Reading Tracker System

Software Requirements Specification

INT222
ADVANCED WEB DEVELOPMENT

Jaspreet-12305973

Prepared for
Continuous Assessment 2
Spring 2025

Table of Contents

1. Introduction

1.1 Purpose

*The purpose of this Software Requirements Specification (SRS) document is to provide a detailed description of the software requirements for the **"Story Verse"** project. This document illustrates the intended purpose, features, and behaviour of the system.*
This SRS is intended for the following audience:

- ***Project Developers (Jaspreet):** To guide the implementation of the MERN stack architecture (MongoDB, Express, React, Node.js).*
- ***Project Evaluators/Instructors:** To assess the scope, complexity, and functionality of the Continuous Assessment 3 project.*
- ***End Users:** To understand the capabilities and limitations of the reading tracker system.*

1.2 Scope

The software product to be produced is named "Story Verse - The Grand Library".

***Product Description:** Story Verse is a web-based Reading Tracker System designed to function as a digital sanctuary for book lovers. Unlike standard e-readers or spreadsheet trackers, Story Verse combines the utility of a database with the immersive aesthetic of a "Dark Academia" library.*

The software will do the following:

- ***Digital Archiving:** Allow users to maintain a comprehensive log of books they have read, are currently reading, or wish to read.*
- ***Interactive Tracking:** Provide visual status indicators (e.g., "Reading," "Finished") and an interactive 5-star rating system.*
- ***Immersive Reading:** Simulate the physical experience of reading a novel through a digital "Flipbook" interface with realistic page-turning animations and side-navigation buttons.*
- ***Communication:** Facilitate user feedback and book requests through a robust "Contact Owl" messaging system.*

The software will NOT:

- *Facilitate e-commerce transactions for buying physical books.*
- *Provide social networking features (e.g., adding friends or chat rooms) in this version.*

***Application & Goals:** The primary goal of Story Verse is to "Curate, Chronicle, and Cherish" a user's literary legacy. The objective is to provide a distraction-free, aesthetically pleasing*

environment that encourages active reading retention through review writing and progress tracking.

1.3 Definitions, Acronyms, and Abbreviations

This subsection provides the definitions of all terms, acronyms, and abbreviations required to properly interpret the SRS.

- ***SRS:*** *Software Requirements Specification.*
- ***MERN:*** *A technology stack comprising MongoDB (Database), Express.js (Backend Framework), React.js (Frontend Library), and Node.js (Runtime Environment).*
- ***API:*** *Application Programming Interface; used here to describe the communication between the React frontend and Node backend.*
- ***DOM:*** *Document Object Model.*
- ***Flipbook:*** *A UI component that simulates the physical turning of book pages.*
- ***CRUD:*** *Create, Read, Update, Delete (The four basic functions of persistent storage).*
- ***GUI:*** *Graphical User Interface.*
- ***Backend:*** *The server-side logic (Node/Express) that processes data.*
- ***Frontend:*** *The client-side interface (React) that the user interacts with.*

1.4 References

This subsection lists the documents and sources referenced in this SRS.

1. *IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications, IEEE Computer Society, 1998.*
2. *Story Verse Project Documentation, GitHub Repository, 2025.*
3. *React.js Official Documentation (react.dev).*
4. *MongoDB Atlas Documentation (mongodb.com/docs).*
5. *Course Syllabus for INT-222, Spring 2025.*

1.5 Overview

This SRS is organized into the following remaining sections:

- ***Section 2 (General Description):*** *Describes the general factors that affect the product and its requirements, including user characteristics and general constraints.*

- ***Section 3 (Specific Requirements):*** *Detailed functional and non-functional requirements, including external interfaces (User, Hardware, Software) and specific functional logic for features like the "Book Reader" and "Tracker".*

- ***Section 4 (Analysis Models):*** *Visual representations of the system, including Data Flow Diagrams (DFD).*

- ***Appendices:*** *Supporting proofs, including GitHub links, deployment links, and client validations.*

2. General Description

2.1 Product Perspective

Story Verse is a self-contained, full-stack web application designed to replace manual reading logs or generic spreadsheet trackers. It functions as a complete Reading Tracker System.

