Project Report

On

PROJECT MANAGEMENT DASHBOARD

Submitted in partial fulfilment of the requirements for the award of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

(Artificial Intelligence & Machine Learning)

by

Ms. M PRASANNA (22WH1A6615)

Ms. S DEEPIKA PRAHARSHINI (22WH1A6623)

Ms. B HEMANYA SAI (22WH1A6636)

Ms. P JAHNAVI (22WH1A6639)

Under the esteemed guidance of

Ms. S Annapoorna

Assistant Professor, CSE(AI&ML)

Department of Computer Science & Engineering

(Artificial Intelligence & Machine Learning)

BVRIT HYDERABAD COLLEGE OF ENGINEERING FOR WOMEN

(AUTONOMOUS)

(Approved by AICTE, New Delhi and Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with A Grade

Bachupally, Hyderabad – 500090

2024-25

Department of Computer Science & Engineering

(Artificial Intelligence & Machine Learning)

BVRIT HYDERABAD COLLEGE OF ENGINEERING FOR WOMEN

(Approved by AICTE, New Delhi and Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with A Grade

Bachupally, Hyderabad – 500090

2023-24

CERTIFICATE

This is to certify that the major project entitled **“Project Management Dashboard”** is a Bonafide work carried out by **Ms. M Prasanna (22WH1A6615), Ms. S Deepika Praharshini (22WH1A6623)**, **Ms. B Hemanya Sai (22WH1A6636), Ms. P Jahnavi (22WH1A6639**) in partial fulfilment for the award of B. Tech degree in **Computer Science & Engineering (AI&ML)**, **BVRIT HYDERABAD College of Engineering for Women, Bachupally, Hyderabad**, affiliated to Jawaharlal Nehru Technological University Hyderabad, Hyderabad under my guidance and supervision. The results embodied in the project work have not been submitted o any other University or Institute for the award of any degree or diploma.

Supervisor
Ms. S Annapoorna
Assistant Professor
Dept of CSE(AI&ML)

Head of the Department
Dr. B. Lakshmi Praveena
HOD & Professor

DECLARATION

We hereby declare that the work presented in this project entitled **"Project Management Dashboard"** submitted towards completion of Project work in III Year of B.Tech of CSE(AI&ML) at **BVRIT HYDERABAD College of Engineering for Women,** Hyderabad is an authentic record of our original work carried out under the guidance of **Ms. S Annapoorna, Assistant Professor, Department of CSE(AI&ML).**

Sign with Date:
M Prasanna
(22WH1A6615)

Sign with Date:
S Deepika Praharshini
(22WH1A6623)

Sign with Date:
B Hemanya Sai
(22WH1A6636)

Sign with Date:
P Jahnavi
(22WH1A6639)

ACKNOWLEDGEMENT

We would like to express our sincere thanks to **Dr. K V N Sunitha, Principal, BVRIT HYDERABAD College of Engineering for Women**, for her support by providing the working facilities in the college.

Our sincere thanks and gratitude to **Dr. B Lakshmi Praveena, Head of the Department, Department of CSE(AI&ML), BVRIT HYDERABAD College of Engineering for Women,** for all timely support and valuable suggestions during the period of our project.

We are extremely thankful to our Internal Guide, **Ms. S Annapoorna, Assistant Professor, CSE(AI&ML), BVRIT HYDERABAD College of Engineering for Women,**
for her constant guidance and encouragement throughout the project.

Finally, we would like to thank our Major Project Coordinator, all Faculty and Staff of CSE(AI&ML) department who helped us directly or indirectly. Last but not least, we wish to acknowledge our **Parents** and **Friends** for giving moral strength and constant encouragement.

M. Prasanna
(22WH1A6615)

S Deepika Praharshini
(22WH1A6623)

B Hemanya Sai
(22WH1A6636)

P Jahnavi
(22WH1A6639)

PROBLEM STATEMENT

The goal is to develop a Project Management Dashboard mobile application that allows users to manage and track their projects effectively. The app should enable users to:

- Create, view, edit, and delete projects easily, allowing for dynamic project management at every stage.
- Add and manage tasks, team members, and expenses related to each project, ensuring efficient team collaboration and task tracking.
- Track the project's budget and expenses, allowing users to keep a real-time check on spending and ensure it stays within allocated limits.
- Implement password protection for secure access to sensitive project details, ensuring that only authorized users can view or edit project information.
- Provide an intuitive, responsive UI that adapts to different screen sizes and ensures a seamless, user-friendly experience.

The app will offer a comprehensive yet easy-to-use platform for project managers and teams to manage the various facets of their projects, ensuring streamlined operations and control over project-related data.

