PRD: MockMate – AI-Powered Mock Interview Matching Platform

1. Problem Statement

Many students preparing for technical interviews struggle to find reliable mock interview partners. Based on discussions with classmates and peers preparing for software engineering interviews, a common pain point is difficulty finding available partners at the right time. Practicing only with friends often leads to scheduling conflicts due to different availability.

Existing solutions are fragmented or inconvenient, and there is no simple platform designed specifically for students to search, filter, and schedule mock interview sessions based on interview type, level, and availability.

This results in inconsistent interview practice and reduced preparation effectiveness.

2. Target Users

Primary Users

Students and software engineering candidates preparing for interviews, including:

- Computer science students preparing for internships
- New graduate software engineering candidates
- Experienced engineers preparing for job transitions

User Characteristics

Goals:

- Practice mock interviews regularly
- Find partners with similar experience level
- Practice specific interview types (behavioral, coding, system design)

Frustrations:

- Cannot find available partners easily
- Scheduling conflicts with friends
- No centralized platform for mock interview coordination

Behaviors:

- Currently rely on friends or classmates
- Communicate via messaging platforms

- Coordinate schedules manually

Needs:

- Easy partner discovery
- Ability to filter partners by interview type and level
- Ability to view partner availability and schedule sessions

3. User Stories (MoSCoW Prioritized)

Must Have

1. User Registration and Login
   As a user, I want to register and log in so that I can use the platform securely.
2. Profile Creation
   As a user, I want to fill in my profile information including interview type, experience level, and availability so that others can find me.
3. Partner Search with Filters
   As a user, I want to search for mock interview partners using filters such as interview type (behavioral, coding, system design) and level (intern, new graduate, experienced) so that I can find suitable partners.
4. View Partner Profile
   As a user, I want to view another user's profile so that I can evaluate if they are a suitable partner.
5. Schedule Mock Interview Session
   As a user, I want to select an available time slot and book a mock interview session so that I can practice interviews.

Should Have

6. Availability Management
   As a user, I want to set my availability so that others can book sessions with me.
7. Session Management
   As a user, I want to view my upcoming scheduled sessions so that I can track my interview practice.

Could Have

8. Meeting Link Storage
   As a user, I want to store a Zoom or Google Meet link for scheduled sessions so that we can conduct the interview.

Won’t Have (Out of Scope)

- AI interview feedback

- Video calling system integration
- Real-time collaborative coding editor
- Notifications system
- Ratings and review system

These features are excluded due to timeline constraints.

4. Success Metrics

The MVP will be considered successful if users can complete the core workflow:

Functional success criteria:

- Users can register and log in
- Users can create and update profiles
- Users can search for partners using filters
- Users can view partner profiles
- Users can schedule mock interview sessions

Technical success criteria:

- System supports multiple users
- Data is stored and retrieved correctly
- Sessions are scheduled and displayed correctly

Usability success criteria:

- Users can complete the partner search and scheduling workflow without errors

5. Technical Constraints

Team Constraints

- Team size: 2 developers
- Timeline: 10 days

This requires limiting scope to essential MVP features only.

Technology Stack

Frontend:

- React

Backend:

- Spring Boot REST API

Database:

- Firebase Firestore

Authentication:

- Firebase Authentication

Meeting Links:

- External Zoom or Google Meet links (user-provided)

System Architecture

- React Frontend
  ↓
  Spring Boot Backend
  ↓
  Firebase Database

Technical Limitations

Due to the 10-day timeline, the system will:

- Use simple REST APIs
- Use Firebase for faster backend development
- Avoid complex real-time or AI features

6. Out of Scope

The following features will NOT be implemented in this MVP:

- AI interview feedback
- Video call implementation
- Real-time chat
- Collaborative coding editor
- Recommendation algorithms
- Notification system
- Mobile native apps

These features may be considered in future versions.