# Project Planning & Management

## Project Proposal

### Overview

The "Movie Recommendation App using React" is designed to provide users with personalized movie recommendations based on genres, trending films, and user preferences. The application will feature an intuitive user interface that enables seamless search, filtering, and browsing of movies. The app will integrate a public movie API, such as The Movie Database API (TMDb), to fetch real-time movie data, ensuring an up-to-date movie catalog.

### Objectives

- Develop a user-friendly movie recommendation platform.  
- Implement filtering and search functionalities for easy navigation.  
- Integrate a third-party API to fetch real-time movie data.  
- Ensure smooth state management using Redux.  
- Optimize performance and responsiveness for a seamless user experience.

### Scope

- Frontend development using React.  
- Backend support via Node.js for API integration and data handling.  
- Implementation of Redux for state management.  
- API integration for fetching movie details, ratings, and images.  
- User-friendly UI with filtering and pagination features.

## Project Plan

### Timeline (Gantt Chart Overview)

Requirement Gathering & Planning – 1 week  
UI/UX Design & Prototyping – 2 weeks  
API Integration & Backend Setup – 2 weeks  
Frontend Development – 4 weeks  
Deployment & Final Review – 1 week

### Milestones

- Week 1: Project initiation, requirements gathering.  
- Week 3: Completion of UI/UX design.  
- Week 5: Backend and API integration finalized.  
- Week 9: Completion of frontend implementation.  
- Week 11: Testing phase completed.  
- Week 12: Deployment and final project review.

### Deliverables

- Wireframes and UI prototypes.  
- Fully functional React-based movie recommendation app.  
- API integration with real-time data fetching.  
- Documentation and project report.

### Resource Allocation

- Frontend Development: React, Redux, CSS, JavaScript (2 Developers)  
- Backend Development: Node.js, API integration (1 Developer)  
- UI/UX Design: Prototyping, wireframes (1 Designer)  
- Testing & Deployment: QA and debugging (1 Tester)

## Risk Assessment & Mitigation Plan

- API Downtime: Implement fallback data caching mechanisms.  
- Performance Issues: Optimize API calls, use lazy loading.  
- UI/UX Challenges: Conduct user feedback sessions for improvements.  
- Timeline Delays: Monitor progress weekly, adjust workload as needed.  
- Security Issues: Implement authentication and secure API calls.

## KPIs (Key Performance Indicators)

- Response Time: Ensure API calls respond within 2 seconds.  
- System Uptime: Maintain 99.9% uptime.  
- User Engagement: Measure daily active users (DAU) and session duration.  
- Search Accuracy: Ensure search and filter accuracy above 90%.  
- Bug Resolution Rate: 95% of reported bugs resolved before deployment.