" displayed	Pass



				Enter different first name, last name, username but existing email 3. Submit registration form	Email: daisy@gmail.com Password: Dais@97 Username: Daisine		, user remains on the register page.	
UT_03	User Authentication module	Verify successful registration with different name, email and password, but same username	Access to any browser and website's registration page	1. Navigate to register page 2. Enter unique First Name, Last Name, valid email, password and duplicate username 3. Submit registration form	First Name: Daina Last Name: Joseph Email: daina@gmail.com Password: Daina@33D Username: Daisy1	Error message: "Email or username already exists"	Error message: "Email or username already exists" is displayed and the user remains in the register page	Pass

UT_04	User Authentication module	Verify error when registering with invalid email	Access to any browser and website's registration page	1. Navigate to register page 2. Enter First Name, Last Name, invalid email, password and username 3. Submit registration form	First Name: Anvi Last Name: Shree Email: anvi@gmail.com Password: anvi.shree Username: Anvi	Error message: "Enter a valid email address."	Error message: "Enter a valid email address." is displayed and the user remains in the register page	Pass
UT_05	User Authentication module	Verify error when registering with invalid name (First/Last name)	Access to any browser and website's registration page	1. Navigate to register page 2. Enter invalid First Name, Last Name, valid email, password and username 3.	First Name: Disha 12 Last Name: Raj Email: disha@gmail.com Password: diiisha Username: Disha_R	Error message: "Name must contain only alphabets."	Error message: "Name must contain only alphabets." is displayed and the user remains in the register page	Pass

				Submit registration form				
UT_06	User Authentication module	Verify error when registering with invalid name (First/Last name)	Access to any browser and website's registration page	1. Navigate to register page 2. Enter First Name, invalid Last Name, valid email, password and username 3. Submit registration form	First Name: Disha Last Name: Raj@1 Email: disha@gmail.com Password: diiisha Username: Disha_R	Error message: "Name must contain only alphabets."	Error message: "Name must contain only alphabets." is displayed and the user remains in the register page	Pass
UT_07	User Authentication module	Verify login with correct credentials	User is registered	1. Go to login page 2. Enter username and password 3. Click submit	Username: Daisy1 Password: Daisy@23	Successful login and user is redirected to profile page	User is logged in successfully and is redirected to profile page	Pass

UT_08	User Authentication module	Verify error for incorrect password during login	User is registered	1. Go to login page 2. Enter username and incorrect password 3. Click submit	Username: Daisy1 Password: diassy@23	Error message: "Invalid credentials"	Error message: "Invalid credentials" is displayed and user remains in the login page.	Pass
UT_09	User Authentication module	Verify error for new unregistered username during login	User is not registered	1. Go to login page 2. Enter new username and password 3. Click submit	Username: Sasha Password: sasha1665	Error message: "Invalid credentials"	Error message: "Invalid credentials" is displayed and the user remains in the login page	Pass
UT_10	Time Capsule Creation module	Verify time capsule creation with valid files	User logged in	1. Navigate to capsule page from the profile 2. Enter Capsule Name,	Capsule Name: Memories, Share with username: Sasha, Files: letter.t	Time capsule created with confirmation message	Time capsule created successfully and user is redirected to dashboard	Pass

				share with username, Upload valid files 3. Set release date 4. Submit	xt, sunflower.jpg Release Date: 2025-01-01			
UT_11	Time Capsule Creation module	Verify error for exceeding file size limit	User logged in	1. Navigate to capsule page 2. Upload oversized file 3. Submit	File: image.png (size > 100MB)	Error message: "File size exceeds limit"	Error message is displayed and the user remains in capsule creation page	Pass
UT_12	Time Capsule Creation module	Verify error for unsupported file type	User logged in	1. Navigate to capsule page 2. Upload unsupported file type 3. Submit	File: video.mov	Error message: "Unsupported file type"	Error message is displayed and the user remains in capsule creation page	Pass
UT_13	Time Capsule	Verify error for repeated	User logged in	1. Naviga	Capsule	Error message:	Error message	Pass

	Creation module	capsule name		te to capsule page from the dashboard 2. Enter duplicate Capsule Name, platform id, share with username, shared user id, Upload valid files 3. Set release date 4. Submit	Name: Memories, Release through: <a href="mailto:daisy@gmail.com">daisy@gmail.com</a> , Share with username: Hasya, shared user id: hasya@gmail.com, Files: wallpaper.jpg Release Date: 2024-12-01	"Capsule already exists."	is displayed and the user remains in capsule creation page	
UT_14	Time Capsule Creation module	Verify error for invalid platform id	User logged in	1. Navigate to capsule page from the dashboard 2. Enter	Capsule Name: Info, Release through: <a href="https://daisy.com">https://daisy.com</a>	Error message: "Invalid platform id."	Error message is displayed and the user remains in capsule creation page.	Pass

				Capsul e Name, invalid platfor m id, share with userna me, shared user id, Upload valid files 3. Set release date 4. Submit	<a href="#">com,</a>  Share with userna me: Lisa, shared user id: lisa@g mail.c om, Fil es: girl.pn g Releas e Date: 2024- 12-10			
UT_ 15	Time Capsule Creation module	Verify error for invalid platform id	User logged in	1. Naviga te to capsule page from the dashbo ard 2. Enter Capsul e Name, valid platfor m id, share with userna	Capsu le Name: Work,  Releas e throug h: <a href="#">kris@g mail.c om,</a>  Share with userna me: Lisa, shared user id:	Error message: "Invalid platform id."	Error message is displayed and the user remains in capsule creation page.	Pass

				me, invalid shared user id, Upload valid files 3. Set release date 4. Submit	78767 21822, Files: history .txt Releas e Date: 2024- 12-25			
UT_ 16	Time Capsule Creation module	Verify error for invalid/past release date	User logged in	1. Naviga te to capsule page 2. Upload files 3. Set Release date to past date 4. Submit	Releas e Date: 2024- 1-25	Error message: "Invalid date."	Error message is displayed and the user remains in capsule creation page.	Pass
UT_ 17	Time Capsule Encryption module	Verify content encryption after capsule creation	User logged in and capsule created	Check storage for encrypt ion on stored conten t	Capsu le conten t: letter.t xt, photo. jpg	Content is encrypted and unreadable in database	Content is encrypte d and unreadab le in database	Pass
UT_ 18	Time Capsule Update	To verify updating of capsule	User logged in and capsule created	1.Chec k for the	Capsu le name:	Capsule release date	Capsule release date is	Pass



	module	release date.		capsule in scheduled capsule page 2.Click on edit capsule release date  3.Submit	Work Update Capsule release date: 2024/12/12	is updated.	updated.	
UT_19	Time Capsule Update module	To verify updating of capsule release date.	User logged in and capsule created	1.Check for the capsule in scheduled capsule page 2.Click on edit capsule release date and enter invalid date  3.Submit	Capsule name: Work Update Capsule release date: 2024/10/10	Error message: Invalid date	Error message is displayed and initial capsule release date is retained.	Pass
UT_20	Time Capsule Retrieval module	Verify retrieval of capsule after release date	Capsule with past release date exists	1. Login 2. Navigate to capsule	Capsule name: Test1	Capsule is decrypted and content is displayed	Capsule content is accessible	Pass

				s 3. Retrie e capsule				
UT_ 21	Time Capsule Retrieval module	Verify restriction of capsule access before release date	Capsule with future release date exists	1. Login 2. Attem pt to access future capsule	Capsu le name: Memo ries	Access denied with message: "Capsule not available until release date"	Error message is displayed	Pass
UT_ 22	Time Capsule Retrieval module	Verify restriction of capsule access to users with whom capsule was not shared	Capsule exists	1. Login 2. Attem pt to access capsule	Capsu le name: Plans	Access denied with message: "This capsule is not shared with you"	Error message is displayed	Pass
UT_ 23	Notifications module	Verify notification on release date arrival	Release date for capsule is today	1. Wait for release date 2. Check user notifica tions	Capsu le name: Projec t	Notificatio n sent to user: "Your capsule is now available"	Notificati on is received	Pass
UT_ 24	User Profile	Verify profile update with valid data	User logged in	1. Naviga te to profile page	Usern ame: Daisy Updat ed	Profile updated successfull y with confirmati	Profile is updated.	Pass

				2. Update profile details 3. Save changes	User Email: daisy22@gmail.com	on		
UT_25	User Profile	Verify profile update with invalid email	User logged in	1. Navigate to profile page 2. Enter invalid email 3. Save changes	Username: Daisy Updated User Email: daisy22@gmail.com	Error message: "Update failed. Please check your inputs."	Error message is displayed.	Pass
UT_26	User Settings	Verify account deletion and data removal	User logged in	1. Navigate to settings 2. Select delete account 3. Confirm deletion	Username: Sameha Password: Sameha	Account deleted, user data and capsules removed	Account is deleted.	Pass
IT-01	User-Capsule Integration module	Verify successful link between user and newly created	User logged in and capsule created	1. Create capsule 2. Check	Username: Sasha Capsule	Capsule linked to user in database	Capsule is linked to user in database	Pass

		capsule		database for capsule association	name: Memories			
IT-02	Capsule-Encryption Integration	Verify capsule encryption on upload completion	Capsule created	1. Create capsule 2. Check database for encryption status	Capsule name: Memories	Capsule content is encrypted and stored securely	Capsule content is encrypted and stored securely	Pass
IT-03	Capsule-Notification Integration	Verify user receives notification upon capsule release	Release date for capsule is today	1. Set capsule with today's release date 2. Wait for notification	Capsule name: Today Release Date: 2024/11/06	Notification received with message: "Your capsule is now available"	Notification received	Pass
ST-01	Full System Test	Verify end-to-end flow from registration to capsule retrieval	New user registers and creates capsule	1. Register 2. Create capsule 3. Retrieve capsule on release date	Email: <a href="mailto:meer@gmail.com">meer@gmail.com</a> , Capsule name: Check, Capsule contents:	User registered, capsule created, content accessible on release date	User registered, capsule created, content accessible on release date	Pass

					letter.txt, photo.jpg			
ST-02	Security Test	Verify only authorized users can access specific capsules	Capsule exists with set access control	1. Attempt unauthorized access to capsule	Capsule name: Project	1. Attempt unauthorized access to capsule	Access denied	Pass
ST-03	Load Test	Verify system performance under high user load	High number of users accessing system	1. Simulate multiple users creating and retrieving capsules	—	System remains responsive without significant delay	System remains responsive without significant delay	Pass

