Project Plan: Development of Library Management System

1. Introduction

This project aims to develop **a Library Management System** that helps users search for and reserve books more smoothly, minimizes overdue penalties through timely reminders, and allows library staff to efficiently manage the book inventory by adding or removing books as needed.

The system will be developed as part of the Software Engineering Project course to provide hands-on experience in managing and executing a full software development lifecycle (SDLC). The project will be completed over 16 weeks, following industry best practices and methodologies.

2. Project Organization

- **Team Members and Roles:**
    - **Project Manager:** Monami Kirjavainen - Responsible for overall project coordination and communication.
    - **Lead Developer:** Riku Toivanen - Oversees software development and code quality.
    - **UI/UX Designer:** Victoria Vavulina - Designs the user interface and experience.
    - **DevOps Engineer:** Riku Kaartoaho - Sets up and maintains development, testing, and production environments.
- **Stakeholders:**
    - **Course Instructor & Product Owner:** Amir Dirin - Provides guidance and evaluates the project.

3. Risk Analysis

- **Technical Risks:**
  - **Integration Issues:** Difficulty in integrating various software components.
  - **Technology Stack Learning Curve:** Delays due to team members' unfamiliarity with new technologies.
- **Project Management Risks:**
  - **Scope Creep:** Potential for additional features being added beyond the original scope.
  - **Time Management:** Risk of falling behind schedule due to underestimated task durations.
  - **Team Coordination**: Miscommunication among group members leading to duplicated work or missing tasks.
- **Mitigation Strategies:**
  - **Regular Code Reviews:** To identify and resolve integration issues early.
  - **Time-Boxing:** Enforcing strict deadlines for each phase to minimize scope creep.
  - **Learning Sessions:** Scheduled training sessions for unfamiliar technologies.
  - **Clear Sprint Planning**: Clearly decide who is responsible for what task during sprint planning.
  - **Frequent Updates:** Use short meetings or tools like Discord to share progress and problems with the whole team, so others can step in if needed.

4. Software Resource Requirements

- **Software:**
  - **Development Tools:**
    - Version Control: **Git with GitHub**
    - Framework: **Next.js 15**
    - Backend as a Service: **Supabase** (PostgreSQL Database, Authentication, S3 Storage Bucket)
  - **Design Tools:**
    - UI/UX Design: **Figma**
  - **Testing Tools:**
    - Automated Testing Framework: **Playwright Unit and E2E Testing**
  - **CI/CD:**

        - **Docker**
        - **Github Actions**
        - **Vercel**
    - **Project Management:**
        - Project Management Software: **Jira**

5. Work Breakdown Structure (WBS)

1. **Requirement Analysis:**

   Gather and document requirements:

   Functional requirements were gathered with the product owner by reviewing the product scope and functionality. The essential functionalities include searching for books, borrowing and returning books and subscribing to groups for notifications about different subjects like book categories. Non-functional requirements such as usability and security were also documented.

   Client/stakeholder review and approval:

   The specific requirements for the project were reviewed by the product owner ensuring the product aligned with the academic objectives of the project.

2. **Design:**

   System architecture design:

   The system was designed to follow a client-server model. Next.js was selected to handle the front-end, while Supabase was chosen for the backend management and database. Supabase's authentication features will be used to handle the user related features such as borrowing and returning books and subscribing for notifications.

   UI/UX design:

   Visualizations for the site were created using figma to represent the sites display structure with positions for the main featues. The design emphazises accessibility and user experience to create a user-friendly product.

3. **Development:**

   Backend development:
   Supabase's will be configured to manage user authentication, database queries, and real-time notifications for users. SQL Will be used to automatically trigger automatic updates for book availability upon returning or borrowing books.

   Frontend development:
   The Next.js framework was chosen to develop the front-end application. It is used for modular components for searching, browsing, borrowing, and returning books.

   Database setup and integration:
   A relational schema was created in Supabase, consisting of tables for books, users, borrowing records, and notification subscriptions. Foreign key constraints were introduced to ensure data integrity. Integration between the frontend and backend was achieved through Supabase-provided APIs.

4. **Testing:**

   Unit testing:
   Playwright will be used to test individual components and functions in isolation. Book searching, borrowing and returning books, and notification logic will be tested and validated with predefined tests to ensure functionality.

   Integration testing:
   The integration within the frontend and backend will be tested using Playwright end-to-end testing. User actions such as searching books, and borrowing or returning them will also be tested to ensure correct results

   User acceptance testing (UAT):
   Trial sessions with users will be done to ensure the features are intuitive and complete. Feedback from these sessions will be used to refine the design and functionality of the product.

5. **Deployment:**

   Setup production environment:

A production environment was established using Vercel for the React frontend and Supabase's managed hosting for the backend and database.

Deployment of application:
The application was deployed and made accessible to users via a cloud-hosted URL. Database migrations were executed to initialize the production schema.

Post-deployment monitoring:
Supabase analytics will be reviewed to monitor performance and user activity.

6. **Documentation:**

   Technical documentation:
   Detailed documentation was prepared, including the system architecture, database schema, API endpoints, and configuration procedures. This ensured that future developers could maintain and extend the system.

   User manuals and training materials:
   A compact user manual will be created to assist users in performing tasks within the site such as searching for books and borrowing or returning them.

6. Project Schedule

- **Milestones:**
  - **Project Kickoff:** Week 1
    - Team formation and role assignment
    - Initial project briefing with supervisor
    - Setup of version control (GitHub) and collaboration tools
  - **Requirement Analysis Complete:** Week 2
    - Gather and document functional and non-functional requirements
    - Review requirements with supervisor and adjust based on feedback
  - **Design Phase Complete:** Week 3
    - System architecture design (React frontend, Supabase backend)

        - UI/UX wireframes and mockups
        - Database schema design in Supabase

    - **Development Phase Complete:** Week 4-6
        - Backend setup in Supabase (tables, authentication, triggers)
        - Frontend implementation of core features (search, catalogue view)
        - Integration of borrowing, returning, and notification subscriptions

    - **Testing Phase Complete:** Week 7
        - Unit testing of individual components using Playwright
        - Integration and end-to-end testing (React + Supabase workflows)
        - User Acceptance Testing (sample group of students)

    - **Deployment and Project Closure:** Week 8
        - Deployment of frontend (Vercel) and backend (Supabase)
        - Post-deployment monitoring and bug fixes
        - Submission of final documentation and project handover

7. Monitoring and Reporting Mechanisms

- **Weekly Meetings:** Regular team meetings will be conducted every **Monday** to review progress, address issues, and update task allocations. These meetings will serve as a platform for collaborative problem-solving and ensuring that all members remain aligned with the project objectives.

- **Progress Reports:** Bi-weekly progress reports will be prepared, summarizing completed tasks, pending activities, and any obstacles encountered. These reports will be shared among team members and the supervisor to maintain transparency and enable timely intervention where necessary.

- **Project Management Tool:** The project will be managed using **Jira**, which will facilitate task tracking, sprint planning, and progress monitoring. Jira's reporting features will allow the team to visualize workload distribution, identify bottlenecks, and measure progress against planned milestones.

- **Stakeholder Reviews:** Monthly reviews with the supervisor will be scheduled to ensure alignment with academic requirements and to accommodate any adjustments in scope or functionality. These sessions will provide opportunities to demonstrate interim progress and receive structured feedback.

- **Final Presentation:** A final presentation, including a live demonstration of the system and submission of the project report, will take place at the conclusion of the course. This will serve as the closing milestone, showcasing the outcomes achieved and reflecting on the project process.