Software Requirement Specification Document for Promodoro Application

Abanoub George, Ibrahim Fawzy, Mai Mahmoud, Nour Bahaa.
Supervised by: Dr. Essam Eliwa.

January 5, 2021

Version	Date	Reason for Change
1.0	18-October-2020	SRS First version
2.0	23-October-2020	Updates in the pdf.
3.0	30-November-2020	SRS Video Comments Functional Requirements
2.0	5-January-2021	Changes in time plan and functional requirements

Table 1: Document version history

GitHub: `https://github.com/BonyGeorge/Pomodoro-Mobile-App`

Contents

1 Introduction **3**
- 1.1 Purpose of this document . . . 3
- 1.2 Scope of this project . . . 3
- 1.3 Overview . . . 3

2 General Description **4**
- 2.1 Product Functions . . . 4
- 2.2 Similar System Information . . . 4
  - 2.2.1 Previous systems . . . 4
- 2.3 User Characteristics . . . 5
- 2.4 User Problem Statement . . . 5

3 Functional Requirements **6**
- 3.1 System Functions . . . 6
- 3.2 Functional Specification . . . 7

4 Interface Requirements **17**
- 4.1 User Interfaces . . . 17
  - 4.1.1 CLI . . . 21
- 4.2 API . . . 21

5 Design Constraints **22**
- 5.1 Software Limitation . . . 22
- 5.2 Android SDK Version . . . 22
- 5.3 External Packages . . . 22

6 Non functional requirements **23**
- 6.1 Security . . . 23
- 6.2 Reliability . . . 23
- 6.3 Maintainability . . . 23
- 6.4 Portability . . . 23
- 6.5 Usability . . . 23

7 Operational Scenarios **24**

8 Application Structure **25**

9 Class Diagram **26**
- 9.1 Inheritance Relationships . . . 26

10 Database design description **29**

11 Widget Tree **30**

12 Github Contribution **31**

13 Time Plan **31**

1 Introduction

1.1 Purpose of this document

The point of this paper is to show full data about the task framework. The archive will depict the trademark focuses and portray the use of the Pomodoro. This archive will help the clients of the application how to utilize it and full data about the undertaking capacities and interfaces.

1.2 Scope of this project

This mobile application for Pomodoro technique. The Pomodoro Technique is a profitability framework that encourages you take the correct number of breaks while as yet completing your work. Generally, it separates your day into 25-minute center meetings followed by five-minute breaks.[3] It's the ideal period of time for absorbing information and completing things—without wearing out.

A committed Pomodoro application eliminates the need to split your day physically—rather, it lets you know precisely when to work and when to take a brief break. Here's a short gander at how the procedure functions, alongside our picks for the 10 best Pomodoro clock applications. Therefore, it will be designed to help the users the best way to try to manage their time in the best way they can. We will use flutter to build this application.This application should be ready within the end of this semester.

1.3 Overview

This application aims to help and manage their users to try to make their best in the 25 minutes of studying without any disturb to their focus. Then, it will give them the appropriate break so they can rest. So by that they can use their time in the most usable way. [2]

General rules:

- In the event that an assignment takes more than 5–7 Pomodoros, separate it
- In the event that it takes short of what one Pomodoro, include it up, and consolidate it with another errand
- When a Pomodoro starts, it needs to ring
- The following of the Pomodoro will make the users to do better
- Login to the administration and keep tabs on your development
- The Pomodoro Technique shouldn't be utilized for exercises you do in your extra time. Appreciate leisure time!

2 General Description

2.1 Product Functions

The Pomodoro Technique will give a straightforward apparatus/measure for improving profitability (your own and that of your colleagues) which can do the accompanying[5]:

- Ease tension connected to starting
- Upgrade center and fixation by eliminating interference's
- Increment consciousness of your choices
- Lift inspiration and keep it consistent
- Support the assurance to accomplish your objectives
- Refine the assessment cycle, both in subjective and quantitative terms
- Improve your work or study measure
- Fortify your determination to continue putting forth a concentrated effort even with complex circumstances

2.2 Similar System Information

This application is a stand alone application. The user need internet only for the first time when he/she install it then they can use it offline. Toward the start of every day select the undertakings you have to finish and put them on the TODO list above[1].

Begin working:

1. Start the Pomodoro clock
2. Work until the Pomodoro rings
3. Enjoy a short reprieve (3-5 minutes)

Continue working, Pomodoro after Pomodoro, until the job that needs to be done is done. Each 4 Pomodoros enjoy a more drawn out reprieve, (15–30 minutes).

2.2.1 Previous systems

1. Pomodoro, a Mobile Robot Platform for Hand Motion Exercising [4]: Moderate or extreme Mobility handicaps adversely influence how individuals perform assignments consistently, accordingly, diminishing their personal satisfaction. So by using Pomodoro mobile robot system, whose objective is to energize clients languishing from decreased portability to participate in treatment by exploiting of their proxemics space. The framework is formed of a non-holonomic stage and a cell phone with a virtual framework, both controlled through hand movement following offering an intelligent answer for performing movements. Subsequently, while the client is diverted teleoperating the Pomodoro, the movement information is caught for later appraisal with a medical services proficient.

2. Cloud Based Algorithm for Task Management based on Pomodoro technique [1]: By using Pomodoro technique it helps the user to manage his/her time and achieve the tasks perfectly. This paper illustrate more details about pomodoro by using a task management solution called (PomaFocus). PomaFocus gives users the ability to: 1) manage and create all tasks you need. 2) Have their days scheduled automatically. 3) Send warning or notifications when a task should begin. 4) Track Pomodoro cycle with timer in application. 5) Automatic rescheduling of tasks if task has not been completed.

3. An implementation to reduce internal/external interruptions in Agile software development using pomodoro technique [6]: These days, to deliver the software product in order time.the Programmer needs to focus in on their work seriously. Despite the fact that, there are many interruptions happening during work time, the product must be delivered by the deadline. Designers may need to concentrate on the best way to improve their concentration and efficiency. This paper focuses on the usage of utilizing the Pomodoro technique.