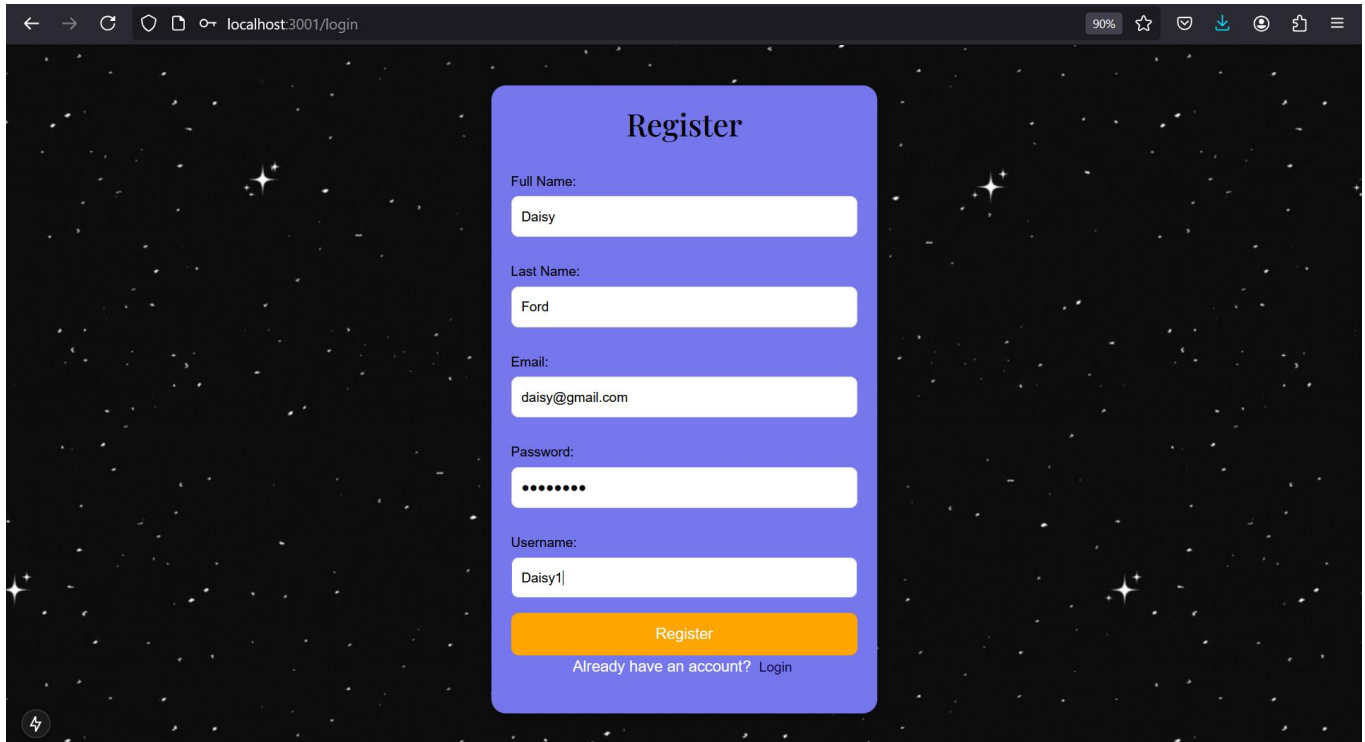
Count of Passed Test Cases = 75

Count of Failed Test Cases = 0

Total Number of Test Cases = 75

## Screenshots:

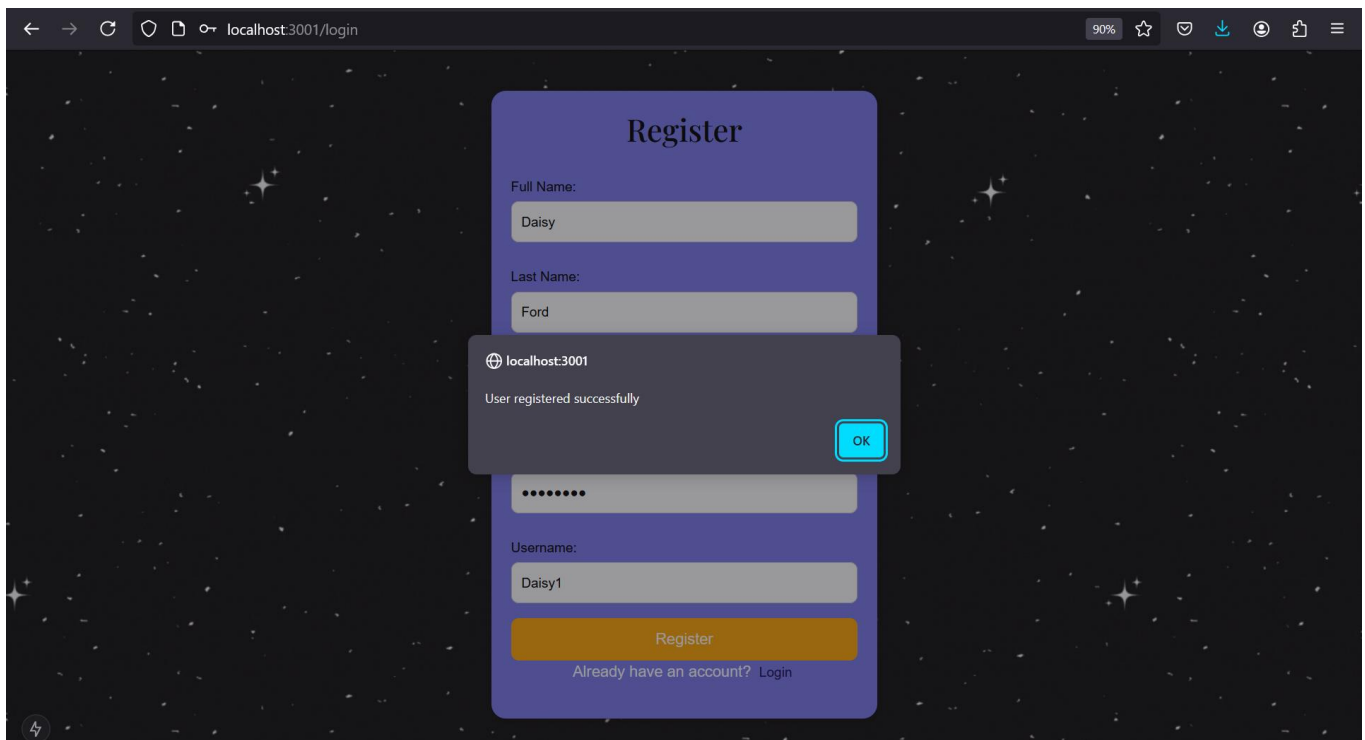
Test Case ID: UT-01



A screenshot of a web browser displaying a registration form titled "Register". The form is set against a dark space background with white stars. The form fields are as follows:

- Full Name: Daisy
- Last Name: Ford
- Email: daisy@gmail.com
- Password: (masked with dots)
- Username: Daisy1

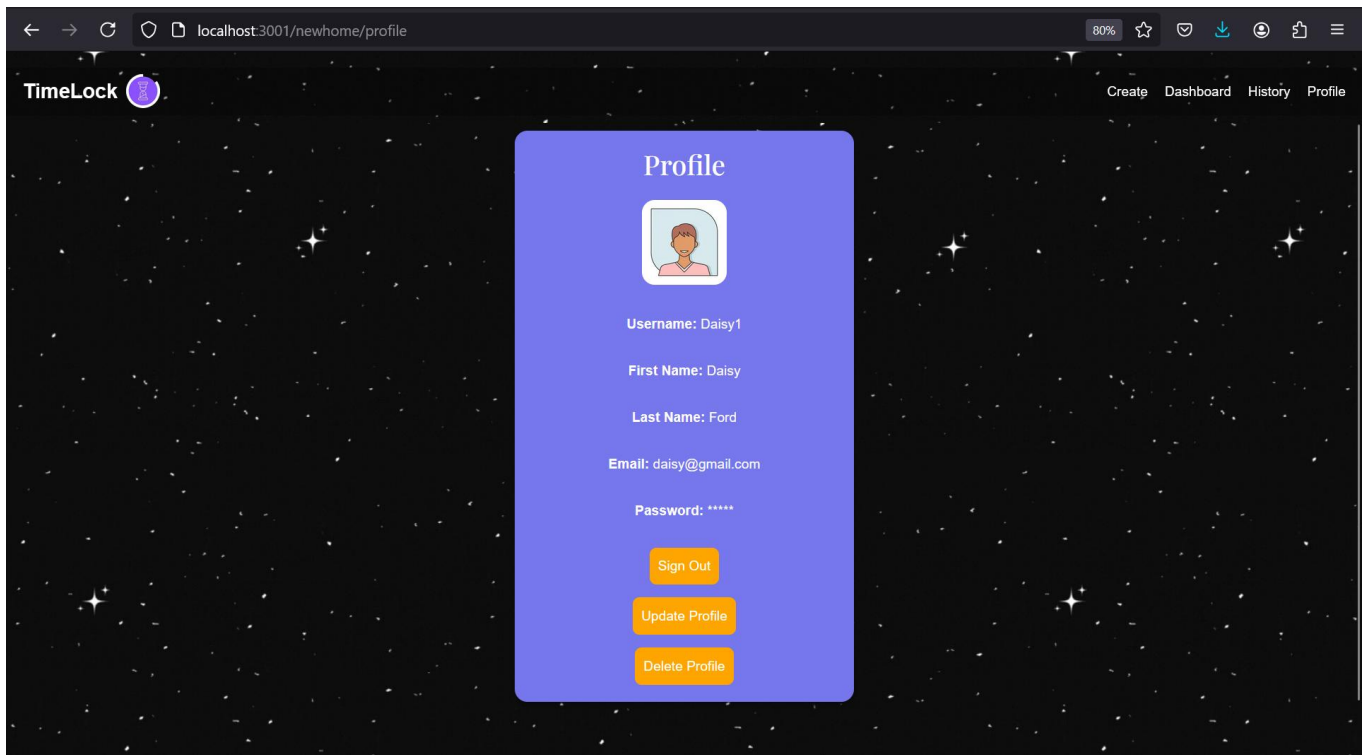
Below the fields is an orange "Register" button. Underneath the button is a link that says "Already have an account? Login". The browser's address bar shows "localhost:3001/login".



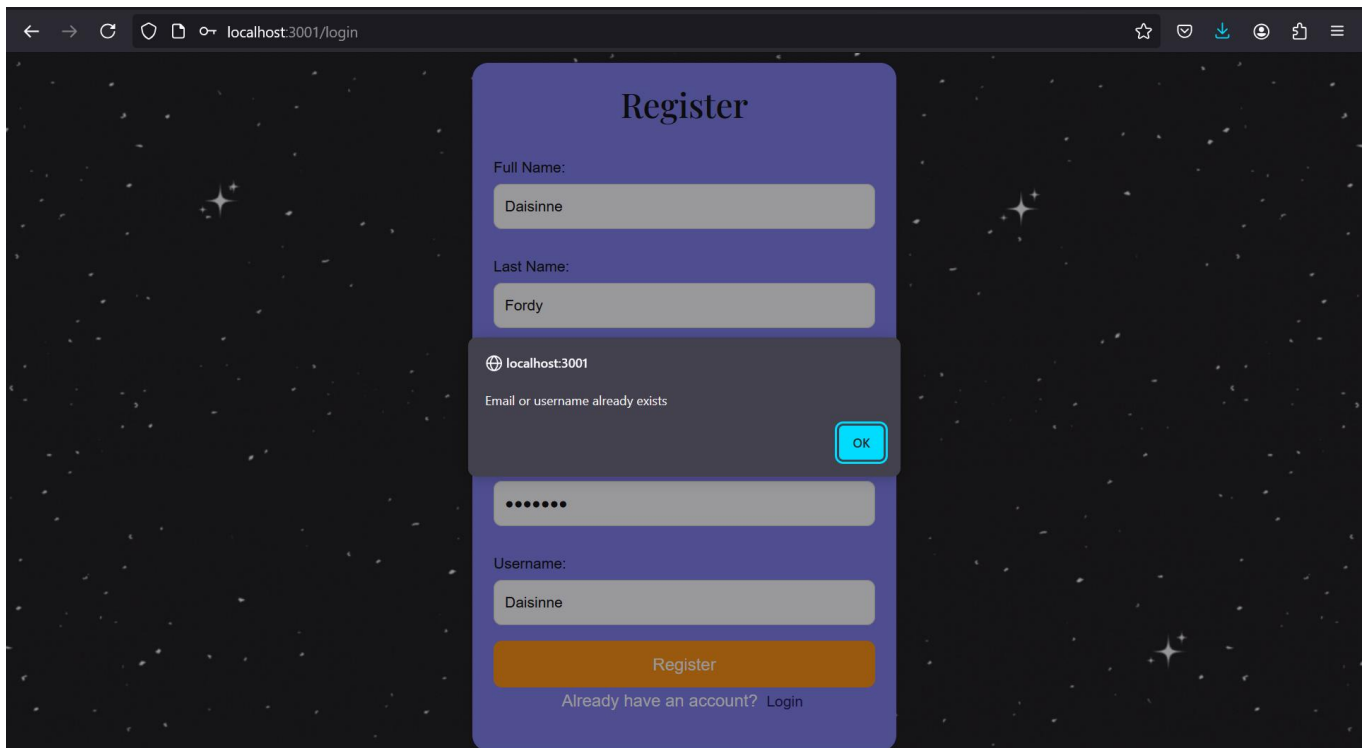
A screenshot of the same web browser displaying the registration form. A dark grey modal box is overlaid on the form, indicating a successful registration. The modal contains the text "User registered successfully" and an "OK" button. The form fields are now disabled (greyed out):

- Full Name: Daisy
- Last Name: Ford
- Password: (masked with dots)
- Username: Daisy1

The "Register" button is also greyed out. The "Already have an account? Login" link remains visible. The browser's address bar shows "localhost:3001/login".



Test Case ID: UT-02 & Test Case ID: UT-03



## Test Case ID: UT-04

Register

Full Name:  
Anvi

Last Name:  
Shree

Email:  
anvi@gmail

Enter a valid email address.

Password:  
.....

Username:  
Anvi

Register

Already have an account? [Login](#)

## Test Case ID: UT-05 &amp; Test Case ID: UT-06

Register

Full Name:  
Disha12

Name must contain only alphabets.

Last Name:  
Raj

Email:  
disha@gmail.com

Password:  
.....

