# **Threads**

Design Document

**Overview:** Threads is a cutting-edge social media platform designed to foster meaningful connections and vibrant communities online. It provides users with a seamless interface to share their interests, engage in discussions, and connect with like-minded individuals.

**Target Audience:** Threads caters to a diverse audience of social media enthusiasts, content creators, professionals, and individuals seeking genuine connections and engaging conversations online.

## **1. Features**

### Core Features:

* Allow users to create profiles with customizable information, including profile pictures, bios, and interests.
* Enable users to create and join communities based on shared interests, hobbies, or professional affiliations.
* Provide a platform for users to share posts, photos, videos, and links with their followers and communities.

### Additional Features:

* Secure user authentication and account management system for seamless login and registration.
* Curate a personalized explore page for users to discover new communities and trending content.
* Implement real-time updates to display live feeds and notifications as activities occur.

## **2. Architecture**

Threads follows a client-server architecture, with the frontend and backend components interacting via RESTful APIs.

### Frontend:

* Framework: Next.js
* State Management: Redux
* Routing: React Router
* Styling: CSS and Tailwind
* HTTP Requests: Axios

### Backend:

* Framework: Express.js
* Database: MongoDB (via Mongoose)
* Authentication: Passport.js
* API Documentation: Postman

### Deployment:

* Frontend: Hosted on a platform like Vercel
* Backend: Hosted on a platform like Heroku

## **3. User Interface Design**

### Home Page:

* Dynamic interface showcasing trending posts, communities, and personalized recommendations.

### Profile Page:

* Customizable user profile with options to edit profile information, view posts, and manage settings.

Community Page:

* Dedicated page for each community displaying recent posts, members, and community details.

## **4. Security Considerations**

* Implement proper input validation and sanitization to prevent injection attacks.
* Use secure authentication mechanisms (if implemented) such as JWT tokens or OAuth.
* Ensure data privacy and protection by encrypting sensitive information and passwords stored in the database.

## **5. Testing Strategy**

* Conduct integration testing to ensure seamless interaction between frontend and backend.
* Test user flows and edge cases to identify and resolve any bugs or issues.

**6. Database Schema:**

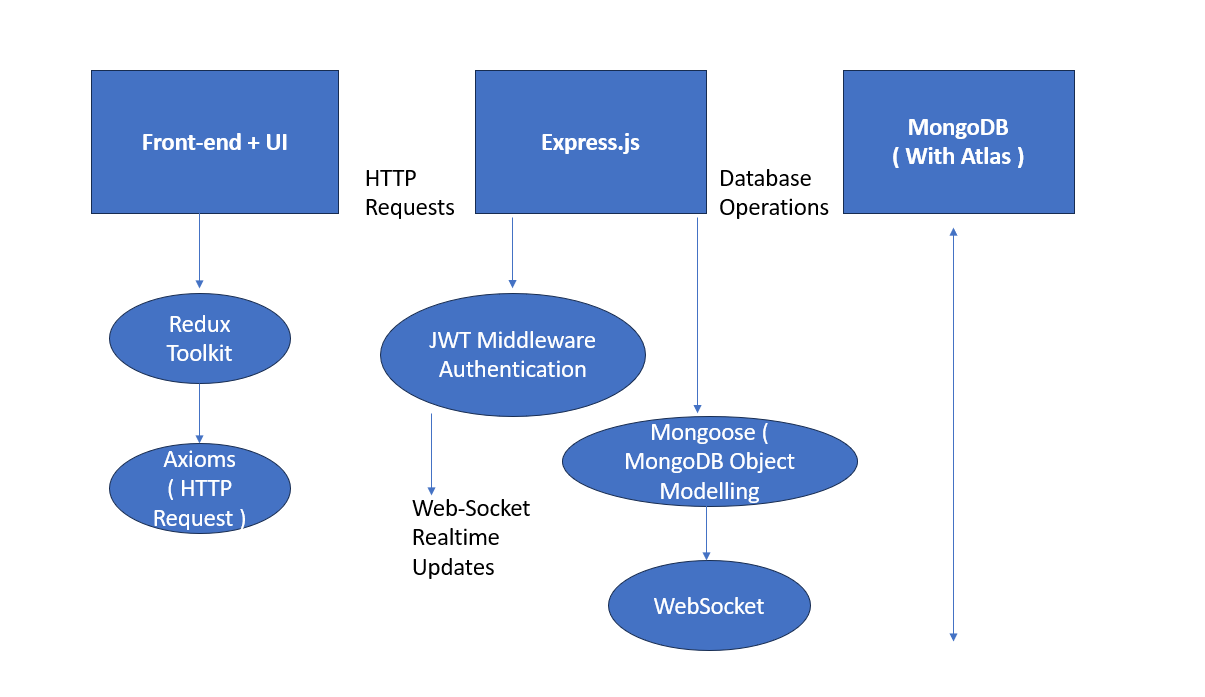
* **Users:** 
  + Unique user ID.
  + Username (unique).
  + Full name.
  + Profile image URL.
  + Bio/description.
  + Array of threads authored by the user.
  + Boolean field indicating whether the user has completed onboarding.
  + Array of communitiesthe user is a member of.
* **Community:**
* Unique community ID.
* Creator's username.
* Community name.
* Community image URL.
* Community description/bio.
* ID of the user who created the community.
* Array of threads posted in the community.
* Array of members who have joined the community.
* **Thread:**
  + Content of the thread.
  + ID of the user who authored the thread.
  + ID of the community the thread belongs to.
  + Timestamp of when the thread was created.
  + Parent ID for thread hierarchy (if applicable).
  + Array of child threads (replies) associated with the thread.

## 

## **7. API Endpoints:**

## Endpoint definitions for User Authentication, User Profiles, Community Management, Content Sharing, Messaging, and Notifications.

## **8. Technology Stack:**



## **9. Conclusion**

Threads aims to redefine the social media experience by providing users with a platform for authentic connections, engaging conversations, and vibrant communities. With its intuitive interface, comprehensive features, and robust architecture, Threads promises to deliver a seamless and enjoyable social networking experience for users across the globe.