4. Can work as alarm : that ask you any questions that already have it's answer to make sure that the user is awake.

2.3 User Characteristics

Anyone can download the application. As it is made with Flutter, they can find it in both App Store and Google Play. The user needs only to have a task to do so that the application can manage it.Therefore , it will give the user track of the work that they completed and the work they need to finalize.

2.4 User Problem Statement

Everyone finds a lot of distractions in their daily routine. And they do have a lot of deadlines for either assignments or work, things that should be delivered in time, so that is when Pomodoro works come, it will manage their time and it support for less distraction . Some people forget about it so that comes the work of the Pomodoro, it benefit them to remind their deadlines. Some people feel pessimistic when they have plenty of things to do in their day and they end up doing nothing, Pomodoro is the solution because it will give hope when they find themselves finished what was needed to be finished in a very organized way in the end of the day. Therefore, they don't feel overworked in the day, as it manage both their work and their break.

3 Functional Requirements

3.1 System Functions

1. The User shall be able to create an account.
2. The User shall be able to login to use the application.
3. The User shall be able to reset password.
4. The User shall be able to add projects.
5. The User shall be able to edit projects.
6. The User shall be able to delete projects.
7. The User shall be able to add tasks.
8. The User shall be able to delete tasks.
9. The User shall be able to assign tasks to a team member.
10. The User shall be able to add member.
11. The User shall be able to remove member.
12. The User shall be able to start Pomodoro timer.
13. The User shall be able to pause Pomodoro timer.
14. The User shall be able to reset Pomodoro timer.
15. The User shall be able to take a short break after each Pomodoro.
16. The User shall be able to take a long break for Pomodoros.
17. The User shall be able to logout.

3.2 Functional Specification

Code	FR1.
Name	SignUp.
Description	User can create an account to start using the mobile application and to login to the system, by using username and password.
Priority	Extreme.
Critical	If the user didn't Sign up he can't access some functions in the system.
Type	void.
Input	Name, E-mail, Password, and confirm password.
Output	Registration Successful, proceed to post-Login page / Registration failed, Try again.
Pre-Condition	User open the application.
Post-Condition	Registration Successful, proceed to login page.
Dependencies	None.
Action	Function takes E-mail and Password and save his/her data for the user to login anytime.

Code	FR2.
Name	Login.
Description	User must login into the system to start using it's main functions, by using his/her E-mail and password.
Priority	Extreme.
Critical	If the user didn't have an account they can't use some functions.
Type	Void.
Input	E-mail, Password.
Output	Login Successful, proceed to post-Login page / Login failed, Try again.
Pre-Condition	No access to some of the system's functions and data.
Post-Condition	Login Successful, proceed to post-login page and grant access to the system's functions and data.
Dependencies	SignUp (FR1).
Action	Function takes E-mail, Password, and check whether the account is valid or not.

Code	FR3.
Name	ResetPassword.
Description	If the user forgot his password the system will send them mail to reset password.
Priority	High.
Critical	If the user forgot his password.
Type	Void.
Input	E-mail and ConfirmEmail.
Output	Password changed successfully / Try again.
Pre-Condition	SignUp.
Post-Condition	None.
Dependencies	SignUp (FR1).
Action	Function takes from the user the password and change it in database.

Code	FR4.
Name	ChangeProfile.
Description	If the user need to change any information about his profile.
Priority	High.
Critical	Change private information of the account.
Type	Void.
Input	args.
Output	Changes done successfully / Try again.
Pre-Condition	Signup.
Post-Condition	None.
Dependencies	Signup (FR1).
Action	Function takes user's args and updates them in the database.

Code	FR5.
Name	AddMember.
Description	If User needs to add any other member to create his team.
Priority	High.
Critical	Add member to team.
Type	Void.
Input	MemberId, MemberName.
Output	Member added successfully.
Pre-Condition	There is a project.
Post-Condition	Add another member.
Dependencies	AddProject (FR7).
Action	Function takes MemberID, MemberName then added the user to the team.

Code	FR6.
Name	RemoveMember.
Description	If there is a problem and the original user needs to remove any other member from his team.
Priority	High.
Critical	Remove member from team.
Type	Void.
Input	UserId.
Output	Member deleted successfully.
Pre-Condition	None.
Post-Condition	None.
Dependencies	AddMember (FR5).
Action	Function takes UserId and delete the user from the database in this team.

Code	FR7.
Name	AddProject.
Description	If the user needs to add project to use it with pomodoros.
Priority	High.
Critical	User can add project.
Type	Void.
Input	ProjectName.
Output	Project added to the list.
Pre-Condition	Project List.
Post-Condition	None.
Dependencies	Login (FR2).
Action	Fuction takes ProjectName and added the project to the user.

Code	FR8.
Name	EditProject.
Description	If the user needs to edit any project from his list.
Priority	High.
Critical	User can edit his project.
Type	Void.
Input	args.
Output	Project edited successfully.
Pre-Condition	Project List.
Post-Condition	None.
Dependencies	Login (FR2).
Action	Function takes args and change the date of selected project in database.

Code	FR9.
Name	DeleteProject.
Description	If the user wants to delete a project.
Priority	High.
Critical	Remove Project from the list.
Type	Void.
Input	ProjectID.
Output	Project deleted successfully.
Pre-Condition	None.
Post-Condition	None.
Dependencies	Login (FR2).
Action	Function takes ProjectID and delete the project from the database to this user.

Code	FR10.
Name	AddTask.
Description	If the user needs to add tasks.
Priority	High.
Critical	User can add tasks.
Type	Void.
Input	args.
Output	Tasks added successfully.
Pre-Condition	User is logged in.
Post-Condition	Task added successfully.
Dependencies	Login(FR2).
Action	Function takes task details and add this task.