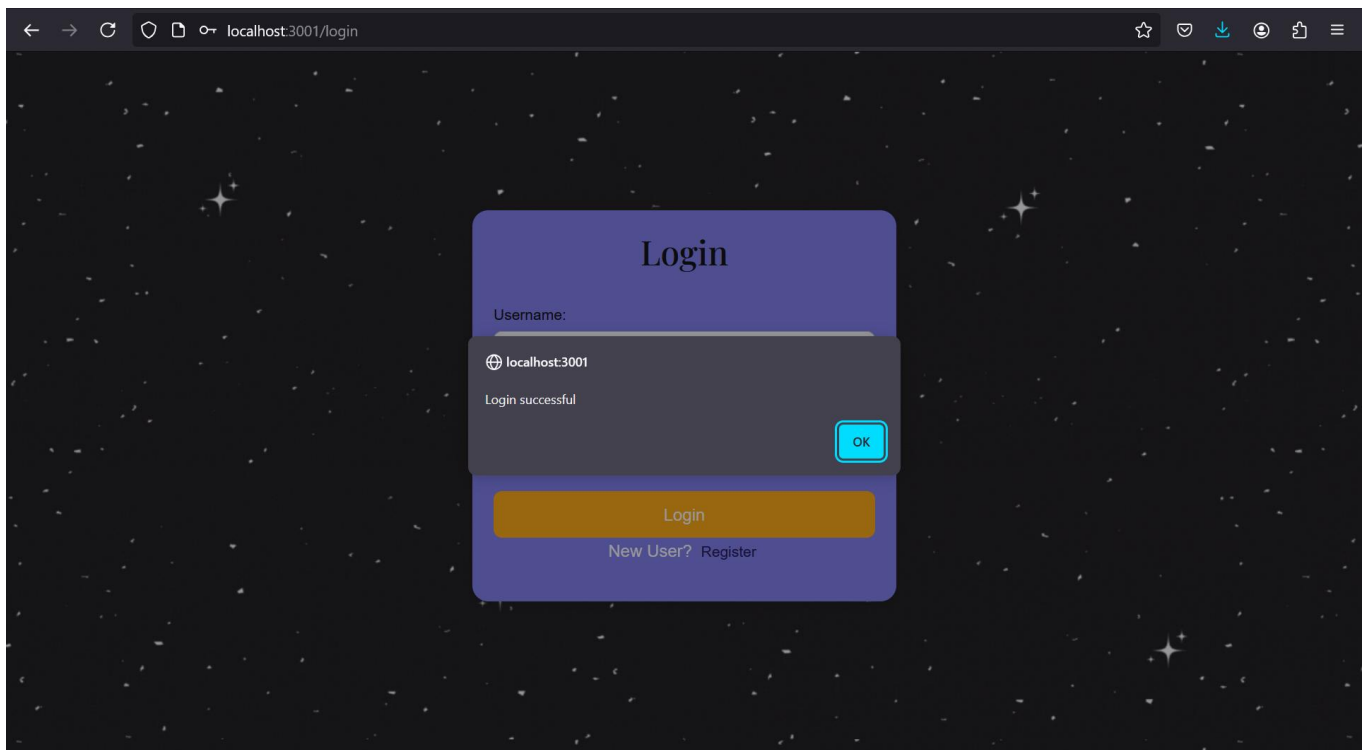
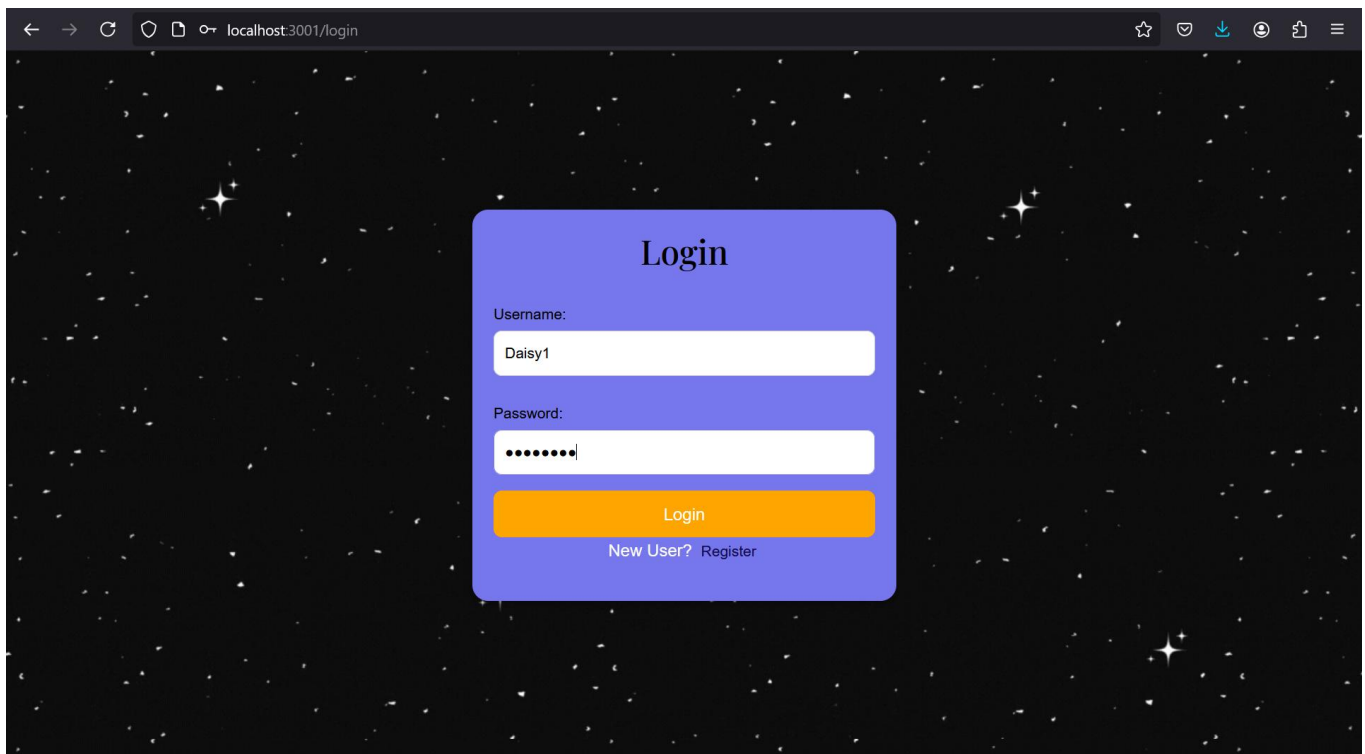
Username:  
Disha\_R

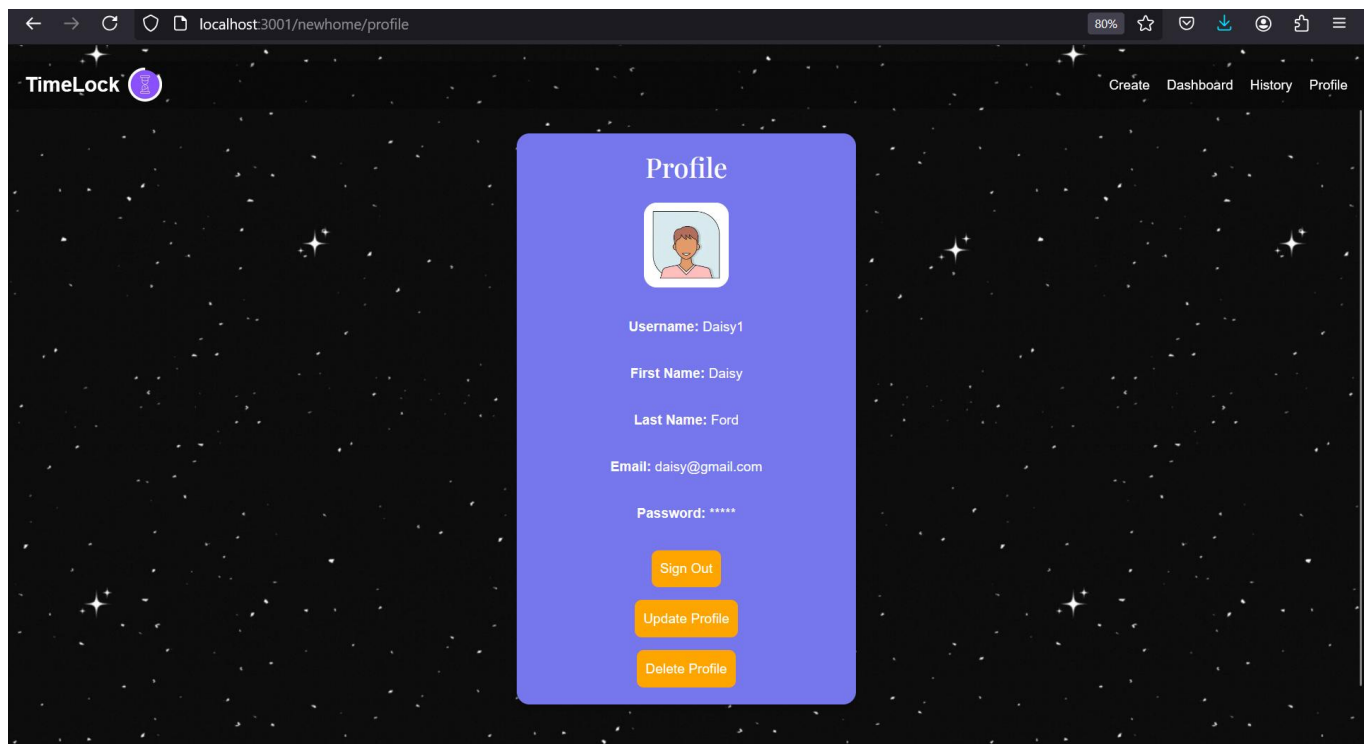
Register

Already have an account? [Login](#)