- ***System Architecture:*** *The product operates on a client-server model. The frontend (Client) is a Single Page Application (SPA) built with React.js, which ensures a smooth, non-reloading user experience. The backend (Server) utilizes Node.js and Express to process API requests, while MongoDB acts as the persistent storage layer for user data and book archives.*

- ***Relationship to Other Systems:*** *Unlike standalone e-readers (like Kindle hardware), Story Verse is browser-based and accessible from any device with internet connectivity. It relies on external libraries such as react-page flip for its core visual simulation features.*

- ***User Interface Perspective:*** *The product prioritizes a "Dark Academia" aesthetic, utilizing specific colour palettes (Deep Wood, Royal Gold) and typography (Cinzel, Crimson Text)*

to simulate the atmosphere of a physical library, distinguishing it from standard, minimalist utility apps.

2.2 Product Functions

The software will perform the following major functions:

1. ***User Authentication:***
    - *Secure Login and Sign-up functionality to protect user archives.*
    - *Session management to keep users logged in across pages.*
2. ***Library Visualization ("The Grand Library"):***
    - *Display a grid of book covers fetched dynamically from the database.*
    - *Visual status indicators (Badges) showing if a book is "Reading," "Read," or "Want to Read."*
3. ***Immersive Reading Simulation:***
    - *Flipbook Engine: A digital reader that simulates realistic page-turning physics.*
    - *Navigation Control: Custom "Next" and "Previous" side buttons for intuitive page flipping.*
4. ***Progress Tracking & Reviewing:***
    - *Status Management: Ability to update the current reading status of any book.*
    - *Star Rating System: An interactive 1-to-5-star rating component that updates instantly.*
    - *Personal Notes: A text area for users to write and save reviews or thoughts on specific books.*
5. ***System Communication ("Contact Owl"):***
    - *A structured contact form allowing users to report bugs (e.g., "The owls are on strike") or request new book additions.*
    - *Feedback mechanisms (Success/Error alerts) to inform users of message delivery status.*

2.3 User Characteristics

The intended users of Story Verse fall into the following categories:

- ***The Avid Reader:***

  - ***Characteristics:*** *Reads multiple books a month, enjoys organizing collections, and values aesthetics.*
  - ***Expectations:*** *Wants a visually pleasing interface and an easy way to track what they have finished.*
- ***The Student / Researcher:***
  - ***Characteristics:*** *Needs to track reading lists for academic purposes.*
  - ***Expectations:*** *Requires reliable data storage for notes and reviews to assist with studies.*
- ***The Administrator (Curator):***
  - ***Characteristics:*** *Has technical access to the backend/database.*
  - ***Expectations:*** *Needs the system to accurately capture user inputs (like new book requests) via the contact form.*
- ***Technical Expertise:*** *Users are expected to have basic computer literacy and familiarity with web browsers (Chrome, Edge, Firefox). No programming knowledge is required to use the frontend.*

2.4 General Constraints

The following items limit the developer's options for designing the system:

1. ***Technology Stack:*** *The project must be developed strictly using the **MERN Stack** (MongoDB, Express, React, Node.js) as per the course curriculum requirements.*
2. ***Hardware Limitations:*** *The application is web-based; therefore, it is constrained by the processing power of the user's device regarding the rendering of 3D flipbook animations.*
3. ***Connectivity:*** *The system requires an active internet connection to fetch book data from the MongoDB cloud database; offline functionality is not currently supported.*
4. ***Development Time:*** *The project must be completed and deployed by the **Continuous Assessment 3** deadline.*
5. ***Design Language:*** *The UI is constrained to a specific "Dark/Vintage" theme, limiting the use of standard bright/modern UI component libraries (like Bootstrap default styles).*

2.5 Assumptions and Dependencies

The requirements stated in this SRS depend on the following factors:

- ***Dependency:*** *The system relies on the **react-page flip** third-party library. It is assumed this library will remain compatible with the current version of React.*
- ***Dependency:*** *The system relies on **MongoDB Atlas** (Cloud) for database hosting. Any downtime on Atlas will affect the application's availability.*
- ***Assumption:*** *It is assumed that the client device has a modern web browser installed that supports **ES6 JavaScript** and **CSS Grid/Flexbox**.*
- ***Assumption:*** *It is assumed that the user will provide valid email formats during the Sign-up and Contact processes.*

3. Specific Requirements

This section contains all the software requirements at a level of detail sufficient to enable designers to design the system and testers to test the system.

3.1 External Interface Requirements

3.1.1 User Interfaces: The user interface (UI) is designed with a "Dark Academia" aesthetic to provide an immersive reading environment.

