Orbit: Personal Well-being Web Application

Fundamentals of Software Engineering

Deliverable 2: Sprint 1 (Iteration 1)

Submission Date: 9th March 2025

Project Team

Dania Waseem	23I-2622
Munaza Tariq	23I-2545

Department of Data Science

National University of Computer and Emerging Sciences
Islamabad, Pakistan

,

Contents

List of Figures

List of Tables

Orbit: Personal Wellbeing Application

0.1 Introduction

This project, Orbit: Personal Well-being Web Application, is designed to help users manage their daily tasks, track habits, reflect on their thoughts, and improve their mental well-being. Many apps focus on only one aspect, like task management or meditation, but Orbit brings everything together in one place. It is built using C and MySQL with Windows Forms, making it a functional and user-friendly desktop application. Users can securely log in, organize tasks, monitor their emotions, and analyze their progress through interactive features.

Orbit offers several key features, including a To-Do List for organizing tasks, Journaling for self-reflection, a Habit Tracker to build positive routines, Meditation Exercises for relaxation, a Mood Tracker to log emotions, and a Monthly Overview with calendars and graphs to track progress.

The scope of Orbit is to provide an easy-to-use and efficient tool for personal growth. The app ensures secure data storage with MySQL, smooth functionality through Windows Forms, and an intuitive interface for all users.

The solution Orbit provides is a single platform for managing both productivity and well-being. Instead of using multiple apps, users can track everything in one place. With a structured C backend, a secure database, and an interactive design, Orbit ensures a seamless and practical experience. It is built to be scalable and adaptable, with future possibilities for AI-driven insights and cloud storage.

Orbit is an efficient, simple, and reliable way to help users stay productive and take care of their mental health, making everyday life more organized and stress-free.

0.2 Sprint 1 Backlog

Module for Sprint 1: Homepage, Login, Signup

0.2.1 User Stories for Sprint 1

- **User Story 1: Homepage**

  - As a *user*, I want to *view a welcoming homepage* so that I can *understand the app's purpose and features*.

    Sub-User Stories:

  - As a *user*, I want to *see a navigation bar* so that I can *easily access other sections*.
  - As a *user*, I want to *read a brief app description* so that I *understand its features*.

- **User Story 2: Login**

  - As a *registered user*, I want to *log in securely* so that I can *access my personal productivity and wellness data*.

    Sub-User Stories:

  - As a *user*, I want to *enter my email and password* so that I can *authenticate myself*.
  - As a *user*, I want to *see an error message* if my credentials are *incorrect*.

- **User Story 3: Signup**

  - As a *new user*, I want to *create an account* so that I can *save my tasks and wellness data*.

    Sub-User Stories:

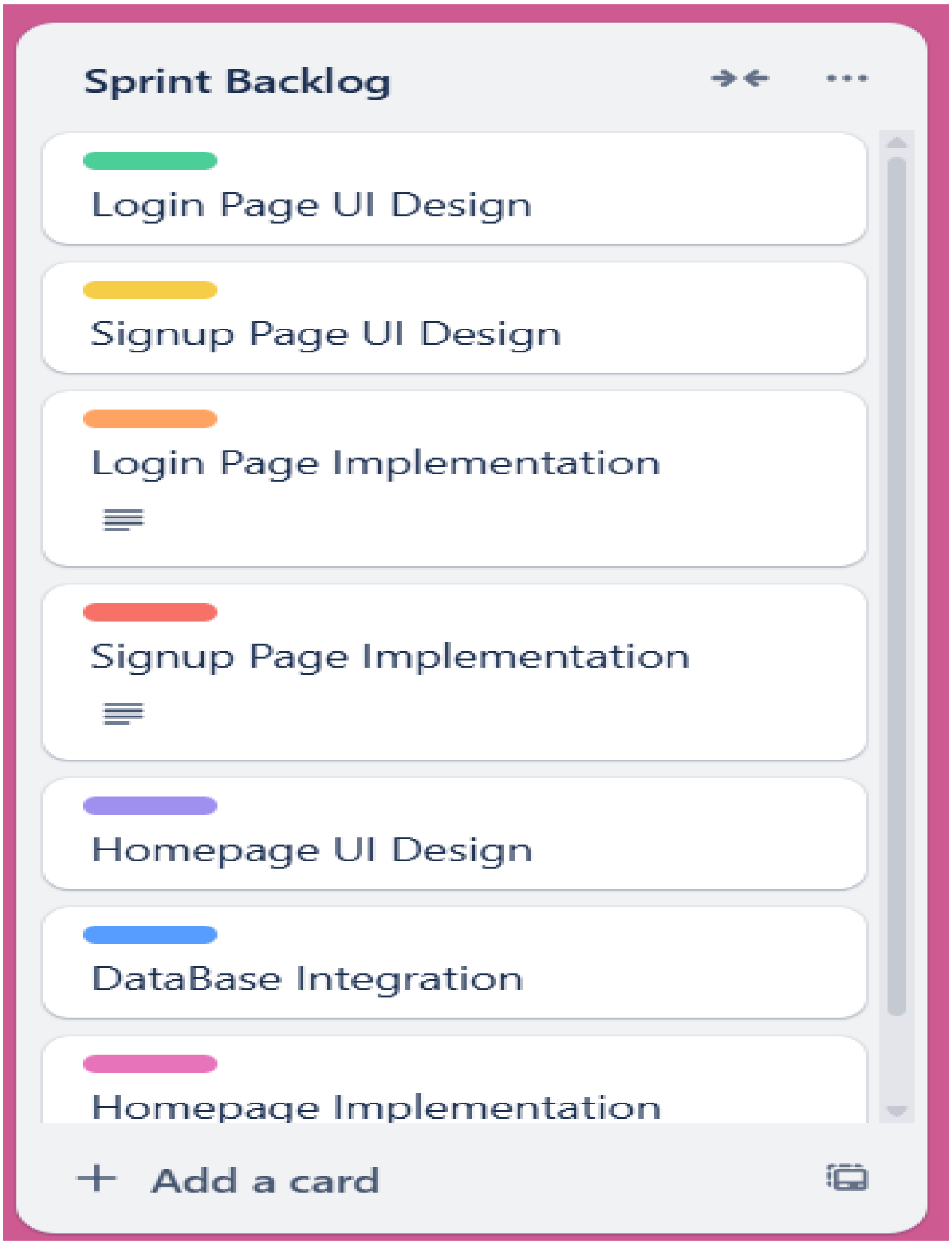
  - As a *user*, I want to *enter my name, email, and password* to *register*.
  - As a *user*, I want to *receive a confirmation email* upon *successful registration*.

0.3 Structured Specifications for User Stories

User Story	ID	Preconditions	Description	Expected Output
Homepage	US-001	None	The user visits the homepage to see an introduction to the app	The user sees a welcoming homepage with navigation and brief app details
Login	US-002	User must be registered	The user enters email and password and clicks login	If credentials are correct, the user is redirected to the homepage; otherwise, an error message appears
Signup	US-003	User is new	The user enters required details and clicks signup	If details are valid, an account is created; otherwise, an error message appears