ABSTRACT

The **Project Management Dashboard** is a Flutter-based app designed to simplify project management by providing a centralized platform to manage project details, tasks, team members, and budgets. The app features a **Project List Screen** that displays active projects, enabling users to add, edit, view, or delete projects securely with password protection. A detailed **Project Details Screen** offers functionality to assign tasks, track expenses, and monitor project progress. The app ensures smooth user experience with responsive layouts, gradient backgrounds, and transparent app bars, complemented by Flutter’s setState () for real-time UI updates. Dialogs handle adding or editing tasks, team members, and expenses, while snackbars provide feedback for errors like incorrect passwords. Designed for security and functionality, the app is ideal for individuals or teams seeking an efficient way to organize and track projects from initiation to completion.

FILE STRUCTURE

```
project_management_dashboard
├── .dart_tool
├── .idea
├── android
├── android
├── build
├── ios
├── lib
│   ├── main.dart
├── linux
├── macos
├── test
│   ├── widget_test.dart
├── web
├── windows
├── .gitignore
├── .metadata
├── analysis_options.yaml
├── project_management_dashboard.iml
├── pubspec.lock
├── pubspec.yaml
└── README.md
```

SOURCE CODE:

```
import 'package:flutter/material.dart';
void main() {
  runApp(ProjectManagementApp());
}
class ProjectManagementApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Project Management Dashboard',
      theme: ThemeData(primarySwatch: Colors.blue),
      debugShowCheckedModeBanner: false, // Remove debug banner
      home: ProjectListScreen(),
    );
  }
}

class ProjectListScreen extends StatefulWidget {
  @override
  _ProjectListScreenState createState() => _ProjectListScreenState();
}

class _ProjectListScreenState extends State<ProjectListScreen> {
  final List<Map<String, dynamic>> projects = [
    {'name': 'The Great Escape ', 'password': 'admin1'},
    {'name': 'Island Fortune', 'password': 'admin2'},
    {'name': 'Dish Genie', 'password': 'admin3'},
    {'name': 'Heavenly Cakes', 'password': 'admin4'},
    {'name': 'Honeybee Breeding', 'password': 'admin5'},
  ];

  String _userName = 'Profile'; // Default username
```

```
void _addProject() {
  showDialog(
    context: context,
    builder: (context) {
      String newProjectName = '';
      return AlertDialog(
        title: Text('Add New Project'),
        content: TextField(
          onChanged: (value) {
            newProjectName = value;
          },
          decoration: InputDecoration(hintText: 'Enter Project Name'),
        ),
        actions: [
          TextButton(
            onPressed: () {
              if (newProjectName.isNotEmpty) {
                setState(() {
                  projects.add({'name': newProjectName, 'password': 'admin'});
                });
              }
              Navigator.of(context).pop();
            },
            child: Text('Add'),
          ),
        ],
      );
    },
  );
}

void _editProfile() {
  showDialog(
      context: context,
      builder: (context) {
        TextEditingController nameController = TextEditingController(
            text:
                _userName); // Pre-fill the controller with the current username
        return AlertDialog(
          title: Text('Edit Profile'),
          content: TextField(
            controller: nameController,
            decoration: InputDecoration(hintText: 'Enter your name'),
          ),
          actions: [
            TextButton(
              onPressed: () {
                if (nameController.text.isNotEmpty) {
                  setState(() {
                    _userName = nameController.text; // Update the profile name
                  });
                }
                Navigator.of(context).pop(); // Close the dialog
              },
              child: Text('Save'),
            ),
            TextButton(
              onPressed: () => Navigator.of(context)
                  .pop(), // Close the dialog without saving
              child: Text('Cancel'),
            ),
          ],
        );
      },
    );
  }
```

```
Widget _buildHeader() {
 return Column(
  children: [
   SizedBox(height: 20), // Spacing above the header
   Text(
    'Project Management Dashboard', // App Name
    style: TextStyle(
     fontSize: 24, // Large font size
     fontWeight: FontWeight.bold, // Bold font
     color: Colors.blueAccent, // Blue text color
    ),
    textAlign: TextAlign.center, // Center align the text
   ),
   SizedBox(height: 10), // Spacing between App Name and Tagline
   Text(
    'Streamlining your projects with ease', // Tagline
    style: TextStyle(
     fontSize: 16, // Smaller font size for the tagline
     color: Colors.grey[600], // Grey text color
    ),
    textAlign: TextAlign.center,
   ),
   SizedBox(height: 20), // Spacing below the header
  ],
 );
}

Widget _buildFooter() {
 return Container(
  color: Colors.blueAccent.withOpacity(0.1), // Light blue background
  padding: EdgeInsets.all(10), // Padding inside the footer
  child: Column(
   mainAxisSize:
     MainAxisSize.min, // Ensures the footer takes minimum space
```

```
      children: [
        Text(
          'Version 1.0.0', // Version Number
          style: TextStyle(
            fontSize: 14, // Font size for the version
            color: Colors.grey[700], // Grey text color
          ),
        ),
        SizedBox(height: 5), // Spacing between the two lines of text
        Text(
          'Powered by Flutter', // Footer note
