

**King Saud University**  
College of Computer and Information Sciences  
Department of Software Engineering  
**SWE 444 - Software Construction Laboratory**



# ACHIVA

## Final Report Template

#	Name	ID
1	Riyam Alsuhailani	443201116
2	Miriam Almogren	443200568
3	Shouq Algureshi	443200473
4	Amal Alharbi	442202306
5	Batool Alsugaih	443200073
6	Waad Almutiri	443200547

Section #	50083
Group #	10
Supervisor(s)	Ms. Nora Alzamel

**Submission Date: 23<sup>rd</sup> of October 2024**

## Table of Contents

Abstract .....	3
Introduction.....	4
Technical Details (Platform – Language - Tools) .....	5
System Analysis .....	7
System architecture.....	7
System database.....	8
Final Product Backlog .....	15
Original product backlog status.....	15
New features added to the product backlog .....	18
Project Plans .....	20
System Features .....	20
Project Burndown Charts .....	25
Sprint 1 .....	26
sprint 2.....	26
sprint 3.....	<b>Error! Bookmark not defined.</b>
Sprint 4 .....	27
Quality Assurance Activates .....	27
Version control .....	28
Checklist.....	28
Defect tracking .....	28
Walkthrough.....	28
Testing .....	28
In each sprint We have tested a user story of the current sprint .....	28
System Screenshots.....	30
Acquired Knowledge .....	48
Future Work .....	49
Conclusion .....	51
References .....	53
Appendices .....	54

## Abstract

Achiva is an innovative goal-tracking and social productivity platform designed to revolutionize how individuals approach personal and professional development. Its primary aim is to create a comprehensive ecosystem where users can set, track, and achieve their goals while leveraging the power of community support and social networking features. In today's fast-paced world, individuals often struggle to maintain consistency in pursuing their goals and may feel isolated in their journey toward self-improvement. Achiva effectively addresses these challenges by providing an intuitive platform where users can break down their goals into manageable tasks while connecting with like-minded individuals. The system not only simplifies goal tracking but also introduces social elements that foster motivation and accountability. Through features like progress sharing, achievement comparisons, and community support, users stay engaged and motivated throughout their journey. Additionally, Achiva's ranking dashboard and reminder system ensure users maintain momentum and celebrate their progress along the way.

## Introduction

### - Problem Statement

In the current digital age, individuals face significant challenges in maintaining focus and motivation while pursuing their personal and professional goals. The isolation of traditional goal setting approaches often leads to decreased motivation and abandoned objectives. Moreover, existing productivity tools typically lack the social engagement features that could provide crucial support and accountability. The absence of a unified platform that combines robust goal tracking capabilities with social networking features creates a gap in the market for comprehensive personal development solutions.

### - The Idea of Our Project

A mobile application designed to transform the goal setting and achievement process by integrating social networking features with traditional productivity tools. Achiva provides a centralized platform for users to set goals, track progress, connect with others, and maintain motivation through community support and engagement.

### - Target Users and Stakeholders.

#### **Stakeholders:**

Our primary stakeholder is our instructor, Ms. Nora Alzamel, as the development of Achiva is being carried out as part of the Software Construction Laboratory course at the university.

#### **Actors:**

The User is the primary actor in Achiva, representing individuals who use the app to set, track, and achieve personal and professional goals. They organize tasks, monitor progress, and engage socially by connecting with friends and sharing achievements. The user's main goal is to stay motivated and productive by leveraging the app's tools and community support.

#### **Targeted users:**

- Personal Development Enthusiasts: Users focused on self-improvement, habit formation, and personal goal achievement.
- Professionals: Individuals looking to manage career-related goals, such as skill development or project completion.
- Students: Users aiming to track academic goals, study schedules, and educational achievements.
- Fitness Enthusiasts: Individuals focused on health and fitness goals, such as exercise routines and dietary plans.

### - Values And Contributions

- Achiva eliminates the isolation often associated with personal goal pursuit.
- Provides an all-in-one platform for goal setting, tracking, and social engagement.

- Enhances motivation through community support and progress sharing.
- Facilitates long-term commitment to personal and professional development.

## **Technical Details (Platform – Language - Tools)**

### **1. Platform**

Flutter serves as our primary development framework, chosen for its robust cross-platform capabilities and extensive widget library. The framework enables us to create a consistent, high-quality user experience across different devices while maintaining efficient development cycles. Flutter's hot reload feature has been particularly valuable for rapid prototyping and iterative development.

### **2. Language**

Dart, as Flutter's primary programming language, provides the perfect balance of performance and developer productivity. Its object-oriented nature and strong typing system ensure code reliability, while its modern syntax facilitates clean, maintainable code development.

### **3. Tools**

#### **3.1 Visual Studio Code**

Visual Studio Code served as our primary integrated development environment (IDE), chosen for its exceptional versatility and developer-friendly interface. As our main development platform, it provided robust support for Flutter and Dart development through its comprehensive extension ecosystem. Key features including Git integration, smart code completion, and powerful debugging capabilities significantly enhanced our development workflow. VS Code's lightweight yet feature-rich nature made it an optimal choice for our team, enabling rapid development cycles and efficient code management.

#### **3.2 Android Studio**

While our project focused on cross-platform development, Android Studio played a vital role in our development stack, particularly for team members working with Android devices. It provided essential Android SDK management, sophisticated debugging tools, and a reliable emulation environment. The IDE's robust Flutter and Dart support ensured seamless development and testing processes, allowing team members to contribute effectively regardless of their preferred development environment.

#### **3.3 Android Studio Emulator**

The Android Studio Emulator proved invaluable for our development and testing phases. It enabled us to test Achiva across various Android device configurations and screen sizes, ensuring consistent functionality and user experience. The emulator's ability to simulate different Android versions and device specifications allowed us to identify and

address platform-specific issues early in the development cycle, significantly improving our application's reliability and compatibility.

### **3.4 Firebase Database**

We implemented Firebase Realtime Database as our primary data storage solution, chosen for its exceptional real-time synchronization capabilities and scalable architecture. This technology was crucial for handling Achiva's social features and goal-tracking functionality, enabling instant updates across devices and seamless data management. The database's flexible schema and robust security rules allowed us to efficiently store and manage user profiles, goals, achievements, and social interactions while maintaining optimal performance and data integrity.

### **3.5 Firebase Authentication with OTP**

For user authentication, we integrated Firebase Authentication with One-Time Password (OTP) functionality. This system provides a secure and user-friendly authentication process, allowing users to verify their identity through SMS codes. The implementation ensures robust security while maintaining a smooth user experience, crucial for our application's adoption and user retention.

### **3.6 Gemini API Integration**

The integration of Google's Gemini API represents a significant enhancement to Achiva's capabilities. We leveraged this advanced AI technology to provide intelligent goal suggestions, personalized motivation messages, and smart progress analysis. The API's natural language processing capabilities enable features such as automated goal breakdown suggestions and personalized achievement recommendations, adding a layer of intelligent assistance to our users' goal-setting journey.

### **3.7 GitHub**

GitHub played a central role in our development workflow, serving as our primary platform for version control and collaborative development. We implemented a structured branching strategy with a stable main branch and feature-specific branches for individual development tasks. This approach, combined with regular code reviews and pull requests, ensured code quality, and maintained project stability. The platform's issue tracking and project management features helped us maintain clear communication and coordinate effectively across the development team.

### **3.8 Trello**

Trello functioned as our agile project management tool, organizing our development process into clear, manageable stages. Our board was structured with columns for 'Backlog', 'Sprint Planning', 'In Progress', 'Testing', and 'Completed', allowing for transparent task tracking and progress monitoring. Each card represented a specific feature or user story, complete with detailed descriptions, checklists, and team member assignments. This systematic approach enabled us to maintain an organized workflow and ensure consistent progress throughout the development cycle.

# System Analysis

## System architecture

We embraced the Model-View-Controller (MVC) Architecture as the foundational structure for Achiva to ensure seamless integration with our project's environment and its specific requirements. Renowned for its effectiveness in organizing and optimizing application development, this architectural pattern excels in separating concerns, dividing the application into three essential components: Model, View, and Controller, each fulfilling distinct roles within the system.

The Model component serves as the cornerstone of data management within Achiva, maintaining vital connections and interactions with the Firebase platform. It efficiently oversees the addition and retrieval of database information, centralizing and enhancing the management of data. Meanwhile, the View component is dedicated to generating the User Interface (UI), carefully designed to meet user expectations. It includes all interactive elements such as text boxes, dropdown menus, and buttons, ensuring a seamless and user-friendly experience. Bridging the Model and the View, the Controller acts as an intermediary, orchestrating the business logic by handling user requests and responding with updates to the View or changes within the Model.