Code	FR11.
Name	DeleteTask.
Description	If the user wants to delete unwanted task.
Priority	High.
Critical	Remove task from list.
Type	Void.
Input	TaskID.
Output	Task deleted successfully.
Pre-Condition	A task exists.
Post-Condition	None.
Dependencies	Login(FR2).
Action	Function takes task id and deletes this task.

Code	FR12.
Name	AssignTask.
Description	User can assign a task for his team.
Priority	Low.
Critical	Assign a task to a specific user.
Type	Void.
Input	Task, UserID.
Output	Task has been Assigned.
Pre-Condition	User created a project and added members.
Post-Condition	None.
Dependencies	AddProject (FR7).
Action	FUnction takes the task and assign it to a specific user.

Code	FR13.
Name	ViewCompletedTask.
Description	If the user wants to view his completed tasks.
Priority	High.
Critical	User has a task and finished it.
Type	Void.
Input	TaskID.
Output	View all the tasks that the user has completed.
Pre-Condition	User had a task.
Post-Condition	None.
Dependencies	AddTask (FR10).
Action	Function returns a screen full of the completed tasks of the user.

Code	FR14.
Name	StartPomodoro.
Description	If the user needs to start the timer using page timer press button start and the function count down 25 min then stops.
Priority	High.
Critical	User start pomodoro.
Type	Void.
Input	None.
Output	None.
Pre-Condition	Page Timer.
Post-Condition	Page Break.
Dependencies	Login (FR2).
Action	Press the button start and the function starts counting 25 min.

Code	FR15.
Name	PausePomodoro.
Description	If the user needs to pause the timer for any reason and want to complete the work later on.
Priority	High.
Critical	Pause the timer.
Type	Void.
Input	None.
Output	Time stopped due to interruption.
Pre-Condition	Start Pomodoro timer.
Post-Condition	Resume the work / Reset Pomodoro and start again.
Dependencies	StartPomodoro (FR14).
Action	Press the button pause and the function stop the count down.

Code	FR16.
Name	ResetPomodoro.
Description	If the user needs to restart the work form the beginning.
Priority	High.
Critical	Re-start the timer.
Type	Void.
Input	None.
Output	None.
Pre-Condition	Pause the timer.
Post-Condition	Page timer again.
Dependencies	PauseTimer (FR15).
Action	Press the button reset and the function starts counting from 25 min again.

Code	FR17.
Name	ShortBreak.
Description	After finishing the work if the user needs to take a short break 5min.
Priority	High.
Critical	Short Break to refresh.
Type	Void.
Input	None.
Output	An alarm to tell the user that he finished his break.
Pre-Condition	Finish the timer.
Post-Condition	Page Break.
Dependencies	StartPomodoro (FR14).
Action	Press the button break after finishing the work and the function takes you to the break page.

Code	FR18.
Name	LongBreak.
Description	Instead of a short break the user can take a long 15 minutes break after finishing four Pomodoros.
Priority	High.
Critical	Take a long break to refresh.
Type	Void.
Input	None.
Output	An alarm after it to tell the user that the break is finished.
Pre-Condition	Finish the timer.
Post-Condition	Page Break.
Dependencies	StartPomodoro (FR14).
Action	Press the button break after finishing the work and the function takes you to the break page.

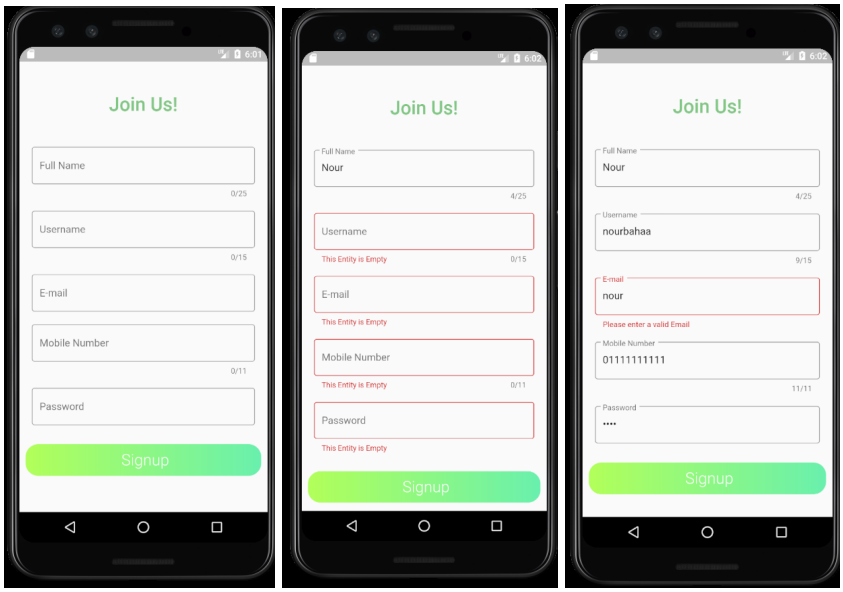
Code	FR19.
Name	Logout.
Description	The user finished their work and exit from the mobile application.
Priority	Extreme.
Critical	To exit from the web application account.
Type	void.
Input	None.
Output	Logged out successfully.
Pre-Condition	Login.
Post-Condition	Exit from application.
Dependencies	Login (FR2).
Action	Function exit the user from the mobile application.

4 Interface Requirements

4.1 User Interfaces

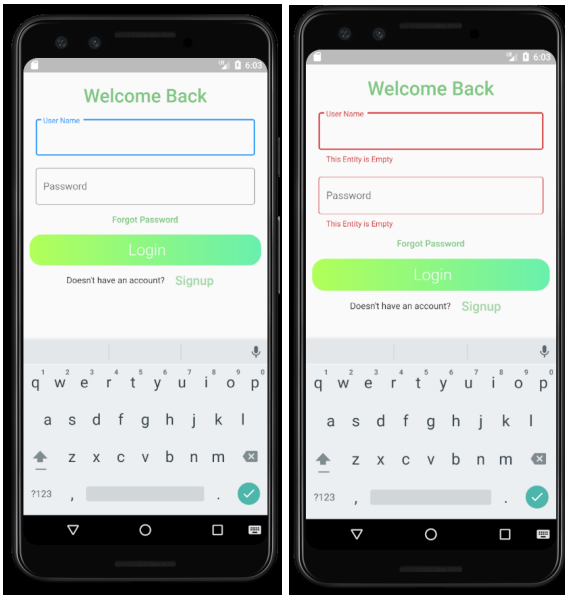
This application will have a great accommodating UI&UX by that they can both understand and use it in the easiest way.

