Software Requirements Specification (SRS)

Project Name: WriteShelf
Prepared By: Seif Eldin 202200973 Ahmed Mostafa 202201114
Date: 11/9/2024

---

1. Introduction

1.1 Purpose

The purpose of this document is to outline the functional and nonfunctional requirements for **WriteShelf**.. It provides a detailed description of the system's functionality and constraints, and serves as a guide for developers, testers, and other stakeholders.

1.2 Scope

WriteShelf is a web application that serves as a digital library for readers and an interactive space for aspiring writers. The platform provides a library of light novels along with a dedicated space where authors can share what they write, and receive feedback on their writing.

1.3 Definitions, Acronyms, and Abbreviations

- **API**: Application Programming Interface
- **UAT**: User Acceptance Testing
- **CRUD:** Create, Read, Update, Delete (standard database operations)
- **UI:** User Interface
- **OAuth:** Open Authorization (for third-party authentication, such as Google/Facebook login)

---

2. System Overview

2.1 Product Perspective

The WriteShelf system is a **standalone system** designed to operate independently. It interacts with various **external services** for third-party authentication (such as Google and Facebook) and email notifications for user communication It consists of:

- **Frontend**: allowing users to browse the library, write and publish stories, view profiles, and receive notifications.
- **Backend**: The server-side processing and business logic,and user authentication, as well as the handling of interactions between the frontend and the database.
- **Database**: The storage for all persistent data,including user profiles, books, tutorials, comments, and interaction history.

2.2 Product Functions

The key functions of the system include:

1. User Authentication: Local and third-party login Oauth (Google/Facebook)
2. Library Search: search books by name
3. Advanced Filter: By genre, theme, author, and title
4. Star review system: readers rate content on a 5 star scale for overall quality
5. Comments for feedback: Users can comment on and rate public works.
6. Follow: to tune into your favorite creator and get updates
7. Feed: Tuned based on the followed creators
8. Push Notifications: Alerts for new feedback, followers, and recommendations.
9. History: Keep track of read, reading, and plan to read books
10. Workshop: Here you can find on going projects if you're a creator
11. Contribution Tracker: Creators can have an activity matrix similar to GIthub's
12. Socials: Creators can link their social media accounts on their profiles

2.3 User Classes and Characteristics

Different types of users who will interact with the system include:

- **Admin Users**: Have full access to all system functions, including managing users and configuring settings.
- **Contributors/Writers and Readers :** Contributors can choose to create and publish written content in the form of stories or articles, edit their work, and engage with other contributors.
- **Guest Users**: Can only view certain sections or content.

2.4 Operating Environment

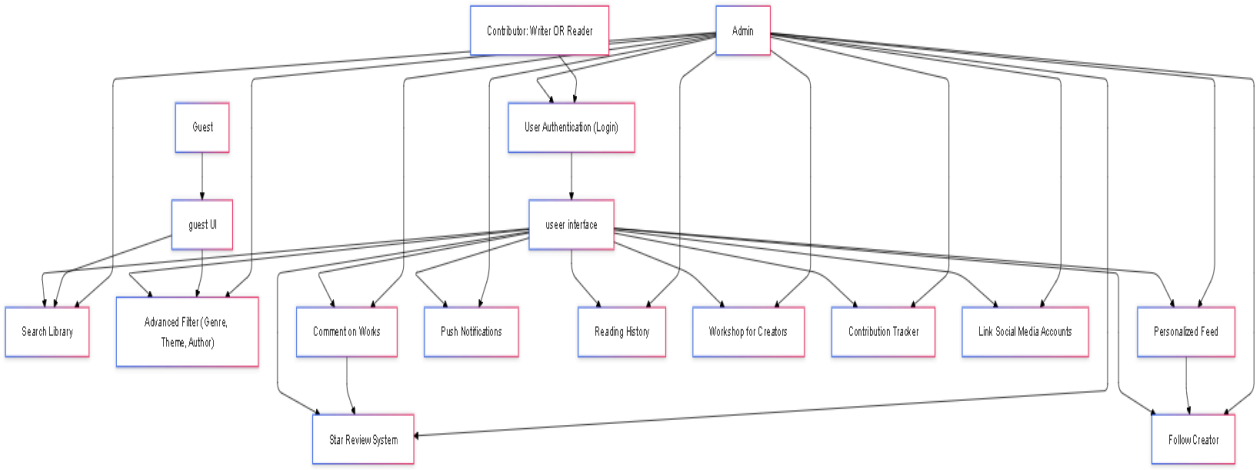
The system will operate in the following environments:

- **Client Side**: Webpage running the dashboard / interface for the webapp
- **Server Side**: Hosted on a VPS / development server running Linux and using the underlying python backend
- **Database**: MongoDB with a possibility to pivot to an SQL based DB like SQLite should limitations be faced.

