**Requirement Gathering and Analysis Phase**

**Data Flow Diagram & User Stories**

|  |  |
| --- | --- |
| Date | 06 July 2024 |
| Team ID | SWTID1720019632 |
| Project Name | HOUSE RENT APP USING MERN |
| Maximum Marks |  |

**Data Flow Diagrams:**

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

**A diagram of a system

Description automatically generated**

**User Stories**

Use the below template to list all the user stories for the product.

| **User Type** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Acceptance criteria** | **Priority** | **Release** |
| --- | --- | --- | --- | --- | --- | --- |
| Renter, Owner | User Authentication | USN-1.1 | As an Owner or Renter, I can register for the application by entering my email, password, and confirming my password. In the Registration page | Form fields: email, password, confirm password, role selection (renter/owner). Successful registration should navigate to dashboard. | Low | Sprint-1 |
| Renter, Owner |  | USN-1.2 | Implement separate registration API’s for handling Owner and Renter Data | API should create user in the database, handle errors, and return success/failure response | Medium | Sprint-1 |
| Renter, Owner |  | USN-1.3 | Integrate frontend with backend for registration | Successful integration should allow new users to register and log in immediately. | High | Sprint-1 |
| Renter, Owner |  | USN-1.4 | As an Owner or Renter, I can login from any device through Login page by entering my registered email, password | API should authenticate user, handle errors, and return user-specific token | Low | Sprint-1 |
|  |  |  |  |  |  |  |
| Renter | Property Listing | USN-2.1 | Renter Should be able to See the List of properties in the property Listing page | Page should display list of available properties with summary information | Low | Sprint-2 |
| Renter |  | USN-2.2 | Implement property listing API | API should retrieve list of properties from the database, handle errors, and return properties | Medium | Sprint-2 |
| Renter |  | USN-2.3 | Integrate frontend with backend for listing the properties posted by owners | Should display properties on the listing page posted by owners | High | Sprint-2 |
| Owner | Property Submission | USN-3.1 | Owner Should be able to submit the Properties details in a form in the property Submission page | Form should contain property name, description, address, price, availability. Successful submission should add property to the database | Low | Sprint-3 |
| Owner |  | USN-3.2 | Implement property submission API | API should add new property to the database, handle errors, and return success/failure response. | Medium | Sprint-3 |
| Owner |  | USN-3.3 | Integrate frontend with backend for submission of property by Owner which will be posted in property listing page of Renters. | On Successful integration should allow owners to submit new properties. | High | Sprint-3 |
|  |  |  |  |  |  |  |
| Renter | Property Details | USN-4.1 | Renters Should be able to see the details of each listed property in The Details Page | Page should display detailed information about a selected property, including owner contact form. | Low | Sprint-4 |
| Renter |  | USN-4.2 | Implement property details API by fetching the data posted by the Owners in the Property Submission Forms | API should retrieve detailed information of a specific property from the database | Medium | Sprint-4 |
| Renter |  | USN-4.3 | Integrate frontend with backend for details | Successful integration should display detailed property information on the details | High | Sprint-4 |
|  |  |  |  |  |  |  |
| Renter | Booking Property | USN-5.1 | Renter should be able to request Booking by filling form details, in the Property Details page and, the form is sent to the property Owner | Forms with renter details, booking dates. Successful submission should create a booking request with status "pending". | Medium | Sprint-5 |
| Renter |  | USN-5.2 | Implement booking API | API should create a new booking in the database, handle errors, and return success/failure response. | Medium | Sprint-5 |
| Renter |  | USN-5.3 | Integrate frontend with backend for booking status update | Successful integration should allow renters to submit booking | High | Sprint-5 |
|  |  |  |  |  |  |  |
| Admin | Admin Dashboard | USN-6.1 | Admin should be able to Handle Users, Owners and the Booking Status of Property | Dashboard should display pending user approval requests with approve/reject buttons. | Medium | Sprint-6 |
| Admin |  | USN-6.2 | Implement admin monitoring API | API should update user approval status in the database | High | Sprint-6 |
| Admin |  | USN-6.3 | Integrate frontend with backend for monitoring tools | successful integration should allow admin to approve/reject user requests. | High | Sprint-6 |
|  |  |  |  |  |  |  |
| Owner | Property Management | USN-7.1 | Owner should be able to Manage Property by approving or rejecting the Renter and updating the Booking Status. | Owners should be able to create, read, update, and delete properties. | Medium | Sprint-7 |
| Owner |  | USN-7.2 | Implement property CRUD API | API should handle property CRUD operations. | High | Sprint-7 |
| Owner |  | USN-7.3 | Integrate frontend with backend for property CRUD | Successful integration should allow owners to manage their properties. | High | Sprint-7 |