Software Requirements Specification

for

A Secure Document Sharing

(TrustVault)

Version 1.0

Prepared by Project Team 11

Name	PRN
Parag Parate	240840320066
Rohit Owal	240840320085
Vaishnavi Kulkarni	240840520095
Ashwini Patil	240840520014
Umesh Pakhare	240840520094

Centre for Development of Advanced Computing

Raintree Marg, Near Bharati Vidyapeeth, Opp. Kharghar Railway Station, Sector 7, CBD Belapur, Navi Mumbai - 400 614 - Maharashtra (India)
Phone: +91-22-27565303
Fax: +91-22-2756-0004

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

INTRODUCTION

1.1 Purpose

The purpose of this document is to define the functional, non-functional, and system requirements for the Secure Document Sharing Platform. This platform aims to provide a secure, role-based system for uploading, managing, and sharing documents. The document will serve as a guideline for the development team and stakeholders, ensuring clarity and alignment on project objectives, features, and technical specifications.

1.2 Problem Statement

Organizations and individuals face significant challenges in securely sharing sensitive documents due to risks like unauthorized access, lack of encryption, and weak permission controls. Traditional file-sharing methods often fail to meet the security and compliance requirements necessary for sensitive data protection.

1.3 Intended Audience and Reading Suggestions

The document is intended for a broad audience, including developers, students, and educational institutes.

For developers, the SRS document serves as a comprehensive guide for the design and development of the system. It outlines the functional and non - functional requirements, performance expectations, and security considerations, providing a clear understanding of the scope of the project. Developers should read the document carefully to ensure that their work aligns with the specified requirements and meets the expectations of all stakeholders.

For students, the SRS document provides an overview of the system's capabilities and the submission and review process, helping them to understand how to use the system effectively. They should read the document to gain a general understanding of the system's functionalities and how they can benefit from it.

For educational institutes, the SRS document outlines the goals and objectives of the project and provides a clear understanding of the system's capabilities, helping them to determine whether the system is suitable for their needs. The document should be read by relevant decision-makers to ensure that the system meets their requirements and expectations.

1.4 Project Scope

The scope of the TrustVault defines the boundaries of the project,outline what the system will and will not do. The scope of the project include the following :

The TrustVault - A Secure Document Sharing Platform is a web-based application designed to enable users to securely share sensitive files with multiple layers of protection. It employs role-based access, robust encryption, and advanced features like secure links, two-factor authentication. The platform targets individuals, organizations, and industries such as healthcare to ensure the safe exchange of sensitive data.

- Secure user authentication and role-based access control.
- Implementation of user friendly interface for users as well as for admin to interact with the system.
- Document management features such as upload, sharing and permissions.
- Receiver user gets email notification if anyone shares a document with him.
- Development of a secure system to share and store the document in encrypted format.
- Advanced security measures including 2FA, file encryption and HTTPS.
- Admin capabilities for user management, activity tracking, and Permission controls.

The platform will support various file types, ensure compliance with data security standards, and provide an intuitive, responsive UI for web users.

1.5 References

SpringBoot

- https://spring.io
- https://www.javascript.com/
- https://developer.mozilla.org/en-US/docs/Web/JavaScript
- https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller
- Spring Boot + React: JWT Authentication with Spring Security - BezKoder

Ms.NET

- ASP.NET Core Documentation Official documentation for building web applications with ASP.NET Core: https://learn.microsoft.com/en-us/aspnet/core/
- Entity Framework Core ORM for database interactions with ASP.NET Core: https://learn.microsoft.com/en-us/ef/core/

2. System Description

2.1 Product Overview

The platform is designed as a secure, role-based CRUD system that allows users to manage documents with features tailored to their specific roles (Admin, User). The platform ensures end-to-end encryption for files, secure sharing through links and comprehensive logging of user activities.

Key components include:

- **Frontend:** Built with ReactJS (Vite) and Material UI for an intuitive user experience.
- **Backend:** Powered by Spring Boot,Ms.Net ensures robust APIs and security features.
- **Database:** MySQL for managing user credentials, documents, permissions, and activity logs.

2.2 Product Features

1. **Secure Authentication and Authorization :**
    - This feature will allow users to log into the system using their credentials and 2FA (using email otp) , and the system will determine their access level and permissions.
2. **Document Management:**
    - Users can upload, view, share, and download documents with robust security measures, ensuring data protection and controlled access.
    - Features include metadata tracking for easy categorization and retrieval, along with version control to maintain and manage changes, ensuring seamless collaboration and transparency.
3. **Access Control:**
    - **Role-Based Access Control (RBAC)**: Assign role-specific permissions for secure and organized control over read, write, delete, and admin access.
    - **ACL-Based Permissions**: Manage shared document access with fine-grained control over permissions for users and groups.
4. **Encryption:**
    - AES encryption for files at rest and HTTPS for data in transit.
5. **Mobile Responsiveness:**
    - Ensure seamless access and optimal user experience across all devices and screen sizes.

2.3 User Documentation

User documentation mainly comprises of Help menu of application. It will give all the minute details about the project, if any user has any query about any module or functionality, one can refer to it and see how to operate the application. This report is the complete documentation of our project. It gives complete details about the project, its functionality, users, software used, hardware requirement, environment and so on.

2.4 Operating Environment

The operating environment for the project consists of the following hardware and software components:

Hardware:

- A machine with at least 8GB of RAM and a fast processor, such as Intel Core i5 or higher, to ensure smooth and efficient execution of the project.