Table 1: Structured Specifications for User Stories

0.4 Scrum Board Snapshots for Sprint 1



Sprint Backlog

Login Page UI Design

Signup Page UI Design

Login Page Implementation

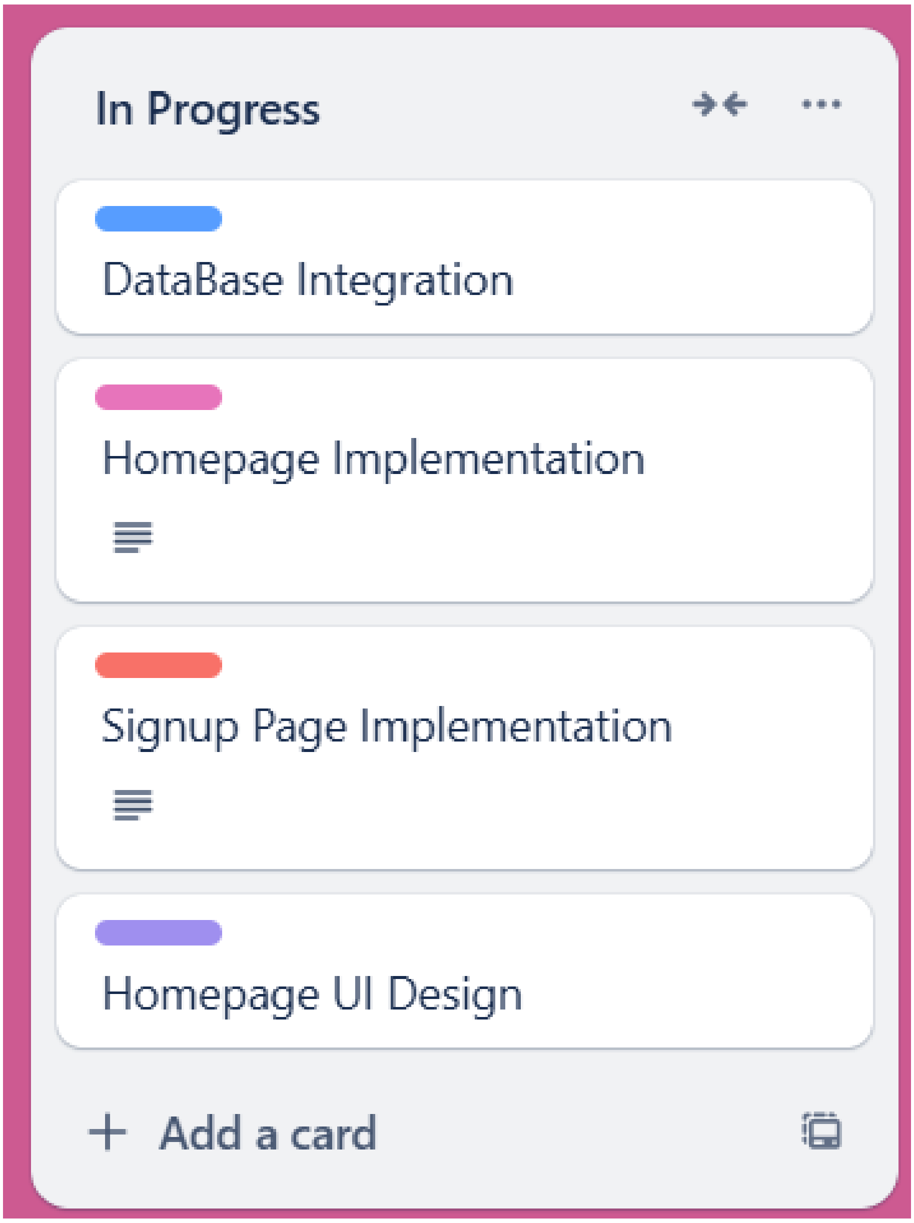
Signup Page Implementation

Homepage UI Design

DataBase Integration

Homepage Implementation

Figure 1: Snapshot 1:Sprint Backlog



In Progress

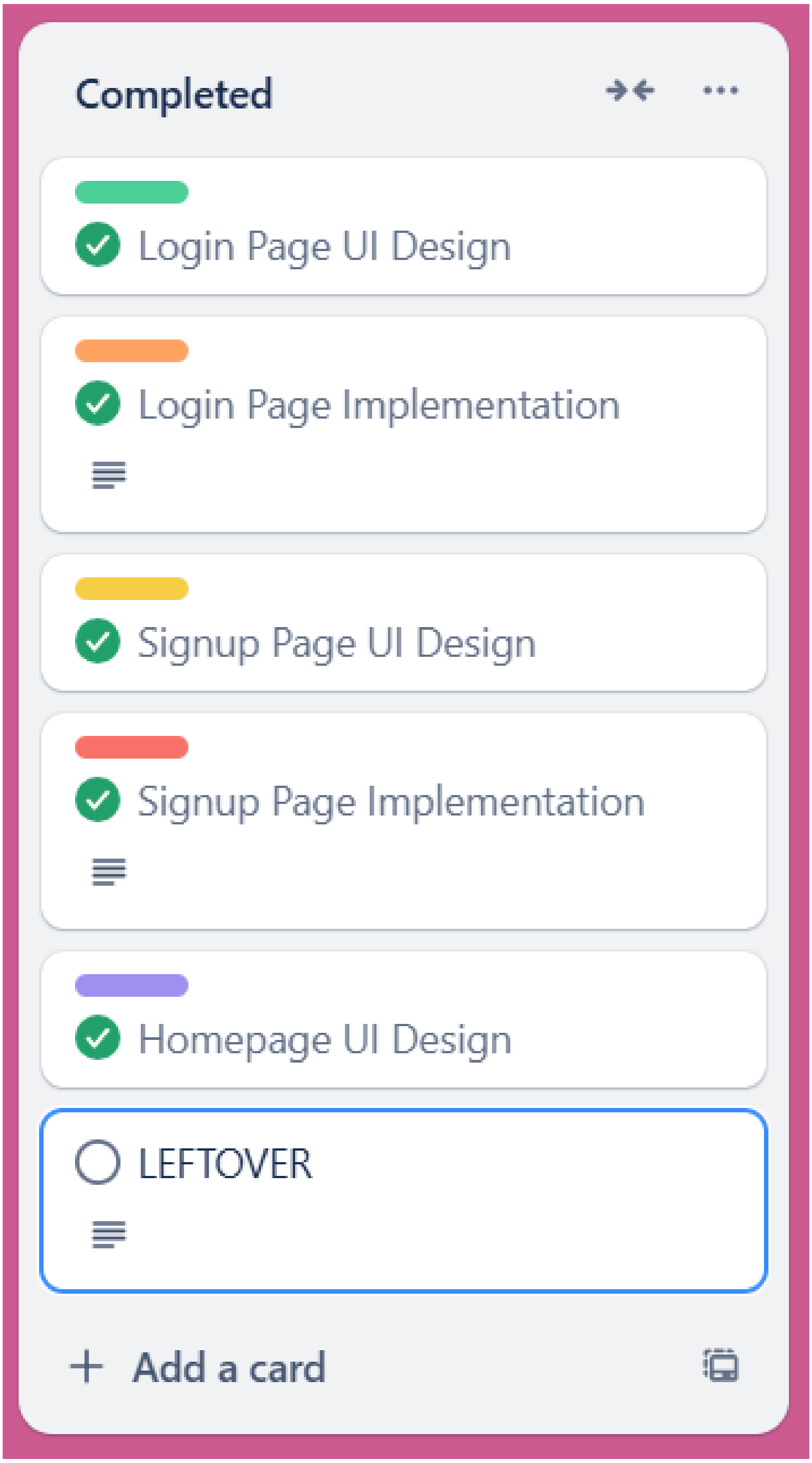
DataBase Integration

Homepage Implementation

Signup Page Implementation

Homepage UI Design

Figure 2: Snapshot 2: Mid-Sprint Progress (Half of User Stories Completed)



Completed

- ✓ Login Page UI Design
- ✓ Login Page Implementation
- ✓ Signup Page UI Design
- ✓ Signup Page Implementation
- ✓ Homepage UI Design
- ○ LEFTOVER

Figure 3: Snapshot 3: Sprint Completion (All Planned Tasks Done)

0.5 NFR Specification Using Natural Language Specification

The following non-functional requirements (NFRs) define key quality attributes of the system. Since this is a school project developed using C and Windows Forms, the focus is on usability, reliability, and maintainability rather than advanced deployment or security concerns.

Performance: The system should respond quickly to user actions. Navigation between forms, opening windows, and saving data should not take more than 1-2 seconds under normal conditions.

Security: Since this is not a publicly deployed application, complex security measures are not needed. However, user passwords should not be stored in plain text, and access to important data should be restricted to authorized users.

Usability: The application should be easy to use, even for users who are not familiar with technology. The interface should follow standard Windows Forms UI patterns, with clear labels, buttons, and instructions.

Maintainability: The code should be well-organized, using meaningful names for variables and functions, and properly commented so that future students or developers can understand and modify it if needed.

Scalability: Since this is a small-scale academic project, the system does not need to handle large amounts of data or multiple concurrent users. However, it should be structured in a way that allows additional features to be added if necessary.

Reliability: The application should work without crashing. If a user enters invalid data or performs an unexpected action, the system should provide a clear error message instead of stopping abruptly.