---

3. Functional Requirements

3.1 Use Case Diagrams / User Stories



[user diagram link]

Use Case 1: User Authentication (Login)

- **Description**: Users log in to access functionalities based on their roles.
- **Actors**: Admin, Contributor (Writer/Reader), Guest.
- **Preconditions**:
  - The user has an existing account for login.
- **Postconditions**:
  - The user gains access to features based on their role.
- **Steps**:
  - Step 1: User navigates to the login page.
  - Step 2: User enters credentials (Admin/Contributor) or accesses as Guest.
  - Step 3: System verifies credentials.
  - Step 4: Authenticated users are redirected to the **User Interface (UI)**; Guest users are redirected to the **Guest UI**.

Use Case 2: Search Library

- **Description**: Users search the digital library for available books.
- **Actors**: Admin, Contributor (Writer/Reader), Guest.
- **Preconditions**:
  - User has access to the platform (logged in or as Guest).
- **Postconditions**:
  - Search results are displayed to the user.
- **Steps**:
  - Step 1: User enters a search term.
  - Step 2: System retrieves matching results from the library database.
  - Step 3: Results are displayed to the user.

Use Case 3: Advanced Filter (Genre, Theme, Author)

- **Description**: Users filter books by genre, theme, or author.
- **Actors**: Admin, Contributor (Writer/Reader), Guest.
- **Preconditions**:
  - User has access to the library search page.
- **Postconditions**:
  - Filtered results are displayed based on selected criteria.
- **Steps**:
  - Step 1: User applies filters on the search page.
  - Step 2: System processes filters and displays relevant results.

Use Case 4: Star Review System

- **Description**: Users can rate books using a star rating system.
- **Actors**: Admin, Contributor (Writer/Reader).
- **Preconditions**:
  - User is logged in and viewing a book.
- **Postconditions**:
  - User's rating is saved and affects the overall rating of the book.
- **Steps**:
  - Step 1: User selects a star rating.
  - Step 2: System saves the rating and updates the book's average rating.

Use Case 5: Comment on Works

- **Description**: Users leave comments on published works.
- **Actors**: Admin, Contributor (Writer/Reader).
- **Preconditions**:
  - User is logged in and viewing a work that allows comments.
- **Postconditions**:
  - Comment is saved and visible to others.
- **Steps**:
  - Step 1: User types a comment and submits it.
  - Step 2: System saves and displays the comment.

Use Case 6: Follow Creator

- **Description**: Users follow their favorite creators to receive updates.
- **Actors**: Admin, Contributor (Writer/Reader).
- **Preconditions**:
  - User is logged in and viewing a creator's profile.
- **Postconditions**:
  - User receives notifications about the followed creator.
- **Steps**:
  - Step 1: User clicks the "Follow" button on a creator's profile.
  - Step 2: System adds the creator to the user's following list.

Use Case 7: Personalized Feed

- **Description**: Users view a feed tuned based on creators they follow.
- **Actors**: Admin, Contributor (Writer/Reader).
- **Preconditions**:
  - User is logged in and follows at least one creator.
- **Postconditions**:
  - User sees a personalized feed of updates.
- **Steps**:
  - Step 1: User navigates to their feed.
  - Step 2: System retrieves updates from followed creators.

Use Case 8: Push Notifications

- **Description**: Users receive notifications for updates or feedback.
- **Actors**: Admin, Contributor (Writer/Reader).
- **Preconditions**:
  - User has notifications enabled.
- **Postconditions**:
  - Notifications appear in the user's dashboard.
- **Steps**:
  - Step 1: System generates a notification based on user activity.
  - Step 2: Notification is sent to the user's feed.

Use Case 9: Reading History

- **Description**: Users track read and currently reading books.
- **Actors**: Admin, Contributor (Reader).
- **Preconditions**:
  - User is logged in.
- **Postconditions**:
  - Reading history is updated.