For developers operating under tight deadlines, MVC offers substantial advantages. Its clear separation of components facilitates parallel development, enabling teams to work simultaneously on different parts, thereby accelerating the overall process. Additionally, MVC's remarkable flexibility is a significant asset within Agile Methodology, supporting iterative development and allowing for frequent, easy modifications. This adaptability is paramount in dynamic environments, ensuring that Achiva can effectively evolve to accommodate both present and future demands.

The accompanying diagram illustrates the architecture of the Achiva system, emphasizing the indirect communication between the View and Model. Within this framework, the Controller mediates all interactions, receiving user input from the View, processing it, and collaborating with the Model while guiding its presentation. The architecture further outlines the integration of the database framework with the Model, facilitating seamless data access and manipulation. Additionally, the diagram incorporates the Gemini API, connected to the Controller as a crucial component.

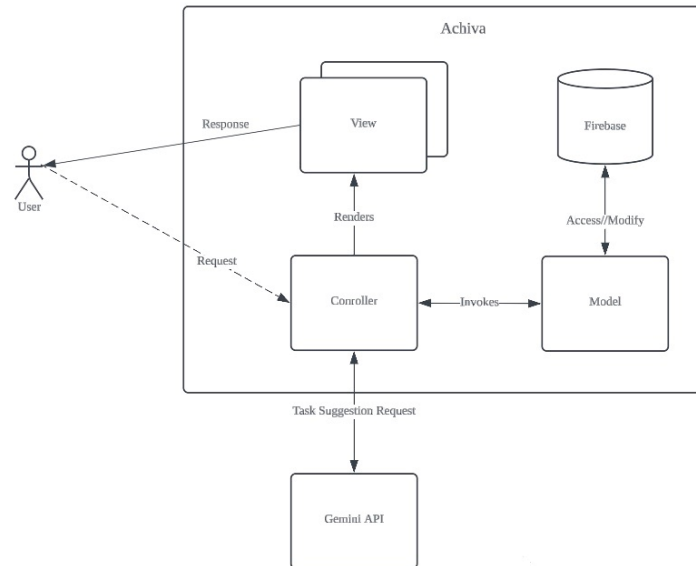


Figure [1] Achiva System Architecture Diagram.

## System database

Achiva has selected Firebase as its database solution, utilizing the Firebase Database, a cloud-based NoSQL platform. This decision stems from Firebase's open-source and cost-effective nature, combined with its rich set of features and services. Here are some compelling reasons why Achiva chose Firebase:

1. **Robust Authentication System:** Firebase offers a comprehensive authentication framework that includes various methods such as phone authentication, ensuring a secure user experience.
2. **Backend Services:** Firebase provides a reliable backend infrastructure that streamlines development, allowing for seamless integration of multiple services without the need for complex server management.
3. **Firestore Storage:** Firebase's Firestore enables efficient storage and retrieval of images and other media, ensuring that data management is both efficient and scalable.

Below is a figure illustrating all the collections utilized by the Achiva system throughout its development phases. Each collection and its corresponding fields are elaborated upon to provide clarity.

Users  
sharedGoal

Figure [2] Achiva Collections.



## User Collection:

The User Collection in our system acts as a central repository for user-specific data, facilitating effective user management and interaction within the platform. This collection contains several sub-collections, each serving distinct purposes.

### 1. RequestsStatus Collection:

This sub-collection tracks the status of user requests. It includes the following fields:

- Status: Indicates the status of the request (e.g., "accepted").
- requestId: A unique identifier for each request (e.g., "00467885-71c2-48ca-a124-45fd620cef40").
- timestamp: Records the date and time when the request was made (e.g., "23 October 2024 at 15:54:53 UTC+3").
- userId: Identifies the user associated with the request (e.g., "WINIYDZKDrerTvOZFE0rzchQZP83").

### 2. Friends Collection:

This sub-collection maintains a list of the user's friends. It contains:

- userId: The unique identifier of the friend (e.g., "U79q1DVjpXdIULIAE2Azu2KteOp2").

### 3. Goals Collection:

This collection outlines specific user goals with relevant details:

- date: The date associated with the goal (e.g., "2024-11-11").
- description: Provides additional context for the goal (currently null).
- endTime: Indicates when the goal is expected to be completed (e.g., "04:58").
- location: The location relevant to the goal (currently null).
- recurrence: Specifies whether the goal follows a recurring pattern (e.g., "No recurrence").
- startTime: The starting time for the goal (e.g., "03:59").
- taskName: The name or title of the goal (e.g., "test").

### 4. Tasks Collection:

This sub-collection captures user tasks and their statuses:

- completed: A Boolean value indicating whether the task has been completed (e.g., false).
- date: The date when the task is scheduled (e.g., "2024-10-29").
- description: A brief overview of the task (currently null).

- endTime: The end time for the task completion (e.g., "9:19 PM").
- location: The location tied to the task (currently null).
- recurrence: Indicates if the task occurs regularly (e.g., "No recurrence").
- startTime: The time when the task is set to start (e.g., "9:18 AM").
- taskName: The title of the task (e.g., "hello").

#### 5. User Profile Information:

This part of the user collection contains essential user profile information:

- email: The user's email address (e.g., "[hanan@gmail.com](mailto:hanan@gmail.com)").
- fname: The user's first name (e.g., "Hanan").
- gender: The user's gender (e.g., "female").
- id: A unique identifier for the user within the system (e.g., "4THS5VBcEDcKSX1wvdLXepkkQoF2").
- lname: The user's last name (e.g., "Khalid").
- phoneNumber: The user's contact number (e.g., "+966544556677").
- photo: A link to the user's profile picture (e.g., "<https://firebasestorage.googleapis.com/...>").
- streak: Represents the user's current streak (e.g., 0).
- streakStartDate: The timestamp indicating when the streak started (e.g., "24 November 2024 at 00:15:34 UTC+3").

This structured approach allows for comprehensive user management, enabling users to track their requests, manage friends, set and achieve goals, and organize tasks effectively.

<div>  &gt; Users &gt; 4THS5VBcEDcK. <div>More in Google Cloud</div> </div>		
<div> <div>(default)</div> <div>Users</div> <div>4THS5VBcEDcKSX1wvdLXepkkQoF2</div> </div>		
Users >	4THS5VBcEDcKSX1wvdLXepkkQoF2 >	RequestsStatus
sharedGoal	<div> <div>76X7qvzPBgkCl0pSkys8EA3...</div> <div>HHFYyBI3VrTp3aczKg9tJNVh...</div> <div>QNiGIXrjuJMnxw0qyAz2WMA...</div> <div>U79q1DVjpXdIULIAE2Azu2Kt...</div> <div>WIN1YDZKDrerTv0ZFE0rzchQ...</div> <div>c4FDYNv72u0EKWxH3BeFIOZ1...</div> <div>ccvx5cv1SwZQWQrvX3QBjONP...</div> <div>1e9jK7zdHWhua1IYgP3q1mfB...</div> <div>nvRvwoVAaYPM8yfKVA3F4P99...</div> <div>sPAASJvLPycQ9LM188e5NPct...</div> </div>	<div> <div>friends</div> <div>goals</div> <div>tasks</div> <div>email: "hanan@gmail.com"</div> <div>fname: "Hanan"</div> <div>gender: "female"</div> <div>id: "4THS5VBcEDcKSX1wvdLXepkkQoF2"</div> <div>lname: "Khalid"</div> <div>phoneNumber: "+966544556677"</div> <div>photo: "https://firebasestorage.googleapis.com/v0/b/achih-alt=media&amp;token=30fbc3e6-f130-472d-8bd8-3fbb0d5adbcc"</div> <div>streak: 0</div> <div>streakStartDate: 24 November 2024 at 00:15:34 UTC+3</div> </div>

Figure [3] Document example of User collection.

## SharedGoal collection:

The SharedGoal user group serves as a collaborative platform for users to work together towards common objectives. This group contains two primary collections, each designed to facilitate goal sharing and task management among participants.

### 1. Goal Invitations Collection

This sub-collection manages the process of inviting users to participate in shared goals. It includes the following fields:

- InvitationID: A unique identifier for each invitation (e.g., "3a3bb03b-40b4-496d-87cf-c6cb42874b8a").
- InviteAt: The timestamp indicating when the invitation was sent (e.g., "23 November 2024 at 17:03:42 UTC+3").
- fromUserID: The user ID of the person sending the invitation (e.g., "U79q1DVjpXdIULIAE2Azu2KteOp2").
- goalID: The ID of the goal associated with the invitation (e.g., "d819b05b-5d95-4f32-9aa1-4860a1649771").