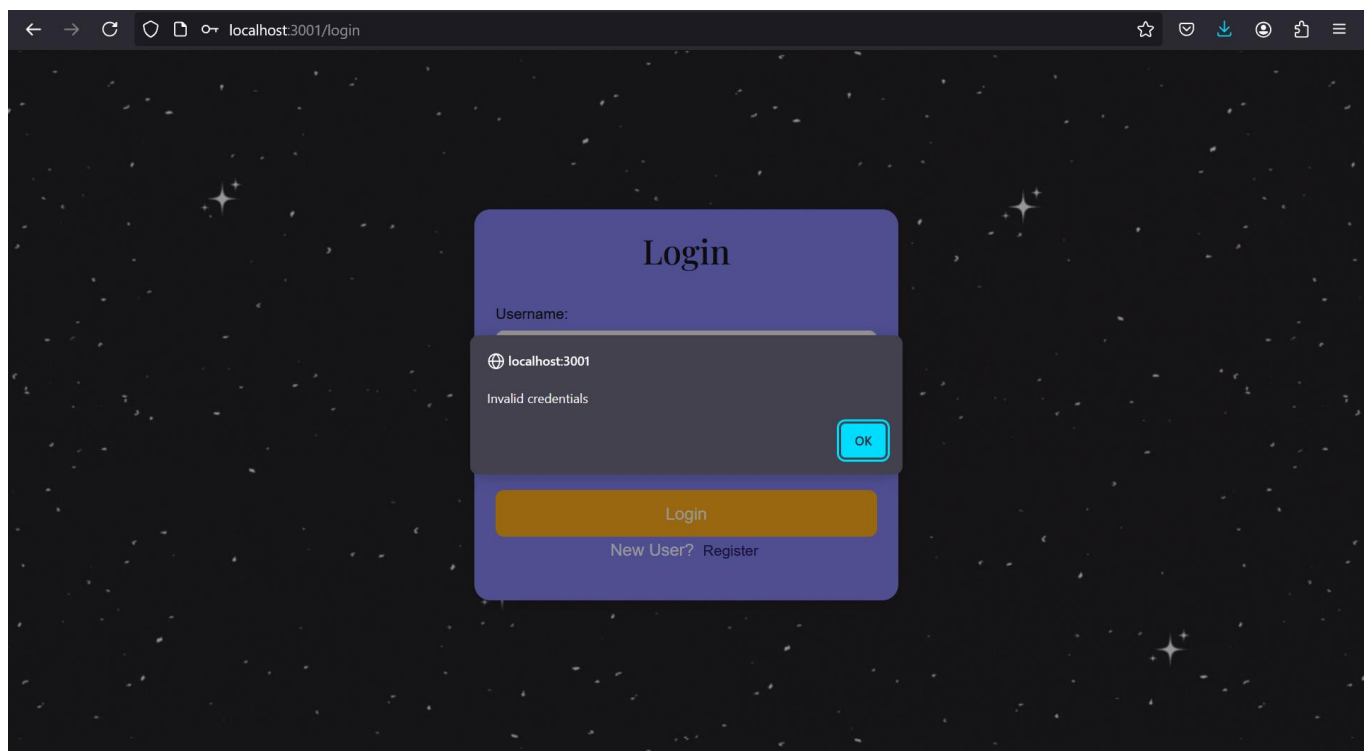


## Test Case ID: UT-07





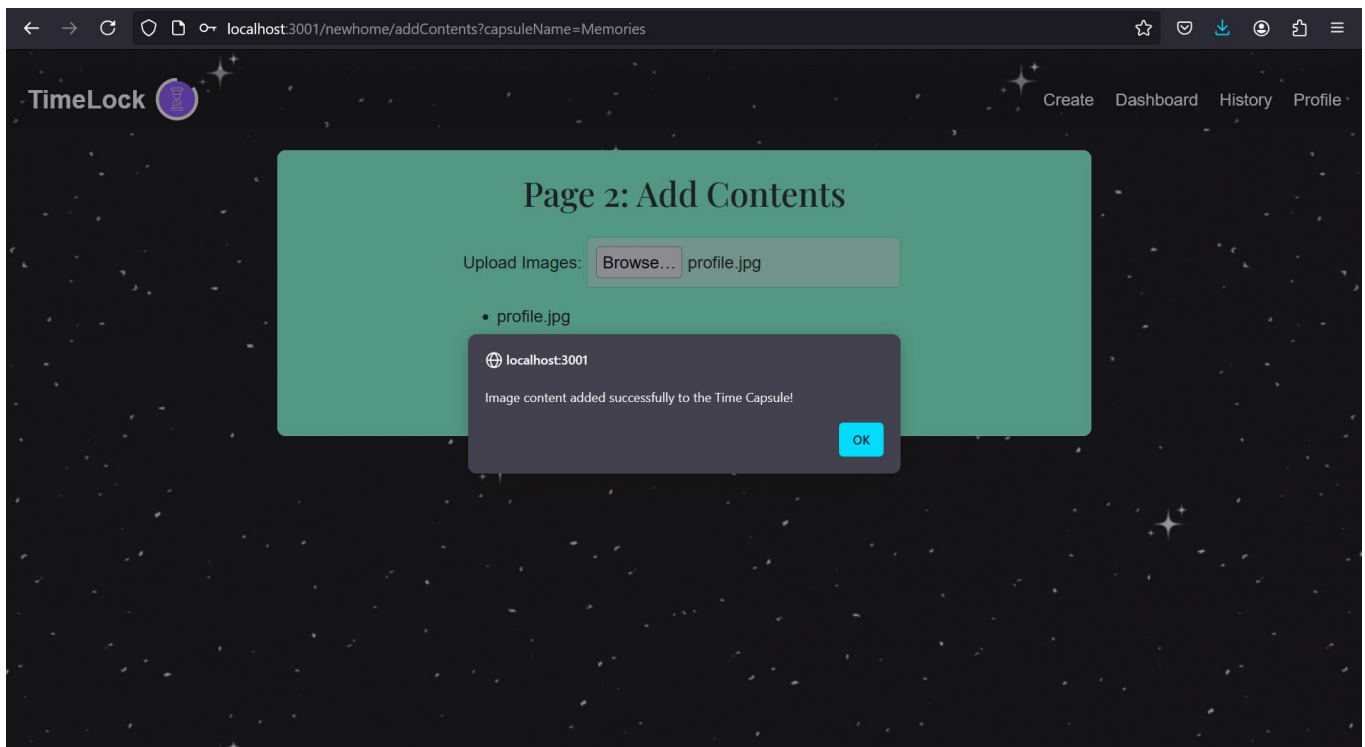
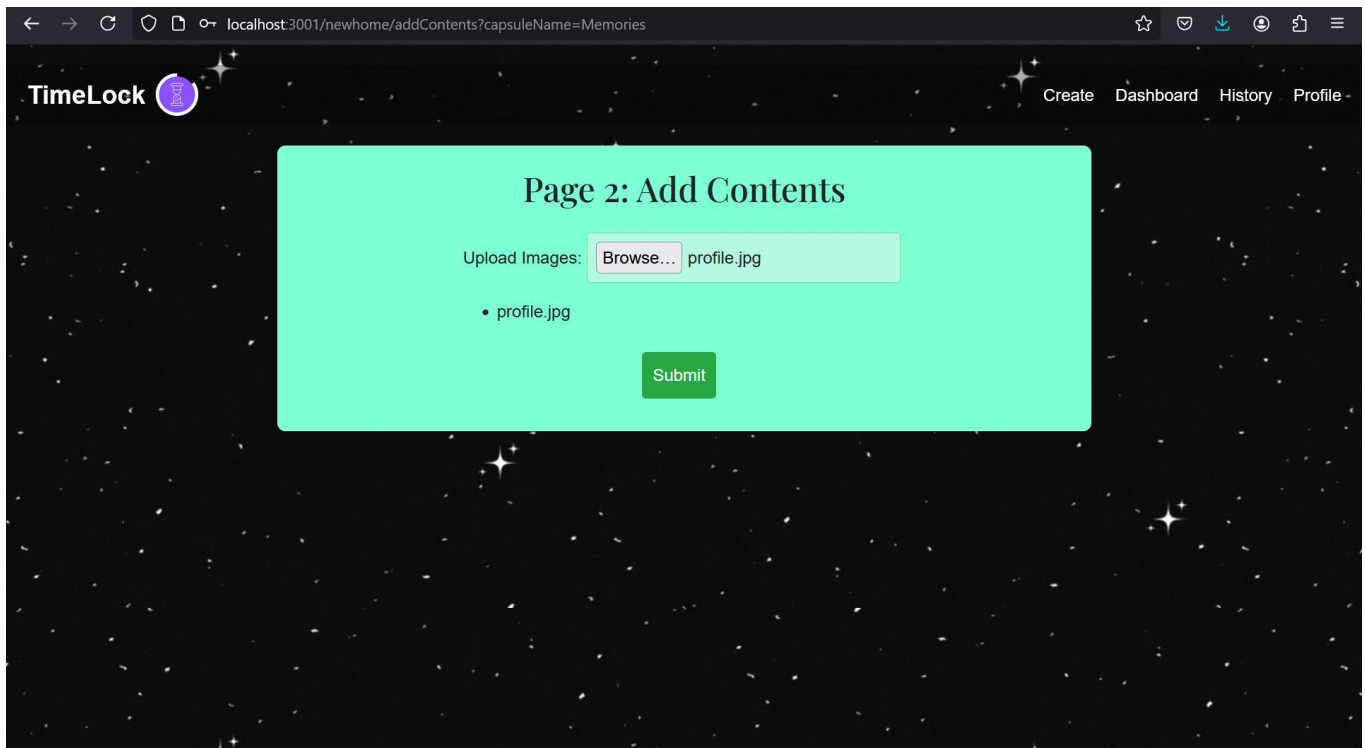
Test Case ID: UT-08 & Test Case ID: UT-09

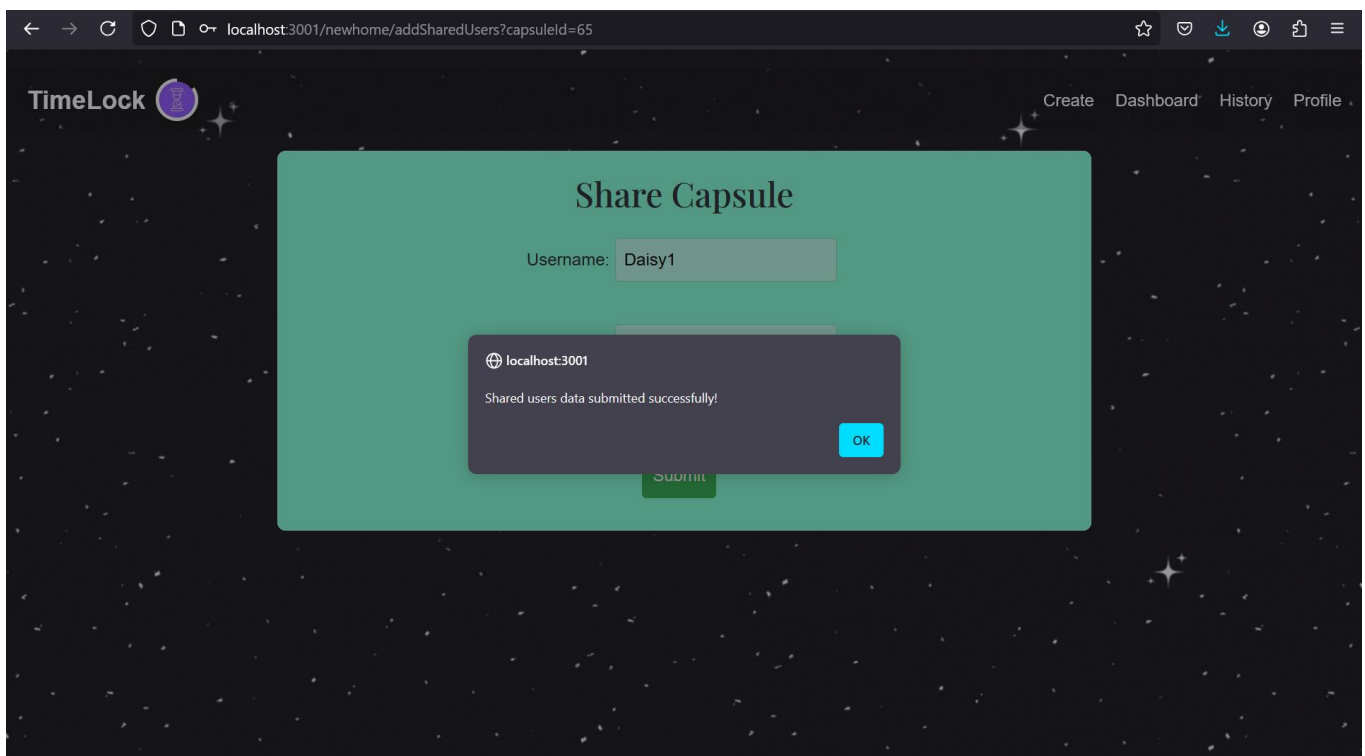
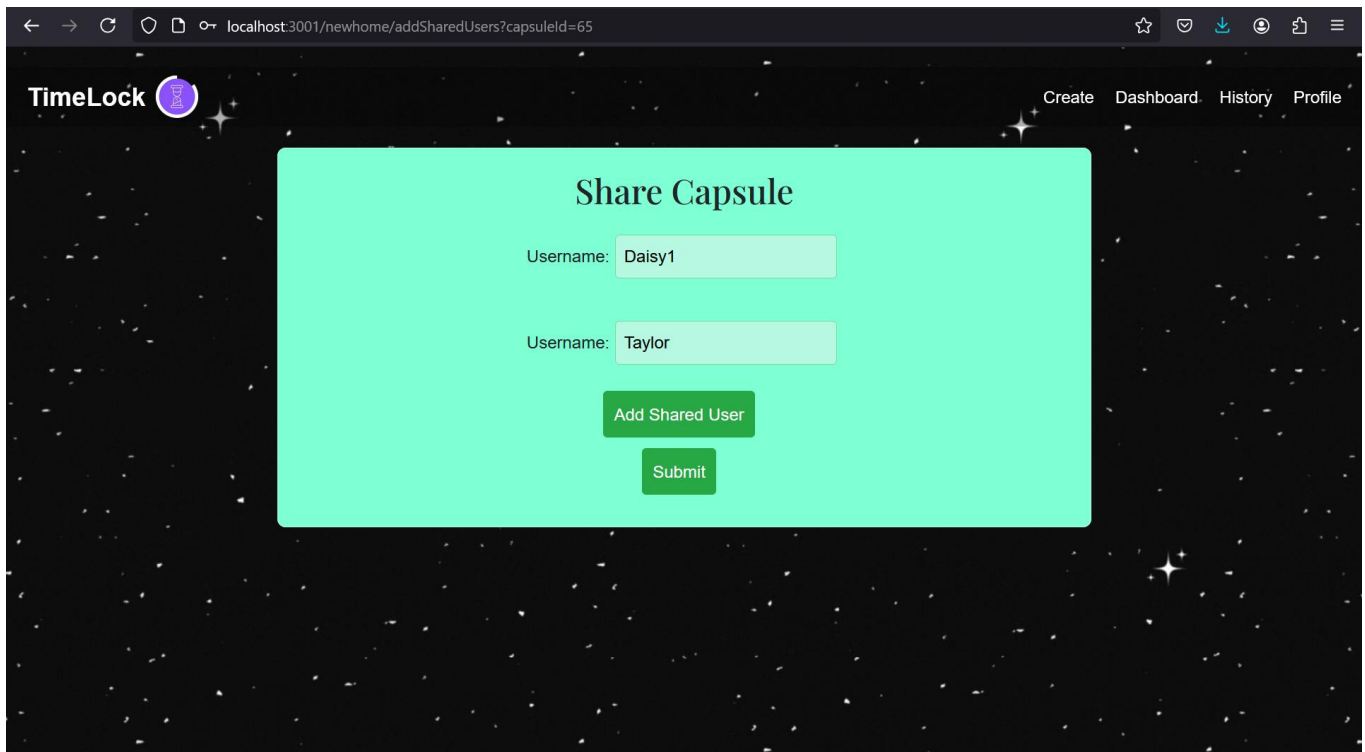


## Test Case ID: UT-10


The screenshot shows a web browser window with the URL `localhost:3001/newhome/createCapsuleForm`. The page has a dark space-themed background with stars. At the top left is the 'TimeLock' logo with a purple circular icon. At the top right are navigation links: 'Create', 'Dashboard', 'History', and 'Profile'. The main content is a light blue rectangular form titled 'Create Capsule'. Inside the form, there are three input fields: 'Capsule Name' with the text 'Memories', 'Release Date' with the date '01/01/2025' and a calendar icon, and 'Release Time' with the time '10:10 am'. At the bottom of the form is a green button labeled 'Create Capsule'.