- **Login/Landing Screen:** A centralized login card set against a full-screen video background of a cozy library with rain effects.
- **My Library (Home):** A responsive grid layout displaying book covers. Each card features the book title, author, and a color-coded status pill (e.g., Orange for "Reading").
- **The Reader:** A full-screen interface simulating an open book. It features parchment-textured pages, "Drop Cap" typography for the first letter, and custom floating "Next/Previous" navigation buttons on the left and right sides.
- **Book Details Dossier:** A two-column layout. The left column displays the book cover and "Open" button; the right column contains metadata and the interactive tracker form (Status dropdown, Star Rating, Review text area).
- **Contact Panel:** A glass-morphism style form with input fields for Name, Email, Subject, and Message, accompanied by instructional icons.

3.1.2 Hardware Interfaces:

 Client Side: The application is browser-based and requires a device (Desktop, Laptop, Tablet) with a minimum screen resolution of 1280x720 to properly render the Flipbook animations. A pointing device (Mouse/Trackpad) or touch interface is required for navigation.
 Server Side: The backend requires a standard web server configuration capable of running **Node.js** (v14 or higher).

3.1.3 Software Interfaces:

 Operating System: The application is platform-independent and runs on any OS (Windows, macOS, Linux, iOS, Android) that supports modern web browsers.
 Database: The system interfaces with **MongoDB** (Atlas Cloud or Local) for storing User and Book collections.
 Libraries:

- **React.js:** For building the component-based UI.
- **Axios:** For handling HTTP requests between Client and Server.
- **React-PageFlip:** For the book turning physics engine.

3.1.4 Communications Interfaces:

 Protocol: The system uses **HTTP/HTTPS** for all client-server communication.
 Data Format: All data is exchanged in **JSON** (JavaScript Object Notation) format.
 API Endpoints: The frontend communicates with specific backend REST endpoints (e.g., GET /library, POST /contact, PUT /update-book/:id).

3.2 Functional Requirements

This section outlines the specific behaviors and functions of the Story Verse system .

- **3.2.1 Functional Requirement #1:** Library Management System This requirement covers the secure access and visualization of the digital book archive.
- **3.2.1.1 Introduction :**The system allows users to securely log in and view their personal collection of books in a dynamic grid layout.
- **3.2.1.2 Inputs**
- **Login:** Email Address, Password.
- **Navigation:** User clicks on "My Library" or specific book cards.
- **3.2.1.3 Processing**
- **Auth:** The system validates credentials against the MongoDB user collection using hashed passwords.
- **Fetching:** Upon loading the Home page, the useEffect hook triggers an Axios GET request to retrieve the book list.
- **Rendering:** The system maps through the data array to display BookCard components with dynamic status badges (Read, Reading, etc.).
- **3.2.1.4 Outputs**
- A personalized dashboard displaying the user's books.
- Access tokens stored in the browser's local storage for session management.
- **3.2.1.5 Error Handling**
- Invalid login attempts return an alert: "Invalid Credentials."
- Database connection failures trigger a "Loading Archives..." placeholder state.
- **3.2.2 Functional Requirement #2:** Reader & Tracking Engine This requirement covers the immersive reading experience and the logic for tracking user progress and feedback.
- **3.2.2.1 Introduction**: The system provides a "Flipbook" interface for reading and a tracking form for updating reading status, ratings, and sending inquiries.
- **3.2.2.2 Inputs**
- **Reader:** Clicks on "Next/Prev" buttons.
- **Tracker:** Status selection (Dropdown), Star Rating (1-5 clicks), Review (Text).
- **Contact:** Name, Email, and Message text.

- **3.2.2.3 Processing**
- **Animation:** The react-pageflip library, controlled by React refs, calculates 3D page turns based on button clicks.
- **Updates:** Clicking "Update Log" sends a PUT request to update the specific book document in the database.
- **Messaging:** Submitting the Contact form sends a POST request to archive the user's inquiry in the "Messages" collection.
- **3.2.2.4 Outputs**
- **Visual:** Realistic page-turning animations.
- **Data:** Updated database records for Book Status and Ratings.
- **Feedback:** Success alerts (e.g., "Entry Updated," "Message Dispatched").
- **3.2.2.5 Error Handling**
- Network errors during updates trigger a "Failed to update log" alert.
- Inquiry transmission failures trigger the custom error: "The owls are on strike."
-

3.5 Non-Functional Requirements

3.5.1 Performance

- The application initial load time (Time to Interactive) shall not exceed 3 seconds on a standard broadband connection.
- The flipbook animation shall run at a minimum of 30 frames per second (FPS) for smoothness.

