Project: Kanban Board for OPT Portal

Team Members and Positions
Rahul Ashok Kodate: Product Manager
Shrimoyee Banerjee: UI/UX Designer

Reviewers
Gayathri Vummenthala: Project Manager
Gerardo Serrano: CEO & Co-founder

Project Proposal.

Project Objective:

Implementing a Kanban Board/Software will provide a centralized platform for managing all volunteers under Bear Brown & Company as well as Abecedarian LLC. This board will facilitate the division, sorting, and progression of projects from inception to completion.

Key Benefits:

1. **Centralized Project Management**: The Kanban Board will serve as a common space for tracking and managing all projects, ensuring that volunteers are assigned to specific tasks and projects efficiently.
2. **Enhanced Organization**: Projects will be systematically divided and sorted, allowing for clear visibility into the status and progress of each task.
3. **Volunteer Assignment**: Volunteers will be sorted into their respective projects, ensuring that each project is adequately staffed and responsibilities are clearly defined.
4. **Regular Updates**: The board will provide real-time updates on project status, enabling consistent monitoring and timely interventions when necessary.

By leveraging the Kanban Board, we will enhance our project management capabilities, improve volunteer coordination, and ensure the successful completion of all initiatives.

Project Scope:

In-Scope:

1. Assign volunteers to specific tasks.

2. Software selection: What kind of software do the Software Engineers of this Project prefer?
3. Set up and Configuration: The Board must consider the hierarchy of the company structure. Create templates and workflows for common project types.
4. User Roles and Permissions: Define the roles of users and assign appropriate permissions. Create user accounts for everyone in the organization.
5. Project and Task Management: Input existing projects and tasks into the Kanban board. Establish a system for creating, updating, and moving tasks through different project phases.
6. Set up automatic notifications for project updates.

Out-of-Scope:

1. Integration with the first part of the OPT portal.
2. Ongoing project management after initial setup and training.

Project Deliverables:

Deliverables

1. Fully configured Kanban Board ready for use.
2. User accounts and roles assigned.
3. The initial set of projects and tasks entered into the system.
4. Regular updates and reporting mechanisms established.

Constraints

1. The project must be completed in 6 weeks.

Risks

1. Potential changes in the team.
2. Technical issues with the Kanban Board during the testing phase.
3. Potential delays.

Potential Project Timeline

Week 1 & 2: Team selection, software, and language selection, initial setup with the workflow.
Week 3 & 4: User account creation, user role assignments, initial project entry.
Week 5 & 6: Training, testing, and Deployment.

Feasibility Study

1. Technical Feasibility:

   Do all the volunteers have the required software and coding capabilities?
2. Operational Feasibility:

   Can we integrate the Board with other parts of the OPT portal?

Team Members

Project Manager: To oversee the development process, manage timelines, and ensure the project stays on track.
Product Manager: To define the product requirements, work with stakeholders to understand needs, and ensure the software meets business objectives.
UI/UX Designer: To design an intuitive and user-friendly interface. This person will create wireframes, mockups, and prototypes.
Frontend Developer: To implement the design and build the user interface. They should be proficient in JavaScript frameworks like React, Vue.js, or Angular.
Backend Developer: To handle the server-side logic, database management, and integration of APIs. They should be skilled in languages like Node.js, Python, Ruby, or Java.

Requirements

1. Functional Requirements

1.1 User Roles & Permissions

- **Administrator**: Full access to all features, including user management and settings.
- **Project Manager**: Access to create, modify, and delete boards, tasks, and manage team members.
- **Team Member**: Access to view and update tasks, move tasks between columns, and comment on tasks.
- **Viewer**: Read-only access to boards and tasks.

1.2 Kanban Board Features

- **Board Creation**: Ability to create multiple Kanban boards.
- **Customizable Columns**: Allow users to create, rename, and arrange columns (e.g., To-Do, In Progress, Done).
- **Task Cards**:
    - **Add/Edit/Delete Tasks**: Users can create, modify, and remove tasks.
    - **Task Details**: Include title, description, assignee, due date, priority, and tags.
    - **Subtasks**: Option to add and track subtasks within a main task.
    - **Attachments**: Ability to attach files and documents to tasks.
    - **Comments**: Users can add comments to tasks for collaboration.
- **Task Movement**: Drag-and-drop functionality to move tasks between columns.
- **Notifications**: Email or in-app notifications for task assignments, due dates, and comments.