          style: TextStyle(
            fontSize: 14, // Font size for the note
            color: Colors.grey[700], // Grey text color
          ),
        ),
      ],
    ),
  );
}

void _editProject(int index) {
  showDialog(
    context: context,
    builder: (context) {
      String password = '';
      String editedName = projects[index]['name'];
      String startDate = '';
      String endDate = '';
      String budget = '';
      return AlertDialog(
        title: Text('Edit Project'),
        content: Column(
          mainAxisSize: MainAxisSize.min,
          children: [
            TextField(
              onChanged: (value) {
                password = value;
              },
              obscureText: true,
              decoration: InputDecoration(hintText: 'Enter Password'),
            ),
            SizedBox(height: 10),
            TextField(
              onChanged: (value) {
                editedName = value;
              },
              decoration: InputDecoration(hintText: 'Project Name'),
            ),
            SizedBox(height: 10),
            TextField(
              onChanged: (value) {
                startDate = value;
              },
              decoration:
                  InputDecoration(hintText: 'Start Date (e.g., 01 Jan 2024)'),
            ),
            SizedBox(height: 10),
            TextField(
              onChanged: (value) {
                endDate = value;
              },
              decoration:
                  InputDecoration(hintText: 'End Date (e.g., 01 Dec 2024)'),
            ),
            SizedBox(height: 10),
            TextField(
              onChanged: (value) {
                budget = value;
              },
              keyboardType: TextInputType.number,
              decoration: InputDecoration(hintText: 'Budget'),
            ),
          ],
        ),
        actions: [
          TextButton(
            onPressed: () {
              if (password == projects[index]['password']) {
                setState(() {
                  projects[index]['name'] = editedName;
                  projects[index]['startDate'] = startDate;
                  projects[index]['endDate'] = endDate;
                  projects[index]['budget'] = budget;
                });
                Navigator.of(context).pop();
              } else {
                ScaffoldMessenger.of(context).showSnackBar(
                  SnackBar(content: Text('Incorrect Password!')),
                );
              }
            },
            child: Text('Save'),
          ),
        ],
       );
      },
      );
      }

      void _deleteProject(int index) {
       setState(() {
      projects.removeAt(index);
    });
  }

  void _navigateToProject(int index) {
    showDialog(
      context: context,
      builder: (context) {
        String password = '';
        return AlertDialog(
          title: Text('Enter Password'),
          content: TextField(
            onChanged: (value) {
              password = value;
            },
            obscureText: true,
            decoration: InputDecoration(hintText: 'Enter Password'),
          ),
          actions: [
            TextButton(
              onPressed: () {
                if (password == projects[index]['password']) {
                  Navigator.of(context).pop();
                  Navigator.of(context).push(MaterialPageRoute(
                    builder: (context) => ProjectDetailsScreen(
                      projectTitle: projects[index]['name'],
                      projectPassword: projects[index]['password'],
                      projectStartDate:
                          projects[index]['startDate'] ?? 'Not set',
                      projectEndDate: projects[index]['endDate'] ?? 'Not set',
                      projectBudget: projects[index]['budget'] ?? '0',
                    ),
                  ));
                } else {
```

```
                ScaffoldMessenger.of(context).showSnackBar(
                  SnackBar(content: Text('Incorrect Password!')),
                );
              }
            },
            child: Text('Enter'),
          ),
        ],
      );
    },
  );
}

@override
Widget build(BuildContext context) {
  return Container(
    decoration: BoxDecoration(
      gradient: LinearGradient(
        colors: [
          const Color(0xFFB2FEFA), // Soft turquoise
          const Color(0xFF0ED2F7), // Light blue
          const Color(0xFFA8DEFF), // Pale sky blue
        ],
        begin: Alignment.topLeft,
        end: Alignment.bottomRight,
      ),
    ),
    child: Scaffold(
      backgroundColor:
          Colors.transparent, // Transparent background for gradient
      appBar: AppBar(
        backgroundColor: Colors.transparent,
        elevation: 0,
        title: Text(
          'Project List',
          style: TextStyle(fontSize: 20, fontWeight: FontWeight.bold),
        ),
        actions: [
          Padding(
            padding: const EdgeInsets.only(
                right: 16.0), // Add some padding on the right for spacing
            child: Row(
              children: [
                // Profile image
                ClipOval(
                  child: Image.asset(
                    "assets/bgg.png",
                    width: 40.0,
                    height: 40.0,
                    fit: BoxFit.cover,
                  ),
                ),
                SizedBox(width: 8.0), // Spacing between image and name
                Text(
                  _userName, // Display the user's name
                  style: TextStyle(fontSize: 16, fontWeight: FontWeight.w500),
