**INDEX**

|  |  |  |
| --- | --- | --- |
| **Sr.no** | **Topic** | **Page no.** |
| **1** | **Introduction**  1.1 Purpose  1.2 Intended Audience and Reading Suggestions  1.3 Product Scope  1.4 References | **2** |
| **2** | **Overall Description**  2.1 Product Perspective  2.2 Product Functions  2.3 User Classes and Characteristics  2.4 Operating Environment  2.5 Design and Implementation  2.6 UML Diagrams | **4** |
| **3** | **External Interface Requirements** 3.1 User Interfaces 3.2 Software Interfaces  3.3 Hardware Interfaces 3.4 Communication Interfaces | **15** |
| **4** | **System features**  4.1 system feature 1  4.2 system feature 2  4.3 system feature 3  4.4 system feature 4  4.5 system feature 5  4.6 system feature 6  4.7 system feature 7  4.8 system feature 8  4.9 system feature 9 | **16** |
| **5** | **Other Non-Functional Requirement** 5.1 Performance Requirements  5.2 Scalability Requirements  5.3 Security Requirements  5.4 Business Rules  5.5 Availability Requirements | **23** |
| **6** | **Appendices** 6.1 Glossary 6.2 Future Enhancements 6.3 References | **25** |

**User Catalog Management System**

**1. Introduction**

**1.1 Purpose**

The User Management System (UMS) is a comprehensive solution designed to manage user data securely and efficiently. It incorporates features like role-based login, password hashing, email notifications, and advanced user data handling. The system ensures robust security, seamless user interaction, and effective administrative functionalities.

**1.2 Intended Audience and Reading Suggestions**

This documentation is intended for:

* Developers: To understand the system architecture and functionalities for further development or maintenance.
* System Administrators: To manage the system effectively.
* Testers: To validate the functionality and identify potential issues.
* Stakeholders: To comprehend the capabilities of the system.

**1.3 Product Scope**

The User Management System aims to streamline user data management with features such as:

* Secure authentication and authorization using Spring JWT.
* Role-based access control for Users and Admins.
* Password security using SHA-512 hashing.
* Email notifications to users when their data is added or updated by the admin.
* Advanced search, sorting, and PDF export functionalities.
* Restricted webpage access post-logout until re-login.

**1.4 References**

* Spring Framework Documentation: <https://spring.io/docs>
* JWT Documentation: <https://jwt.io>
* SHA-512 Hashing: <https://www.ietf.org/rfc/rfc6234.txt>

**2. Overall Description**

**2.1 Product Perspective**

The UMS is a standalone web application built using Spring Boot, designed to simplify user management tasks for administrators. It integrates with an email service for notifications and utilizes JWT for secure token-based authentication. The architecture follows modern design principles, ensuring scalability and maintainability.

**2.2 Product Functions**

The primary functions of the UMS include:

* User data management with CRUD operations.
* Authentication using Spring JWT.
* Role-based access control.
* Password hashing for enhanced security.
* Email notifications.
* Search, sort, and filter capabilities for user lists.
* PDF export of user data.
* Secure session management.

**2.3 User Classes and Characteristics**

* Admin: Can manage user data, view lists, and perform advanced actions like exporting data and sending notifications.
* User: Can access their data and perform limited interactions based on their role.
* System Auditor: Reviews logs and ensures compliance with security protocols.

**2.4 Operating Environment**

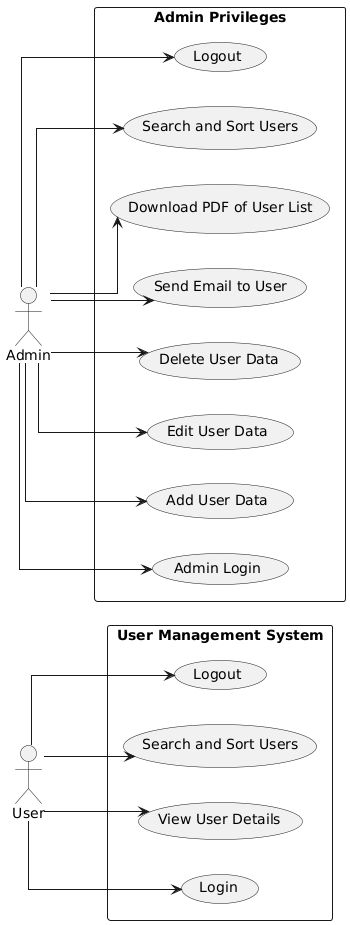
* Backend: Spring Boot framework.
* Frontend: Any web browser supporting modern standards (e.g., Chrome, Firefox).
* Database: Relational database (e.g., MySQL or PostgreSQL).
* Email Service: SMTP-compatible email server.
* Server Environment: Linux-based servers for deployment.