Software:

- ReactJS for the frontend development.
- Spring Boot for the backend development
- MySQL for database management.
- JWT Authentication for security.

Other Applications:

- Code editor (such as Visual Studio Code, Eclipse)
- Git version control software
- Command line interface (CLI) or terminal
- A browser for testing the application.
- Postman for testing APIs.
- MySQL Workbench or another database management tool

3.User Classes / Characteristics:

3.1 Users

- **End Users:** Upload, manage, and securely share documents.
- **Admins:** Manage users, monitor logs, and oversee the system.

3.2 User Roles

- **Admin:** Complete access to users, logs, and document management.
- **User:** Manage their own files and access shared files.

4. System Features

4.1 Description and Priority

- Description: Users and Admin will be able to log in to the web app after validating using otp and their unique credentials.
- Priority: High

4.2 Stimulus/Response Sequences

- After successful login user and admin redirect to their respective dashboards.

4.3 User Dashboard

- Users are able to upload the documents.
- Document is stored in an encrypted format in the database.
- Users are able to search, sort and share the documents using the link.

4.4 Admin Dashboard

- Admin is able to manage the documents and users.
- Admin has authority to change roles of the users.
- Admin has access to edit and delete the users and documents.

5. Security and Authentication

- **Authentication:** 2FA, OTP-based login, bcrypt for password hashing.
- **Authorization:** Role-based access control (RBAC).
- **Encryption:** AES for file encryption, HTTPS for secure data transit.
- **Activity Monitoring:** Logs for uploads, downloads, sharing, and edits.

6. Other Nonfunctional Requirements

6.1 Performance Requirements

The performance requirements of the project should outline the expected speed, reliability, scalability, and efficiency of the system. This may include the maximum response time for user actions, the expected uptime percentage, the ability to handle increasing numbers of users and submissions, and the resource usage of the system. These requirements should be realistic and achievable, taking into account factors such as hardware limitations and network bandwidth. They should also be measurable and verifiable, allowing the system to be tested and evaluated against the defined standards.

6.2 Safety Requirements

The system should be reliable and have minimal downtime to ensure that students can submit assignments and receive feedback in a timely manner. The user interface should be intuitive and user-friendly to prevent errors or misunderstandings while using the app. The app should have robust error handling mechanisms to deal with unexpected situations and prevent any harm to the user or loss of data.

6.3 Security Requirements

The project would include measures to protect the confidentiality, integrity, and availability of sensitive information such as students' assignments, grades, and personal information. Some of the security requirements are:

• Authentication: A secure authentication system that ensures only authorized users can access the system.

• Authorization: An authorization mechanism to determine what actions a user can perform within the system based on their role and permissions.

• Encryption: Data in transit and at rest should be encrypted to protect sensitive information from being intercepted or accessed by unauthorized parties.

• Access Control: A mechanism to control access to sensitive information within the system, including the ability to set permissions for different users and roles.

6.4 Software Quality Attributes

The software quality attributes for the project include the following:

• Usability: The user interface should be intuitive, easy to navigate and user-friendly, allowing users to easily submit and review assignments.

• Reliability: The system should be reliable and provide accurate results, ensuring that assignments are submitted and reviewed accurately and efficiently.

• Performance: The system should perform efficiently and respond quickly to user requests, allowing for fast and seamless assignment submissions and reviews

• Scalability: The system should be scalable, allowing for an increasing number of users and assignments as the demand grows. • Maintainability: The system should be maintainable, with easy-to-update code and well-documented processes, ensuring that updates and improvements can be made quickly and efficiently.

• Security: The system should have strong security features, protecting sensitive information and ensuring the privacy and confidentiality of user data.