Figure 1: Landing Frame of the application.



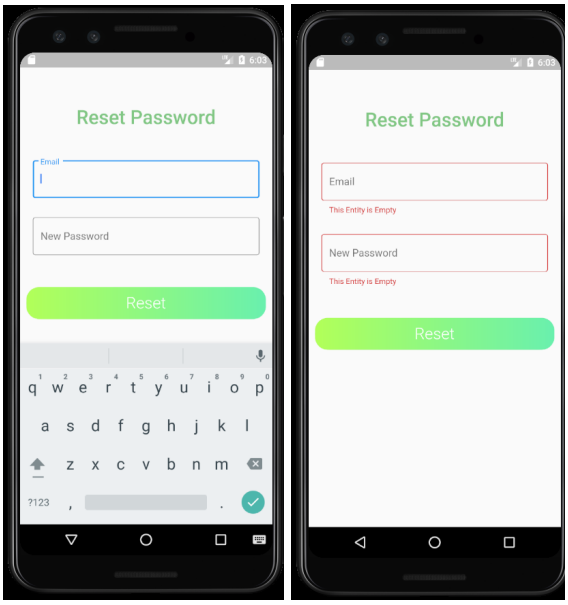
(a) Sign Up (b) Sign up Validation (c) Email Validation

Figure 2: User interface



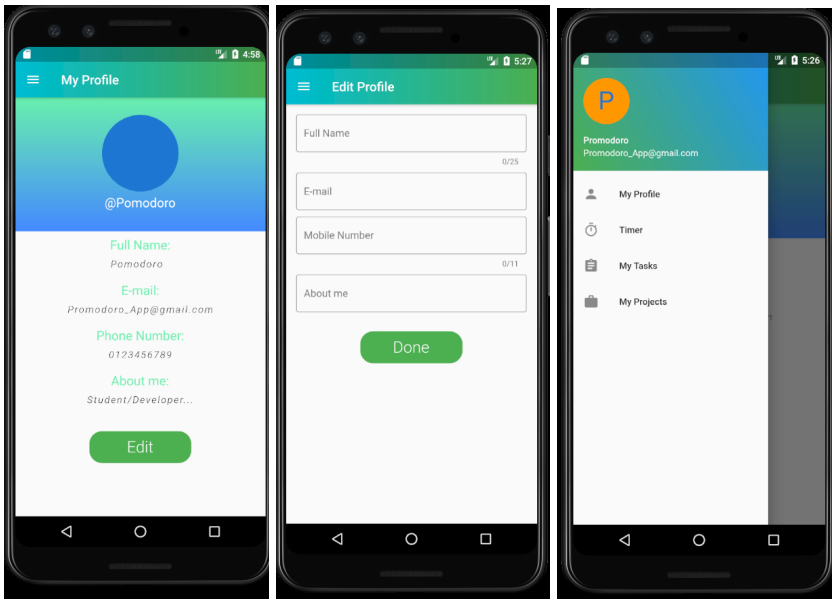
(a) Login (b) Login Validation

Figure 3: Login Pages



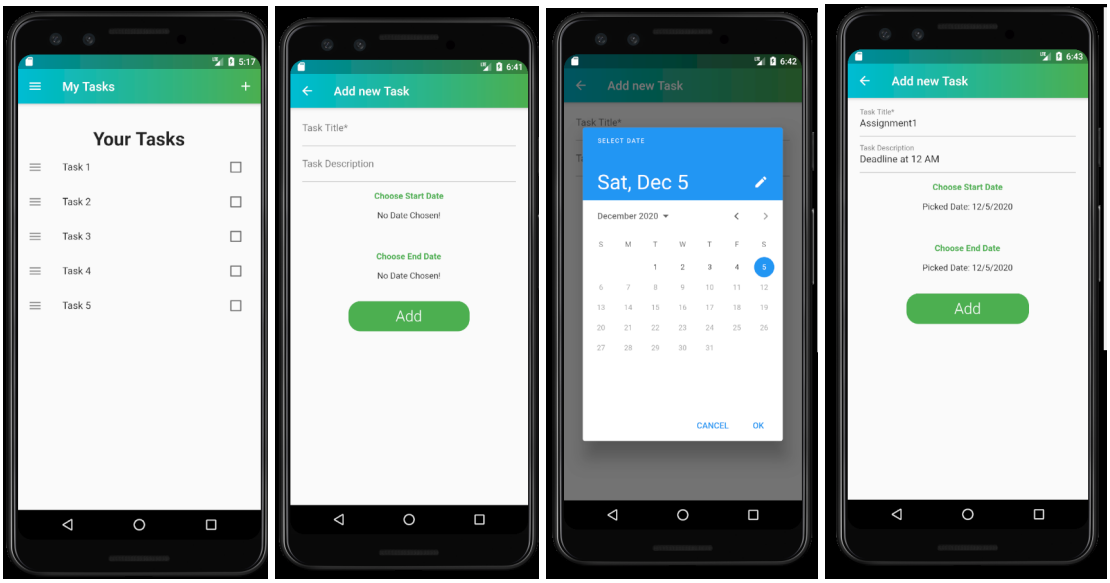
(a) Password (b) Validation

Figure 4: Reset Password Pages



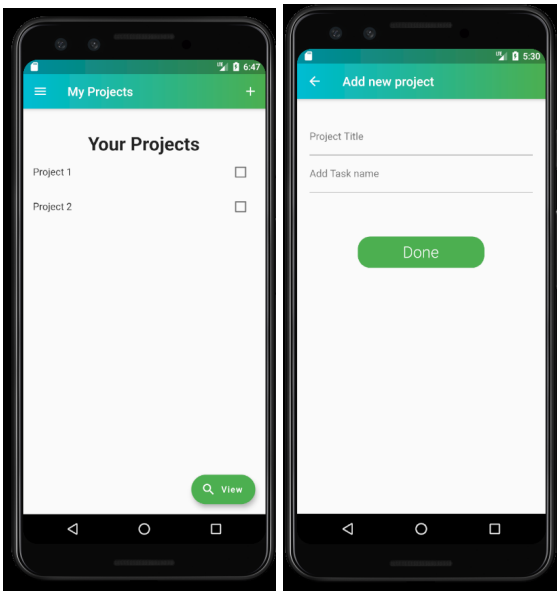
(a) Profile (b) Edit (c) Drawer

Figure 5: Profile Pages



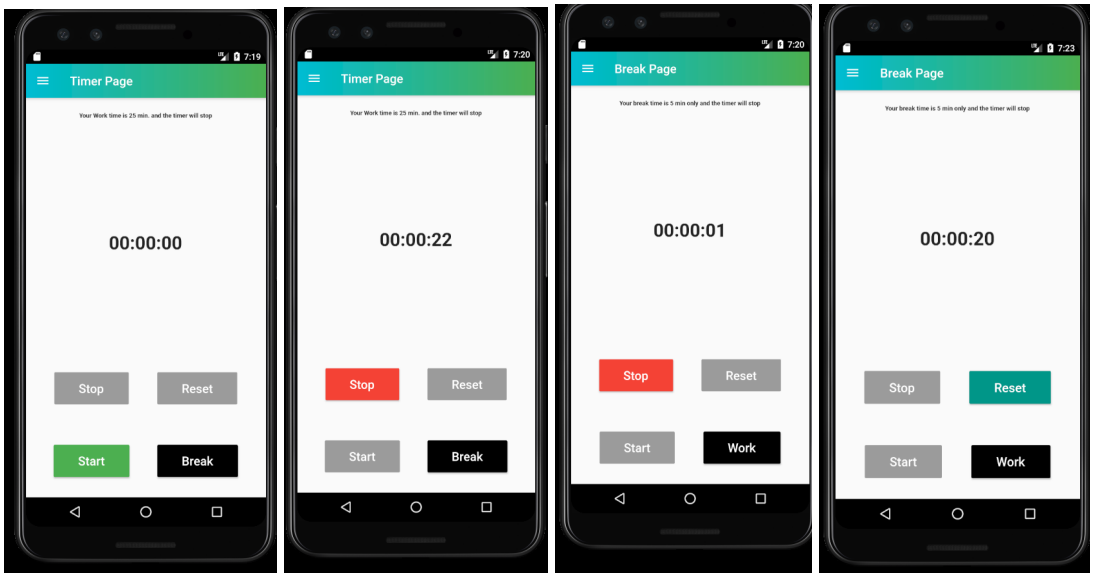
(a) Task (b) Add Task (c) Task Time (d) UI Task

Figure 6: Task Pages



(a) Project (b) Add Project

Figure 7: Project Pages



(a) Work (b) Stop Work (c) Stop Break (d) Reset Time

Figure 8: Task Pages

4.1.1 CLI

No Command Line Interface is used.

4.2 API

1. Google Calendar API.
2. Google OAuth API.
3. Google Maps API.
4. reCaptcha API.
5. Firebase.

5 Design Constraints

5.1 Software Limitation

The mobile application should be accessible by any Android device with operating system (Oreo or higher) and on any IOS device.

5.2 Android SDK Version

Android SDK Oreo API 29.

5.3 External Packages

Table 2: Pomodoro External Packages

#	Package	Version
1	Cupertino Icons	1.0.0
2	Slide Countdown Clock	1.0.3