3.5.2 Reliability

- The system shall implement try/catch blocks around all asynchronous API calls to prevent the application from crashing due to server timeouts.

3.5.3 Availability

- The system shall be available 24/7 via the deployed URL, barring scheduled maintenance or MongoDB Atlas downtime.

3.5.4 Security

- **Authentication:** User passwords shall not be stored in plain text (hashing required).
- **Access Control:** Unauthenticated users attempting to access /home or /book/:id shall be automatically redirected to the Login page via localStorage verification.

3.5.5 Maintainability

- The code shall be organized into a modular folder structure (/components, /assets, /pages) to allow for easy updates and debugging.

3.5.6 Portability

- The web application shall be responsive and function correctly on different browser engines (Chromium, Gecko, WebKit).

3.7 Design Constraints

1. **Aesthetic Consistency:** All UI components must adhere to the defined colour palette (Gold #FFD700, Dark Wood #1a0f0a) to maintain the "Story Verse" brand identity.
2. **No External Animation Libraries (CSS):** For the "About Page" scroll effects, the system is constrained to use the native Intersection Observer API rather than heavy libraries like GSAP, to optimize performance.

3.9 Other Requirements

Catchall section for any additional requirements.

4. Analysis Models

4.1 Data Flow Diagrams (DFD)

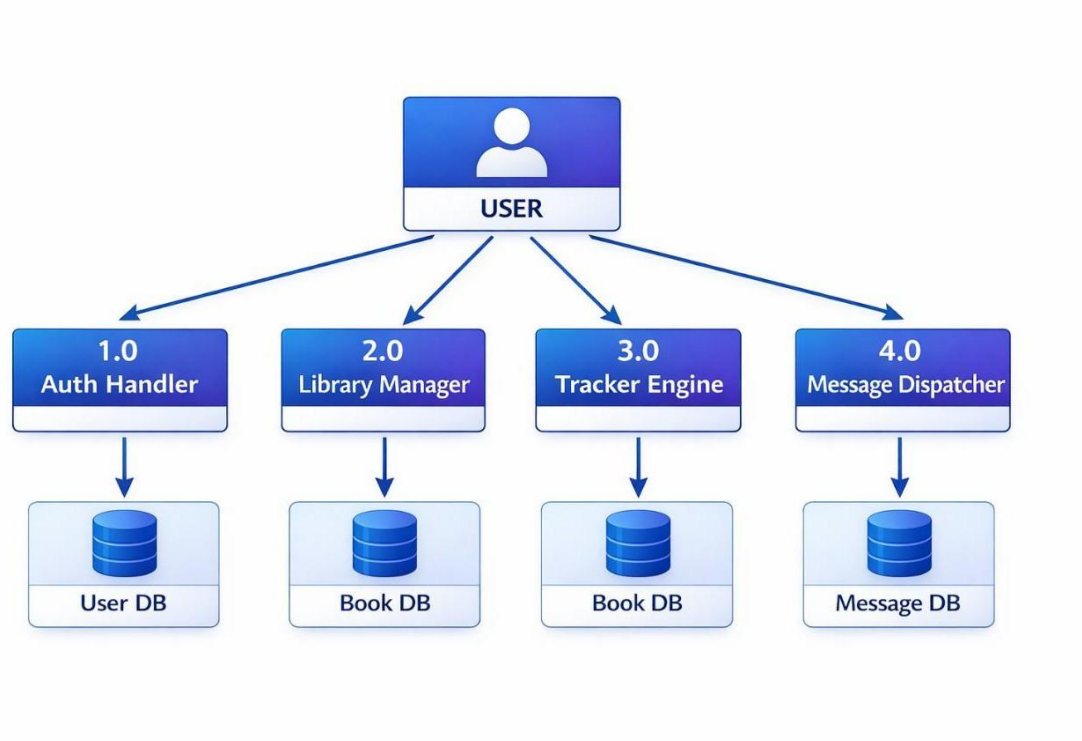
The Data Flow Diagram (DFD) maps the flow of information for the Story Verse Reading Tracker System. It illustrates how data enters the system from external entities (Users), how it is processed by the MERN stack backend, and where it is stored in the MongoDB database.

4.1.1 DFD Level 0 (Context Diagram): The Context Diagram represents the entire Story Verse system as a single process interacting with external entities.

- **External Entity:** User
- **Process:** Story Verse System
- **Data Flow (Input):** Login Credentials, Book Status Updates, Search Queries, Contact Messages.
- **Data Flow (Output):** Authentication Token, Book Library Data, Success/Error Alerts, Flipbook Animations.