- sharedID: A unique identifier linking the invitation to the shared goal (e.g., "3e971987-d280-44b7-ae9e-068c8e1c60be").
- status: The current status of the invitation (e.g., "pending").
- toUserID: The ID of the user who is invited (e.g., "HHFYyBI3VrTp3aczKg9tJNVhYc93").

## 2. Tasks Collection

This sub-collection contains tasks related to the shared goal, which are managed by the group. It includes:

- createdAt: The timestamp for when the task was created (e.g., "24 November 2024 at 02:09:07 UTC+3").
- createdBy: The user ID of the individual who created the task (e.g., "U79q1DVjpXdIULIAE2Azu2KteOp2").
- date\*: The date the task is scheduled for (e.g., "2024-11-24").
- description: Details regarding the task (currently null).
- endTime: The time by which the task should be completed (e.g., "3:08 AM").
- location: The location relevant to the task (currently null).
- recurrence: Whether the task is recurring (e.g., "No recurrence").
- startTime: The time when the task is set to begin (e.g., "2:08 AM").
- taskName: The title of the task (e.g., "shinternaladd").

Additional task details include:

- goalID: The ID of the goal this task is related to (e.g., "d819b05b-5d95-4f32-9aa1-4860a1649771").
- name: The specific name assigned to the task (e.g., "imboored").
- notasks: The number of related tasks (e.g., 6).

## 3. Participants Collection

This collection provides information about the members of the shared goal group, detailing their roles and join dates:

- joinedAt: The timestamp indicating when the user joined the shared goal (e.g., "23 November 2024 at 17:03:47 UTC+3").
- role: The role of the user within the shared goal (e.g., "owner").

Final Fields

Additional metadata associated with the shared goal includes:

- sharedID: A unique identifier linking the entire shared goal (e.g., "3e971987-d280-44b7-ae9e-068c8e1c60be").
- visibility: A boolean indicating whether the shared goal is visible to others (e.g., true).

🏠 > sharedGoal > 2a81ee3e-9a0f-... <a href="#">More in Google Cloud</a> ▾		
🏠 (default)	📁 sharedGoal ▾	📄 2a81ee3e-9a0f-4b61-b89e-ee57812cec76
Users	2a81ee3e-9a0f-4b61-b89e-... >	createdBy: "U79q1DVjpXdIULIAE2Azu2KteOp2"
sharedGoal >	3b56ded9-ac6f-452c-86c8-...	date: "2024-11-30"
	3e971987-d280-44b7-ae9e-...	goalID: "339ac6a9-7fd7-4a3d-b6eb-b1222501a05d"
	3eb2a64e-847b-4d69-a806-...	name: "Flower Event"
	4a97e6fe-2604-4b66-91b6-...	notasks: 0
	6106cfc5-3b0d-49d5-875b-...	▼ participants
	8e6f955b-ffe8-49bf-98c4-...	▼ U79q1DVjpXdIULIAE2Azu2KteOp2
	928e3928-cb20-4084-ad38-...	joinedAt: 23 November 2024 at 22:26:20 UTC+3
	e1ad1c09-703b-40d0-9dfa-...	role: "owner"
	ee6644ee-f5af-4c67-9b72-...	sharedID: "2a81ee3e-9a0f-4b61-b89e-ee57812cec76"
		visibility: true

Figure [4] Document example of SharedGoal collection.



## Final Product Backlog

### Original product backlog status

Feature ID	Brief Description	Status	Comment
1	User can sign up using email, first name, last name, gender, phone number, photo(optional).	Delivered	We have decided to switch to mobile number login, so we have removed the password from the sign-up process.
2	User can log in using his/her phone number	Delivered	We have decided to switch to mobile number sign up
3	User can reset the password when she/he forgets it using OTP.	Not Delivered	We have decided to switch to mobile number login, so we have removed the password
4	User can view her/his profile.	Delivered	
5	User can delete her/his profile.	Delivered	
6	User can edit his/her profile information (first name, last name, phone number, photo, email and gender).	Delivered	We added gender and email.
7	User can log out from her/his account.	Delivered	
8	User can view her/his goals (as a roadmap).	Delivered	
9	User can add goals (Name, Date, visibility, Tasks).	Delivered	

10	User can edit an existing goal (Name, Date, visibility, Tasks).	Delivered	
11	User can delete an existing goal.	Delivered	
12	User can add tasks (Name, description (optional), location (optional), date, time, duration (optional), repeating (optional)) for each goal.	Delivered	
13	User can edit tasks (Name, description (optional), location (optional), date, time, duration (optional), repeating (optional)) for each goal.	Delivered	
14	User can delete tasks for each goal.	Delivered	
15	User can view each task for each goal.	Delivered	
16	User can check off tasks when he/she completes them.	Delivered	
17	User can post a photo with task information (optional) to friends' feeds when he/she checks off a task for goals that are visible to his/her friends.	Delivered	
18	User can view his/her streak.	Delivered	
19	User can view his/ her progress for each goal.	Delivered	



20	User can view the countdown for each goal.	Delivered	
21	User can search for a specific user.	Delivered	
22	User can view other users' profiles.	Delivered	
23	User can add other users.	Delivered	
24	User can view a list of her/his add requests.	Delivered	
25	User can accept/reject her/his add requests.	Delivered	
26	User can view the list of her/his friends.	Delivered	
27	User can remove a specific added friend.	Delivered	
28	User can view posts by his/her friends that are posted in the friends feed.	Delivered	
29	User can react (using emojis only) to his/her friends' posts.	Delivered	
30	User can view a ranking dashboard for comparing his/her productivity with his/her friends.	Delivered	
31	User can share his/her productivity on social media platforms.	Delivered	
32	Users can chat with an AI assistant bot for task suggestions and goal-setting advice, powered by the OpenAI API.	Delivered	
33	System can send a notification when it's time to complete the task.	Delivered	One hour before the task starts.
34	Install VSCode IDE.	Delivered	
35	Install Android Studio IDE.	Delivered	

36	Install flutter SDK.	Delivered	
37	Install dart extension in VSCode.	Delivered	
38	Install flutter extension in VSCode.	Delivered	
39	Install Firebase CLI.	Delivered	
40	Firebase Configuration.	Delivered	
41	Firebase authentication using OTP integration	Delivered	
42	Install Git and GitHub desktop.	Delivered	
43	Configuration GitHub repository for version control.	Delivered	
44	OpenAi API integration.	Delivered	
45	Connect a physical Android device to Android Studio.	Delivered	
46	Set up an emulator.	Delivered	
47	Learn dart programming language.	Delivered	
48	Learn flutter framework.	Delivered	
49	Learn how to integrate OpenAi API.	Delivered	
50	Learn how to use firebase.	Delivered	
51	Learn how to use GitHub.	Delivered	
52	Learn how to integrate firebase authentication using OTP.	Delivered	

**New features added to the product backlog.**

Feature ID	Brief Description	Status	Comment
1	User shall be able to add the same task repeatedly for each goal on a weekly basis to save time and add tasks efficiently.	Delivered	The team realized the feature is more useful and preferred to the user.
2	User shall be able to view the status of their requests to know if they have been accepted or rejected.	Delivered	This feature was requested by the product owner.
3	User shall be able to delete posts to remove posts that are no longer relevant to their tasks.	Delivered	This feature was requested by the product owner.
4	User shall be able to reject/accept incoming collaboration requests to decide which shared goals they are cooperating on.	Delivered	This feature was requested by the product owner.
5	User shall be able to view an onboarding page when signing up for the first time to understand the app's main features effectively.	Delivered	The team realized the feature is more useful and preferred to the user.

# Project Plans

## System Features

Sprint no.	Release Date	Features	Description
Sprint 1	26/9/2024	Sign up	The user shall be able sign-up using phone number, emails, gender, name photo(optional).
		Log in	The user shall be able to log in using his/her phone number.
		Logout	The user shall be able to logout.
		View goal	The user shall be able to view each goal as a cards.
		Add task	The user shall be able to add tasks (Name, description (optional), location (optional), date, time, duration (optional)) for each goal.
		Add goal	The user shall be able to add goals (Name, Date, visibility, Tasks).
		View profile	The user shall be able to view his/her profile.

		Edit profile	The user shall be able to edit his/her profile.
		Delete account	The user shall be able to delete his/her account.
Sprint 2	17/10/2024	View requests	The user shall be able to view incoming requests.
		Accept/Reject requests	The user shall be able to Accept/Reject incoming requests.
		Goal Redundancy	The user shall be able to have the option to add the same task repeatedly for each goal(every week ).
		View goal (as a roadmap)	The user shall be able to view each goal (as a roadmap)
		View task	The user shall be able to view each task associated with each goal.