This screenshot shows the same 'Create Capsule' form as above, but with a success message overlay. The message box is dark gray with a white border and contains the text 'localhost:3001' and 'Time Capsule Created!'. It has a blue 'OK' button. The 'Create Capsule' button remains visible at the bottom of the form.





←→↻🔍localhost:3001/newhome/addContents?capsuleName=Memories🛡️⬇️👤📦☰

TimeLock

CreateDashboardHistoryProfile

Create Capsule

Capsule Name:

Release Date: 📅

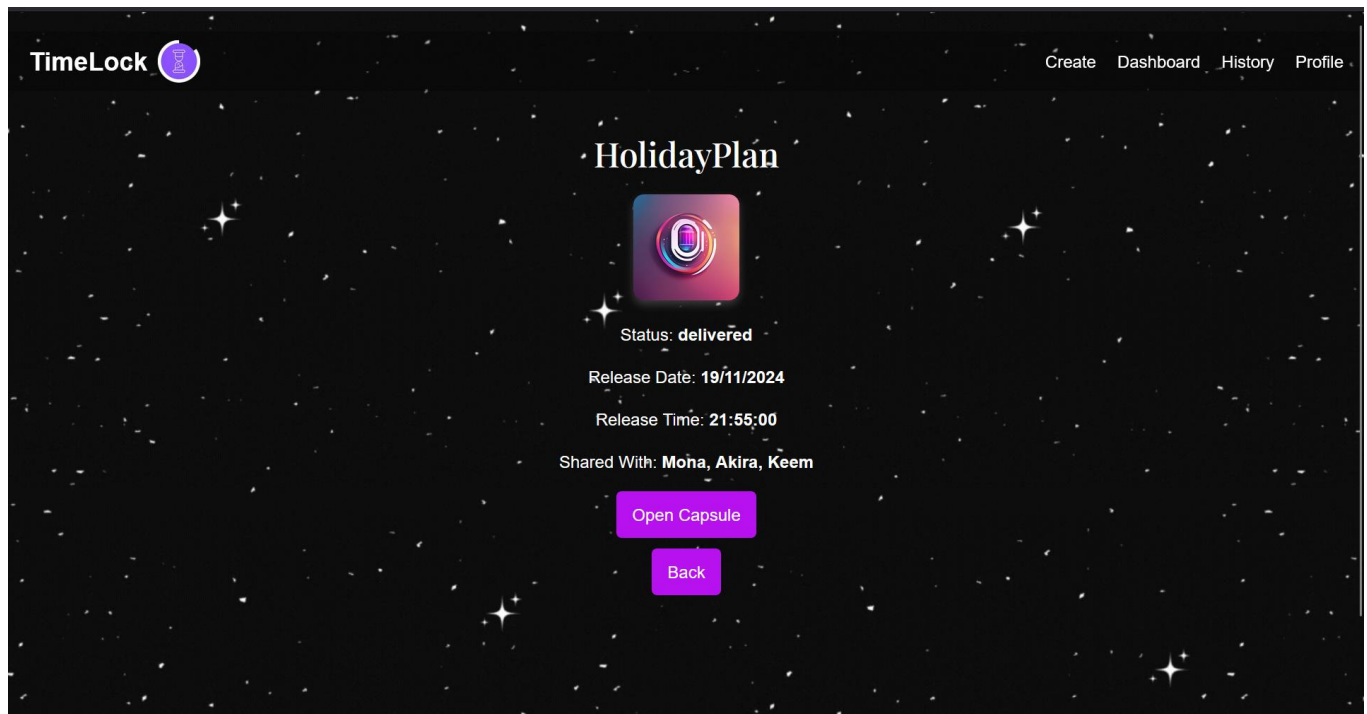
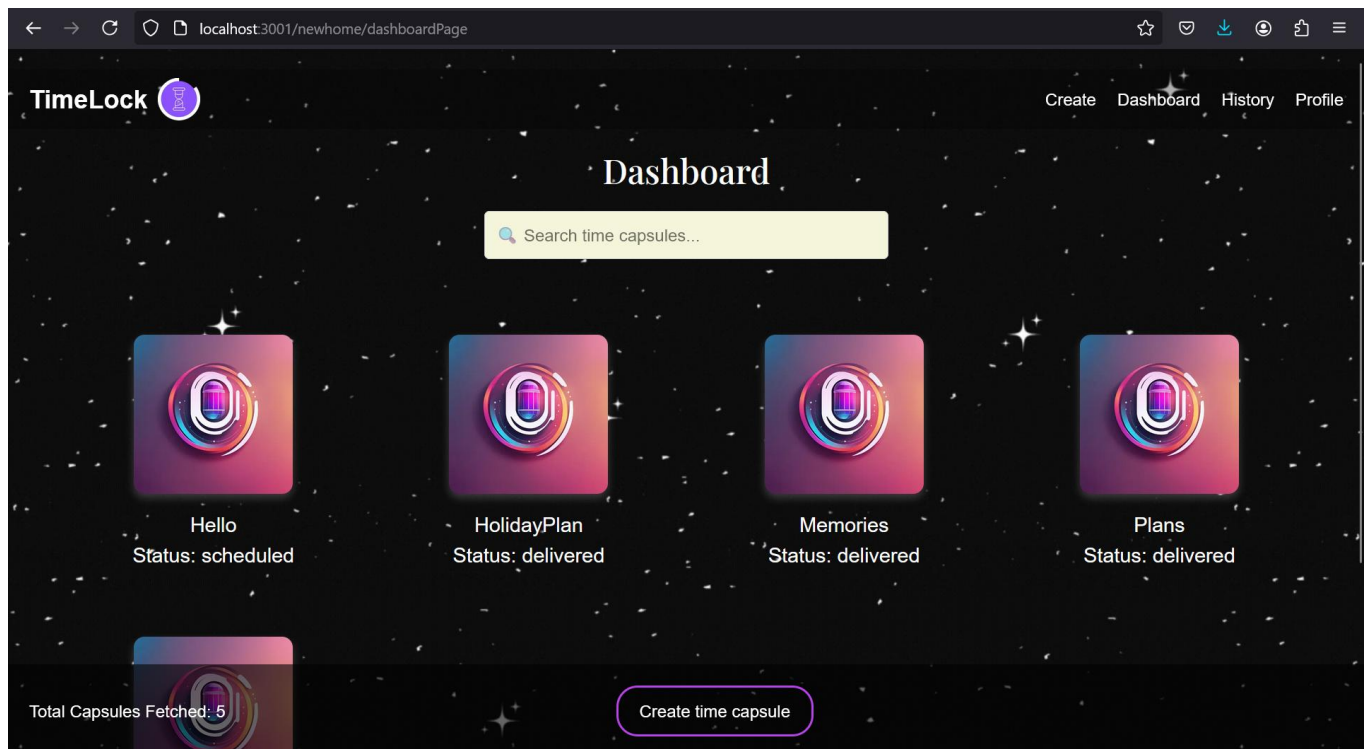
Release Time:

An error occurred while submitting the form.

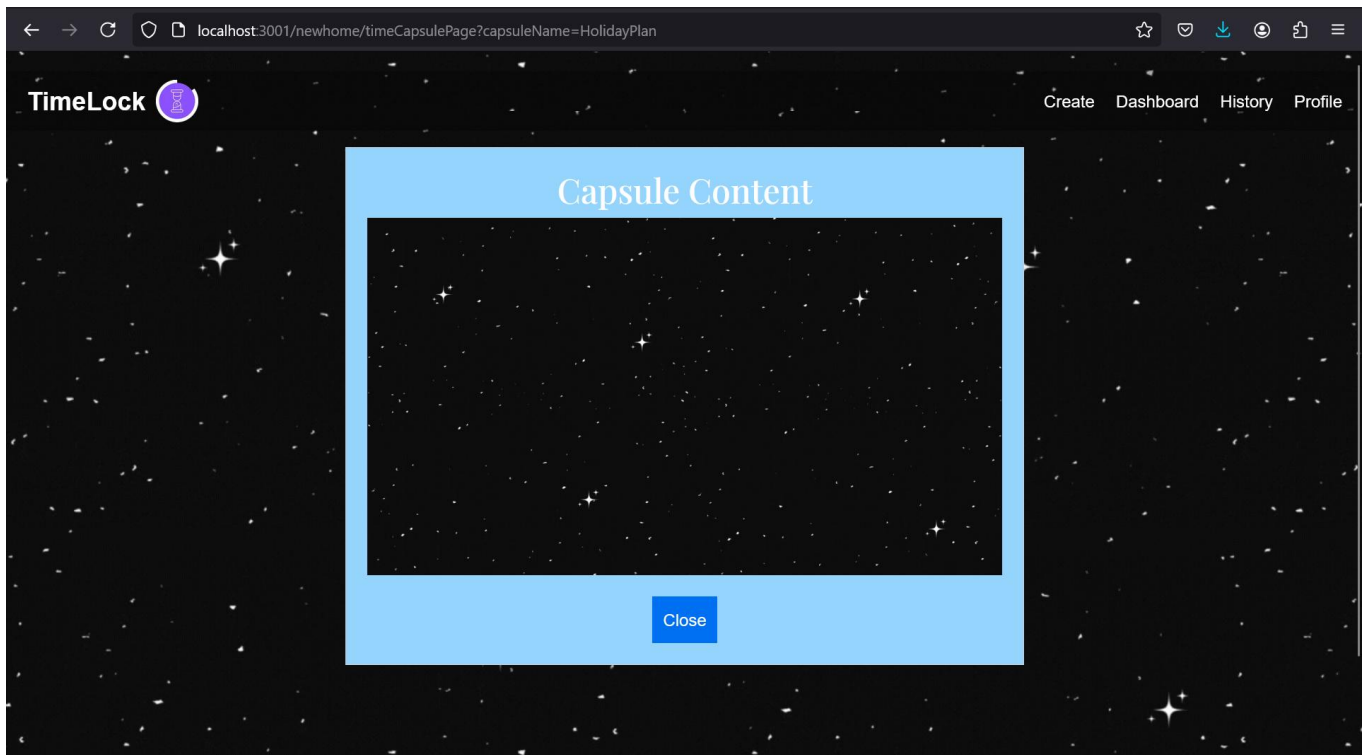
Create Capsule

[illegible]