4.1.2 DFD Level 1 (Detailed System Flow) The Level 1 DFD decomposes the main system into specific functional processes corresponding to the functional requirements defined in Section 3.2.

- **Process 1.0: Authentication Handler**
  - **Narrative:** The user submits credentials (Email/Password). This process verifies the input against the **User Collection** in MongoDB. If valid, it returns an access token to the user.
- **Process 2.0: Library Manager**
  - **Narrative:** Upon accessing the "My Library" page, this process requests book data. It queries the **Book Collection** and returns a list of book objects (Title, Cover, Author, Status) to be displayed in the grid.
- **Process 3.0: Tracker Engine**
  - **Narrative:** When a user updates a book's status or rating, this process receives the new values. It executes an update command (PUT request) to modify the specific record in the **Book Collection**. It then confirms the success to the user.
- **Process 4.0: Message Dispatcher**
  - **Narrative:** This process handles inquiries from the Contact page. It accepts the message payload (Name, Email, Subject, Body) and stores it in the **Messages Collection** for administrator review.



5. **Github link - https://github.com/jaspreetgithubit/Reading-Tracker-System.git**

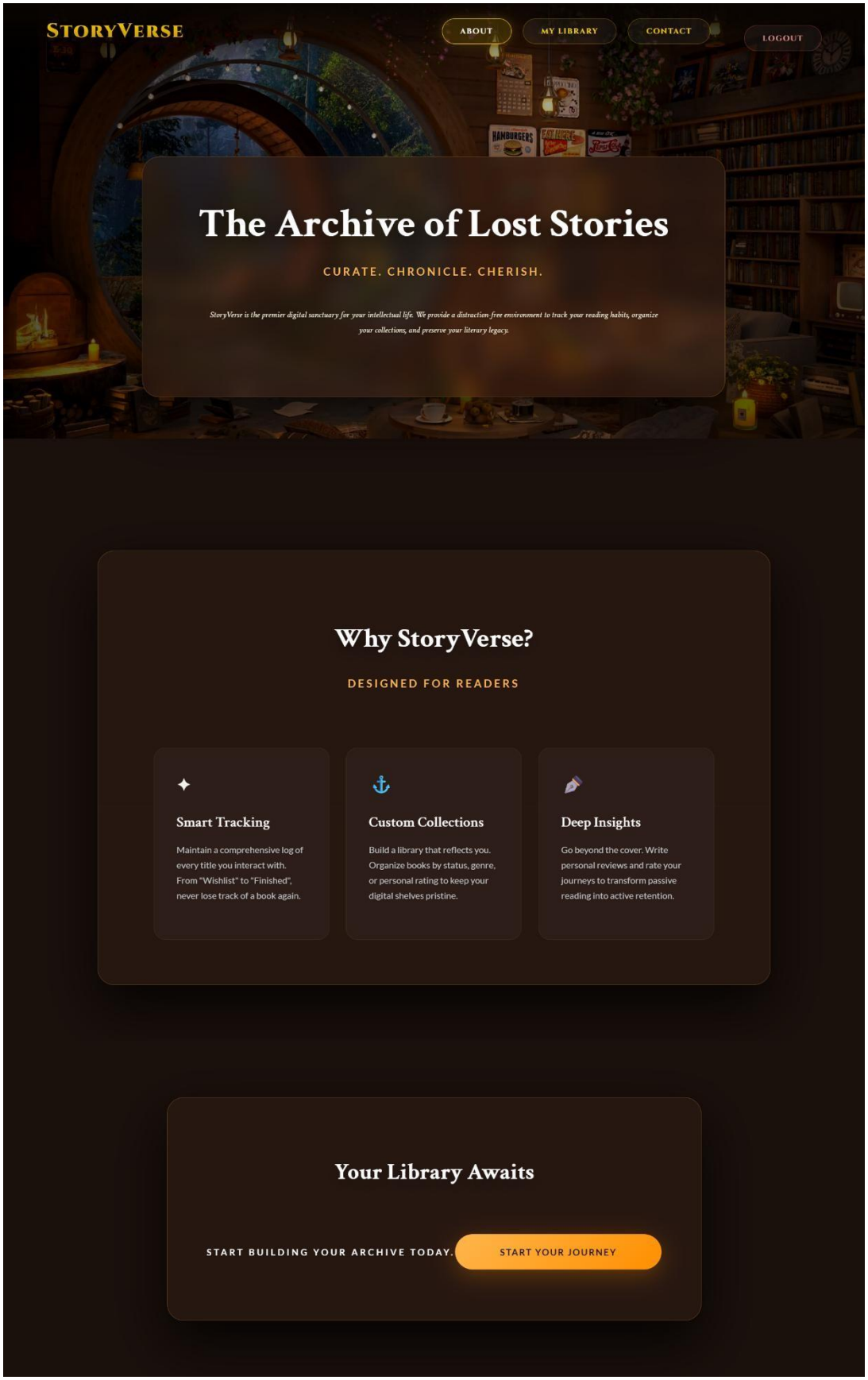
6. **Linkdln link - https://www.linkedin.com/posts/jaspreet-069ab4285_fullstackdevelopment-mernstack-reactjs-activity-7407828324955332608-dDvk?utm_source=share&utm_medium=member_desktop&rcm=ACoAAEVlXlEBYQkuYSij7TmnoJhjuREtZVmLw5Y**