		Check/Uncheck task	The user shall be able to check and uncheck tasks when I complete them
		View friends feed	The user shall be able to be able to view the friends' feed
		Friends search	The user shall be able to search for a user so he can find his/her friends.
		Add friends	The user shall be able to add other users to make them his/her friends.
			The user shall be able to view other users' profiles.
		View friends list	The user shall be able to view a list of my friends
		Remove friends	The user shall be able to be able to remove friends
		Post a photo	The user shall be able to post a

			photo with content (optional) when he/she check off a task for goals that are visible to his/her friends.
		React	The user shall be able to react (using emojis only) to his/her friends' posts.
Sprint 3	02/11/2024	View productivity	The user shall be able to view his/her productivity compared to his/her friends through a ranking dashboard.
		View requests status	The user shall be able to view his/her requests' status after his/her accept or reject it.
		ChatGPT	The user shall be able to interact with a ChatGPT bot.
		View streak	The user shall be able to view his/her streak

		View countdown	The user shall be able to view the countdown for each goal
		View progress	The user shall be able to view his/her progress for each goal
		Share achievements	The user shall be able to share his/her achievements on social media
Sprint 4	23/11/2024	Delete goals	The user shall be able to be able to delete goals
		Edit goals	The user shall be able to be able to edit goals (Name, Date, visibility, Tasks).
		Share goal	The user shall be able to share a goal with my friends, so that we can cooperate in a goal as a team
		Reject/Accept collaboration requests	The user shall be able to be able to reject/ accept incoming collaboration requests



		Delete post	The user shall be able to delete posts
		Edit task	The user shall be able edit tasks (Name, description (optional), location (optional), date, time, duration (optional), repeating (optional)).
		Delete task	The user shall be able to delete tasks
		Notifications	The user shall be able to receive notifications for task reminders
		Onboarding page	The user shall be able to view an onboarding page so that he/she can understand the app's main features.

## Project Burndown Charts

Here is each sprint, start and completion date with its burndown chart.

We used a different chart extension in sprint 2 and 3 since we faced some technical problems with Trello

## Sprint 1

It started from 8/9/2025 to 26/9/2025

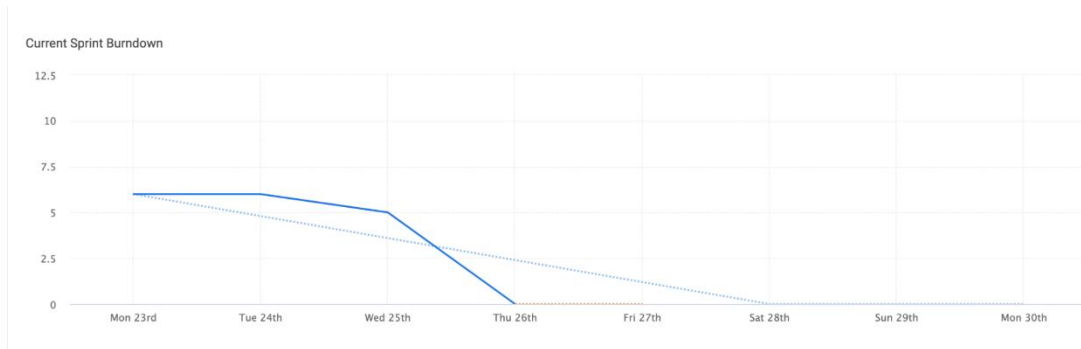


Figure [5] Sprint 1 Burndown chart

## sprint 2

It started from 29/9/2025 to 17/10/2025



Figure [6] Sprint 2 Burndown chart

### sprint 3

It started from 20/10/2025 to 3/11/2025

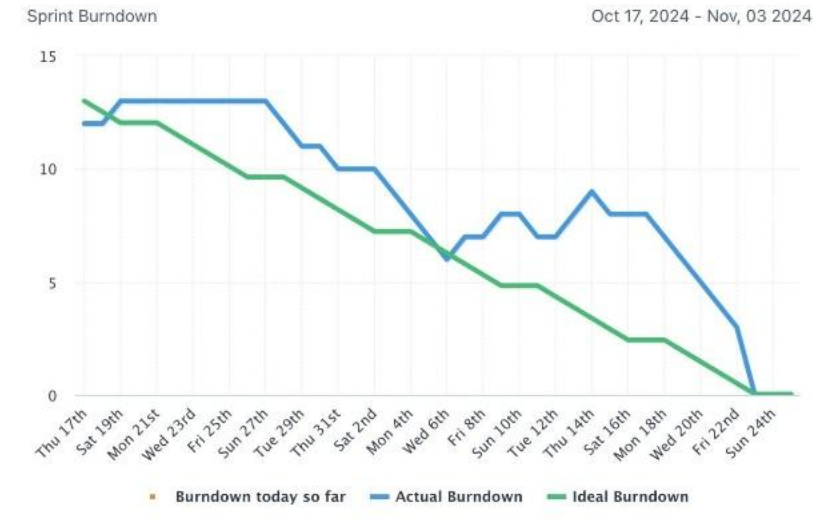


Figure [7] Sprint 3 Burndown chart

### Sprint 4

It started from 4/11/2025 to 24/11/2025

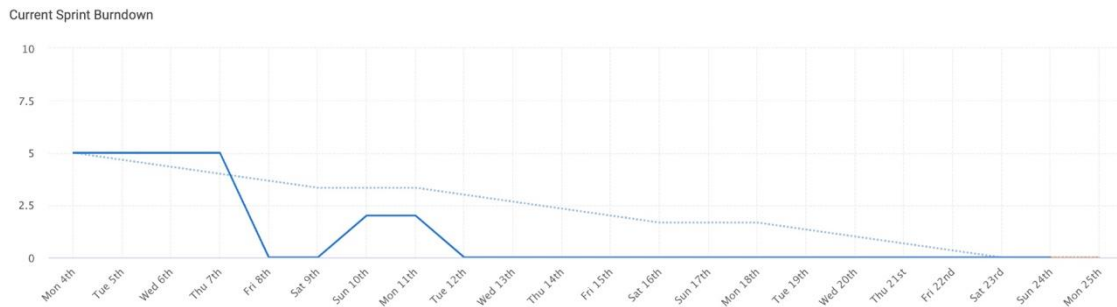


Figure [8] Sprint 4 Burndown chart

### Quality Assurance Activates

Mention at least **four** quality assurance activities (including testing as one activity) that the team followed throughout the project lifetime. Provide a clear description for each.

*In our project, we have conducted several QA activities managing defects, versions, and team management through the following activities.*

## Version control

The project utilized GitHub as its primary version control platform, which served as a centralized system for managing code development across multiple sprints. The team implemented a branching strategy where different team members worked on separate branches, allowing for development while minimizing integration conflicts. This approach enabled ease of integration, with team members able to work independently and merge their changes into the main codebase in a controlled manner. Through GitHub, we maintained clear version history, tracked changes, and managed sprint releases.

## Checklist

Checklist was a great activity we implemented that acted as a monitoring tool for team activities and deliverables. It provided a clear view of ongoing progress and team productivity levels also it helped team members to track their assigned tasks and mark completions.

## Defect tracking

It's a well-known activity to track and monitor defects and ensure fixing them by creating a defect log. It served as both a current tracking system and a historical reference, enabling the team to monitor fix progress, prevent recurring issues, and maintain accountability in the debugging process.

## Walkthrough

Prior to each sprint release, We conducted walkthroughs that served two primary purposes. First, we reviewed and validated fixes implemented based on feedback from previous sprints, ensuring that identified issues had been addressed. Second, We examined all user stories planned for the current sprint, to identify potential issues, and ensure alignment with project requirements. This proactive approach helped maintain quality standards and reduce potential issues before it reached release.

## Testing

In each sprint We have tested a user story of the current sprint

**Sprint 1:** “As a user, I would like to sign up using phone number, email, gender, name photo(optional), so that I can create a new account” [link](#)

**Sprint 2:** “As a user, I want to post a photo with content (optional) when I check off a task for goals that are visible to my friends so that I can share my achievements.” in the following [link](#).

**Sprint 3:** “As a user, I would like to be able to interact with a ChatGPT bot, so that I can ask for task suggestion” in the following [link](#).

**Sprint 4:** “As a user, I would like to be able to edit goals (Name, Date, visibility, Tasks), so that I can have the flexibility to change them anytime” in the following [link](#).