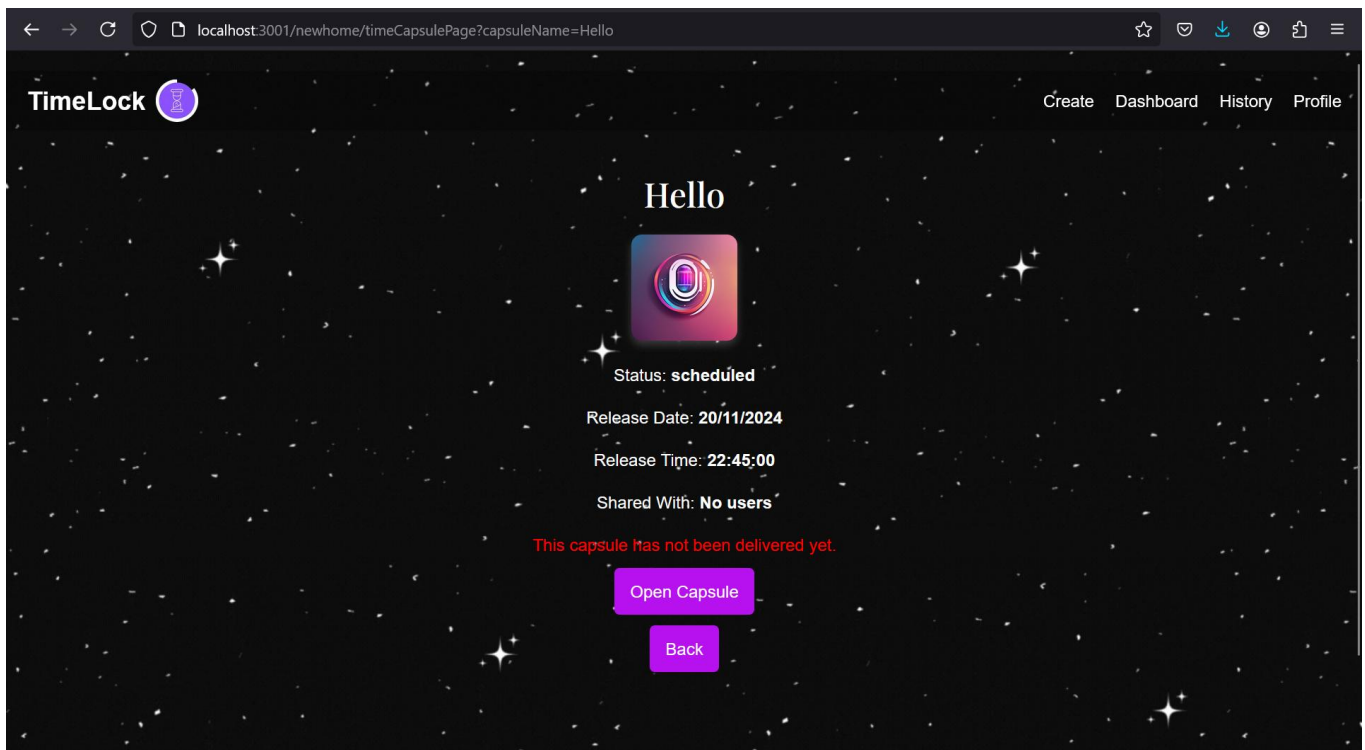
## Test Case ID: UT-20





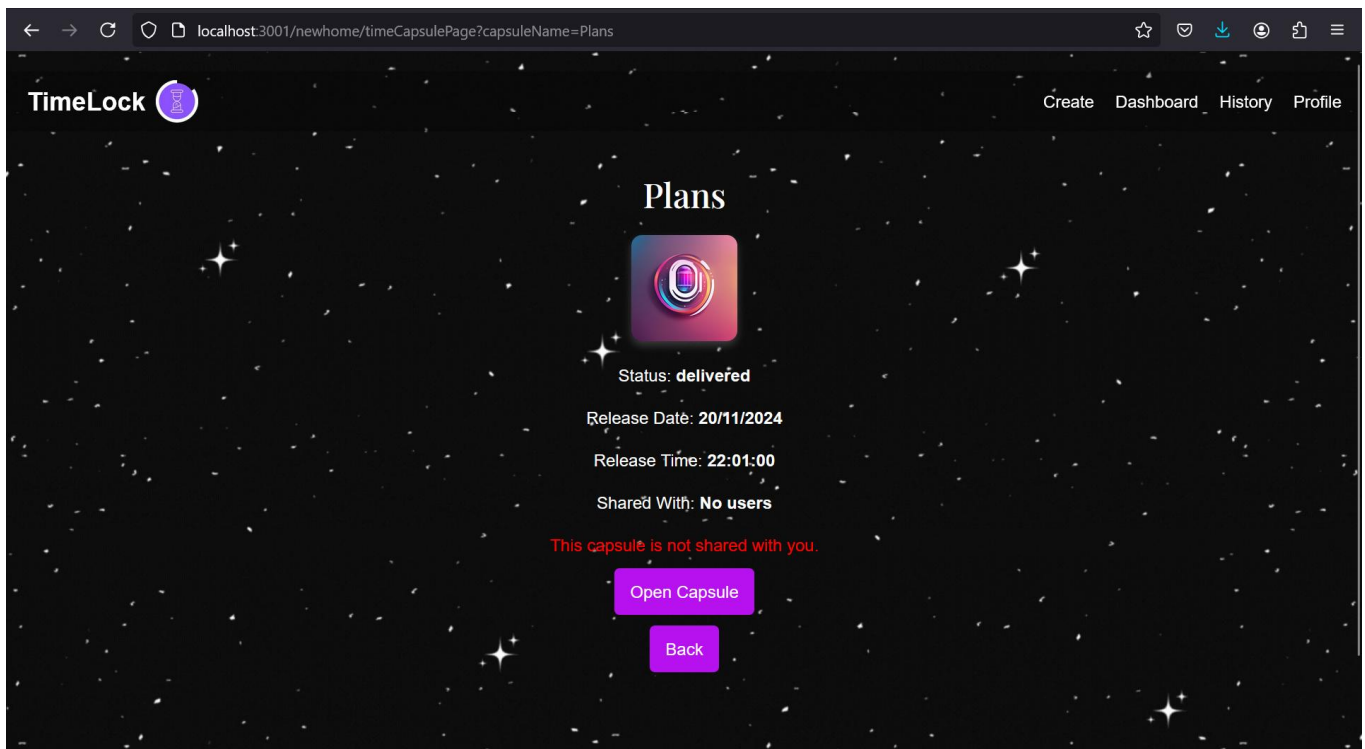


Test Case ID: UT-21

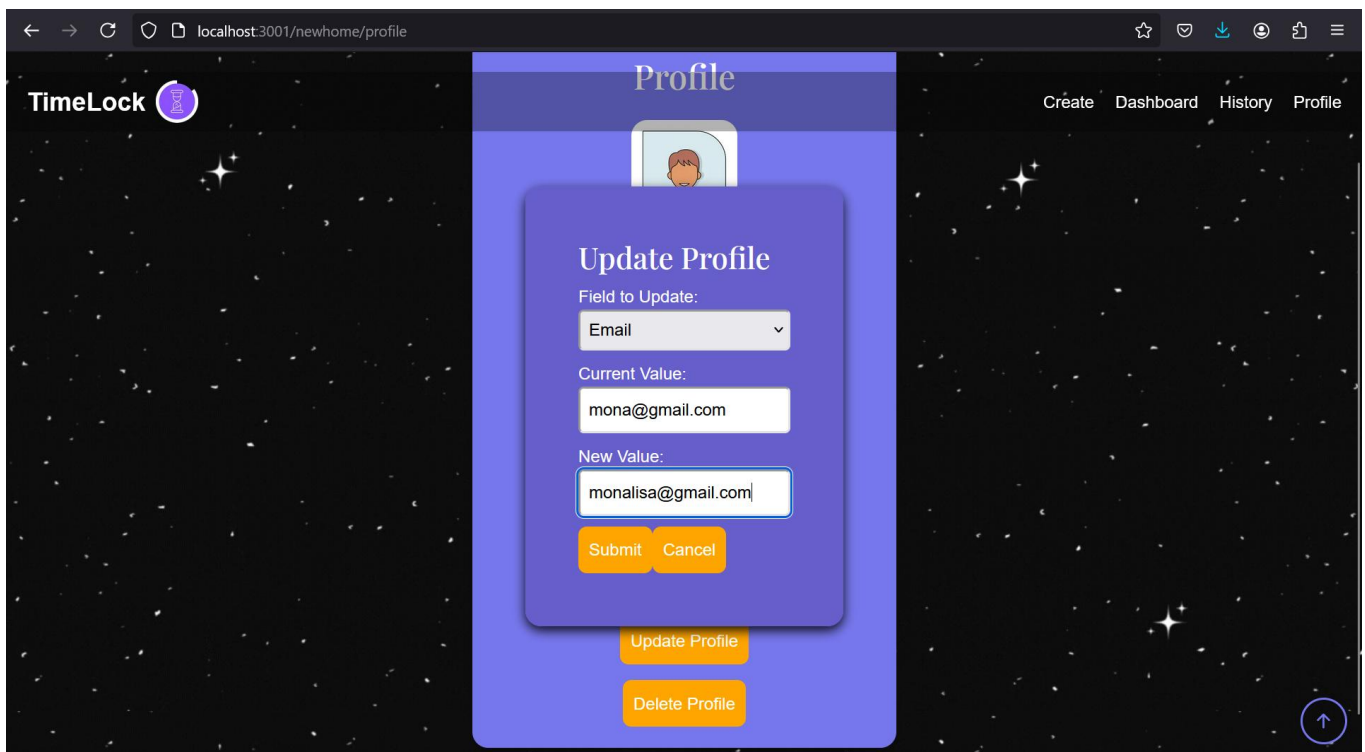


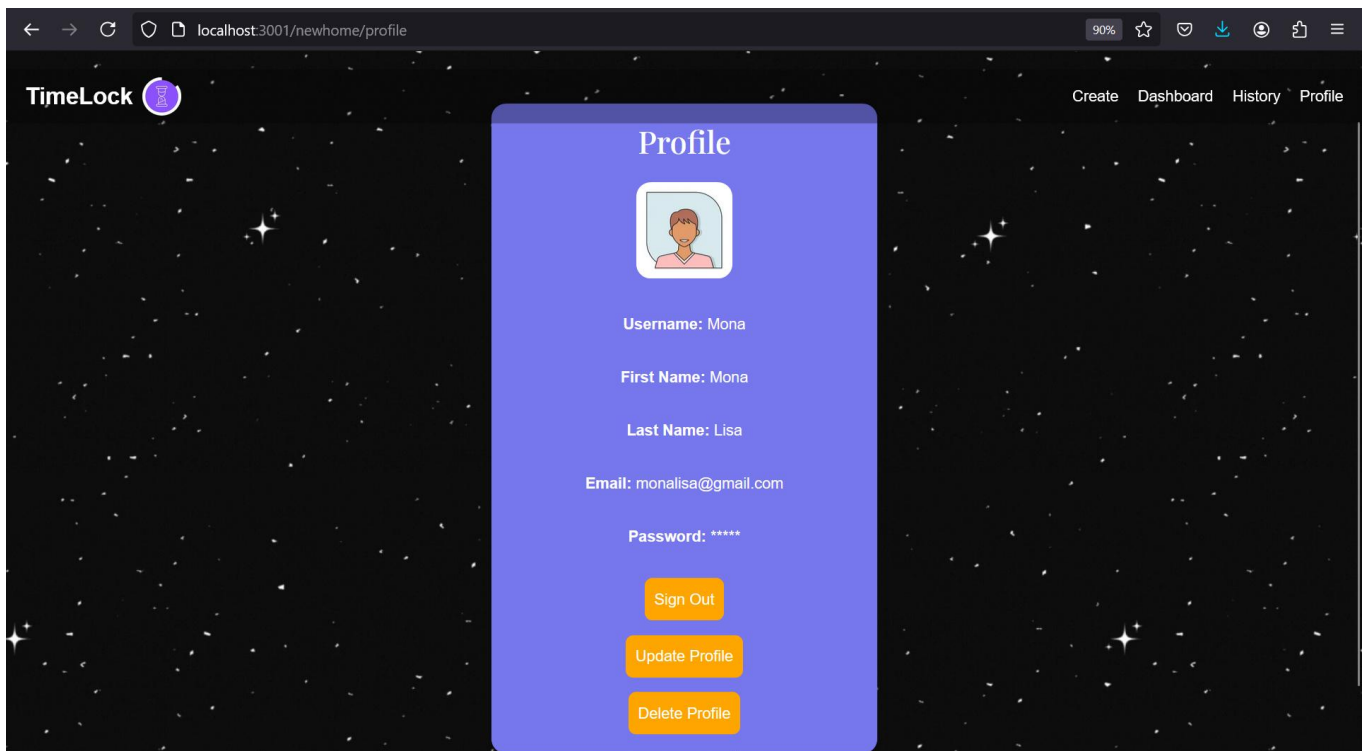
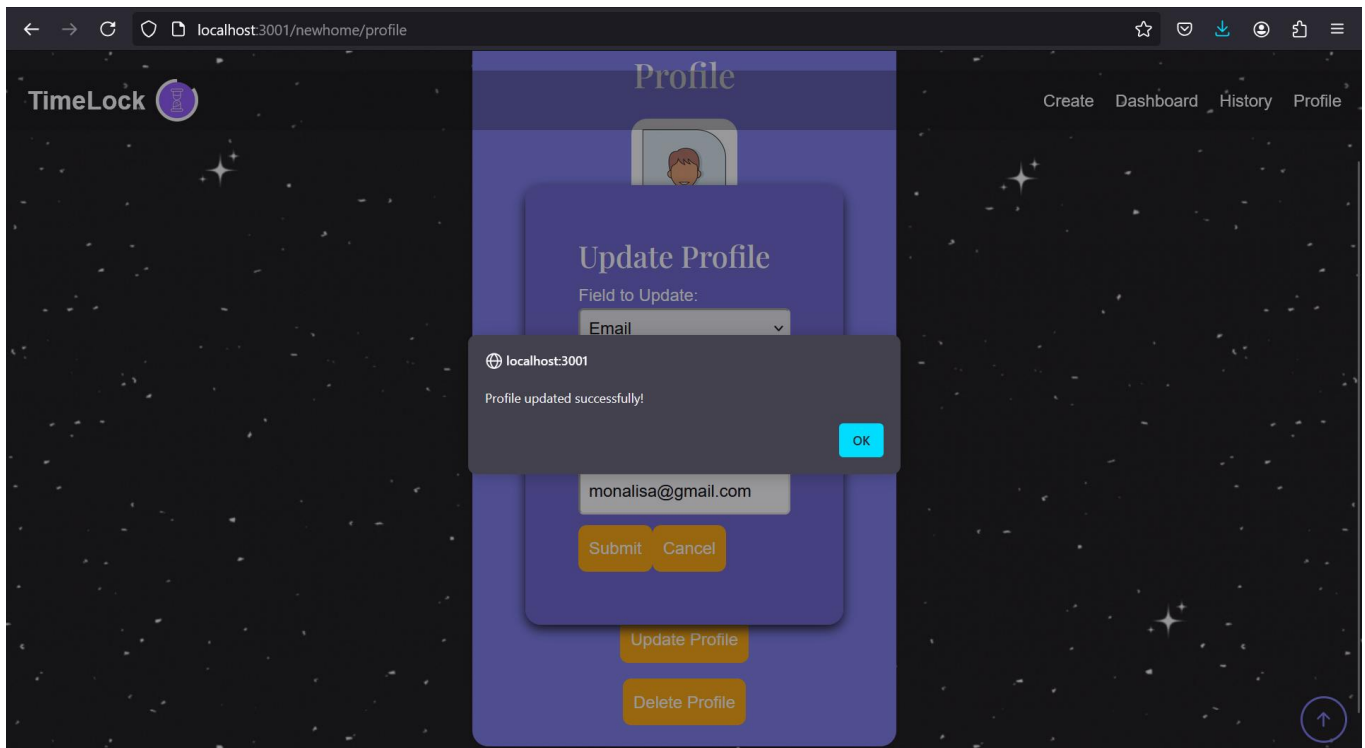