**2.5 Design and Implementation Constraints**

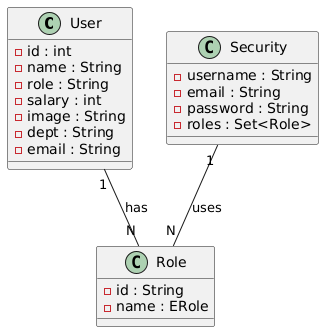
* Follows RESTful API principles.
* Secure password handling using SHA-512.
* Limited to two roles: Admin and User.
* System scalability for future integration with external systems.

**2.6 UML Diagrams:**

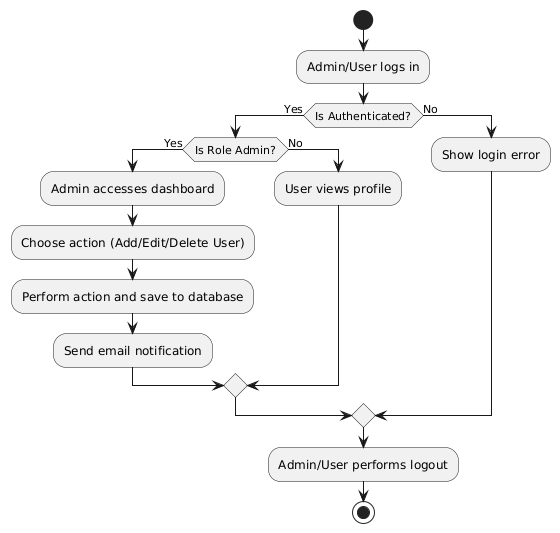
**Use Case Diagram :-**

****

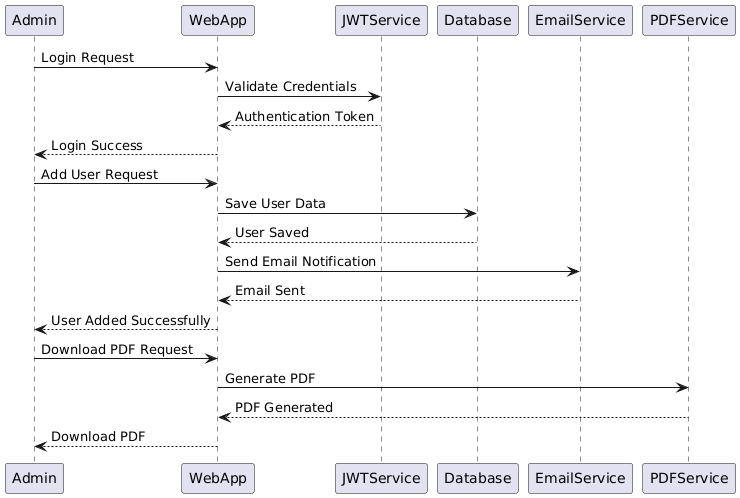
**Class Diagram :-**

****

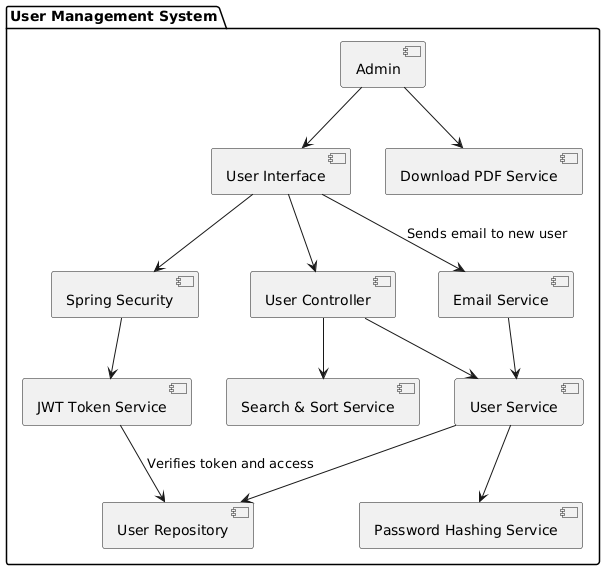
**Activity Diagram :-**

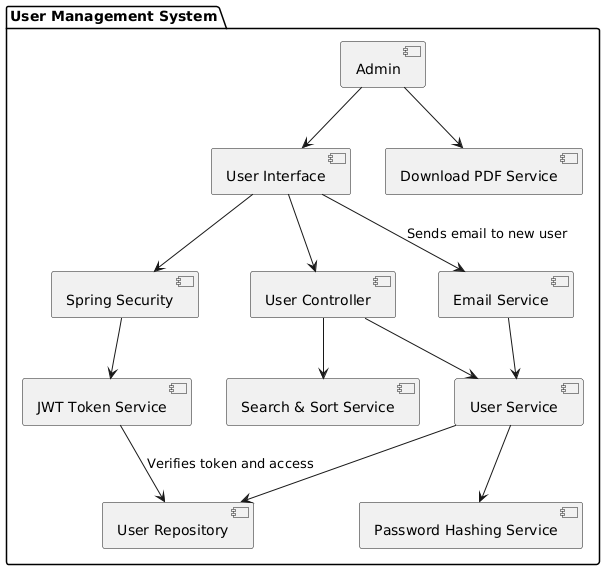
****

**Sequence Diagram :-**

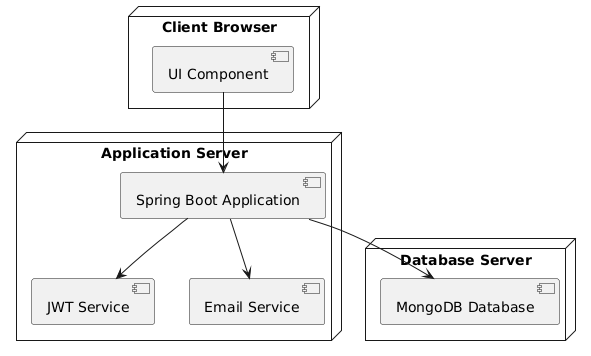