- **Steps**:
    - Step 1: User marks a book as read or currently reading.
    - Step 2: System records and displays the history in the user's profile.

Use Case 10: Workshop for Creators

- **Description**: Creators access an area to develop ongoing projects.
- **Actors**: Admin, Contributor (Writer).
- **Preconditions**:
    - User is logged in as a Creator.
- **Postconditions**:
    - User accesses tools to manage projects.
- **Steps**:
    - Step 1: User navigates to the workshop section.
    - Step 2: System provides access to project management tools.

Use Case 11: Contribution Tracker

- **Description**: Tracks creator activities, showing contributions.
- **Actors**: Admin, Contributor (Writer).
- **Preconditions**:
    - User is logged in as a Creator.
- **Postconditions**:
    - Contribution stats are displayed in the creator's profile.
- **Steps**:
    - Step 1: System records activities (e.g., writing).
    - Step 2: Activity data updates the contribution tracker.

Use Case 12: Link Social Media Accounts

- **Description**: Users link social media accounts to their profiles.
- **Actors**: Admin, Contributor (Writer).
- **Preconditions**:
    - User is logged in.
- **Postconditions**:
    - Social media links appear on the user's profile.
- **Steps**:
    - Step 1: User links social media account.
    - Step 2: System saves the link and updates the profile.

3.2 Feature Requirements

Feature 1: User Authentication (Login)

Description: Allows users to log into the system to access features based on their roles.

- **Inputs**: Username and password, or third-party OAuth token (Google/Facebook).
- **Outputs**: Access to the main UI or guest UI based on role; error message if login fails.
- **Error Handling**: If credentials are invalid, display an error message prompting the user to try again. If third-party authentication fails, notify the user and allow them to retry.

Feature 2: Library Search

Description: Enables users to search for books by entering keywords.

- **Inputs**: Search query (text) entered by the user.
- **Outputs**: List of books matching the search criteria.
- **Error Handling**: If no books match the query, display a "No results found" message. If the database query fails, display a generic error and prompt the user to retry.

Feature 3: Advanced Filter (Genre, Theme, Author)

Description: Allows users to filter search results by genre, theme, or author.

- **Inputs**: Filter options selected by the user.
- **Outputs**: Filtered list of books matching the selected criteria.
- **Error Handling**: If no books match the selected filters, display a message stating "No results found." If filters fail to apply, refresh the filters and display an error.

Feature 4: Star Review System

Description: Allows users to rate books on a 5-star scale.

- **Inputs**: Star rating (1-5 stars) chosen by the user.
- **Outputs**: Updated average rating of the book.
- **Error Handling**: If saving the rating fails, display a notification and allow the user to retry.

Feature 5: Comment on Works

Description: Enables users to leave comments on published works.

- **Inputs**: Text comment entered by the user.
- **Outputs**: Visible comment on the work's page, accessible to all users.
- **Error Handling**: If the comment cannot be saved, display an error message and allow the user to retry.

Feature 6: Follow Creator

Description: Lets users follow creators to receive updates.

- **Inputs**: "Follow" action taken on a creator’s profile.
- **Outputs**: Creator added to the user’s followed list, resulting in future notifications for updates.
- **Error Handling**: If the follow action fails, display an error message and allow the user to try again.

Feature 7: Personalized Feed

Description: Provides a customized feed with updates from followed creators.

- **Inputs**: Followed creator updates, user’s followed list.
- **Outputs**: Feed page displaying new content from followed creators.
- **Error Handling**: If loading fails, display a message prompting the user to refresh the page.

Feature 8: Push Notifications

Description: Sends users notifications about feedback, updates, and recommendations.

- **Inputs**: User actions, feedback, new updates from followed creators.
- **Outputs**: Notification appearing on the user’s dashboard.
- **Error Handling**: If notifications fail to send, display a warning and offer a manual refresh option.

Feature 9: Reading History

Description: Tracks books the user has read, is currently reading, or plans to read.

- **Inputs**: User actions marking a book as "Read," "Reading," or "Plan to Read."
- **Outputs**: Updated reading history on the user's profile.
- **Error Handling**: If an update to reading history fails, display an error message and retry option.

Feature 10: Workshop for Creators

Description: A dedicated space for creators to work on and manage ongoing projects.

- **Inputs**: Creator’s project information and updates.
- **Outputs**: Access to project management tools and visible ongoing projects.
- **Error Handling**: If the workshop fails to load, prompt the user to refresh; display error for failed project updates.