## System Screenshots

- Sign Up/Sign In

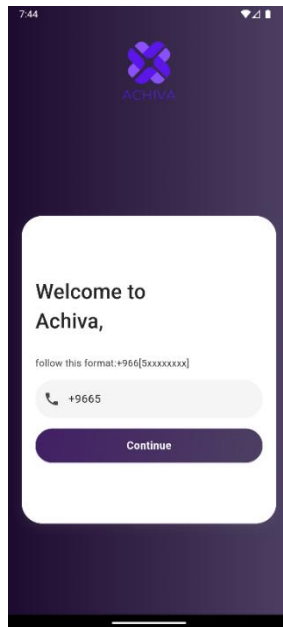


Figure [9] Phone Number to sign up or sign in

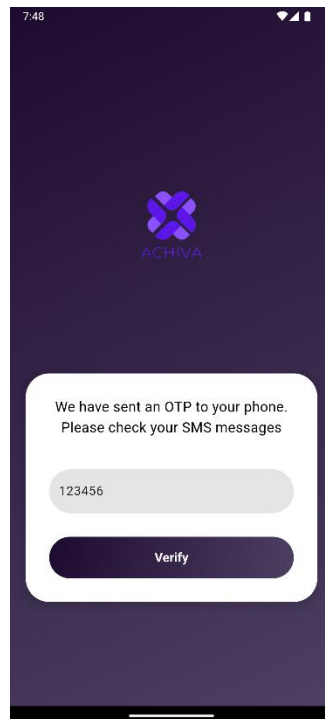
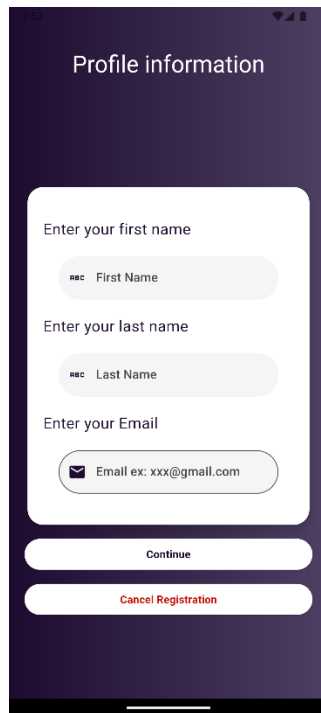
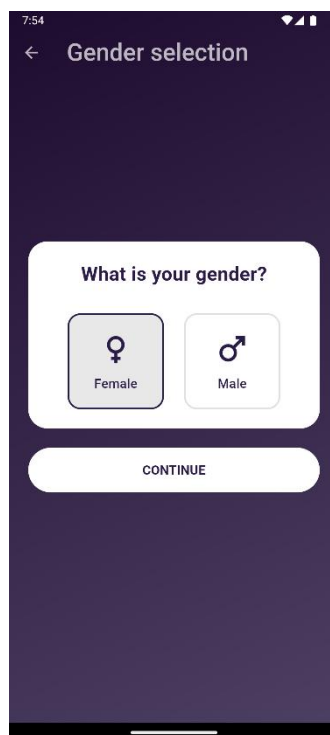


Figure [10] OTP to authenticate phone number



A mobile app screen titled "Profile information" with a dark purple background. It features a white rounded rectangle containing three input fields: "Enter your first name" with a placeholder "First Name", "Enter your last name" with a placeholder "Last Name", and "Enter your Email" with a placeholder "Email ex: xxx@gmail.com". Below these fields are two buttons: a white "Continue" button and a red "Cancel Registration" button.

Figure [11] Profile information form



A mobile app screen titled "Gender selection" with a dark purple background. It features a white rounded rectangle containing the question "What is your gender?" and two buttons: "Female" with a female symbol and "Male" with a male symbol. Below these buttons is a white "CONTINUE" button.