3	Splashscreen	1.3.3
4	Percent Indicator	2.1.5
5	Flutter Inner Drawer	0.5.7+2
6	Loading	1.0.2
7	int1 (DateTime)	0.15.8

6 Non functional requirements

6.1 Security

User's data could only be changed and accessed by him information and each user must not access the database of any other user and each user shall access the system with his basic information, his username and password.

6.2 Reliability

Our Pomodoro application depends on real-time database, the database update process must roll back all related updates when any update drops, because it should always be up to date.

6.3 Maintainability

Our Pomodoro application is maintained through using a list of design patterns and The system will include records in the database, which includes many relations between the database tables to give the flex-ability to the application.

6.4 Portability

Our Pomodoro application should be working on cross-platforms devices such as Android, IOS but the user must download it first.

6.5 Usability

Our Pomodoro application should be provided with an easy interface so anyone can use it and doesn't find any difficulty in it.

7 Operational Scenarios

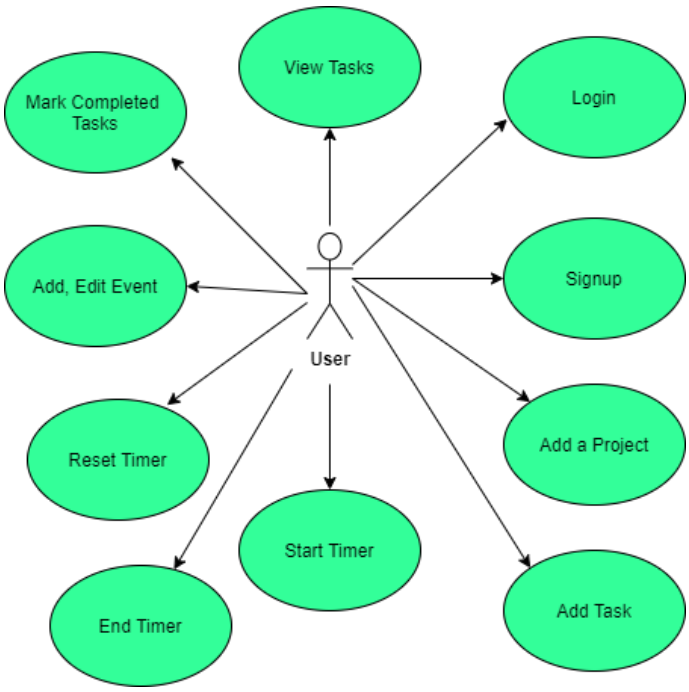


Figure 9: Our Scenario case.