                ),
              ],
            ),
          ),
        ],
      ),

      body: Column(
        children: [
          _buildHeader(), // This is where the header widget is placed
          Expanded(
            child: ListView.builder(
                      itemCount: projects.length,
                      itemBuilder: (context, index) {
                        return ListTile(
                          title: Text(projects[index]['name']),
                          trailing: Row(
                            mainAxisSize: MainAxisSize.min,
                            children: [
                              IconButton(
                                icon: Icon(Icons.edit),
                                onPressed: () => _editProject(index),
                              ),
                              IconButton(
                                icon: Icon(Icons.delete),
                                onPressed: () => _deleteProject(index),
                              ),
                            ],
                          ),
                          onTap: () => _navigateToProject(index),
                        );
                      },
                    ),
                  ),
                  _buildFooter(), // Add the footer at the bottom
                ],
              ),

              floatingActionButton: FloatingActionButton(
                onPressed: _addProject,
                child: Icon(Icons.add),
              ),
            ),
          );
        }
      }
```

```
class ProjectDetailsScreen extends StatefulWidget {
  final String projectTitle;
  final String projectPassword;
  final String projectStartDate;
  final String projectEndDate;
  final String projectBudget;

  ProjectDetailsScreen({
    required this.projectTitle,
    required this.projectPassword,
    required this.projectStartDate,
    required this.projectEndDate,
    required this.projectBudget,
  });

  @override
  _ProjectDetailsScreenState createState() => _ProjectDetailsScreenState();
}

class _ProjectDetailsScreenState extends State<ProjectDetailsScreen> {
  late TextEditingController _projectNameController;
  late TextEditingController _startDateController;
  late TextEditingController _endDateController;
  late TextEditingController _statusController;
  late TextEditingController _budgetController;
  late TextEditingController _spentController;
  late TextEditingController _taskController;
  late TextEditingController _teamMemberController;
  List<Map<String, dynamic>> tasks = [];
  List<Map<String, dynamic>> expenses = [];
  List<Map<String, dynamic>> teamMembers = [];
  double _rating = 0.0;
  double amountSpent = 0.0;
double balanceAmount = 0.0;

@override
void initState() {
  super.initState();
  _projectNameController = TextEditingController(text: widget.projectTitle);
  _startDateController = TextEditingController(text: widget.projectStartDate);
  _endDateController = TextEditingController(text: widget.projectEndDate);
  _statusController = TextEditingController(text: 'In Progress');
  _budgetController = TextEditingController(text: widget.projectBudget);
  _spentController = TextEditingController(text: '0.0');
}

@override
void dispose() {
  _projectNameController.dispose();
  _startDateController.dispose();
  _endDateController.dispose();
  _statusController.dispose();
  _budgetController.dispose();
  _spentController.dispose();
  super.dispose();
}

void _addExpense() {
  showDialog(
    context: context,
    builder: (context) {
      String expenseName = '';
      double expenseAmount = 0.0;
      return AlertDialog(
        title: Text('Add New Expense'),
        content: Column(
          mainAxisSize: MainAxisSize.min,
  children: [
    TextField(
      onChanged: (value) {
        expenseName = value;
      },
      decoration: InputDecoration(hintText: 'Expense Name'),
    ),
    TextField(
      keyboardType: TextInputType.number,
      onChanged: (value) {
        expenseAmount = double.tryParse(value) ?? 0.0;
      },
      decoration: InputDecoration(hintText: 'Expense Amount'),
    ),
  ],
),
actions: [
  TextButton(
    onPressed: () {
      if (expenseName.isNotEmpty && expenseAmount > 0.0) {
        setState(() {
          expenses
              .add({'name': expenseName, 'amount': expenseAmount});
          amountSpent += expenseAmount;
          balanceAmount =
              double.parse(_budgetController.text) - amountSpent;
          _spentController.text = amountSpent.toStringAsFixed(2);
        });
      }
      Navigator.of(context).pop();
    },
    child: Text('Add'),
  ),
],
        );
      },
    );
  }

void _addTask() {
  showDialog(
    context: context,
    builder: (context) {
      String taskName = '';
      String taskDescription = '';
      return StatefulBuilder(
        builder: (context, setState) {
          return AlertDialog(
            title: Text('Add New Task'),
            content: Column(
              mainAxisSize: MainAxisSize.min,
              children: [
                TextField(
                  onChanged: (value) {
                    setState(() {
                      taskName = value;
                    });
                  },
                  decoration: InputDecoration(
                    hintText: 'Task Name',
                    errorText: taskName.isEmpty
                        ? 'Task Name cannot be empty'
                        : null,
                  ),
                ),
                TextField(
                  onChanged: (value) {
                    setState(() {
```