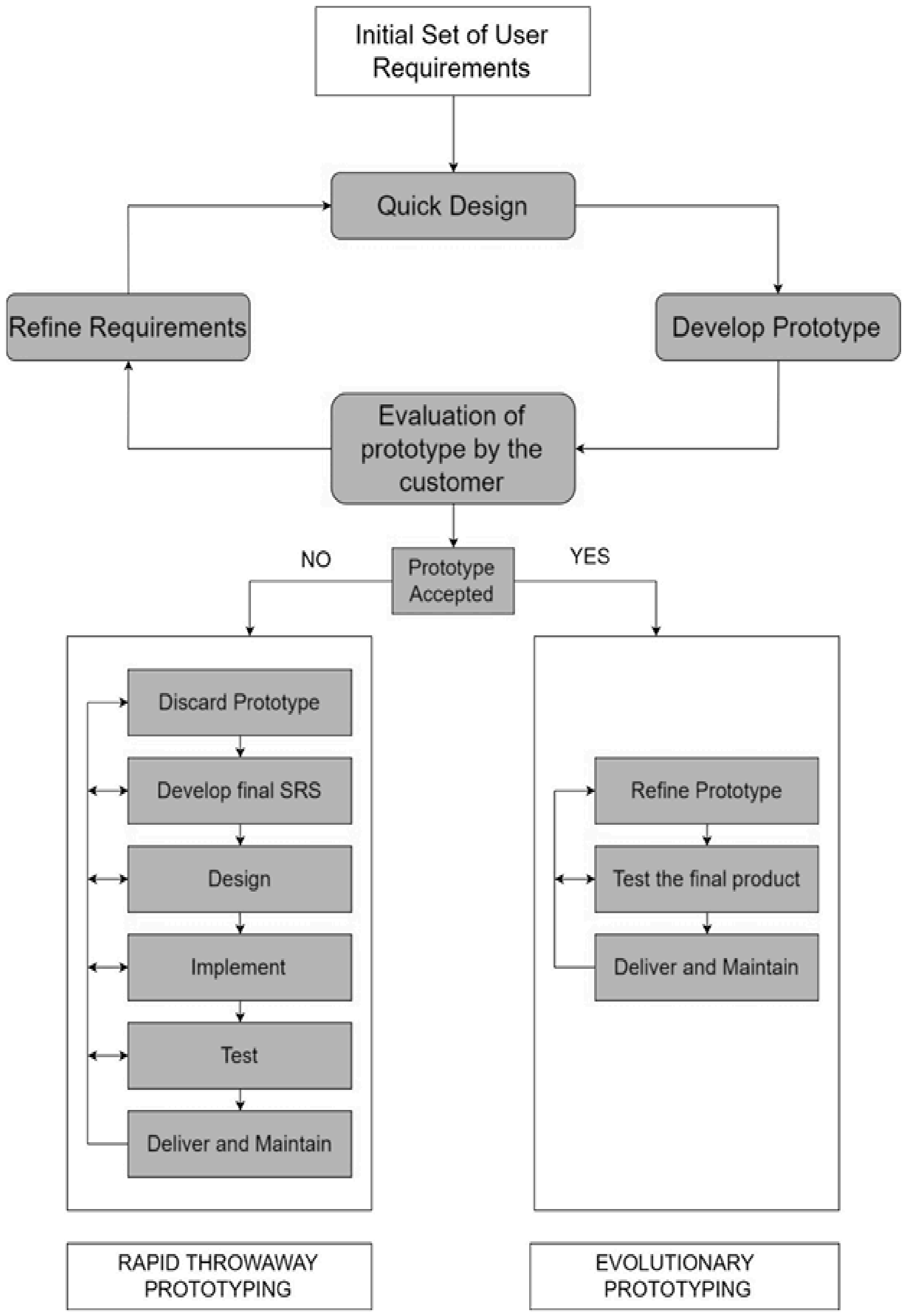
• Compliance: The system should comply with relevant regulations and standards, ensuring that it operates within the bounds of the law and industry best practices.

Appendix A: Glossary

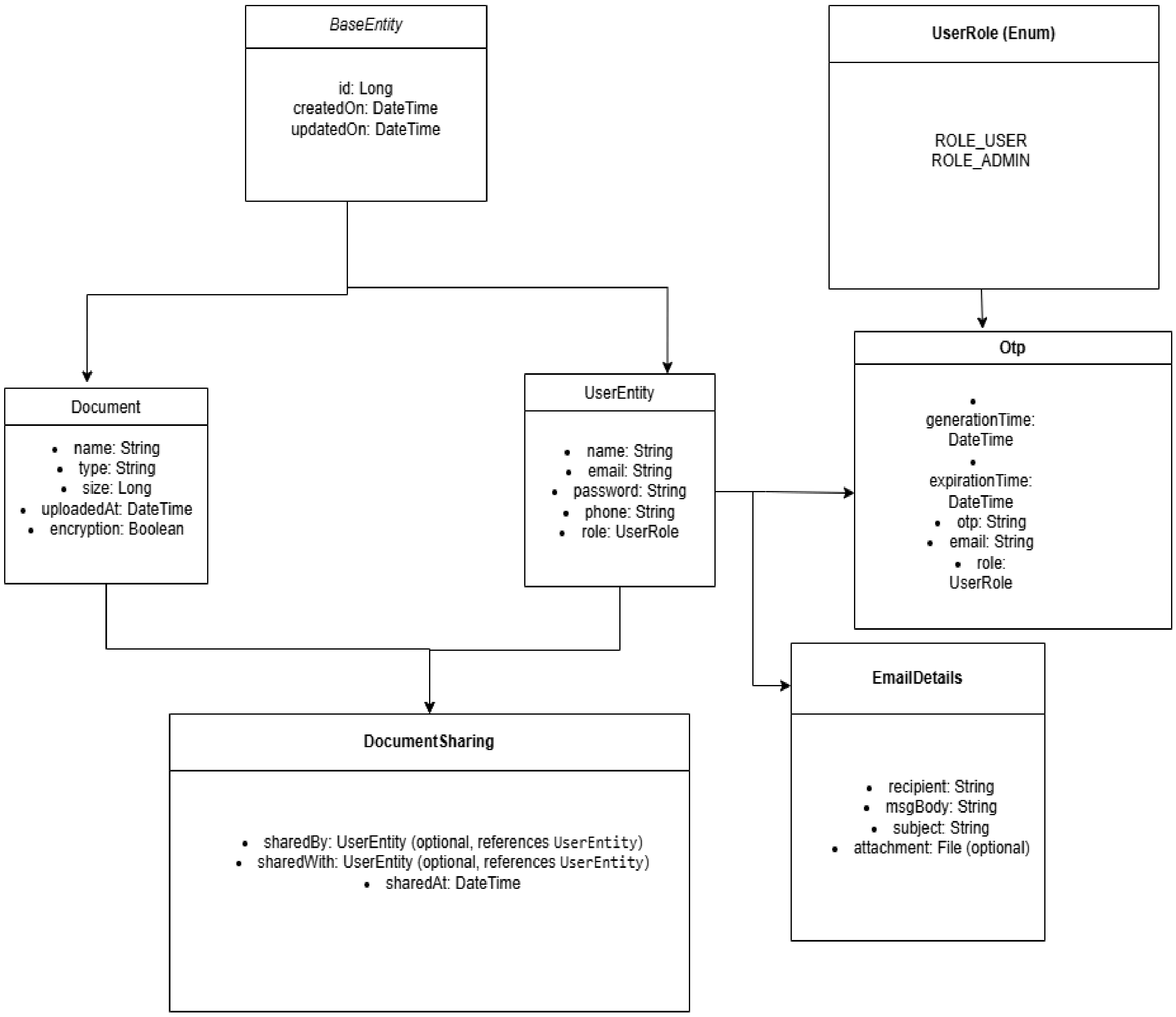
Sr. No.	Abbreviation	Full Form
1.	API	Application Programming Interface
2.	AWS	Amazon Web Service
3.	CLI	Command line Argument
4.	GB	Gigabyte
5.	HTML	Hypertext Markup Language
6.	HTTP / HTTPS	Hypertext Transfer Protocol / Hypertext Transfer Protocol Secure
7.	ID	Identification
8.	JS	JavaScript
9.	JWT	Java Web Token
10.	OS	Operating System
11.	RAM	Random Access Memory
12.	SQL	Structured Query Language
13.	SRS	Software Requirement Specification
14.	URL	Uniform Resource Locator
15.	ER	Entity Relationship

