Project Name: QuickTask

Objective:

QuickTask is a simplified task management app designed to help users organize their tasks efficiently.

This project uses Flutter and Back4App by building a basic task management app with essential features.

Features:

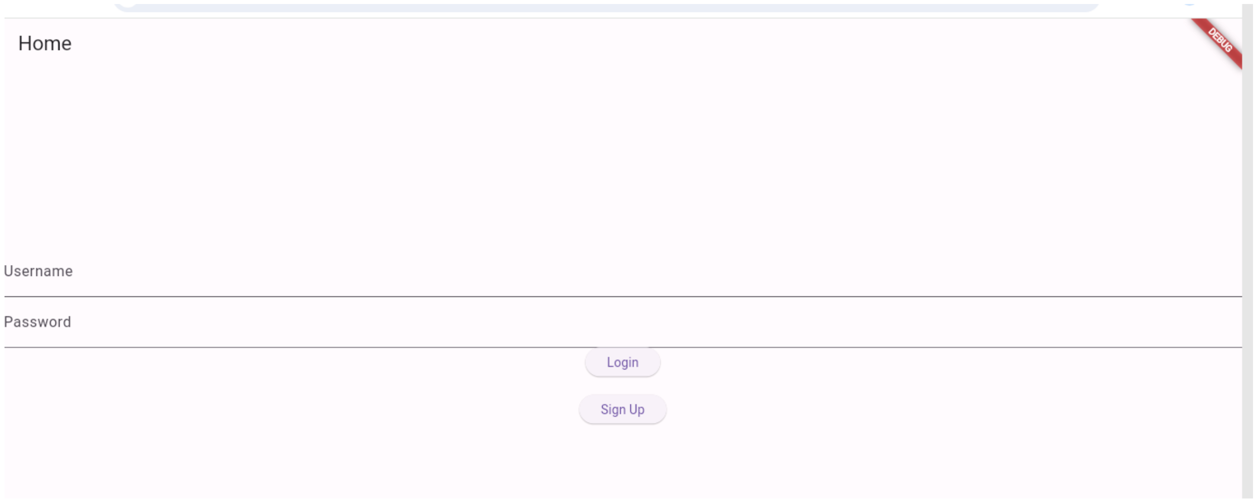
1. User Authentication:

- Implement basic user authentication using Back4App.

- Allow users to sign up and log in to the app securely.

User Authentication is implemented as below

- Code Implementation: Implemented user authentication using Back4App and Parse SDK.
- Code Location: Authentication functionality is primarily implemented in the login_screen.dart , authentication_service.dart and SignupScreen.dart files.
- Authentication Methods: Used ParseUser.login() and ParseUser.signUp() methods for login and signup, respectively.
- Error Handling: Basic error handling for authentication failures is implemented using SnackBar widgets to display error messages.



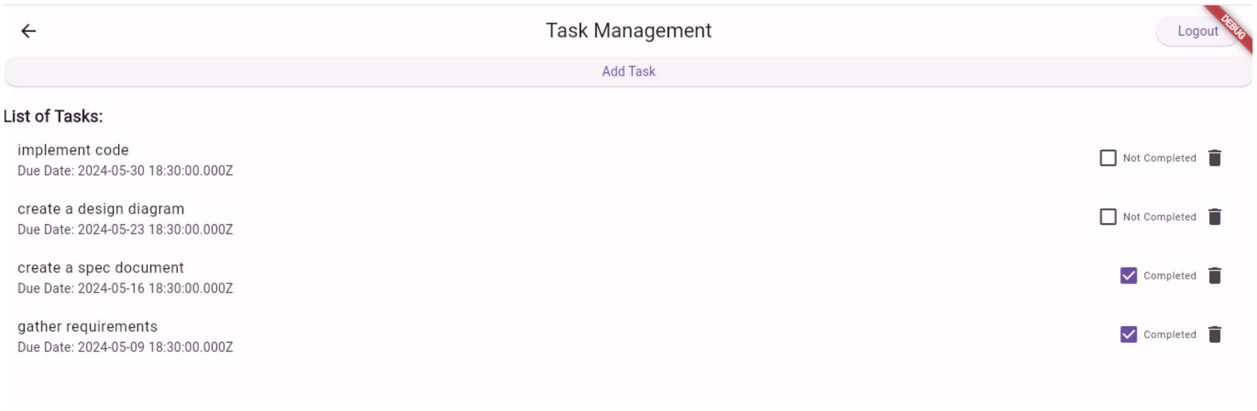
2. Task Management:

- Users can add, view, and delete tasks.

- Each task should have a title and a due date.

Task Management is implemented as below

- Code Implementation: Task management features include adding, viewing, editing, and deleting tasks.
- Code Location: Task-related functionality is implemented in multiple files, including task_management.dart, task_list_screen.dart, task_model.dart, task_service.dart, add_task_screen.dart and corresponding UI screens.
- CRUD Operations: Implemented CRUD operations for tasks using Parse SDK methods like ParseObject.save(), ParseObject.delete(), etc.



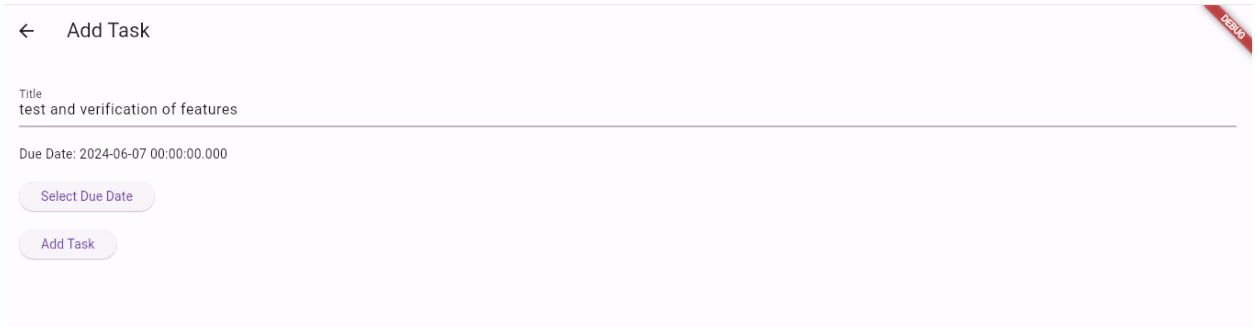
List of Tasks:

implement code
Due Date: 2024-05-30 18:30:00.000Z
Not Completed

create a design diagram
Due Date: 2024-05-23 18:30:00.000Z
Not Completed

create a spec document
Due Date: 2024-05-16 18:30:00.000Z
Completed

gather requirements
Due Date: 2024-05-09 18:30:00.000Z
Completed



Title
test and verification of features

Due Date: 2024-06-07 00:00:00.000

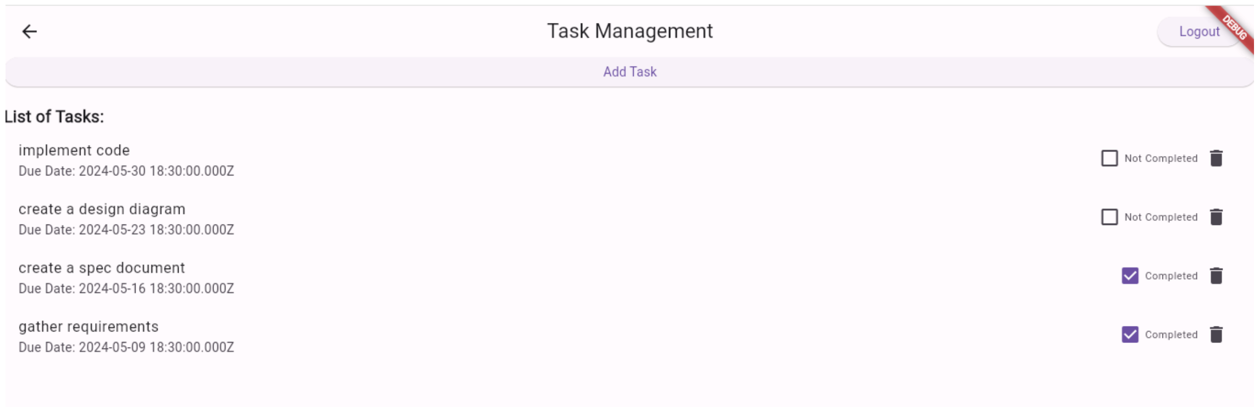
3. Task Status:

- Tasks can be marked as completed or incomplete.

- Provide an option to toggle the status of tasks.

Task Status is implemented as below

- Task Status: Added functionality to mark tasks as completed or incomplete and toggle their status.
- Code Location: implemented in task_list_screen.dart



List of Tasks:

implement code
Due Date: 2024-05-30 18:30:00.000Z
Not Completed

create a design diagram
Due Date: 2024-05-23 18:30:00.000Z
Not Completed

create a spec document
Due Date: 2024-05-16 18:30:00.000Z
Completed

gather requirements
Due Date: 2024-05-09 18:30:00.000Z
Completed

4. UI Design and Enhancements:

- Code Implementation: Implemented a simple and minimalistic user interface using Flutter widgets.
- Code Location: UI design elements are distributed across various Dart files corresponding to different screens and components.
- Styling and Layout: Applied basic styling and layout enhancements to improve the user experience.
- Widget Reusability: Encouraged widget reusability and modularity to maintain a clean and organized codebase.

5. Backend Integration:

- Code Implementation: Integrated the frontend Flutter application with Back4App's backend services.
- Code Location: Backend integration logic is primarily found in task_service.dart file.
- Parse SDK Usage: Utilized Parse SDK methods to communicate with Back4App's backend, including fetching tasks, adding tasks, etc.
- Error Handling: Implemented error handling for backend communication errors to provide a seamless user experience.

6. Configuration and Setup Guide

1. Flutter Environment Setup:

- Install Flutter: Installed the Flutter SDK on the development machine by following the official Flutter installation guide.
- IDE Setup: Visual Studio Code

- Flutter Packages: Ensured that the necessary Flutter packages are installed by running flutter pub get in the project directory.

2. Back4App Configuration:

- Account Creation: Sign up for a Back4App account on the Back4App website.
- App Creation: Created a new **“QuickTask”** App on Back4App's dashboard and noted down the Application ID and Client Key.
- Parse SDK Setup: Configured the Parse SDK in Flutter project by providing the Application ID and Client Key in the Parse.initialize() method.

3. GitHub Repository Setup:

- Repository Creation: Created a private **CrossPlatformAppDev** repository on GitHub to store the project files.
- Push Code: Pushed the project code to the GitHub repository using Git commands

4. Testing and Debugging:

- Emulator/Device Setup: Setted up an Android or iOS emulator and connectd a physical device for testing.
- Run Application: Run the Flutter application using flutter run command in the project directory to launch the app on the emulator , device, chrome browser.
- Debugging: Used the debugging features of your IDE to troubleshoot and fix any issues encountered during testing.

Conclusion

- The QuickTask project demonstrates the implementation of a basic task management app using Flutter and Back4App. By following the code implementation and configuration guide provided above, developers can set up, configure, and run the project successfully for testing and deployment purposes.

GitHub Repository - https://github.com/bharatodedara/CrossPlatformAppDev

Video-
https://drive.google.com/file/d/1OhTvAYpykXIgOOJ96eKLwC0bOI0jdvV8/view?usp=drive_link