The User shall be able to:

1. Signup and Login.
2. Add projects
3. Add tasks
4. View tasks
5. Mark completed tasks
6. Add/Edit event
7. Start timer
8. Rest timer
9. End timer

8 Application Structure

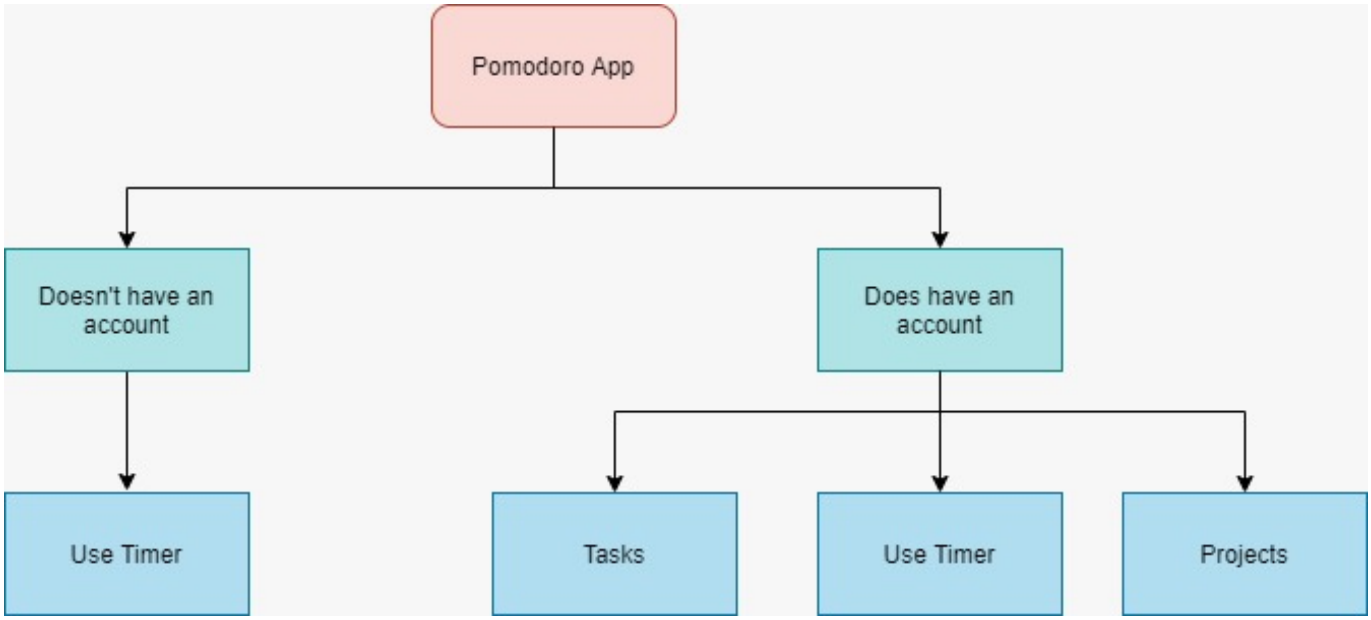


Figure 10: Application Structure Diagram

9 Class Diagram

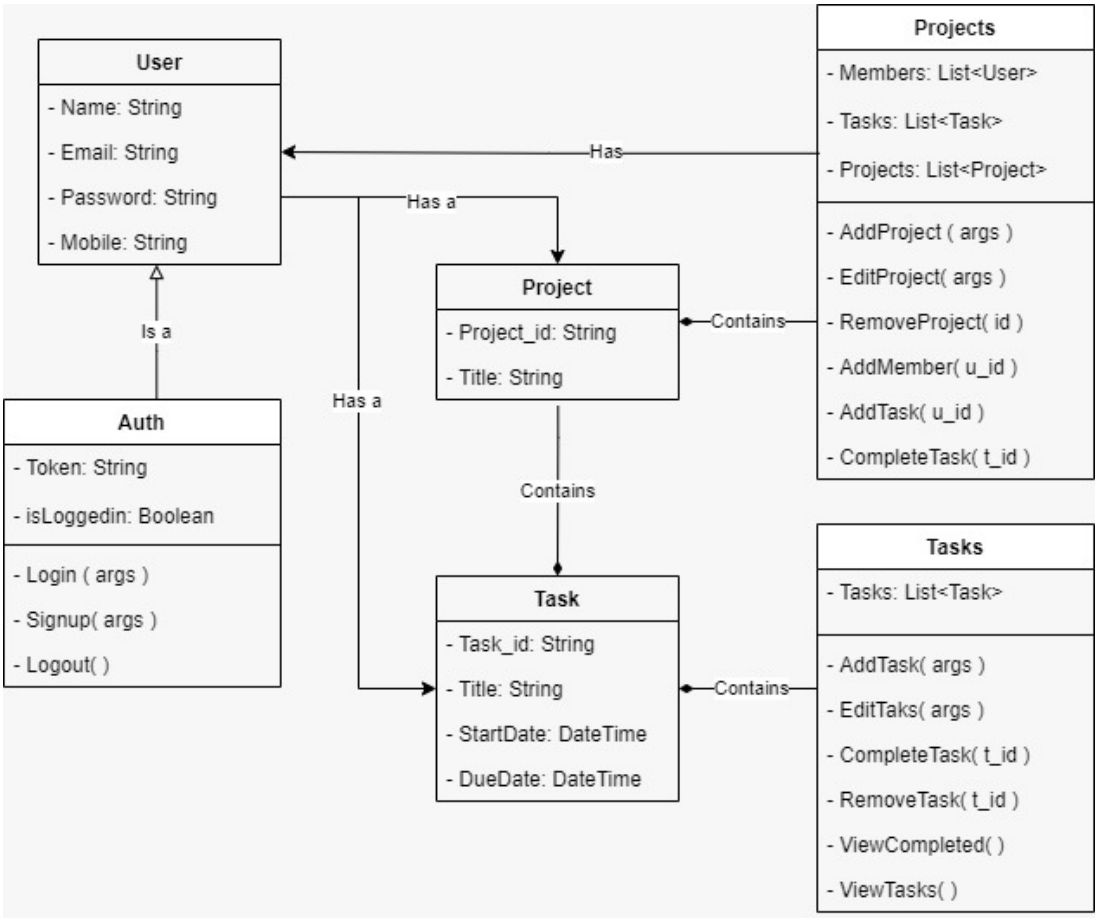


Figure 11: Class Diagram

9.1 Inheritance Relationships

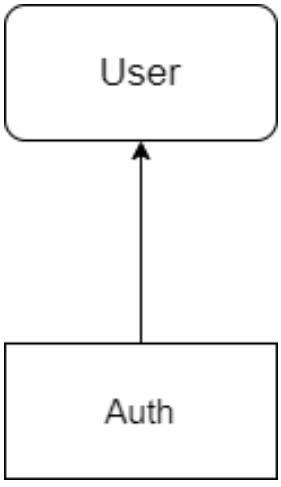


Figure 12: Class Inheritance

Table 1: User

Abstract or Concrete:	Concrete.
List of Superclasses	None.
List of Subclasses	Auth.
Purpose	Store personal data of a user.
Collaborations	Extends with Auth and Aggregate with Projects, project, Task.
Attributes	Name:string, Email:string, Password:string, Mobile: String.
Operations	None.
Constraints	Store all the data of the user.

Table 2: Auth

Abstract or Concrete:	Concrete.
List of Superclasses	User.
List of Subclasses	None.
Purpose	To give the user the exact authorization to use the application.
Collaborations	Extends from user.
Attributes	Token: String, isLoggedin: Boolean.
Operations	Login(args), Signup(args), Logout().
Constraints	Carries the functionallity and checks if the user is logged in.

Table 3: Project

Abstract or Concrete:	Concrete.
List of Superclasses	None.
List of Subclasses	None.
Purpose	Hold the data of the project details.
Collaborations	Aggregate with User and Associate with Task, projects.