Figure [12] Select Gender page

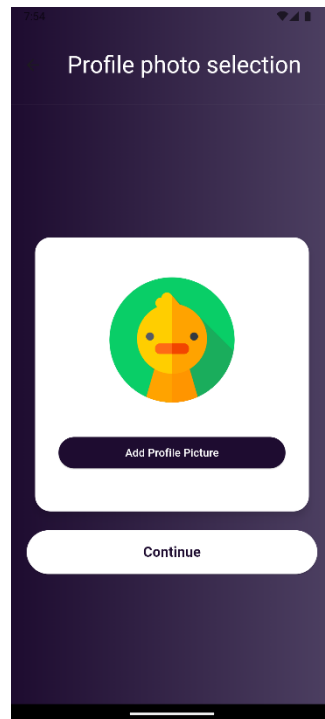


Figure [13] Add profile photo page

- **View The Onboarding Process For New App Users**





Figure [14] View The Onboarding 1

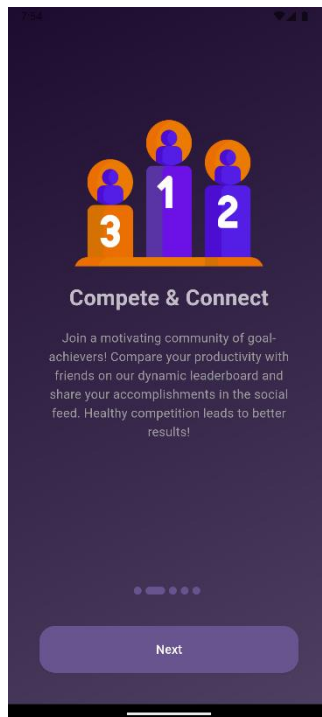


Figure [15] View The Onboarding 2



Figure [16] View The Onboarding 3

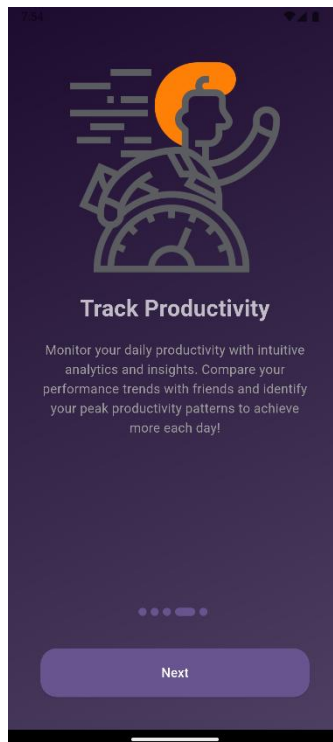


Figure [17] View The Onboarding 4

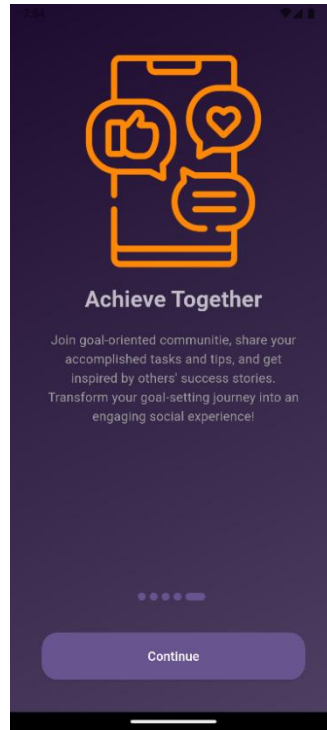


Figure [18] View The Onboarding 5

- **View Profile/ Edit Profile**

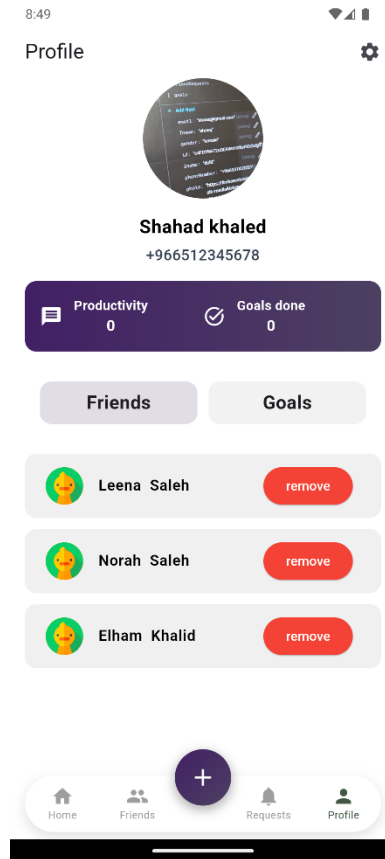


Figure [19] View Profile 1

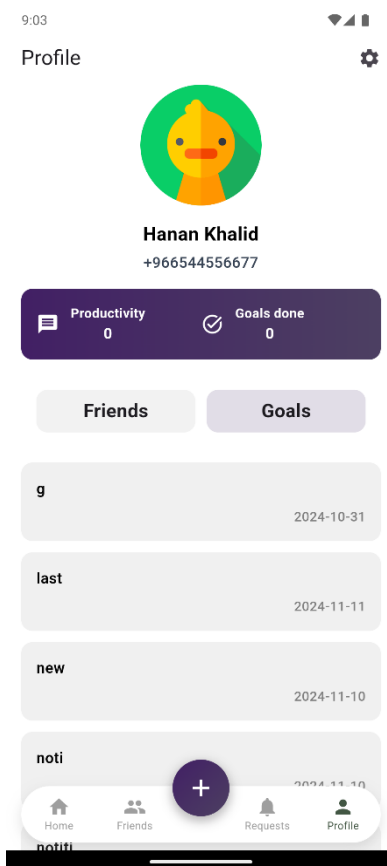


Figure [20] View Profile 2

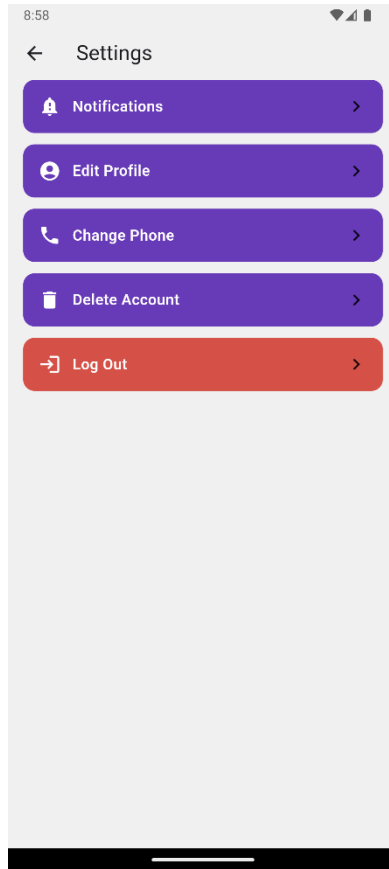


Figure [21] Edit Profile

- **Add Goal/Add Task**

8:16

← Add goal

Goal Name

Get fit

7/100

End Date: 2025-02-28

Shared goal

Visibility

Next: Add Tasks

Figure [22] Add goal

8:17

← Add Tasks

Task Name \*

Workout

7/50

Description Location

Cardio for 30 mins gym

Recurrence Day \*

Weekly recurrence 23.11.2024

Time-From \* Time-To \*

2:00 PM 3:25 PM

Save

Figure [23] Add task



- **View Goals/View Tasks**

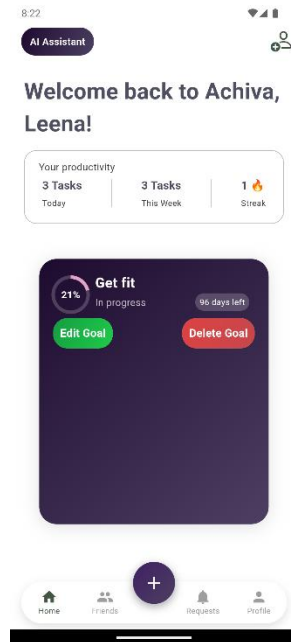


Figure [24] View Goals

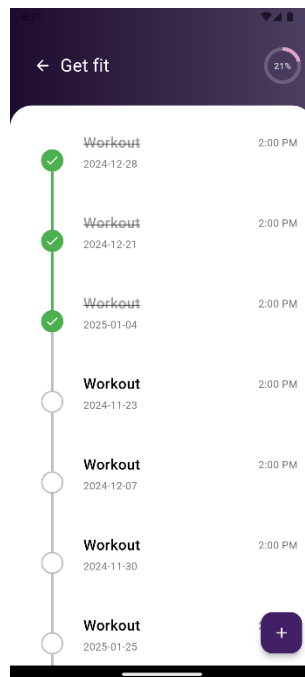


Figure [25] View Task

- **Receive Task Reminder Notification**

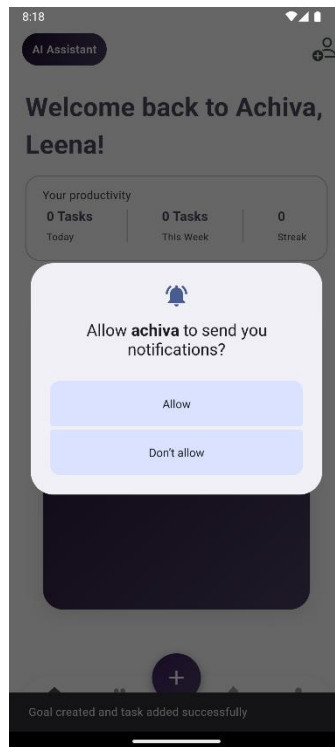
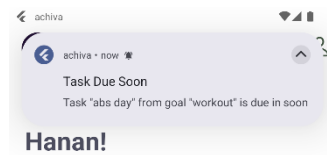


Figure [26] Take Permission Figure.



[27] Task Reminder Notification

- **Search For Friends/ Request Add Friend**

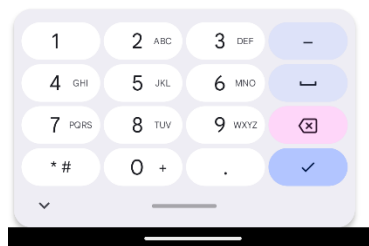
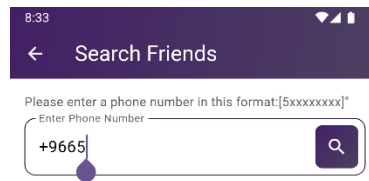


Figure [28] Search for friends 1.

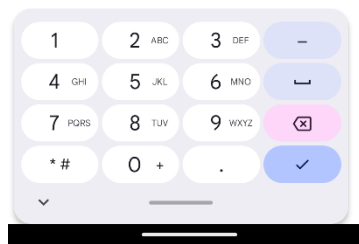
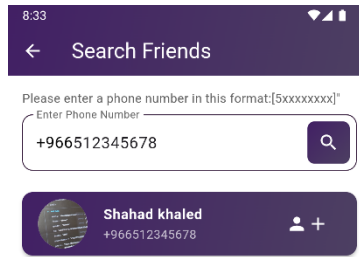