Compatibility: The software should run on Windows 10 or later without requiring additional installations beyond the .NET Framework and any necessary libraries for Windows Forms.

Backup and Recovery: The system does not need an automated backup feature. However, users should be able to save and reload their data manually to avoid accidental loss.

Compliance: Since this is a school project, there are no strict legal or regulatory requirements. However, the project should follow good programming practices to ensure quality and clarity.

0.6 Implementation

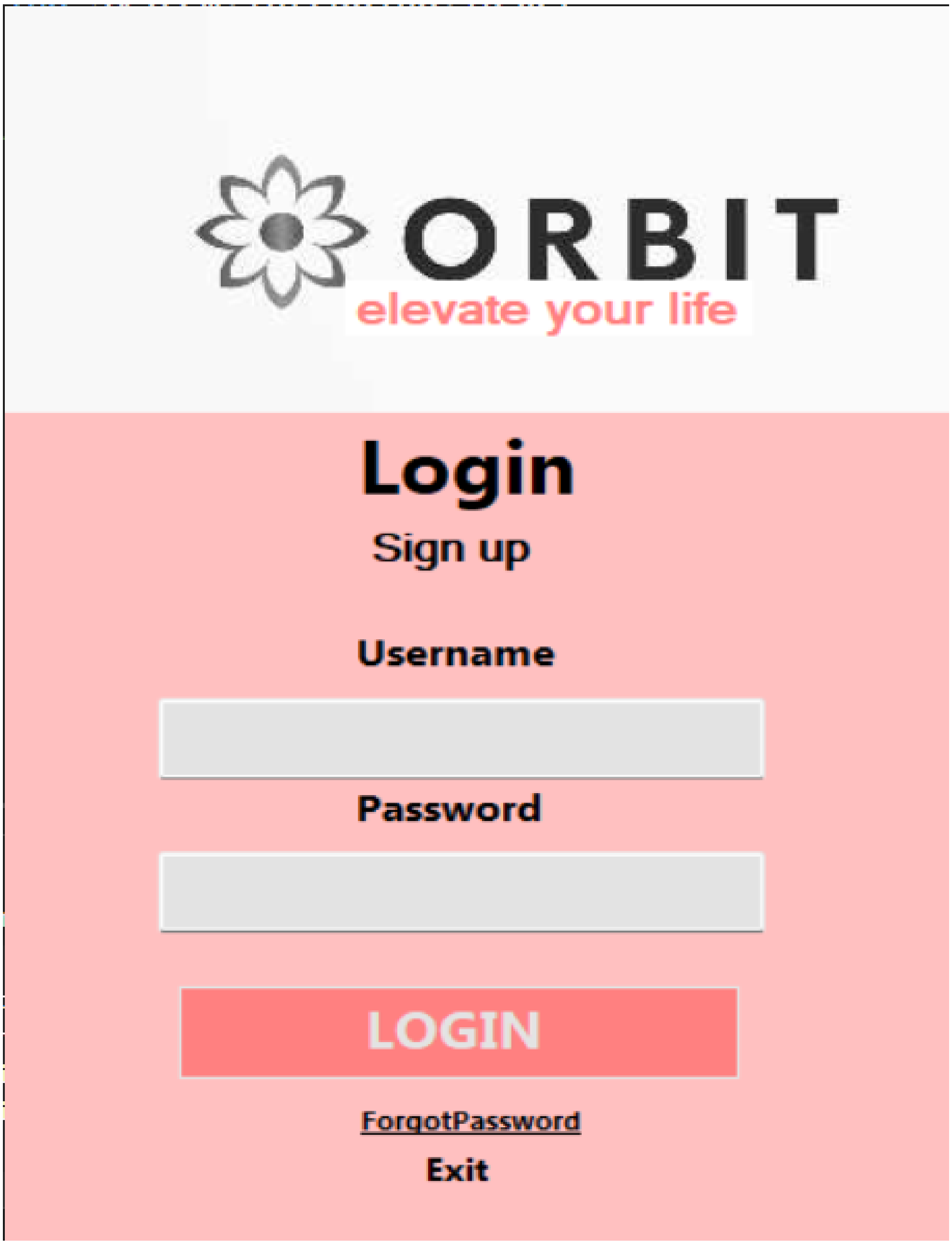


Figure 4: Iteration 1 - Feature Implementation 1 (login page)