Attributes	ProjectId: String, Tilte: String.
Operations	None.
Constraints	Store all the data of a single project.

Table 4: Task

Abstract or Concrete:	Concrete.
List of Superclasses	None.
List of Subclasses	None.
Purpose	Hold the data of the task details.
Collaborations	Aggregate with User and Associate with both tasks and project.
Attributes	TaskId: String, Title: String, StartDate: DateTime, DueDate: DateTime.
Operations	None.
Constraints	Store all the data of a single task.

Table 5: Projects

Abstract or Concrete:	Concrete.
List of Superclasses	None.
List of Subclasses	None.
Purpose	Carries the data of all the projects & it's functionality.
Collaborations	Aggregate wit User and Associate wit project.
Attributes	Members: List <User>, Tasks: List <Task>, Projects: List <Project>.
Operations	AddProject (args), EditProject (args), RemoveProject (id), AddMember (uId), AddTask (uId), CompleteTask (tId).
Constraints	Store all the data & functionality of the user projects.

Table 6: Tasks

Abstract or Concrete:	Concrete.
List of Superclasses	None.
List of Subclasses	None.
Purpose	Carries the data of all the tasks & it's functionality.
Collaborations	Associate with Task.
Attributes	Tasks: List <Task>.
Operations	AddTask (args), EditTasks (args), CompleteTask (tId), RemoveTask (tId), View-Completed (), ViewTasks ().
Constraints	Store all the data & functionality of the user tasks.