Appendix B: Analysis Models

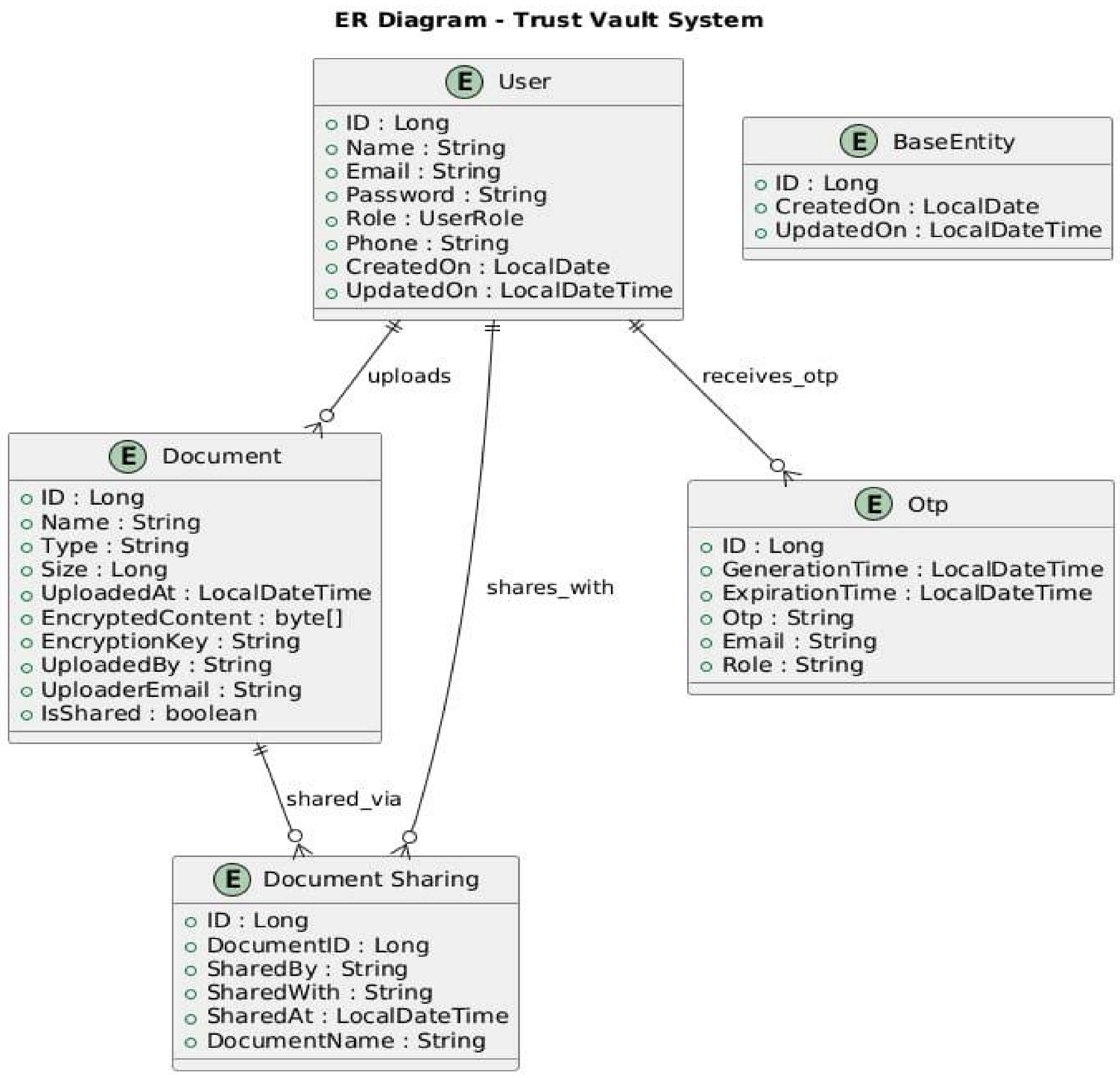
- **Software Development Approach in Our System**