## Test Case ID: UT-22

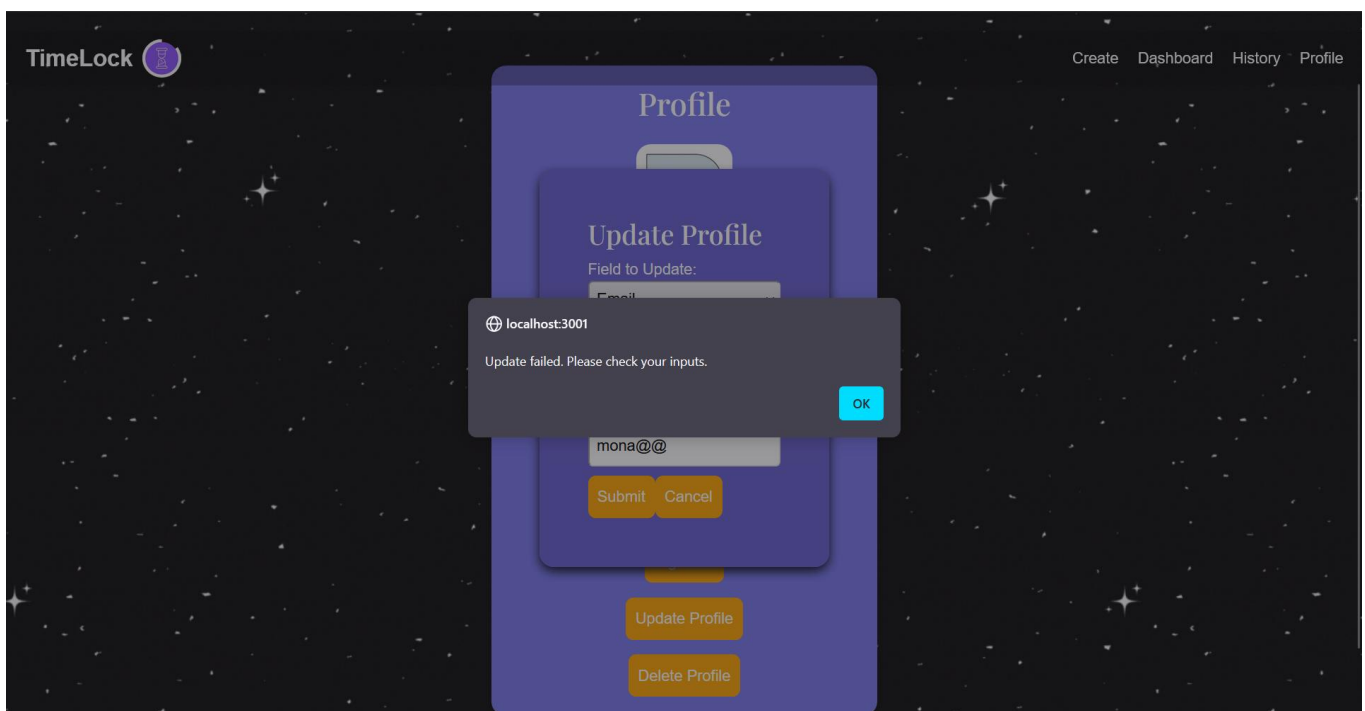
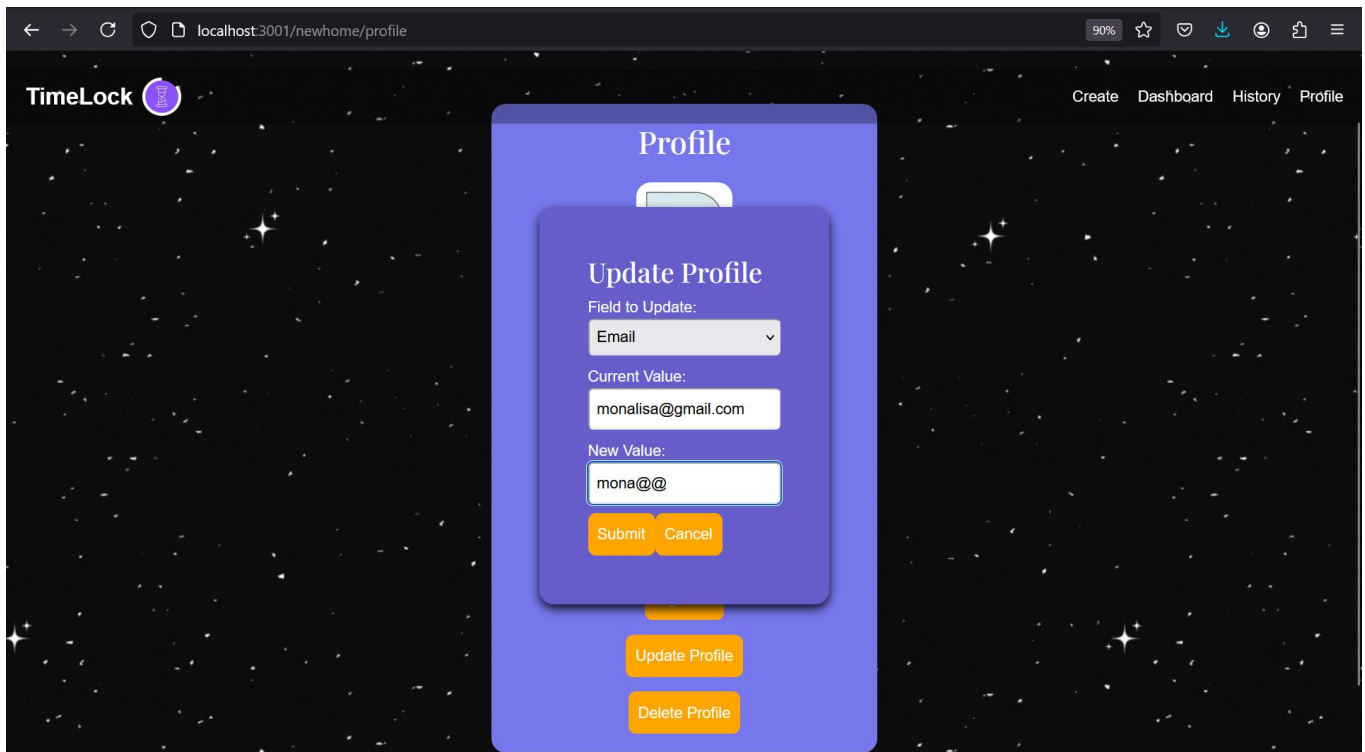


## Test Case ID: UT-24

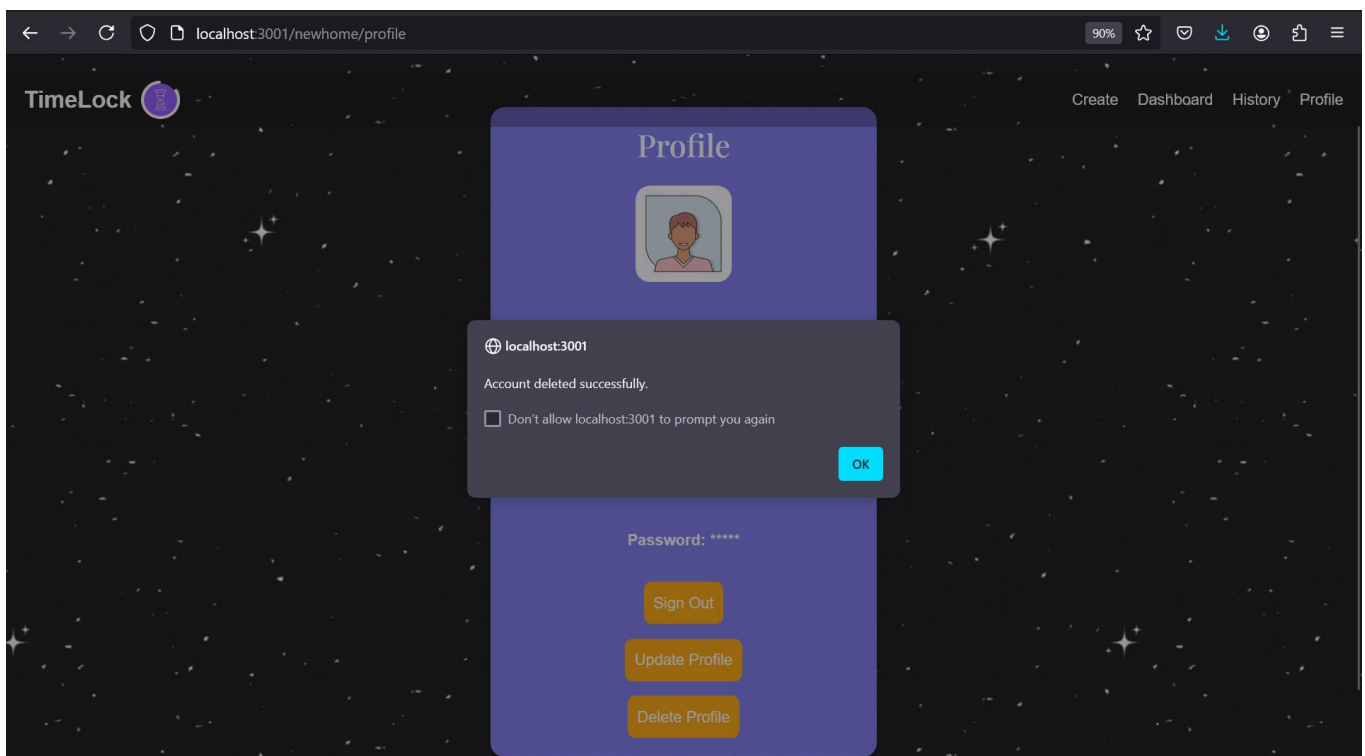
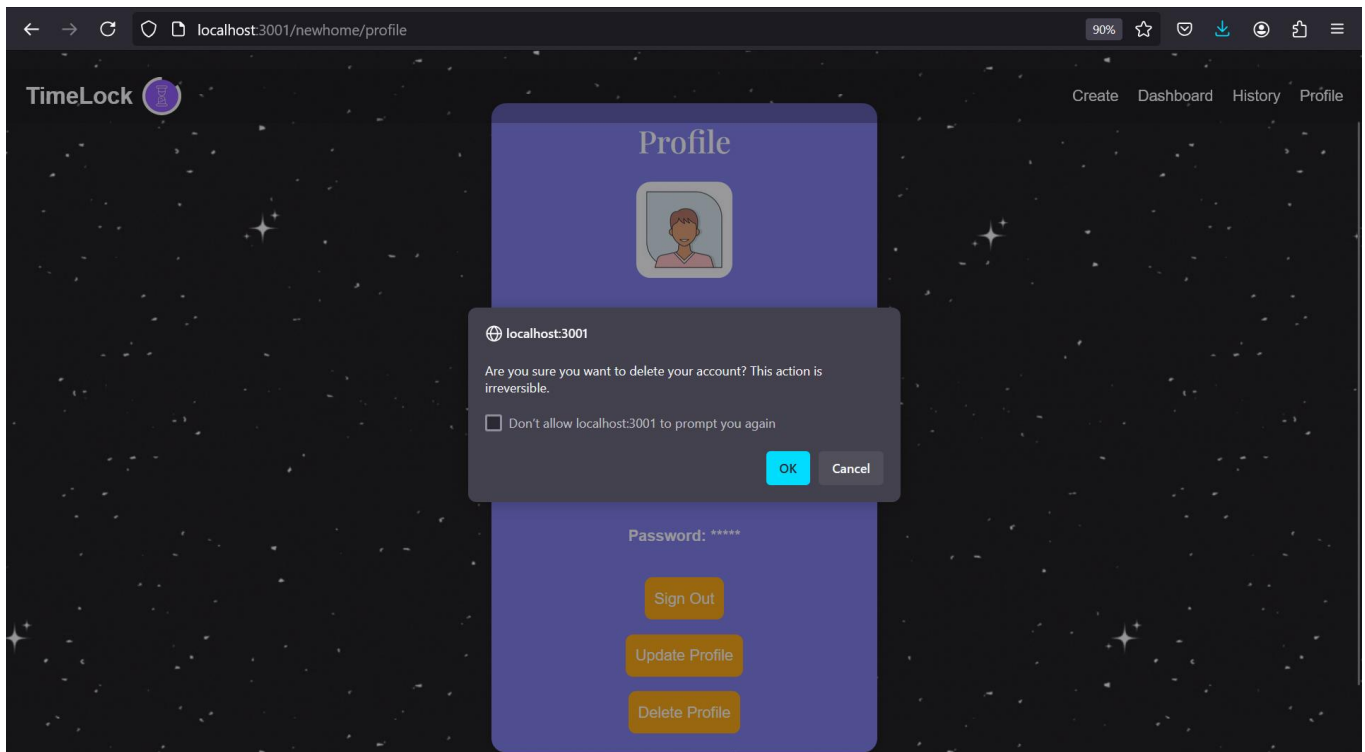