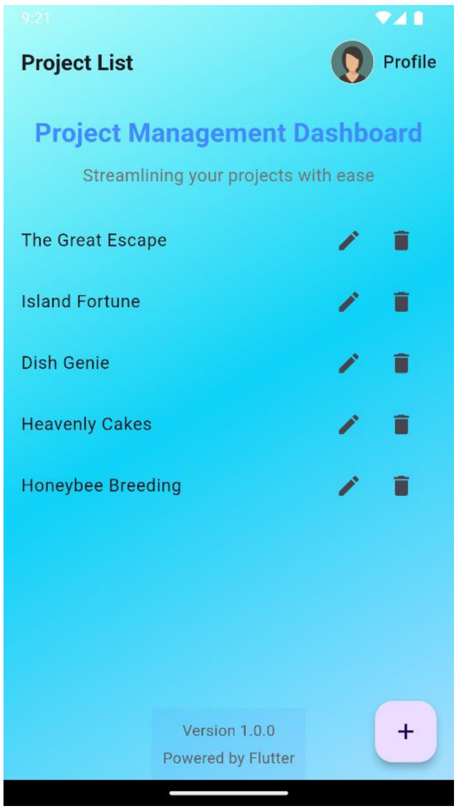
```
                  taskDescription = value;
                });
              },
              decoration: InputDecoration(
                hintText: 'Task Description',
                errorText: taskDescription.isEmpty
                    ? 'Task Description cannot be empty'
                    : null,
              ),
            ),
          ],
        ),
        actions: [
          TextButton(
            onPressed: () {
              if (taskName.isNotEmpty && taskDescription.isNotEmpty) {
                setState(() {
                  tasks.add({
                    'name': taskName,
                    'description': taskDescription,
                  });
                });
                Navigator.of(context).pop();
              }
            },
            child: Text('Add'),
          ),
        ],
      );
    },
  );
},
);
}
```

```
void _addTeamMember() {
  showDialog(
    context: context,
    builder: (context) {
      String teamMemberName = '';
      String taskAssigned = '';
      return StatefulBuilder(
        builder: (context, setState) {
          return AlertDialog(
            title: Text('Add Team Member'),
            content: Column(
              mainAxisSize: MainAxisSize.min,
              children: [
                TextField(
                  onChanged: (value) {
                    setState(() {
                      teamMemberName = value;
                    });
                  },
                  decoration: InputDecoration(
                    hintText: 'Team Member Name',
                    errorText: teamMemberName.isEmpty
                        ? 'Team Member Name cannot be empty'
                        : null,
                  ),
                ),
                TextField(
                  onChanged: (value) {
                    setState(() {
                      taskAssigned = value;
                    });
                  },
                  decoration: InputDecoration(
                    hintText: 'Assigned Task',
                    errorText: taskAssigned.isEmpty
                        ? 'Assigned Task cannot be empty'
                        : null,
                  ),
                ),
              ],
            ),
            actions: [
              TextButton(
                onPressed: () {
                  if (teamMemberName.isNotEmpty && taskAssigned.isNotEmpty) {
                    setState(() {
                      teamMembers.add({
                        'name': teamMemberName,
                        'task': taskAssigned,
                      });
                    });
                    Navigator.of(context).pop();
                  }
                },
                child: Text('Add'),
              ),
            ],
          );
        },
      );
    },
  );
}

@override
Widget build(BuildContext context) {
  return Container(
    decoration: BoxDecoration(
      gradient: LinearGradient(
        colors: [
          const Color(0xFFB2FEFA), // Soft turquoise
          const Color(0xFF0ED2F7), // Light blue
          const Color(0xFFA8DEFF), // Pale sky blue
        ],
        begin: Alignment.topLeft,
        end: Alignment.bottomRight,
      ),
    ),
    child: Scaffold(
      backgroundColor: Colors.transparent, // Make the Scaffold background transparent
      appBar: AppBar(
        title: Text(widget.projectTitle),
        backgroundColor: Colors.transparent, // Transparent AppBar background
        elevation: 0, // Remove AppBar shadow
      ),
      body: SingleChildScrollView(
        child: Column(
          children: [
            // Project Name
            ListTile(
              title: Text('Project Name:'),
              subtitle: Text(_projectNameController.text),
              trailing: IconButton(
                icon: Icon(Icons.edit),
                onPressed: () {
                  showDialog(
                    context: context,
                    builder: (context) {
                      String newName = _projectNameController.text;
                      return AlertDialog(
                        title: Text('Edit Project Name'),
                content: TextField(
                  onChanged: (value) {
                    newName = value;
                  },
                  decoration: InputDecoration(
                    hintText: 'Enter New Name',
                    errorText: validateProjectName(newName),
                  ),
                ),
                actions: [
                  TextButton(
                    onPressed: () {
                      if (validateProjectName(newName) == null) {
                        setState(() {
                          _projectNameController.text = newName;
                        });
                        Navigator.of(context).pop();
                      }
                    },
                    child: Text('Save'),
                  ),
                ],
              );
            },
          );
        },
      ),
    ),

    // Start Date
    ListTile(
      title: Text('Start Date:'),
      subtitle: Text(_startDateController.text),
      trailing: IconButton(
    icon: Icon(Icons.edit),
    onPressed: () {
      showDialog(
        context: context,
        builder: (context) {
          String newStartDate = _startDateController.text;
          return AlertDialog(
            title: Text('Edit Start Date'),
            content: TextField(
              onChanged: (value) {
                newStartDate = value;
              },
              decoration: InputDecoration(
                hintText: 'Enter New Start Date',
                errorText: validateDate(newStartDate),
              ),
            ),
            actions: [
              TextButton(
                onPressed: () {
                  if (validateDate(newStartDate) == null) {
                    setState(() {