****

**Component Diagram :-**

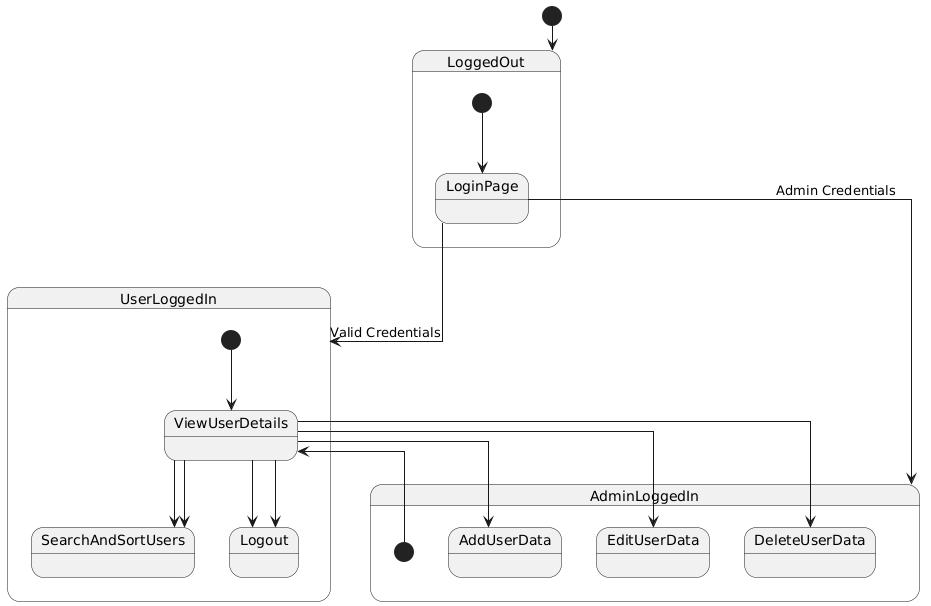
****

**  
ER Diagram :-**

**Deployment Diagram :-**

****

**State Diagram :-**

****

**Object Diagram :-**

**A diagram of a user flow

Description automatically generated**

**3. External Interface Requirements**

**3.1 User Interfaces**

* Login page with role-based access.
* User dashboard displaying relevant user information.
* Admin dashboard with advanced functionalities like search, sort, and export.
* Error pages for unauthorized access and system errors.