1.3 Reporting and Analytics

- **Task Tracking**: Track the status and progress of tasks.
- **Performance Metrics**:
- **Analytical Report:** Generate Reports on all the projects based on timeline, number of tasks, on-time tasks, etc.

1.4 Integrations

- **Calendar Integration**: Sync tasks with external calendars (Google Calendar, Outlook).
- **API Access**: Provide API access for custom integrations.

1.5 User Interface

- **Dashboard**: Centralized dashboard displaying boards, tasks, and notifications.
- **Search and Filter**: Ability to search and filter tasks by keywords, assignee, due date, and status.

2. Non-Functional Requirements

2.1 Performance

- **Load Time**: The application should load within 2 seconds.
- **Scalability**: The system should handle multiple boards and tasks without performance degradation.

2.2 Security *think about it*

2.3 Usability

- **Intuitive Design**: The interface should be user-friendly and require minimal training.

2.4 Reliability

- **Uptime**: The system should have a minimum uptime of 99.9%.
- **Backup and Recovery**: Implement automated data backup and recovery processes.

2.5 Compatibility

- **Browser Compatibility**: The application should be compatible with major browsers (Chrome, Firefox, Safari, Edge).
- **Device Compatibility**: The system should be responsive and work on desktops, tablets, and mobile devices.
-

3. Technical Requirements

3.1 Technology Stack

- **Frontend**: React, Vue.js, or Angular.
- **Backend**: Node.js, Python (Django/Flask), or Ruby on Rails.
- **Database**: MySQL, PostgreSQL, or MongoDB.
- **Hosting**: Cloud-based hosting on AWS, Azure, or Google Cloud.
- **Version Control**: Use Git for version control.

3.2 Development Environment

- **CI/CD Pipeline**: Set up Continuous Integration/Continuous Deployment for automated testing and deployment.
- **Testing Frameworks**: Use tools like Jest, Mocha for unit testing, and Selenium for end-to-end testing.
- **Development Tools**: Set up necessary development tools, including IDEs, code linters, and debuggers.

4. Documentation Requirements:

User Documentation: Provide user manuals and guides for different roles.

API Documentation: Document APIs for developers and third-party integrations.

Technical Documentation: Maintain detailed technical documentation for future maintenance and updates.

5. Support and Maintenence:

- Set up support and issue reporting systems.
- Define a maintenance plan for regular updates and debugging.

Project Plan

Timeline

https://docs.google.com/document/d/14p77LFWKYYqZuQuZxC_tnrB0lvubwdKndJPJ2N8hSjI/edit?usp=sharing

Design, Develop, and Deploy the Software

Design

UI/IX Designer: Create wireframes and prototypes. Plan the architecture of the software including the tech stack and data flow.

Develop

Front-End Engineer: Build the user interface.

Back-End Engineering: Develop the server-side logic and database.

Integrate APIs as needed.

Testing

Unit Testing: Test individual components for functionality.

Integration Testing: Test unit components together to ensure logic, flow, and seamless working.

User Testing: Conduct testing with actual users. Make necessary adjustments.

Deployment

Deploy the software in the staging environment for final testing. Followed by deploying on the production environment after successful testing.

Monitor and Maintain

Monitor Performance: Continuously monitor the software's performance and address any issues.

Regular Updates: Plan for regular updates and improvements based on user feedback and changing requirements.

Documentation and Training

Create Documentation: Document the software, including user guides and technical documentation.

Train Users: Provide training sessions for users to ensure they can effectively use the software.

Evaluate and Improve

Collect Feedback: Gather feedback from users and stakeholders.

Continuous Improvement: Implement improvements based on feedback and monitor for ongoing performance and user satisfaction.