Figure [29] Search for friends 2

- **View Incoming Requests/View Request Status**

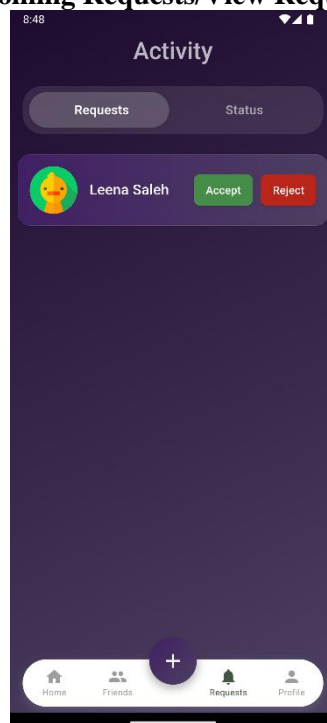


Figure [30] View incoming requests

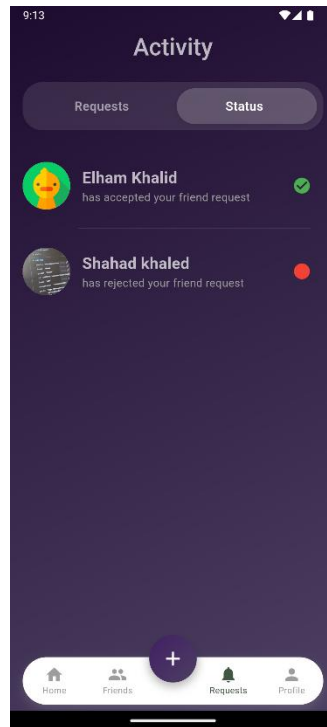


Figure [31] View requests status

- **View Friends Posts/React To Friends Posts**

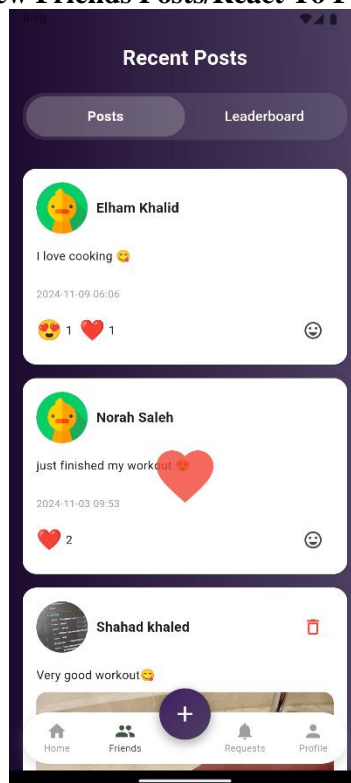


Figure [32] View friends



Figure [33] React to friends posts

- **View Leaderboard/Share Leaderboard**

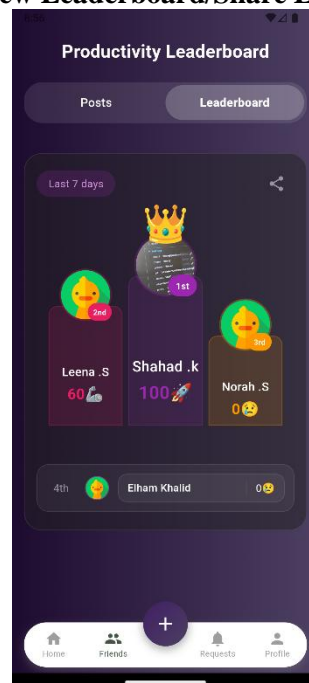


Figure [34] View leaderboard



Figure [35] Share leaderboard

- **Chat With AI Assistant**



Figure [36] Chat with AI 1.

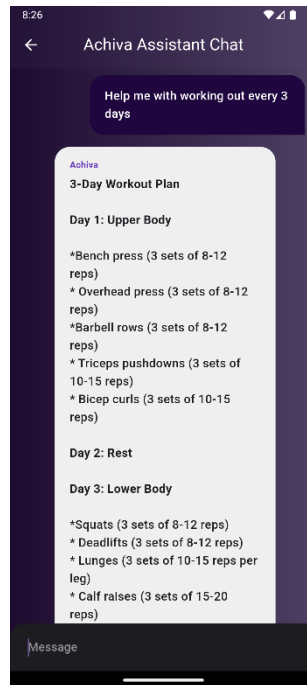


Figure [37] Chat with AI 2

## Acquired Knowledge

Our team has gained plenty of important knowledge and abilities during this project. Below is a thorough summary of everything we learned:

### Management of Projects

#### 1. Effective Planning

Recognized the importance of a comprehensive project plan. Learned to define clear objectives, tasks, deliverables, and deadlines for better work division and organization.

#### 2. Team Collaboration

Improved teamwork skills by fostering a positive environment. Enhanced conflict resolution abilities and communication practices for productive collaboration

#### 3. Risk Control

We came to understand that every undertaking has possible hazards and problems. Early detection, impact analysis, and contingency planning were essential to the success of our project.



#### **4. Managing Time**

We were professionals at setting priorities and making sure we were working quickly to fulfill deadlines.

### **Development and Technical Knowledge**

#### **1. Flutter Framework**

We learned a lot about Flutter, which is Google's UI toolkit for creating natively built apps. We discovered how to use different Flutter libraries, control app state, and make custom widgets.

#### **2. Firebase**

We discovered how to incorporate Firebase, a NoSQL cloud database, enabling real-time data syncing and storage. We became acquainted with Firebase's range of services, such as Firestore and Authentication.

#### **3. Programming with Darts**

We had to become familiar with the language because Flutter uses Dart. We studied its control flow statements, data kinds, and syntax.

#### **4. Design of UI/UX**

We developed our ability to create user-friendly and intuitive interfaces. This required knowing the fundamentals of UI/UX design and how they relate to creating mobile applications.

All things considered; our team learned a lot from this project. We enhanced our comprehension of project management and teamwork in addition to honing our technical skills. It was a demanding yet beneficial experience that will surely help our upcoming group projects.

### **Future Work**

To improve the user experience and functionality of the application, we have identified the following enhancements and future work areas:

#### **1. Increase Gamification Features**

- Badge System:

- Introduce a system where users can earn badges for achieving specific milestones or completing tasks.
- Display earned badges on the user profile to encourage engagement and provide a sense of accomplishment.

## **2. Improved Organization**

- Goal Categorization:
  - Allow users to categorize goals using labels and assign colors to differentiate between goal types.
  - Enable filtering and sorting based on these labels and categories for easier goal management.

## **3. Advanced Search Capabilities**

- Post Filtering by Friends:
  - Enable users to filter posts by their friends to quickly locate relevant content.
  - Add advanced filtering options such as date range, type of content, or reactions.

## **4. User Interface (UI) Enhancements**

- Dark Theme:
  - Introduce a dark mode option to enhance accessibility and provide a better viewing experience in low-light environments.
  - Allow seamless switching between light and dark themes in the app settings.
- Home Screen Widget:
  - Add a customizable widget for users to access their goals directly from their device home screen.
  - Display summary information, shortcuts to specific goals, or daily tasks in the widget.

## 5. AI-Powered Features

- Task Suggestion and Autofill:
  - Enable auto-filling of task details, such as time, category, or location, to save user effort.

## 6. Platform Support

- iOS System Compatibility:
  - Expand the app's availability by supporting the iOS platform.
  - Ensure a consistent and optimized experience across both Android and iOS devices.
  - Integrate with iOS-specific features such as widgets, notifications, and Apple Sign-In where applicable.

## 7. Enhanced Performance

- Faster Search Mechanisms:
  - Optimize search algorithms to provide quicker results.
  - Implement indexing techniques for posts, goals, and user data to reduce latency.

By implementing these features, we aim to improve user engagement, streamline the app's usability, and provide a more personalized and efficient experience.

## Conclusion

Achiva represents an innovative approach to personal and professional development through its mobile application platform. By seamlessly integrating task management with social networking features, the system creates a unique environment where individual goal achievement becomes a shared journey. The Flutter-based Android application empowers users not only to set and track their goals but also to draw inspiration and support from a community of like-minded individuals.

The application's strength lies in its ability to transform traditional goal-setting into an engaging, social experience. Through features like progress tracking, friend connections, and achievement sharing, Achiva turns personal development into a collaborative endeavor. This combination of productivity tools and social elements creates a powerful

platform that motivates users to stay committed to their goals while building meaningful connections with others on similar journeys.

As personal productivity continues to evolve in our increasingly connected world, Achiva stands as a testament to how technology can bridge the gap between individual achievement and community support. By bringing together goal-oriented individuals in a supportive digital environment, Achiva is poised to make a meaningful impact on how people approach and accomplish their personal and professional objectives.

## References

1. [Online]. Available: <https://waverleysoftware.com/blog/why-use-flutter-pros-and-cons/>.
2. [Online]. Available: <https://relevant.software/blog/top-8-flutter-advantages-and-why-you-should-try-flutter-on-your-next-project/>.
3. Should-try-flutter-on-your-next-project/.
4. [Online]. Available: <https://www.educative.io/answers/what-is-visual-studio-code>.
5. [Online]. Available: <https://firebase.google.com/docs/firestore>.

## Appendices

Write any additional information or data that supports or extends your report.

checklist

Grammarly

منصة سطر اعدادية

CoderHub

الترجمة - تم...

Typing Practice

IBM Academic Init...

Enterprise Manage...

المصلحة الرئيسية - تم...

Slack | Setup Welc...

RESOURCES

# to-dos

Events

# general

TEXT CHANNELS

الانكسار

مصادر

docs

# to-dos

VOICE CHANNELS

General

Shouq Online

Settings

# to-dos

Welcome to #to-dos!

This is the start of the #to-dos channel.