**3.2 Software Interfaces**

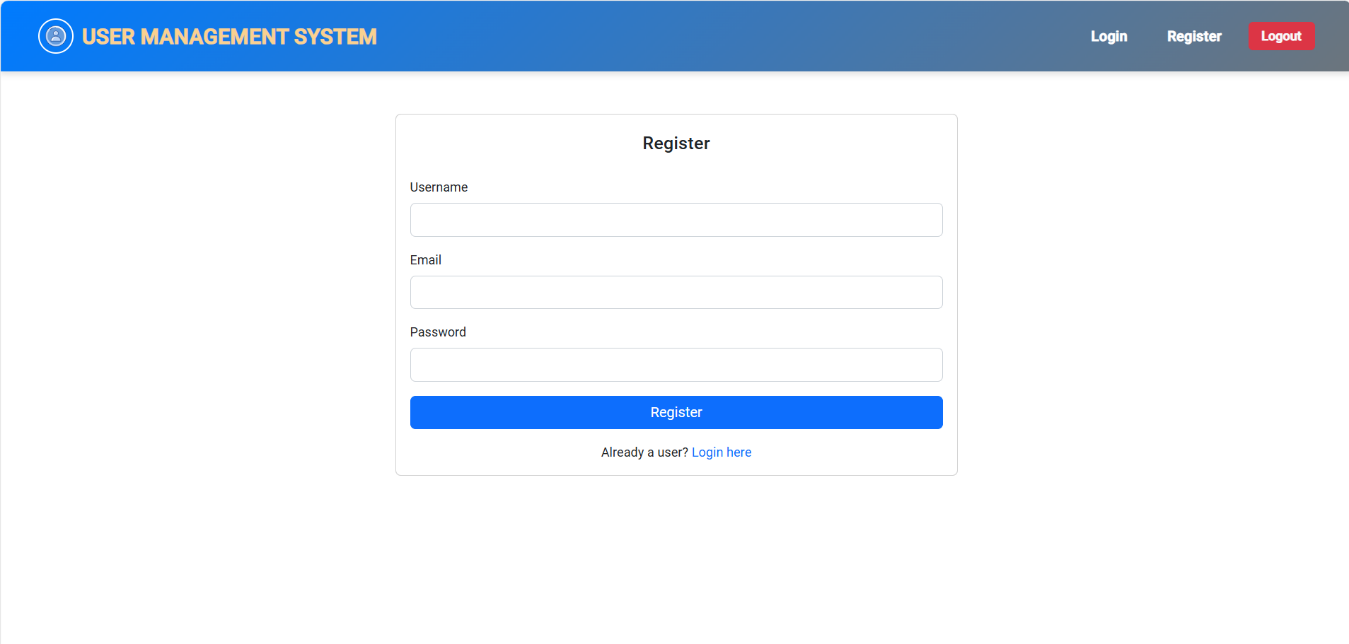
* JWT for authentication and authorization.
* Database for storing user and role information.
* SMTP for email notifications.
* Integration with third-party libraries for PDF generation and password hashing.

**3.3 Hardware Interfaces**

* Server requirements include 4 CPU cores and 8GB RAM for optimal performance.
* Database server with at least 100GB storage capacity for user records.

**3.4 Communication Interfaces**

* HTTP/HTTPS protocols for client-server communication.
* Secure connections for email notifications.