A. Appendices

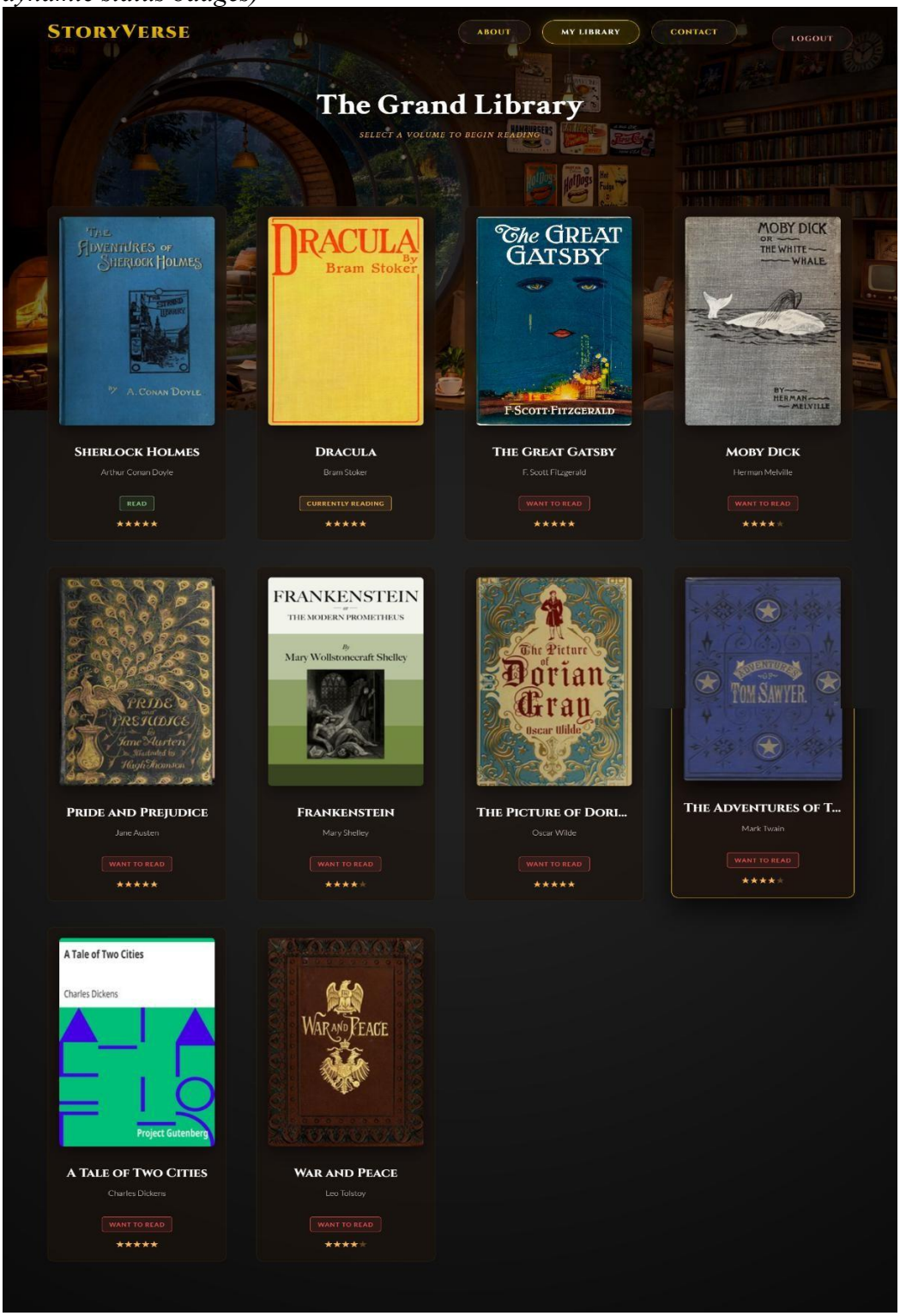
This section includes additional information and visual references to support the SRS.