Feature 11: Contribution Tracker

Description: Displays an activity matrix tracking the creator’s contributions.

- **Inputs**: Creator activities, such as writing and project updates.
- **Outputs**: Activity matrix on the creator's profile, showing contribution history.
- **Error Handling**: If tracking data fails to load, display an error and allow a manual refresh.

Feature 12: Link Social Media Accounts

Description: Allows users to link their social media accounts to their profile.

- **Inputs**: Social media account information.
- **Outputs**: Linked social media icons on the user’s profile.
- **Error Handling**: If linking fails, show an error message and retry option.
-

---

4. Non-Functional Requirements

4.1 Performance Requirements

- The system should respond to user requests within 3 seconds for typical actions (e.g., searching, loading feeds, posting comments)..
- It must support up to 1,000 concurrent users without significant degradation in performance, especially during peak usage times..

4.2 Security Requirements

- User data must be encrypted in transit using HTTPS and at rest using encryption methods.

- Access to specific system functionalities should be role-based, ensuring only authorized users (Admin, Contributor, Guest) can access certain features.

4.3 Usability Requirements

- The user interface should be intuitive and user-friendly, with consistent navigation, tooltips, and user guidance.
- Accessibility features should be included to support users with disabilities, following [insert accessibility standard, e.g., WCAG 2.1].

4.4 Reliability and Availability Requirements

- The system must have a minimum uptime of 99.9% to ensure availability for users.
- In case of a failure, the system should recover within 5 minutes to minimize user disruption and prevent data loss.

- Regular backups should be performed hourly, with data redundancy to avoid potential loss of content.

4.5 Scalability

- The system should be able to scale to support 100000 users or up to 1 terabyte of content, including text, images, and user interaction history.

4.6 Compatibility

- The software should be compatible with modern web browsers, including Chrome, Firefox, Edge, and Safari, ensuring a consistent experience across platforms.
- The system should be compatible with mobile devices.

---

5. System Models

5.1 Use Case Diagrams

user diagram link

5.2 Data Flow Diagrams

Data Flow Diagrams

5.3 Class Diagrams

Class Diagram link

---

6. External Interface Requirements

6.1 User Interfaces

- The WriteShelf system should feature an intuitive and user-friendly UI with the following major components:

    1. **Home Screen**: Displays a summary of user activity, including recent books, recommended content, and recent notifications

    2. **Profile Page**: Allows users to view and edit their profile information,view their history projects or books being read,manage social media links, and view their followers or following list.

    3. **Library Search and Filter Page**: Enables users to search for books by title, author, genre, or theme and apply advanced filters.

    4. **Book Details Page**: Displays book information, star rating, comments, and related books. Logged-in users can rate and comment on books here.
    5. **Dashboard**:

        -For **Readers**: Provides options to search the library, access saved books, track reading history, and view a personalized feed based on followed authors.

        -For **Writers**: Allows access to the workshop area, contribution tracker, and options for managing ongoing projects.

        -For **Admin Users**: Includes administrative controls for managing users, reviewing content, and monitoring site activity.

.6.2 API Interfaces

- The system will provide APIs for [insert functionalities, e.g., data retrieval, user management].
- APIs will follow the REST/GraphQL protocol.

6.3 Hardware Interfaces

- The WriteShelf system will primarily be web-based and does not directly interact with external hardware. However, it should be adaptable to hardware integrations in the future if required (e.g., printers for printing book excerpts, barcode scanner for book search ,etc.).

---

7. Other Requirements

7.1 Legal and Regulatory Requirements

The WriteShelf system must comply with data protection and privacy regulations, including:

- **GDPR** (General Data Protection Regulation) for handling personal data of EU users.
- **DMCA** (Digital Millennium Copyright Act) compliance for user-generated content management and copyright infringement monitoring.

7.2 Documentation Requirements

The following documentation should be provided:

- **User Manual**: A guide covering user account setup, navigating the platform, and utilizing all user functionalities.
- **API Documentation**: Technical documentation covering API endpoints, request and response formats, and usage examples for developers.

7.3 Data Backup Requirements

The WriteShelf system should:

- **Back up data daily** to ensure minimal data loss in case of system failure.
- Store backups for a minimum of **30 days** in a secure storage solution, allowing for data restoration if required.