                      _startDateController.text = newStartDate;
                    });
                    Navigator.of(context).pop();
                  }
                },
                child: Text('Save'),
              ),
            ],
          );
        },
      );
    },
 ),
),

// End Date
ListTile(
 title: Text('End Date:'),
 subtitle: Text(_endDateController.text),
 trailing: IconButton(
  icon: Icon(Icons.edit),
  onPressed: () {
   showDialog(
    context: context,
    builder: (context) {
     String newEndDate = _endDateController.text;
     return AlertDialog(
      title: Text('Edit End Date'),
      content: TextField(
       onChanged: (value) {
        newEndDate = value;
       },
       decoration: InputDecoration(
        hintText: 'Enter New End Date',
        errorText: validateDate(newEndDate),
       ),
      ),
      actions: [
       TextButton(
        onPressed: () {
         if (validateDate(newEndDate) == null) {
          setState(() {
           _endDateController.text = newEndDate;
          });
          Navigator.of(context).pop();
         }
```

```
              },
              child: Text('Save'),
            ),
          ],
        );
      },
    );
  },
),
),

// Budget
ListTile(
  title: Text('Budget:'),
  subtitle: Text(_budgetController.text),
  trailing: IconButton(
    icon: Icon(Icons.edit),
    onPressed: () {
      showDialog(
        context: context,
        builder: (context) {
          String newBudget = _budgetController.text;
          return AlertDialog(
            title: Text('Edit Budget'),
            content: TextField(
              onChanged: (value) {
                newBudget = value;
              },
              decoration: InputDecoration(
                hintText: 'Enter New Budget',
                errorText: validateBudget(newBudget),
              ),
            ),
            actions: [
              TextButton(
                onPressed: () {
                  if (validateBudget(newBudget) == null) {
                    setState(() {
                      _budgetController.text = newBudget;
                    });
                    Navigator.of(context).pop();
                  }
                },
                child: Text('Save'),
              ),
            ],
          );
        },
      );
    },
  ),
),

// Amount Spent
ListTile(
  title: Text('Amount Spent:'),
  subtitle: Text(_spentController.text),
),

// Balance Amount
ListTile(
  title: Text('Balance Amount:'),
  subtitle: Text(balanceAmount.toStringAsFixed(2)),
),

// Add Expense button
ElevatedButton(
  onPressed: _addExpense,
  child: Text('Add Expense'),
),
SizedBox(height: 20),

// Add Task button
ElevatedButton(
  onPressed: _addTask,
  child: Text('Add Task'),
),
SizedBox(height: 20),

// Add Team Member button
ElevatedButton(
  onPressed: _addTeamMember,
  child: Text('Add Team Member'),
),

// Tasks List
ListTile(
  title: Text('Tasks'),
  subtitle: Text(tasks.isEmpty ? ' ' : ''),
  trailing: IconButton(
    icon: Icon(Icons.arrow_forward),
    onPressed: () {
      showDialog(
        context: context,
        builder: (context) {
          return AlertDialog(
            title: Text('Tasks'),
            content: Column(
              mainAxisSize: MainAxisSize.min,
              children: tasks
                  .map((task) => Text(
                      '${task['name']} - ${task['description']}'))
                  .toList(),
            ),
          );
        },
      );
    },
  ),
),

// Team Members List
ListTile(
  title: Text('Team Members'),
  subtitle: Text(teamMembers.isEmpty ? ' ' : "),
  trailing: IconButton(
    icon: Icon(Icons.arrow_forward),
    onPressed: () {
      showDialog(
        context: context,
        builder: (context) {
          return AlertDialog(
            title: Text('Team Members'),
            content: Column(
              mainAxisSize: MainAxisSize.min,
              children: teamMembers
                  .map((member) => Text(
                      '${member['name']} - ${member['task']}'))
                  .toList(),
            ),
          );
        },
      );
    },
  ),
),
            // Expenses List
            ListTile(
              title: Text('Expenses'),
              subtitle: Text(expenses.isEmpty ? ' ' : ''),
              trailing: IconButton(
                icon: Icon(Icons.arrow_forward),
                onPressed: () {
                  showDialog(
                    context: context,
                    builder: (context) {
                      return AlertDialog(
                        title: Text('Expenses'),
                        content: Column(
                          mainAxisSize: MainAxisSize.min,
                          children: expenses
                              .map((expense) => Text(
                                  '${expense['name']} - \$${expense['amount']}'))
                              .toList(),
                        ),
                      );
                    },
                  );
                },
              ),
            ),
          ],
        ),
      ),
    ),
  );
}

// Validation functions
```

```
String? validateProjectName(String? value) {
  if (value == null || value.isEmpty) {
    return 'Project name cannot be empty';
  }
  return null;
}

String? validateDate(String? value) {
  // Regex for date validation (MM/DD/YYYY)
  RegExp regExp = RegExp(r'^(0[1-9]|1[0-2])\/([0-2][0-9]|3[01])\/\d{4}$');
  if (value == null || value.isEmpty) {
    return 'Date cannot be empty';
  } else if (!regExp.hasMatch(value)) {
    return 'Enter a valid date (MM/DD/YYYY)';
  }
  return null;
}

String? validateBudget(String? value) {
  if (value == null || value.isEmpty) {
    return 'Budget cannot be empty';
  }
  if (double.tryParse(value) == null || double.parse(value) <= 0) {
    return 'Enter a valid positive number for budget';
  }
  return null;
}
}
```