Edit Channel

September 22, 2024

Shouq Today at 1:27 AM

Friday : integration

Shouq Today at 4:51 AM

Saturday : interface, view goal/edit email

Sunday : device testing

Monday : report

Tuesday, wednesday for any errors

Message #to-dos

Version control

[GitHub repository link](#)

Defect tracking

Achiva sprint 3 defect tracking

File Edit View Insert Format Data Tools Extensions Help

100%

123

Calibri

11

B

I

Z

Y

Share

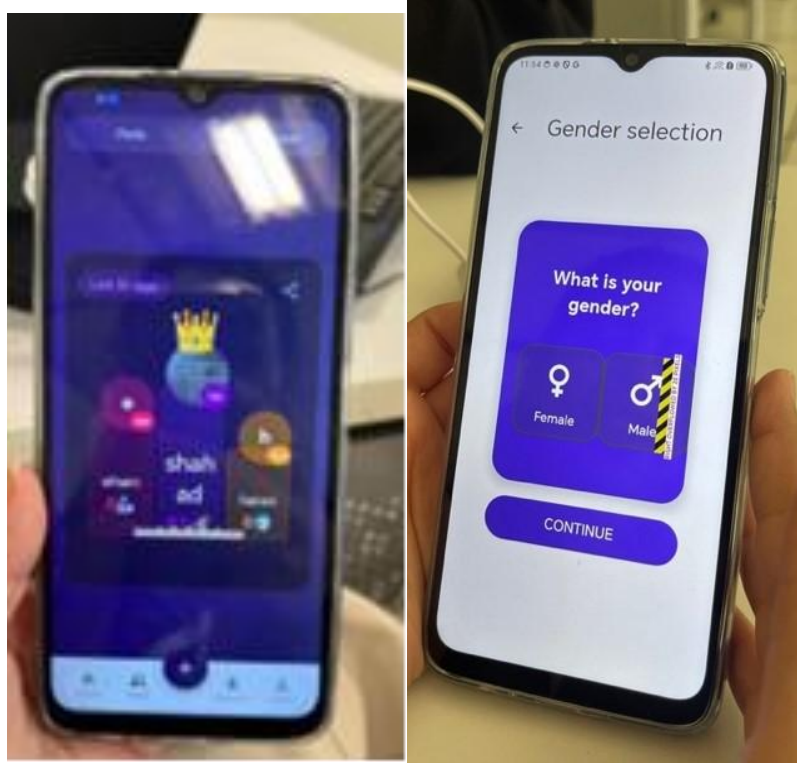
Defect ID	Detected Date	Status	H/M/L Priority	H/M/L Severity	Subject	Reported by	Defect / Issue description	Environment	Assigned to
1	21/10/2024	3-Closed	2-High	4-Low	productivity land board overflow	Shouq Alqureshi	productivity land board overflow from the button effecting the shared emage	QA	Mariam
2	21/10/2024	3-Closed	3-Medium	3-Medium	streak counting is not logically complete	Shouq Alqureshi	streak counting timing needs declaring to by post per day	Dev	waad
3	21/10/2024	3-Closed	1-Critical	1-Critical	chatbot exception	Shouq Alqureshi	chatbot throws an error	QA	Riyam,batool
4	21/10/2024	3-Closed	2-High	4-Low	request status overflow	Riyam	request status overflow from the right side	Dev	Shouq Alqureshi
5	21/10/2024	3-Closed	4-Low	4-Low	the shared photo has a transparent background	mariam	the shared photo has a transparent background	QA	Shouq

PROJECT DETAILS

DEFECT TRACKER

Dropdowns

walkthrough



Notification code

```

1  import 'dart:ffi'; // Unused import: 'dart:ffi' is not used, try removing the import directive.
2
3  import 'package:flutter/services.dart';
4  import 'package:flutter_local_notifications/flutter_local_notifications.dart';
5  import 'package:permission_handler/permission_handler.dart';
6  import 'package:rxdart/rxdart.dart';
7  import 'package:timezone/data/latest.dart' as tz;
8  import 'package:timezone/timezone.dart' as tz;
9
10 import '../core/network/cache_network.dart'; // Unused import: '../core/network/cache_network.dart' is not used, try removing the import directive.
11
12 class LocalNotification {
13   static final FlutterLocalNotificationsPlugin flutterLocalNotificationsPlugin = FlutterLocalNotificationsPlugin();
14   static final onClickNotification = BehaviorSubject<String>();
15
16   // on tap notification
17   static void onTapNotification(NotificationResponse notificationResponse) {
18     onClickNotification.add(notificationResponse.payload!);
19   }
20
21   static Future init() async {
22     flutterLocalNotificationsPlugin = FlutterLocalNotificationsPlugin();
23     // Initialize the plugin: android needs to be added as a drawable resource to the Android head project
24     const AndroidInitializationSettings initializationSettingsAndroid =
25       AndroidInitializationSettings('@mipmap/ic_launcher');
26     final DarwinInitializationSettings initializationSettingsDarwin =
27       DarwinInitializationSettings();
28     final LinuxInitializationSettings initializationSettingsLinux =
29       LinuxInitializationSettings(
30         defaultActionName: 'Open notification');
31     final InitializationSettings initializationSettings = InitializationSettings(
32       android: initializationSettingsAndroid,
33       iOS: initializationSettingsDarwin,
34       macOS: initializationSettingsDarwin,
35       linux: initializationSettingsLinux);
36     flutterLocalNotificationsPlugin.initialize(initializationSettings,
37       onDidReceiveNotificationResponse: onTapNotification,
38       onDidReceiveBackgroundNotificationResponse: onTapNotification);
39   }
40
41   static Future<bool> requestExactAlarmPermission() async {
42     final platform = const MethodChannel('dev.flutter/flutter_local_notifications_plugin');
43     final bool? result = await platform.invokeMethod<bool>('requestExactAlarmPermission');
44     return result ?? false;
45   }
46 }

```

```
AndroidManifest.xml build.gradle main.dart 2 add_redundance_tasks.dart add_task_independently_page.dart add_task_page.dart local_notification.dart 9 x ...
lib > utilities > local_notification.dart > LocalNotification > scheduleTaskHourNotification
12 class LocalNotification {
13   static Future<bool> requestExactAlarmPermission() async {
44     final bool? result = await platform.invokeMethod<bool>('requestExactAlarmPermission');
45     return result ?? false;
46   }
47
48
49   static Future showNotification({
50     required String title,
51     required String body,
52     required String payload
53   }) async {
54     const AndroidNotificationDetails androidNotificationDetails =
55       AndroidNotificationDetails('your channel id', 'your channel name',
56         channelDescription: 'your channel description',
57         importance: Importance.max,
58         priority: Priority.high,
59         ticker: 'ticker');
60     const NotificationDetails notificationDetails =
61       NotificationDetails(android: androidNotificationDetails);
62     await _flutterLocalNotificationPlugin.show(
63       0, title, body, notificationDetails,
64       payload: payload);
65   }
66
67   static Future cancelNotification({
68     required String taskName,
69     required String goalName,
70   }) async {
71     await _flutterLocalNotificationPlugin.cancel(2, tag: taskName + goalName);
72   }
73
74
75   // to schedule a local notification
76   static Future showScheduleNotification({
77     required String title,
78     required String body,
79     required String payload,
80   }) async {
81     tz.initializeTimeZones();
82     await _flutterLocalNotificationPlugin.zonedSchedule(
83       2,
84
```

```
AndroidManifest.xml build.gradle main.dart 2 add_redundance_tasks.dart add_task_independently_page.dart add_task_page.dart local_notification.dart 9 x ...
lib > utilities > local_notification.dart > LocalNotification > scheduleTaskHourNotification
12 class LocalNotification {
13   static Future showScheduleNotification({
85     title,
86     body,
87     tz.TZDateTime now(tz.local).add(const Duration(seconds: 5)),
88     NotificationDetails(
89       android: AndroidNotificationDetails(
90         'channel 3', 'your channel name',
91         tag: title + body,
92         channelDescription: 'your channel description',
93         importance: Importance.max,
94         priority: Priority.high,
95         ticker: 'ticker'), // AndroidNotificationDetails // NotificationDetails
96       androidScheduleMode: AndroidScheduleMode.exactAllowWhileIdle,
97       uiLocalNotificationDateInterpretation:
98         UILocalNotificationDateInterpretation.absoluteTime,
99       payload: payload);
100   })
101
102   static Future scheduleTaskDueNotification({
103     required String taskName,
104     required DateTime dueDate,
105     required String goalName,
106   }) async {
107     tz.initializeTimeZones();
108
109     final scheduledDate = dueDate.subtract(const Duration(hours: 1));
110
111     final PermissionStatus notificationStatus = The value of the local variable 'notificationStatus' isn't used.-Try removing the variable or using it.
112     await Permission.notification.request();
113
114     final PermissionStatus notificationStatus = The value of the local variable 'notificationStatus' isn't used.-Try removing the variable or using it.
115     await Permission.scheduleExactAlarm.request();
116     // await requestExactAlarmPermission();
117     final bool? permissionGranted = await _flutterLocalNotificationPlugin The value of the local variable 'permissionGranted' isn't used.-Try removing the variabl
118     .resolvePlatformSpecificImplementation<AndroidFlutterLocalNotificationsPlugin>()
119     ?.requestExactAlarmPermission();
120
121     await _flutterLocalNotificationPlugin
122       .resolvePlatformSpecificImplementation<
123         AndroidFlutterLocalNotificationsPlugin>()
124       ?.requestExactAlarmPermission();
125
126
```