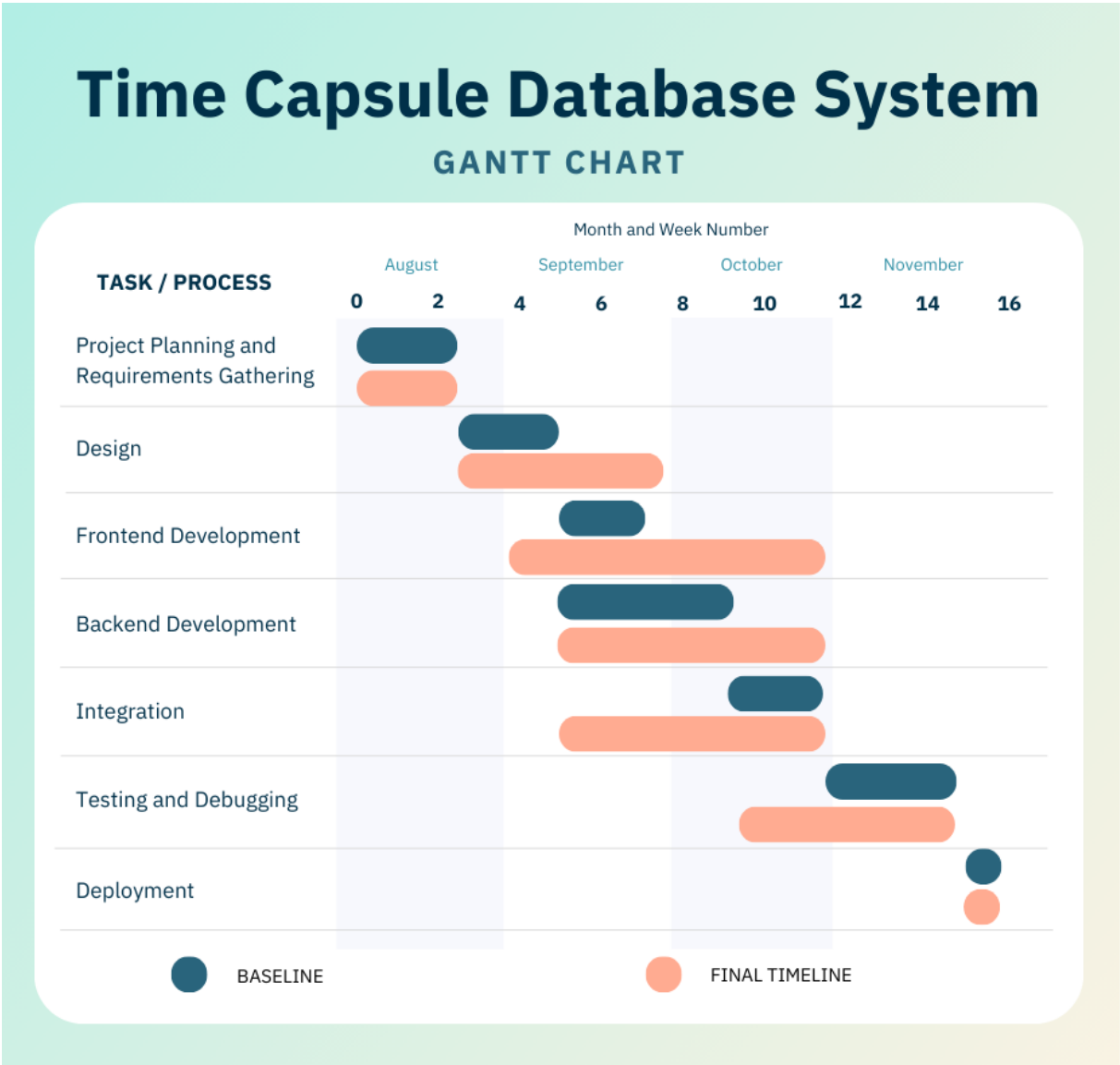
Test Case ID: UT-25



Test Case ID: UT-26



8. Final Gantt Chart (Baseline and Final Timelines)



9. Conclusions

The **Time Capsule Database System (TCDS)** successfully addresses the need for a secure, user-friendly platform to store and preserve digital memories for future retrieval. By combining strong encryption, a scalable SQL database, and an intuitive Next.js-based interface, the system ensures that users' digital content remains safe and accessible only at the designated time.

The system's incremental development approach enables efficient feedback integration and adaptability to future enhancements, such as adding support for video files or new user roles. TCDS also prioritizes performance, availability, and compliance with global data privacy laws like GDPR, making it a robust and reliable solution. Moving forward, the platform's modular architecture ensures maintainability and scalability, enabling seamless integration of new features and accommodating a growing user base.

By emphasizing security, usability, and customization, TCDS is well-positioned to serve individuals and organizations alike, ensuring that their time capsules are preserved and accessible for generations to come.

**10. Appendix A: Glossary of Abbreviations and Acronyms**

Administrator:	The technical user responsible for maintaining the system and managing encryption protocols.
AES-256:	Advanced Encryption Standard with a 256-bit key, used for securely encrypting the stored content.
Authentication:	The process by which the system verifies user identity before granting access to content.
Business User:	An organization using TCDS to store important documents and records for future use.
Decryption:	The process of converting encrypted data back into its original form, typically done at the time of release.
Encryption:	The process of encoding data in such a way that only authorized users can access it upon decryption.
GDPR:	General Data Protection Regulation, a law governing data privacy and security in the European Union.
General User:	An individual using TCDS to store personal digital content.
Next.js:	A web framework used for the front-end development of the TCDS platform.

- Release Date:** The future date set by users when the time capsule content will be made available.
- SQL:** Structured Query Language, the database system used to manage stored data.
- TCDS:** Time Capsule Database System, the project that stores and releases encrypted digital content at a future date.
- Time Capsule:** A digital collection of files, such as letters, photos, and documents, that is encrypted and set to be opened at a future date.
- User:** Any individual or organization that uses TCDS to create, manage, or retrieve time capsules.
- User Interface (UI)** The visual interface through which users interact with the TCDS platform.

## 11. Appendix B: RTM (Final version)

Sl. No	Requirement ID	Brief Description of Requirement	Architecture Reference	Design Reference	Code File Reference	Test Case ID	System Test Case ID
1	REQ-1	Users must be able to log in with their credentials (username and password).	TCDS-Authentication-V1.0	UserAuth Manager	login/page.tsx, login/route.ts	TC-01	STC-01

2	REQ-2	Users must be authorized to access only their own data and content.	TCDS-Authentication-V1.1	AccessControlModule	Profile/page.tsx, getUserProfile/route.ts	TC-02	STC-02
3	REQ-3	The system must allow users to create a time capsule with a title.	TCDS-Creation-V1.0	CapsuleCreationModule	createCapsuleForm/page.tsx, time-capsule/route.ts	TC-03	STC-03
4	REQ-4	Users can upload content (images in PNG, JPG, or GIF formats) to the time capsule.	TCDS-FileUpload-V1.0	FileUploadHandler	addContents/page.tsx, addContents/route.ts	TC-04	STC-04
5	REQ-5	The system must validate and set a future release date and time for the time capsule.	TCDS-DateValidator-V1.1	DateTimeModule	createCapsuleForm/page.tsx, time-capsule/route.ts	TC-05	STC-05



6	REQ-6	Users can add up to 3 shared users per time capsule.	TCDS-AccessControl-V1.2	SharedUserModule	addSharedUsers/page.tsx, addSharedUsers/route.ts	TC-06	STC-06
7	REQ-7	Once the capsule is sealed, it remains inaccessible until the release date and time is reached.	TCDS-Seal-V1.3	CapsuleStatusModule	timeCapsulePage/page.tsx, getTimeCapsules/route.ts	TC-07	STC-07
8	REQ-8	The system must ensure that only shared users can view the released capsule.	TCDS-AccessControl-V1.4	ViewCapsuleModule	timeCapsulePage/page.tsx, getTimeCapsules/route.ts	TC-08	STC-08
9	REQ-9	The system must prevent unauthorized	TCDS-Authentication-V1.5	SecurityManager	timeCapsulePage/page.	TC-09	STC-09

		access or modification of time capsules.			tsx, getTimeCapsules/route.ts		
10	REQ-10	The system must log actions like capsule creation, content upload, and capsule release.	TCDS-AuditLog-V1.0	AuditLog Module	History/page.tsx, audit-logs/route.ts	TC-10	STC-10

## 12. Appendix C: Technology stack and References [Books, Links to web pages/portals, tools use]

### Technology Stack

- Frontend Development:
  - Framework: Next.js
  - UI Libraries: Material-UI, Tailwind CSS, React Bootstrap
  - Design Tools: Figma, Canva
  - Language: TypeScript
- Backend Development:
  - Server Framework: Node.js
  - API Development: Express.js
  - Authentication: OAuth2.0, JWT (JSON Web Token)

- Encryption: AES-256 for data security
    - Language: TypeScript
  - Database:
    - Type: Relational Database
    - Platform: MySQL
    - Tools: SQL Workbench for database design and management
  - Testing:
    - API Testing: Postman
    - Automated Testing: Selenium
    - Unit Testing: Jest
  - Project Management:
    - Task Tracking: Trello
    - Version Control: Git
    - Repository Hosting: GitHub
  - Deployment:
    - Platform: Vercel for frontend hosting, AWS for backend services
    - CI/CD: GitHub Actions for automated testing and deployment
  - Additional Tools Used:
    - Code Editor: Visual Studio Code
    - Design and Diagramming: StarUML, Lucidchart
- 

## References

- Books:
  - "Web Development with Node and Express" by Ethan Brown

- "Modern SQL: A Hands-On Guide to Relational Databases" by Markus Winand
- Online Resources:
  - Next.js Documentation: <https://nextjs.org/docs>
  - Node.js Official Website: <https://nodejs.org/>
  - MySQL Documentation: <https://dev.mysql.com/doc/>
  - Material-UI Documentation: <https://mui.com/material-ui/getting-started/>
- Tools Documentation:
  - Figma: <https://www.figma.com/>
  - Canva: <https://www.canva.com/>
  - Lucidchart: <https://www.lucidchart.com/>
  - StarUML: <https://staruml.io/>
  - Postman: <https://www.postman.com/>
  - Selenium: <https://www.selenium.dev/>