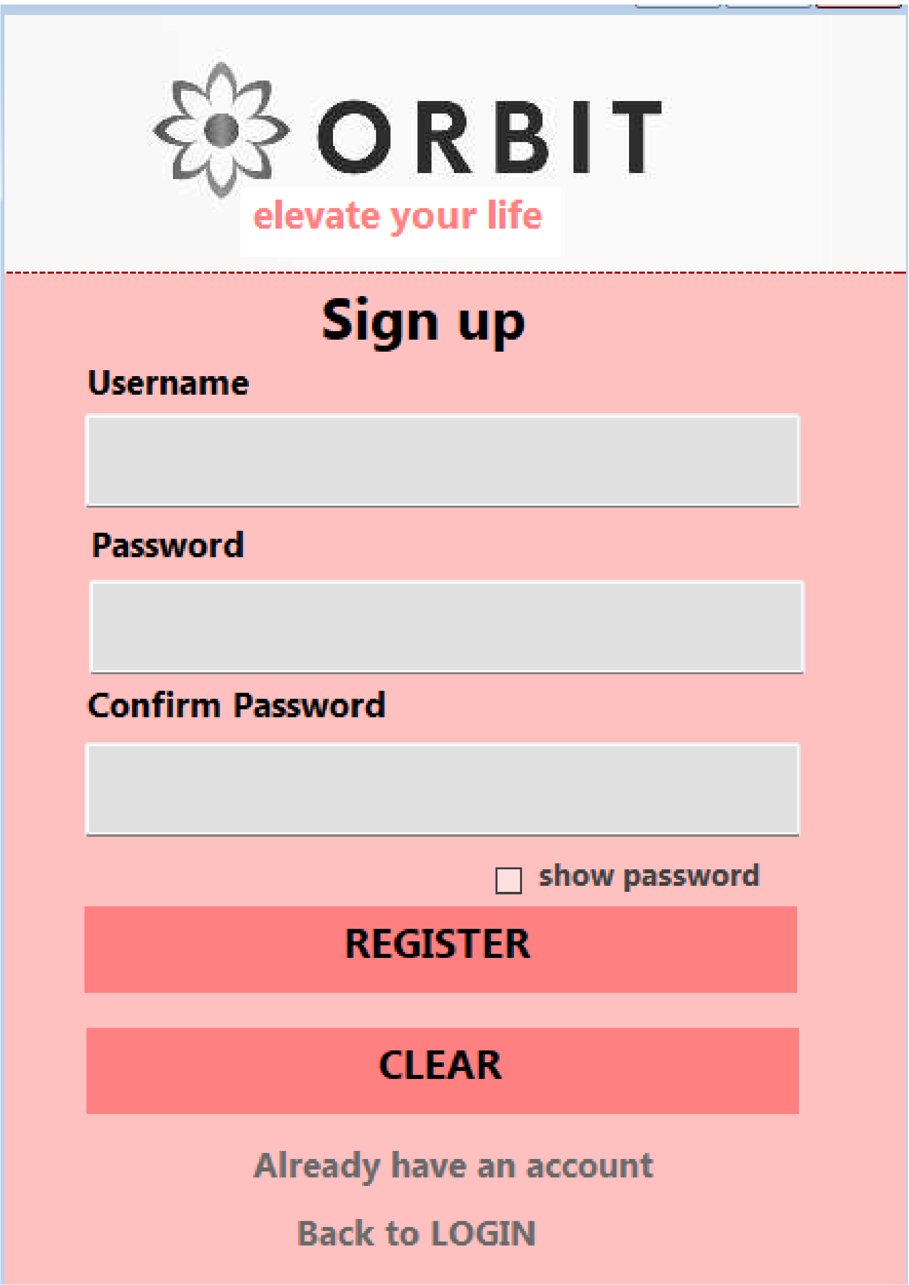


Figure 5: Iteration 1 - Feature Implementation Screenshot 2

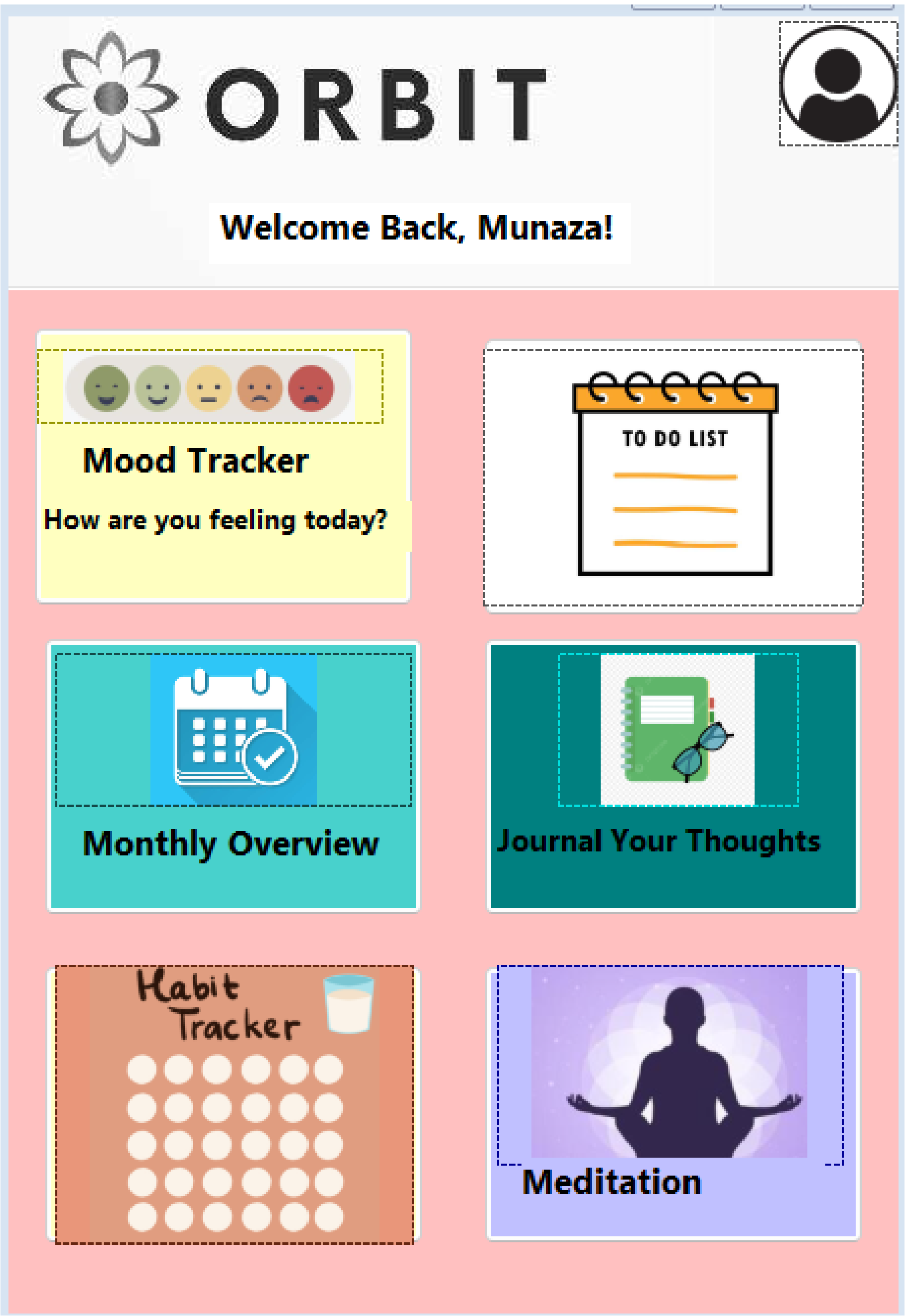


Figure 6: Iteration 1 - Feature Implementation Screenshot 3

0.7 Work Division

Team Member	Roles and Responsibilities
Dania Waseem(Team Lead)	- Managed Trello (Scrum Board) and LaTeX documentation - Designed Signup UI - Designed Homepage UI
Munaza Tariq	- Designed Login UI - Implemented Login Page - Implemented Signup Page

Table 2: Work Division