**Smart Neighborhood Exchange**

#### **Project Objectives**

The **Smart Neighborhood Exchange** web application enables local communities to collaboratively share and manage resources like tools, space, and equipment. It offers a smart scheduling and reservation system designed to optimize resource usage, reduce unnecessary purchases, and strengthen community connections through shared access to items and spaces.

#### **Feature Descriptions**

1. **Resource Sharing and Scheduling**
   1. **User Story:** As a community member, I want to list my tools and equipment for sharing, so others can reserve them without conflict.
   2. **Acceptance Criteria:**
      1. Users can list items (tools, appliances, equipment) with availability status.
      2. Reservations are handled by a calendar/booking system, preventing scheduling conflicts.
2. **Community Events and Spaces**
   1. **User Story:** As a user, I want to reserve spaces like gardens or meeting rooms, so I can organize community events or workshops.
   2. **Acceptance Criteria:**
      1. Users can list and book community spaces, with descriptions and availability settings.
      2. Events are visible to other users, with an option to join or express interest.
3. **Smart Availability and Notifications**
   1. **User Story:** As a user, I want to be notified when a reservation is confirmed or when my booking is near expiration.
   2. **Acceptance Criteria:**
      1. Notifications are sent for booking confirmations, conflicts, or upcoming due dates.
      2. Real-time notifications and reminders for both lenders and borrowers.
4. **User Reviews and Profiles**
   1. **User Story:** As a participant, I want to review others and be reviewed after exchanges, building trust within the community.
   2. **Acceptance Criteria:**
      1. Users have profiles with names, ratings, and participation history.
      2. A review system allows users to rate each other and add comments.
5. **Security and Trust**
   1. **User Story:** As a user, I want my data and exchanges to be secure, ensuring trust within the app.
   2. **Acceptance Criteria:**
      1. Users are verified through a simple ID check.
      2. Secure handling of user data, with clear visibility controls for privacy.

#### **Technical Specifications**

* **Backend**: Python (Flask)
* **Frontend**: HTML, CSS, JavaScript (React or Vue.js)
* **Database**: SQLite
* **APIs**:
  + RESTful APIs for handling CRUD operations (Create, Read, Update, Delete) of resources and user information.
  + Notification system API for real-time updates.
* **Security Requirements**:
  + Server-side data validation for all user inputs.
  + Protection against SQL injection, XSS, and CSRF vulnerabilities.
  + User verification with basic ID checks and profile privacy controls.

#### **Timeline and Milestones**

|  |  |  |
| --- | --- | --- |
| **Milestone** | **Tasks** | **Timeline (Days)** |
| Day 1-3: Project Planning & Setup | Create PRD, design wireframes, and set up the development environment and version control. | 3 |
| Day 4-6: Database Design & Backend Development | Design database schemas for users, resources, messages, and reviews. Implement user authentication. | 3 |
| Day 7-9: Backend Development Continued | Develop CRUD operations for resources, implement messaging and review systems. | 3 |
| Day 10-12: Frontend Development | Create web pages for homepage, dashboard, profile, listing details, and integrate with backend APIs. | 3 |
| Day 13-14: Frontend Interactivity | Add form validation, user feedback, and complete the messaging interface. | 2 |
| Day 15: Testing | Conduct unit testing and verify backend functionality, frontend responsiveness, and UI/UX. | 1 |
| Day 16: Validation & Security | Finalize data validation, protect against security threats (e.g., SQL injection, XSS, CSRF). | 1 |
| Day 17: Documentation | Create database schemas, ER diagrams, user manual, and requirements.txt for setup. | 1 |
| Day 18: Final Review | Conduct final review, prepare presentation, and ensure project completion. | 1 |