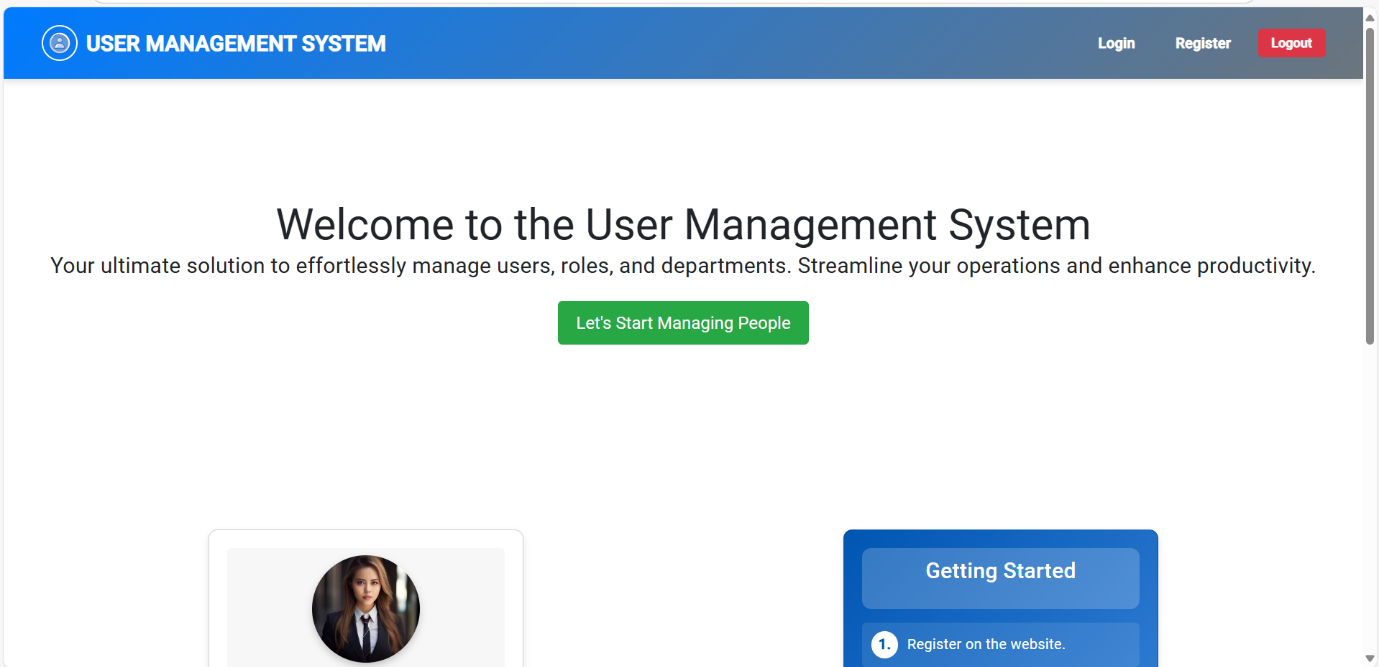
**Register**

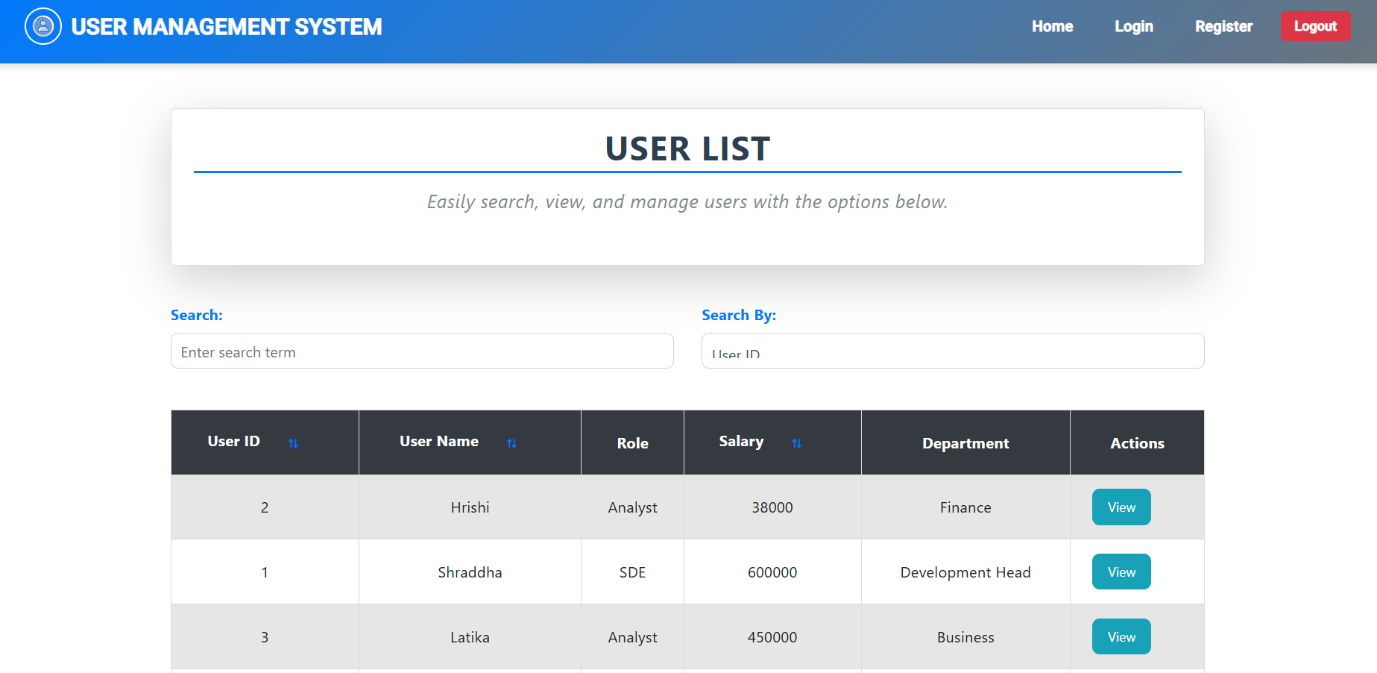
**A screenshot of a computer

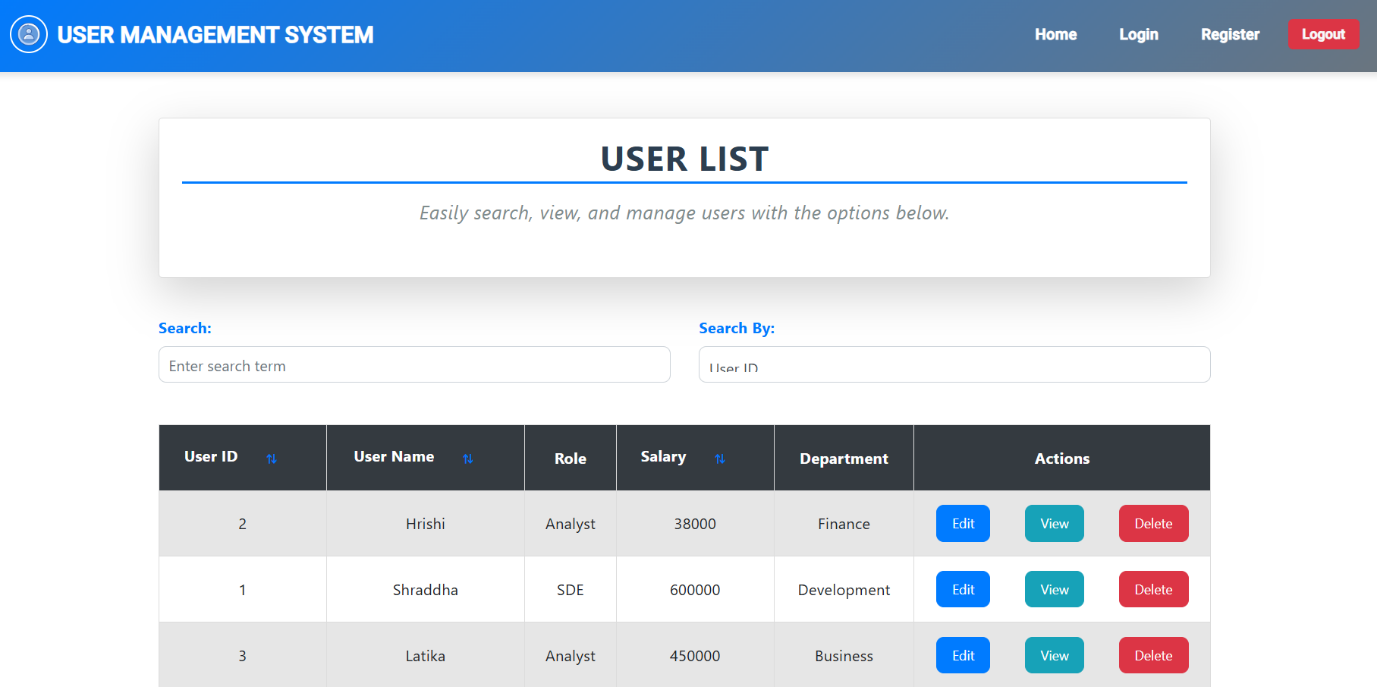
Description automatically generated**

**Login**

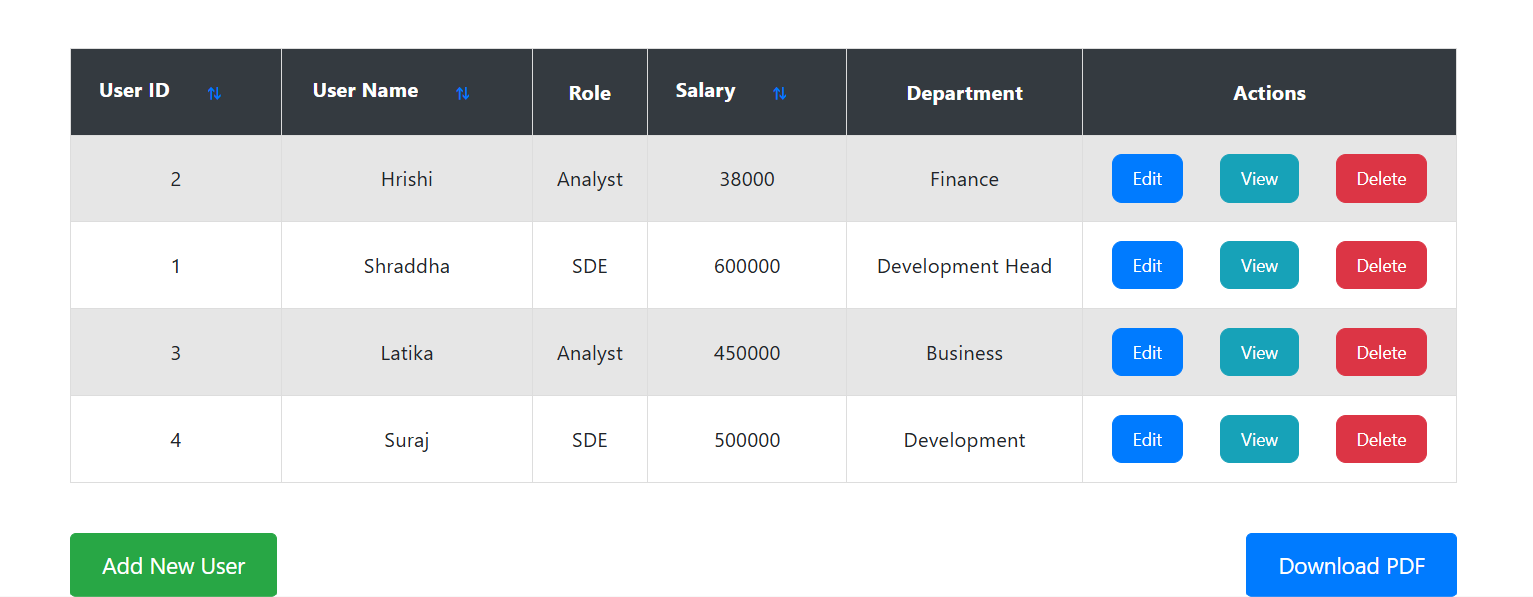
**A screenshot of a computer

Description automatically generated Front Page**

** Home Page :-User View**

****

**Home Page :- Admin View 1**

** Home Page :- Admin View 2**

**A screen shot of a computer

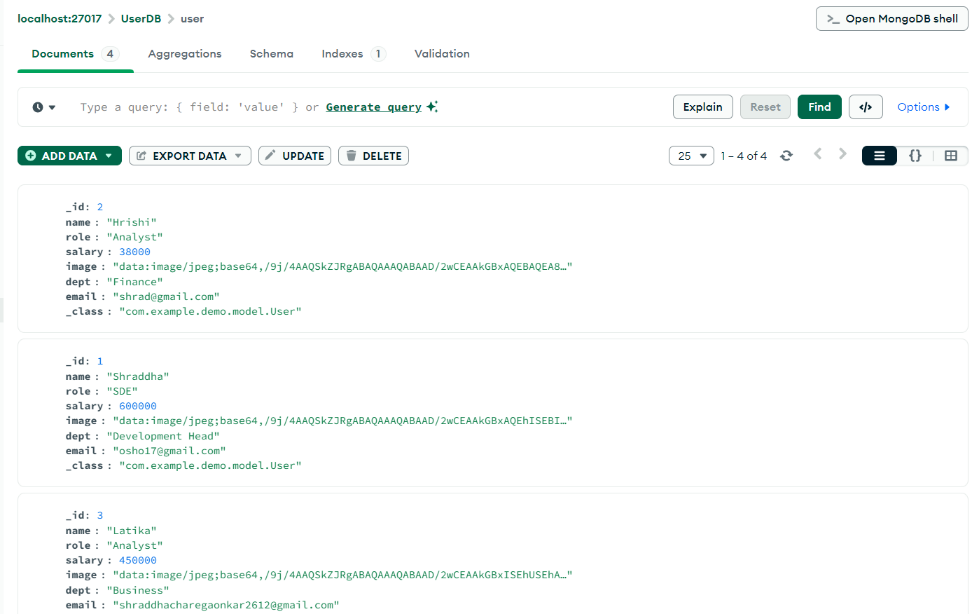
Description automatically generated**

**Edit Page**

**A screenshot of a computer

Description automatically generated**

**View Page**

****

**User Database – user collection**

**A screenshot of a computer

Description automatically generated**

**Security Database – users collection**

**A screenshot of a computer

Description automatically generated**

**Security Database – roles collection**

**4. System Features**

**4.1 System Feature 1**: Secure Authentication and Authorization

* Spring JWT ensures token-based login.
* Role-based access control for Admin and User roles.