OUTPUT:


9:21

Project List Profile

Project Management Dashboard

Streamlining your projects with ease

The Great Escape

Island Fortune

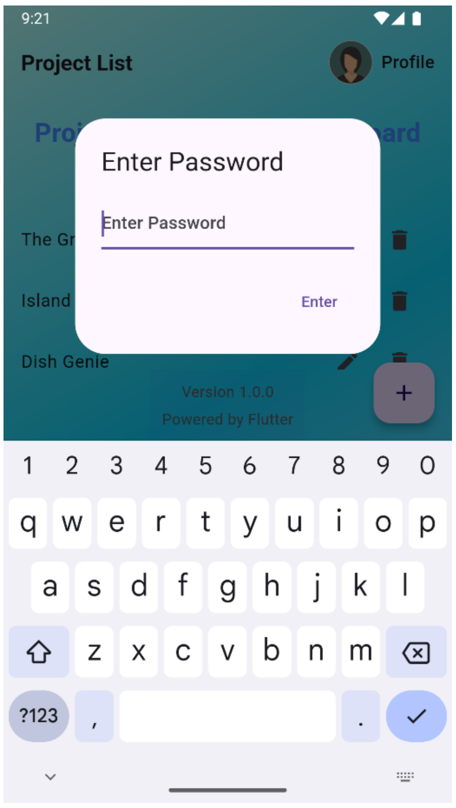
Dish Genie

Heavenly Cakes

Honeybee Breeding

Version 1.0.0

Powered by Flutter


9:21

Project List Profile

Enter Password

Enter Password

Enter

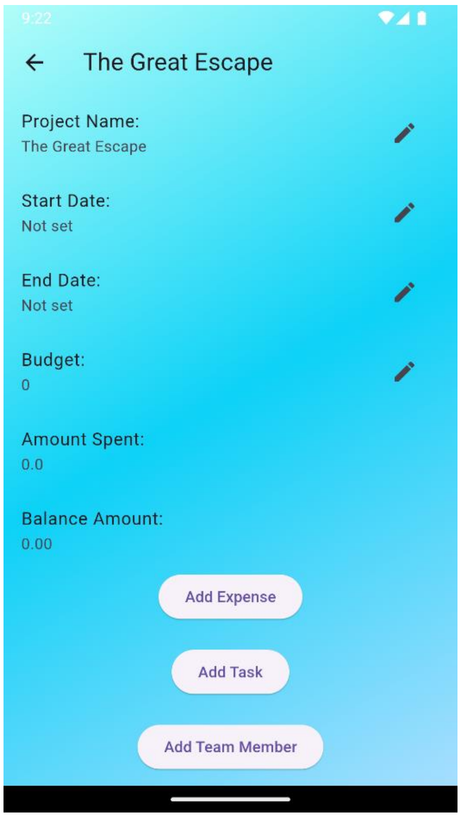
The Gr

Island

Dish Genie

Version 1.0.0

Powered by Flutter


9:22

The Great Escape

Project Name:
The Great Escape

Start Date:
Not set

End Date:
Not set

Budget:
0

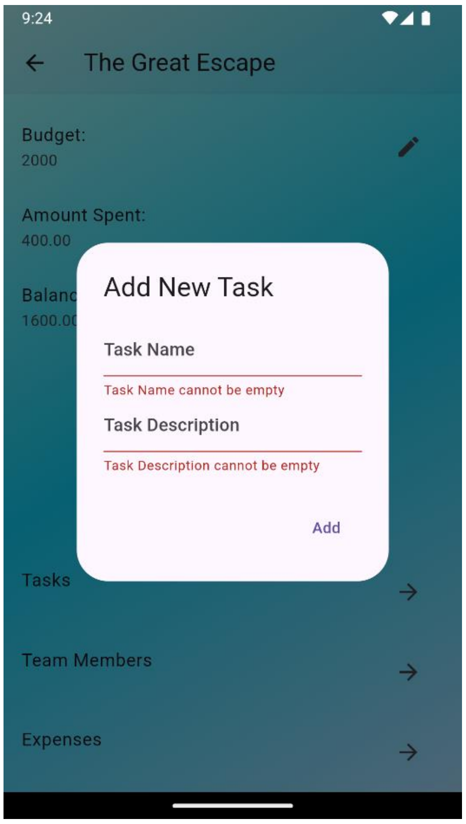
Amount Spent:
0.0

Balance Amount:
0.00

