EDGE PROJECT: Department of ICT,MBSTU

Deliverable

Project title: TODO Application Proposal

Submitted by: Abdullah Al- Mamun

Supervisor Name: Dr. Ziaur Rahman

December 31, 2024

Contents

1 Abstract

The TODO Application is a cross-platform mobile and web application that allows users to manage their tasks efficiently. Built using Django as the backend and Flutter for the frontend, the application offers a user-friendly interface to add, edit, delete, and organize tasks. Key features include categorizing tasks, setting reminders, and a prioritization system. This app aims to help users enhance productivity by keeping their tasks organized and accessible across devices.

2 Introduction

The TODO Application addresses the need for an efficient task management tool that is simple, intuitive, and accessible on multiple platforms. Many task management solutions

exist, but few offer a seamless user experience with a centralized backend for data consistency. This project leverages Django for a robust backend API and Flutter for a responsive, crossplatform frontend. Key features include:

- Creating and managing personal task lists.
- Categorizing and prioritizing tasks.
- Setting reminders for deadlines.
- Accessing tasks from multiple devices.

This app is targeted at individuals and professionals seeking an organized way to manage their daily tasks.

3 Completion Plan

3.1 Phase 1: Planning and Requirements Gathering

- Define Objectives: Outline the core features (e.g., task creation, categorization, reminders).
- User Stories: Define use cases for different user interactions with the app.
- Technology Stack: Confirm the tech stack, including Django REST Framework, PostgreSQL (or SQLite), and Flutter.

3.2 Phase 2: Design

- UI/UX Design: Create wireframes and mockups for user interface elements.
- Database Design: Develop the schema for tasks, categories, reminders, and users.
- Architecture: Establish a system architecture outlining Django's backend structure and Flutter's frontend structure.

3.3 Phase 3: Development

- Backend Development (Django):

  Set up a Django project and configure initial settings. Create models for User, Task, Category, and Reminder. Build RESTful APIs with Django REST Framework for task management, user interactions, and reminders.

- Frontend Development (Flutter):

  Design screens for login, task management, and task details. Implement state management for handling user actions. Connect to Django APIs for data synchronization.

3.4 Phase 4: Testing

- Unit Testing: Develop and run unit tests for Django models, views, and Flutter components.
- Integration Testing: Test the integration between Django APIs and the Flutter frontend.
  - User Testing: Conduct user testing to collect feedback and identify usability improvements.
- Bug Fixes: Resolve any issues identified during testing.

3.5 Phase 5: Deployment

- Deploy Backend: Set up the production environment and deploy the backend application.
- Deploy Frontend: Release the Flutter app on the App Store and Google Play for mobile users.
- Monitoring: Implement logging and monitoring for performance tracking and error detection.

3.6 Phase 6: Maintenance

- Ongoing Support: Provide support and fix any issues reported by users.
- Feature Enhancements: Regularly improve the application with user-requested features or optimizations.

4 Proposed Application Details

4.1 Features:

- User Registration and Login: Secure user authentication, including OAuth or social logins.
- Task Management: Allow users to add, edit, and delete tasks with detailed descriptions and due dates.
- Categorization: Enable task categorization by tags (e.g., Work, Personal, Urgent).
- Prioritization and Reminders: Allow users to set priority levels and reminders for important tasks.
- Task Filtering and Search: Provide search and filter options for easy access to specific tasks.
- User Profiles: Users can view and edit their profiles, track completed tasks, and set preferences.
- Cross-Device Synchronization: Sync tasks across multiple devices using Django APIs.
- Responsive Design: Ensure the app's usability on both mobile and web platforms.

4.2 Technical Stack:

- Backend: Django with Django REST Framework
- Frontend: Flutter (for mobile and web)

- Database: PostgreSQL for production; SQLite for development
- Hosting: Use Heroku, AWS, or DigitalOcean for deploying the Django backend
- Other Tools: Firebase for push notifications (for task reminders)

5 Discussion

5.1 Challenges:

- Data Consistency: Ensuring that task data is synchronized across all devices.
- Scalability: Designing the backend to handle high user demand as the user base grows.
- Reminder Notifications: Implementing timely reminders across multiple devices/platforms.

5.2 Benefits:

- Improved Productivity: Helps users organize and prioritize their tasks.
- Accessibility: Provides cross-device access with a consistent user experience.
- Personalization: Allows users to customize and categorize tasks based on their preferences.

5.3 Future Enhancements:

- Collaboration Feature: Allow multiple users to collaborate on tasks or projects.
- Voice Commands: Integrate with voice assistants for hands-free task creation.
- Analytics: Offer productivity analytics for tracking completed tasks over time.

6 Conclusion

The TODO Application aims to create an effective solution for managing tasks across multiple platforms with a centralized backend for data consistency. With Django's backend capabilities and Flutter's responsive UI, this application will serve as a robust and accessible productivity tool for users across different devices.