- **Class Diagram**

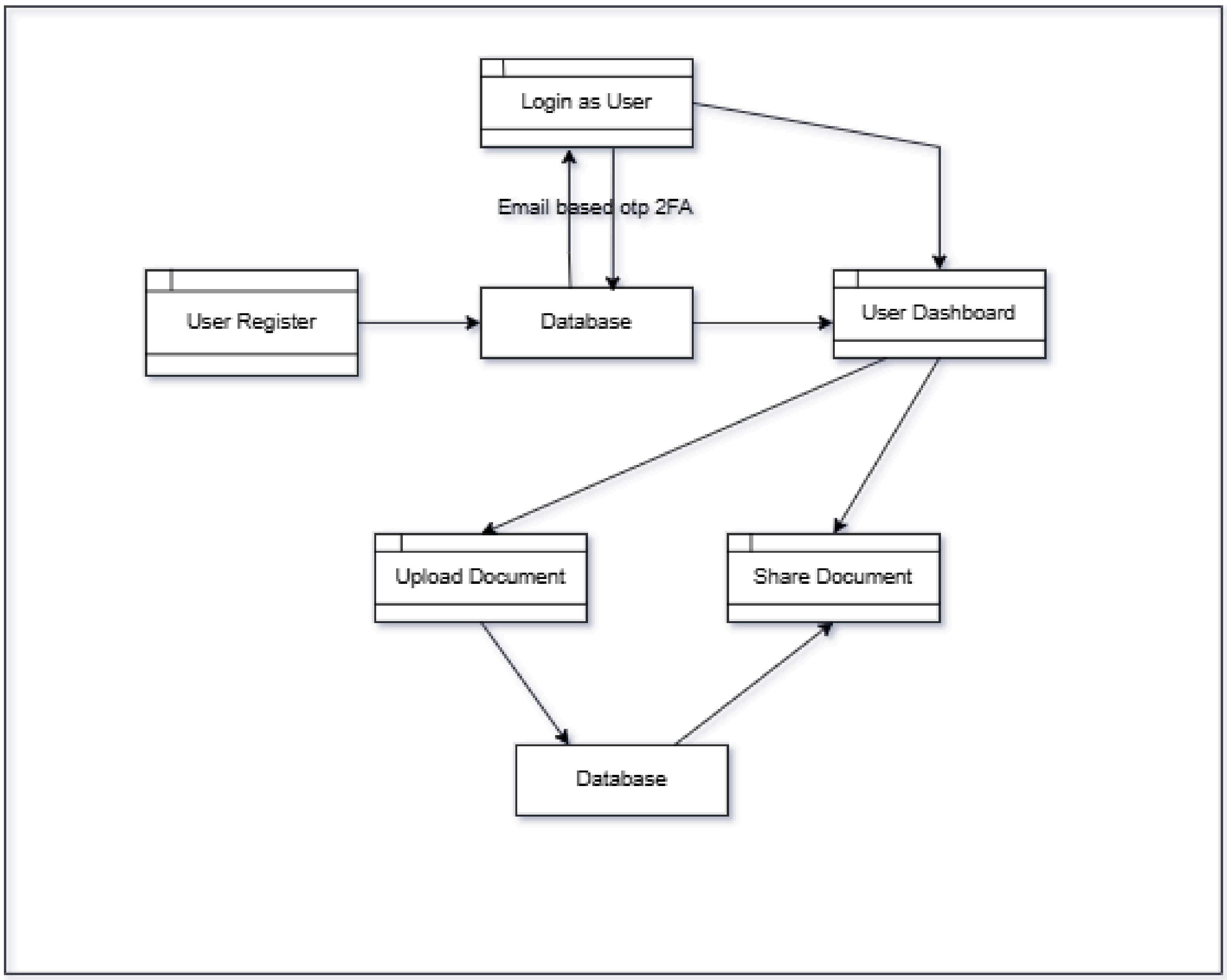


- **ER-Diagram :**

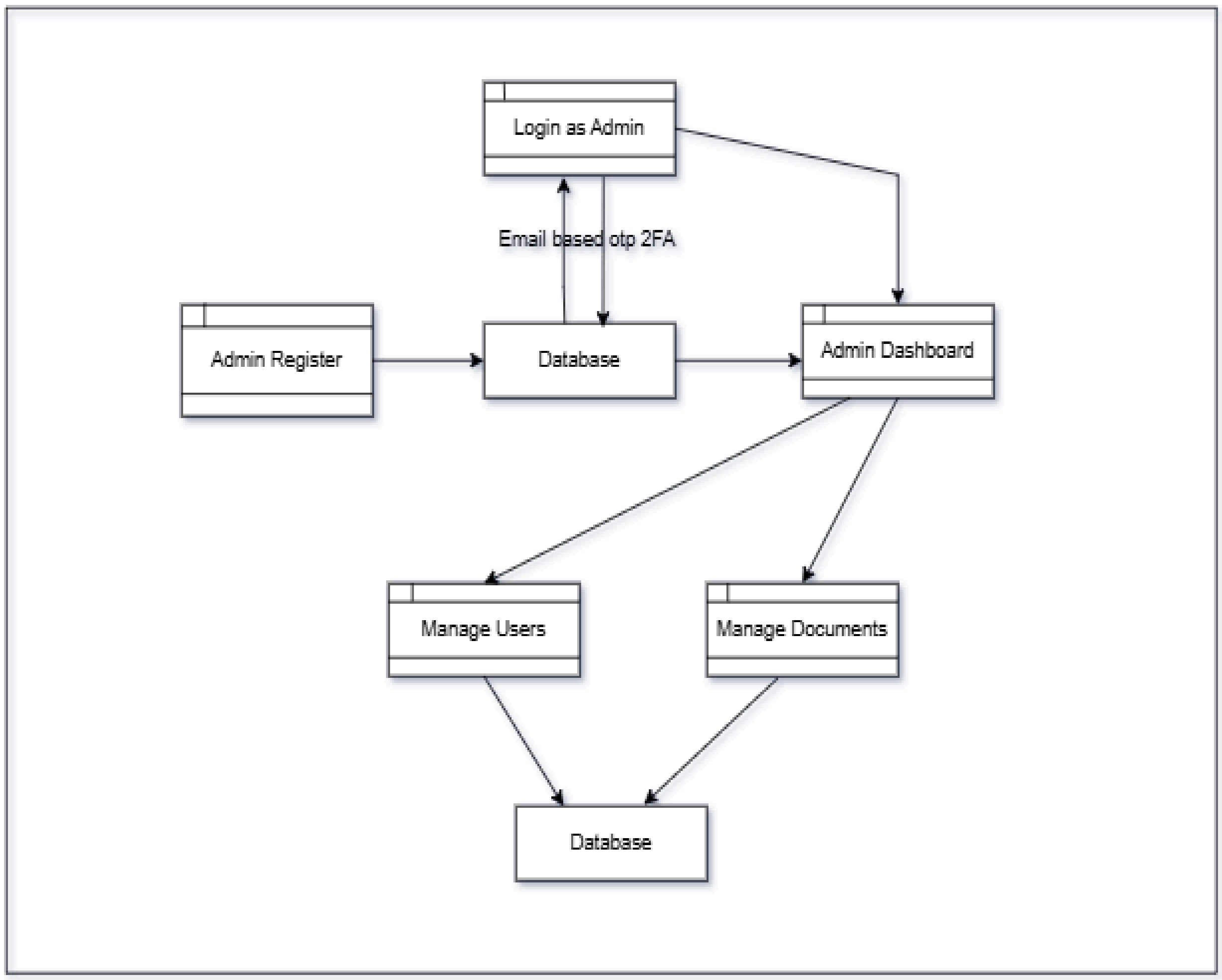

ER Diagram - Trust Vault System
E User
ID : Long
Name : String
Email : String
Password : String
Role : UserRole
Phone : String
CreatedOn : LocalDate
UpdatedOn : LocalDateTime
E BaseEntity
ID : Long
CreatedOn : LocalDate
UpdatedOn : LocalDateTime
uploads
receives_otp
E Document
ID : Long
Name : String
Type : String
Size : Long
UploadedAt : LocalDateTime
EncryptedContent : byte[]
EncryptionKey : String
UploadedBy : String
UploaderEmail : String
IsShared : boolean
shares_with
E Otp
ID : Long
GenerationTime : LocalDateTime
ExpirationTime : LocalDateTime
Otp : String
Email : String
Role : String
shared_via
E Document Sharing
ID : Long
DocumentID : Long
SharedBy : String
SharedWith : String
SharedAt : LocalDateTime
DocumentName : String