Add Expense

Add Task

Add Team Member


9:24

The Great Escape

Budget:
2000

Amount Spent:
400.00

Balanc
1600.0

Add New Task

Task Name

Task Name cannot be empty

Task Description

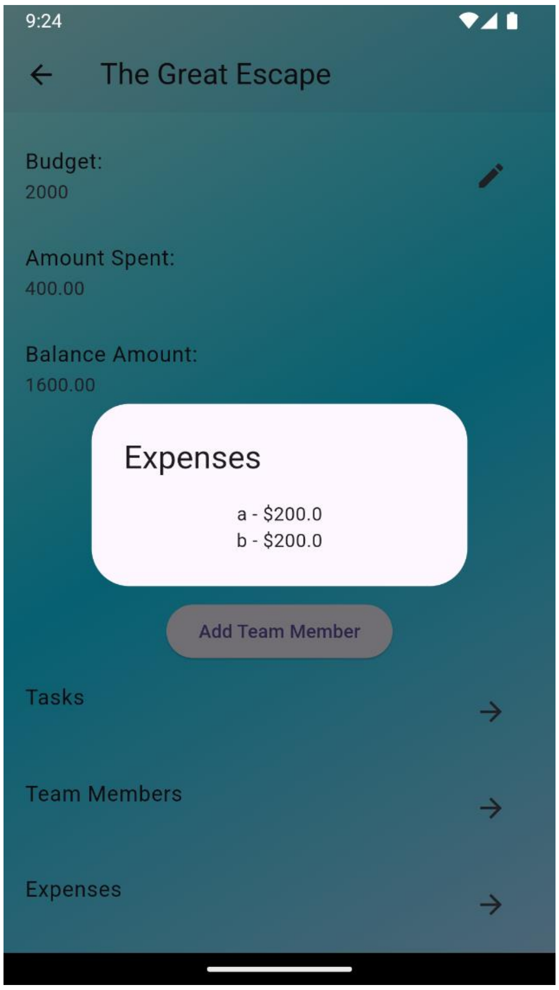
Task Description cannot be empty

Add

Tasks

Team Members

Expenses



9:24

The Great Escape

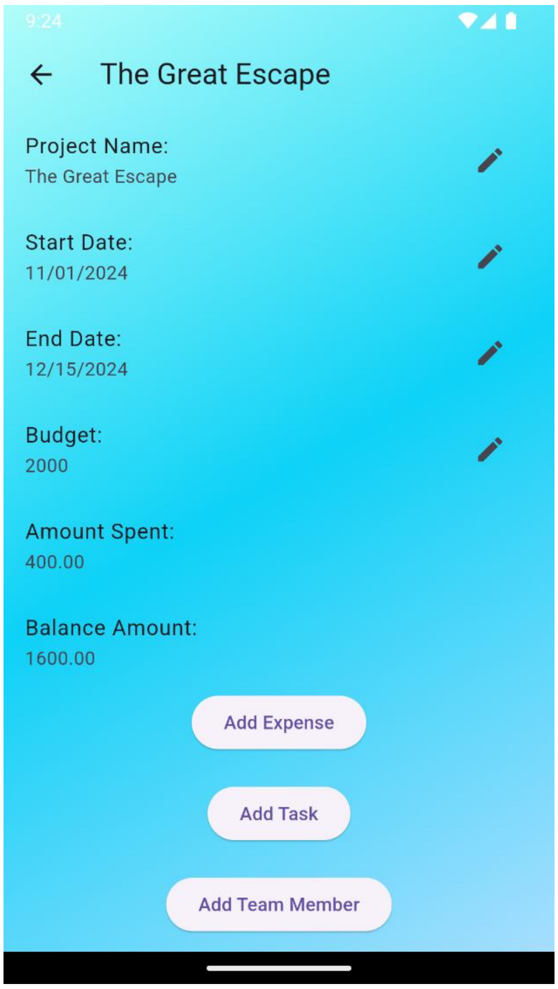
Budget:
2000

Amount Spent:
400.00

Balance Amount:
1600.00

Expenses

a - $200.0
b - $200.0



9:24

The Great Escape

Project Name:
The Great Escape

Start Date:
11/01/2024

End Date:
12/15/2024

Budget:
2000

Amount Spent:
400.00

Balance Amount:
1600.00