A.1 Appendix 1: User Interface Screenshots

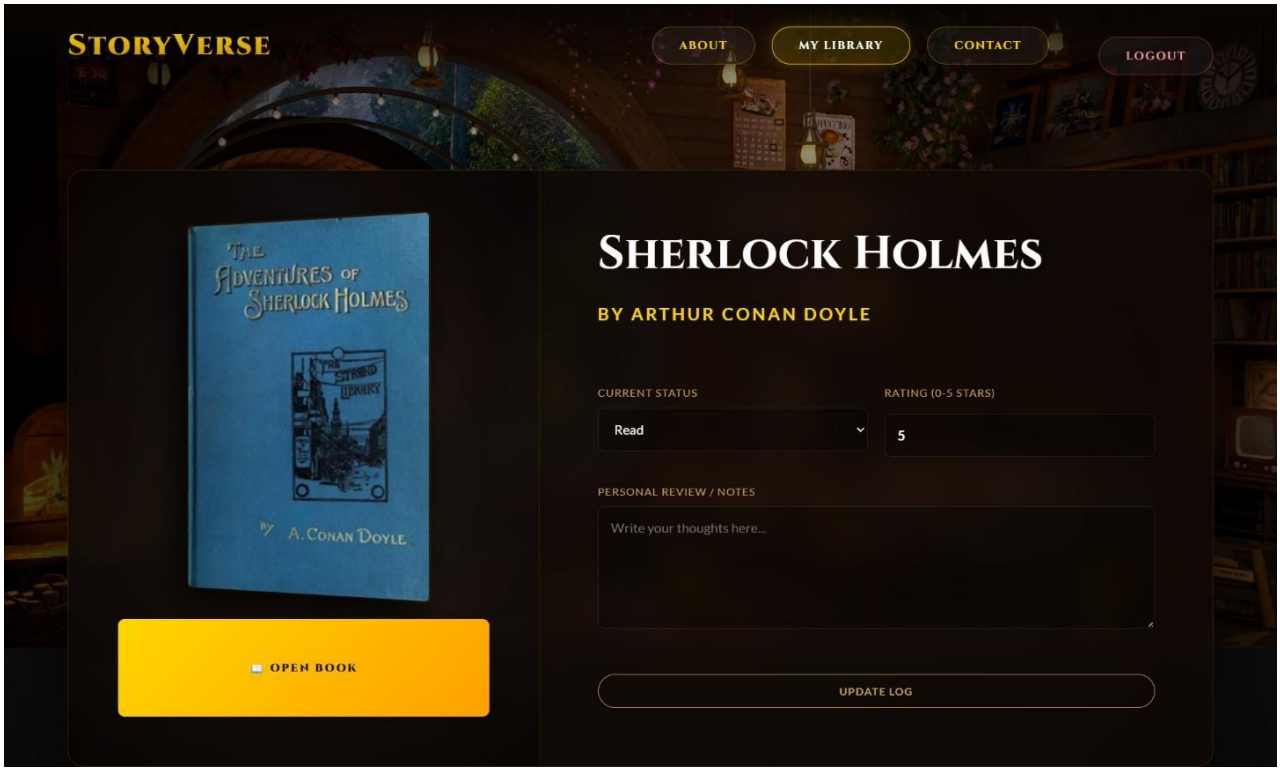
- **Figure 1:** The Archive (About Page) *(Displays the cinematic scroll animations and "Vision" cards)*

- 
StoryVerse
ABOUT
MY LIBRARY
CONTACT
LOGOUT
The Archive of Lost Stories
CURATE. CHRONICLE. CHERISH.
StoryVerse is the premier digital sanctuary for your intellectual life. We provide a distraction-free environment to track your reading habits, organize your collections, and preserve your literary legacy.
Why StoryVerse?
DESIGNED FOR READERS
Smart Tracking
Maintain a comprehensive log of every title you interact with. From "Wishlist" to "Finished", never lose track of a book again.
Custom Collections
Build a library that reflects you. Organize books by status, genre, or personal rating to keep your digital shelves pristine.
Deep Insights
Go beyond the cover. Write personal reviews and rate your journeys to transform passive reading into active retention.
Your Library Awaits
START BUILDING YOUR ARCHIVE TODAY.
START YOUR JOURNEY