- **Data Flow Diagram :**

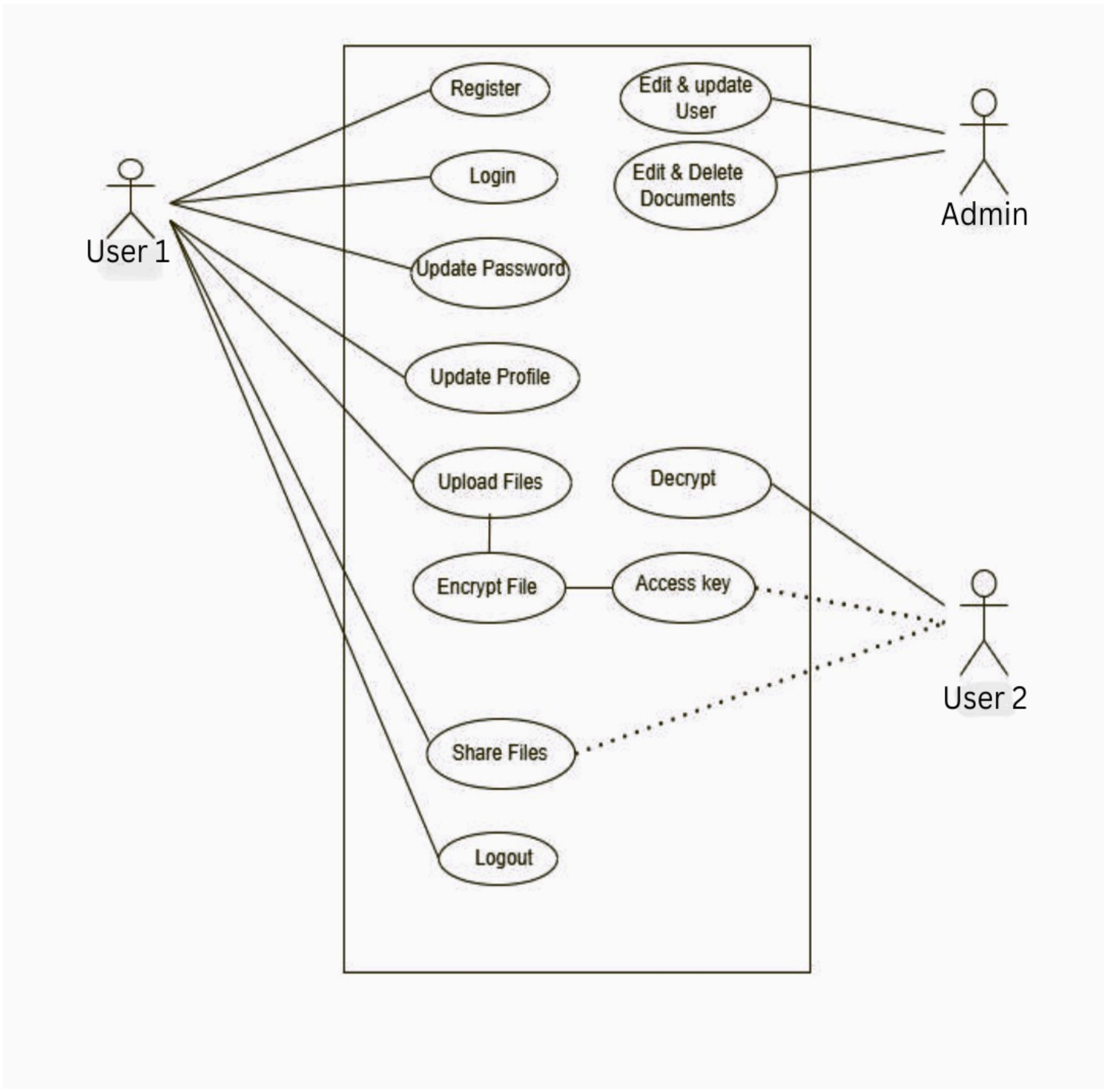
1. User Module



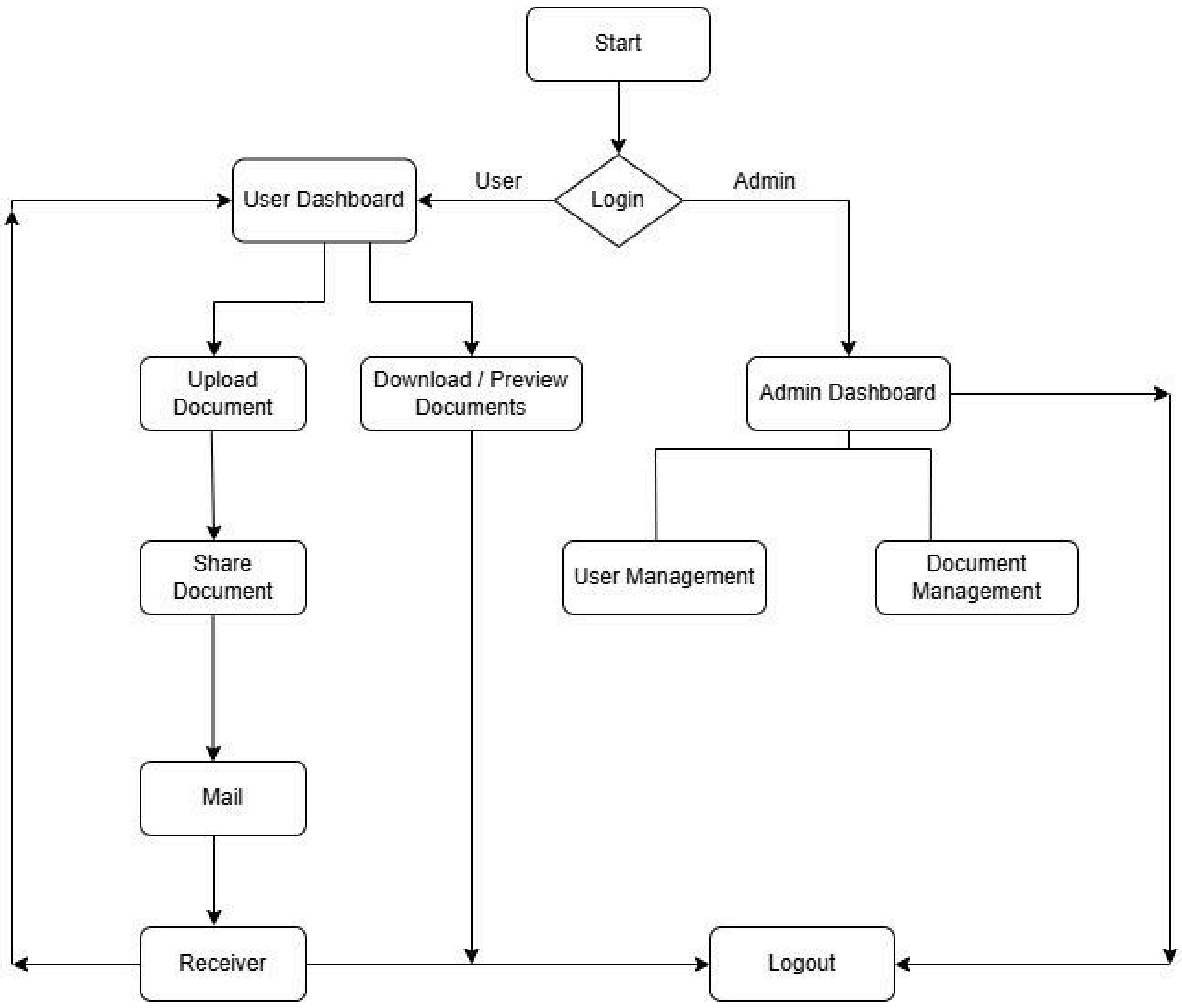
2. Admin Module



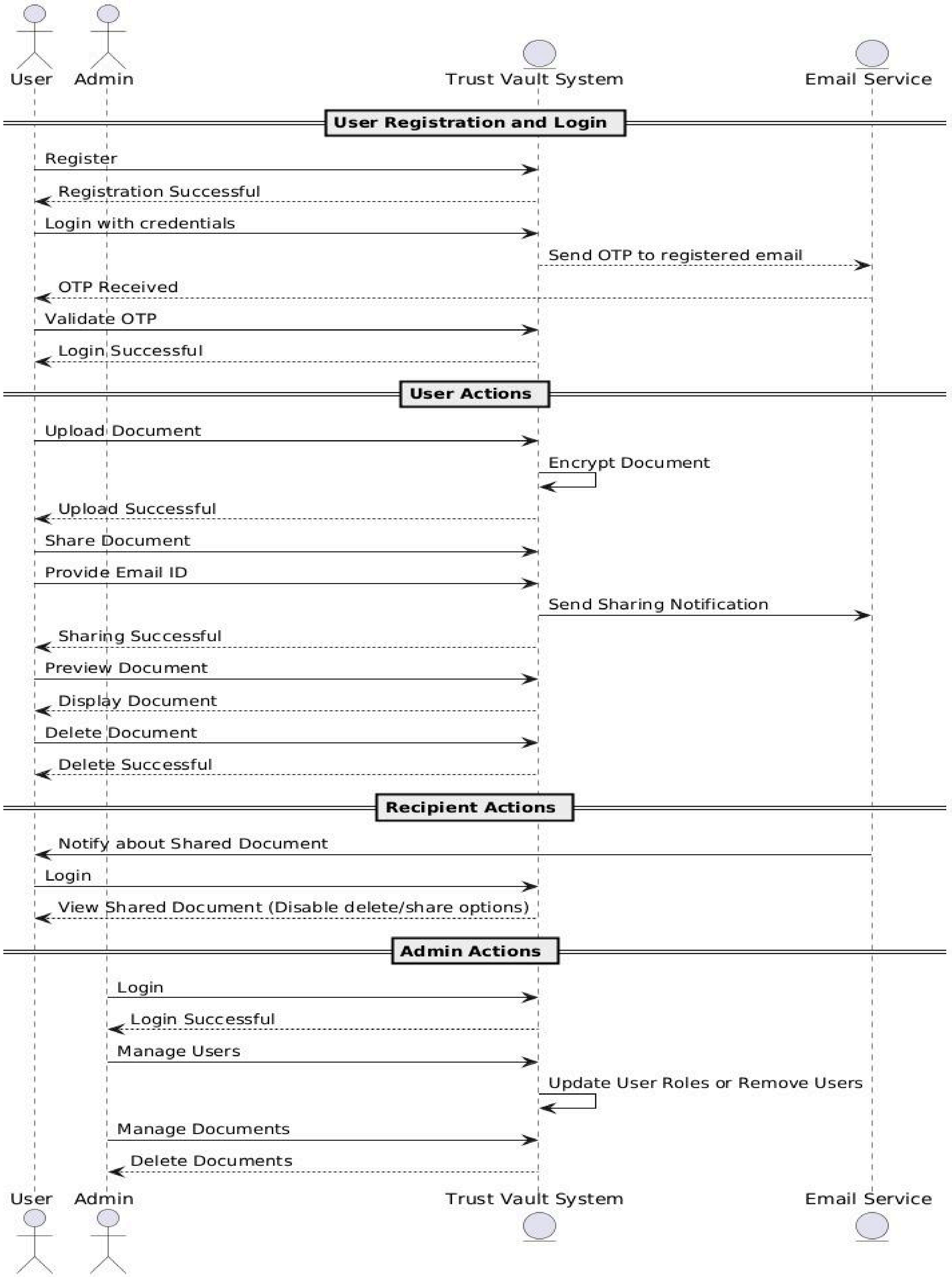
- **Use case Diagram :**



- **Activity Diagram :**



- **Sequence Diagram :**


User
Admin
Trust Vault System
Email Service
User Registration and Login
Register
Registration Successful
Login with credentials
Send OTP to registered email
OTP Received
Validate OTP
Login Successful
User Actions
Upload Document
Encrypt Document
Upload Successful
Share Document
Provide Email ID
Send Sharing Notification
Sharing Successful
Preview Document
Display Document
Delete Document
Delete Successful
Recipient Actions
Notify about Shared Document
Login
View Shared Document (Disable delete/share options)
Admin Actions
Login
Login Successful
Manage Users
Update User Roles or Remove Users
Manage Documents
Delete Documents
User
Admin
Trust Vault System
Email Service