* Supports session expiration and token refresh mechanisms.

**4.2 System Feature 2**: Password Security

* User passwords are hashed using SHA-512 before storage.
* Enhances security by preventing password exposure.
* Includes validation for password strength.

**4.3 System Feature 3**: Email Notifications

* Admins can notify users via email when their data is added or updated.
* Uses a secure SMTP connection.
* Customizable email templates for various notifications.

**4.4 System Feature 4**: Data Management and Accessibility

* Admins can search, sort, and filter user data based on multiple criteria.
* Users can view their data securely.
* Provides pagination for efficient data handling.

**4.5 System Feature 5**: PDF Export and Session Management

* Admins can export user lists to PDF format.
* Users are restricted from accessing pages post-logout until re-login.
* Supports audit logging for session activity.

**4.6 System Feature 6**: Advanced Search and Sorting

* Multi-criteria search functionality for user data.
* Sorting options by name, role, department, and salary.
* Filters for date ranges and role-specific data.

**4.7 System Feature 7**: Role Management

* Admins can assign or modify roles for users.
* Supports the addition of new roles for future scalability.

**4.8 System Feature 8**: Logging and Monitoring

* Tracks all system activities for security audits.
* Logs user login/logout events and data changes.
* Monitoring dashboards for admin oversight.

**4.9 System Feature 9**: Error Handling

* Displays user-friendly error messages for system issues.
* Detailed error logs for debugging and resolution.

**5. Other Nonfunctional Requirements**

**5.1 Performance Requirements**

* The system should handle up to 1000 concurrent users.
* Email notifications should be sent within 5 seconds of triggering.
* PDF export should process lists of up to 10,000 records within 10 seconds.

**5.2 Scalability Requirements**

* System should allow integration with additional authentication methods like OAuth.
* Supports scalability for databases with millions of user records.

**5.3 Security Requirements**

* Ensures data encryption during transmission and at rest.
* Implements measures to prevent common vulnerabilities like SQL injection and CSRF.

**5.4 Business Rules**

* Only Admins can manage user data and perform advanced actions.
* Users must have unique email IDs for account creation.
* Passwords must meet complexity requirements.
* System must comply with GDPR and similar data protection regulations.

**5.5 Availability Requirements**

* System uptime should be at least 99.9%.
* Scheduled maintenance should be announced 24 hours in advance.

**6. Appendices**

**6.1 Glossary**

* JWT: JSON Web Token, a compact token format used for secure authentication.
* SHA-512: A cryptographic hash function used for password security.
* SMTP: Simple Mail Transfer Protocol, used for email communications.
* CRUD: Create, Read, Update, Delete operations.

**6.2 Future Enhancements**

* Integration with mobile applications.
* Support for multi-factor authentication.
* Real-time user activity monitoring.

**6.3 References**

* Spring Boot Documentation: <https://spring.io/projects/spring-boot>
* SHA-512 Algorithm: <https://en.wikipedia.org/wiki/SHA-2>
* PDF Libraries: <https://pdfbox.apache.org/>