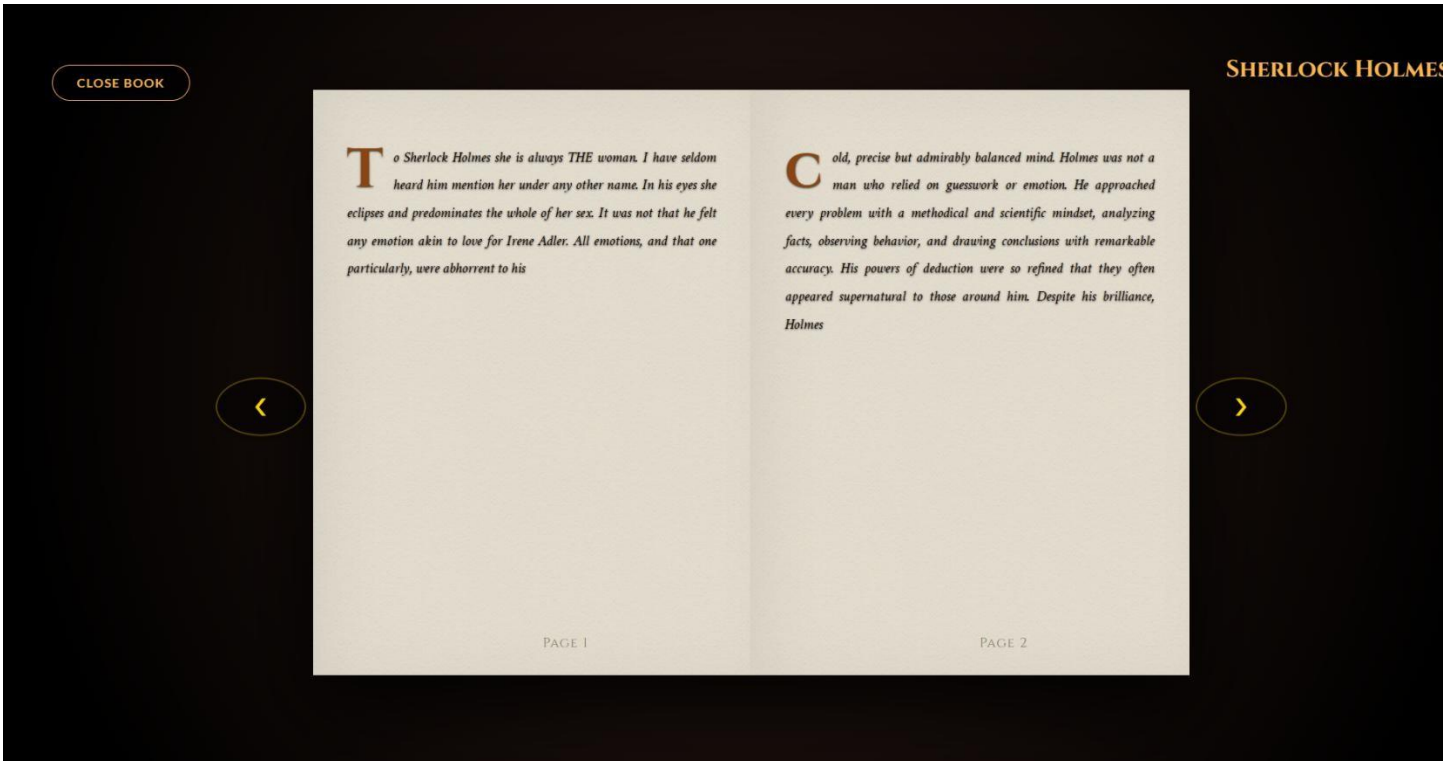
- **Figure 2:** The "Grand Library" (Home Page) *(Displays the user's book collection with dynamic status badges)*



- **Figure 3:** Book Details & Tracker *(Displays the rating system and status update controls)*



- **Figure 4:** The Immersive Reader *(Displays the flipbook simulation with side navigation)*



- **Figure 5:** Contact Owl (Inquiry Form) *(Displays the user feedback mechanism)*