10 Database design description

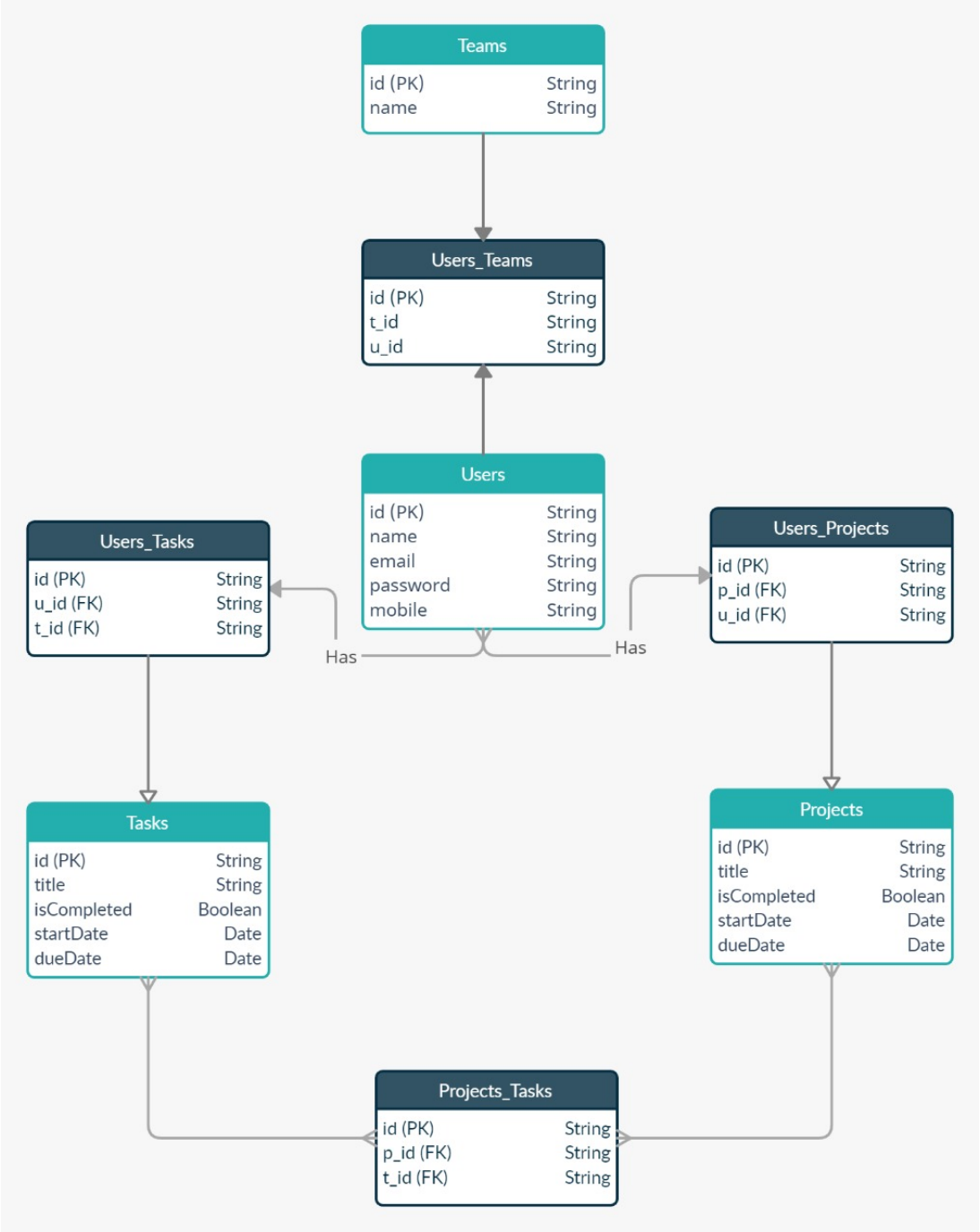
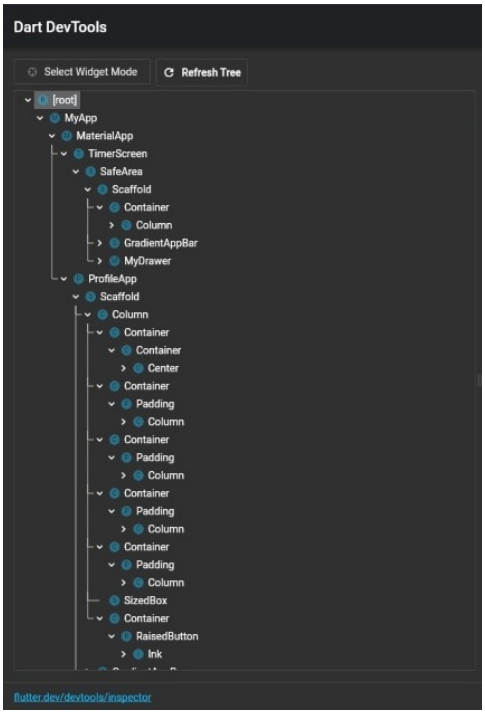
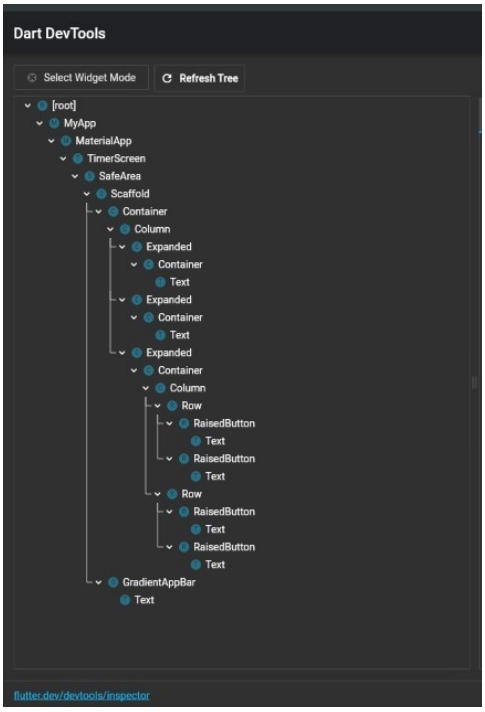


Figure 13: Database Schema design

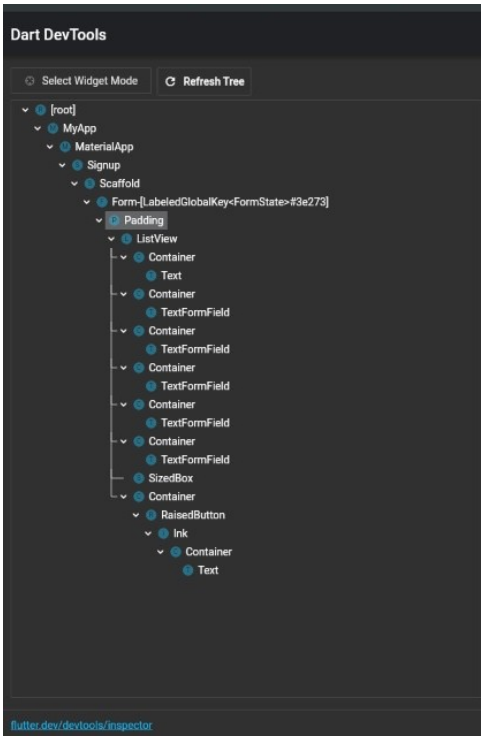
11 Widget Tree



(a) Profile Page Tree



(b) Timer Page Tree



(a) Sign Up Page Tree

12 Github Contribution

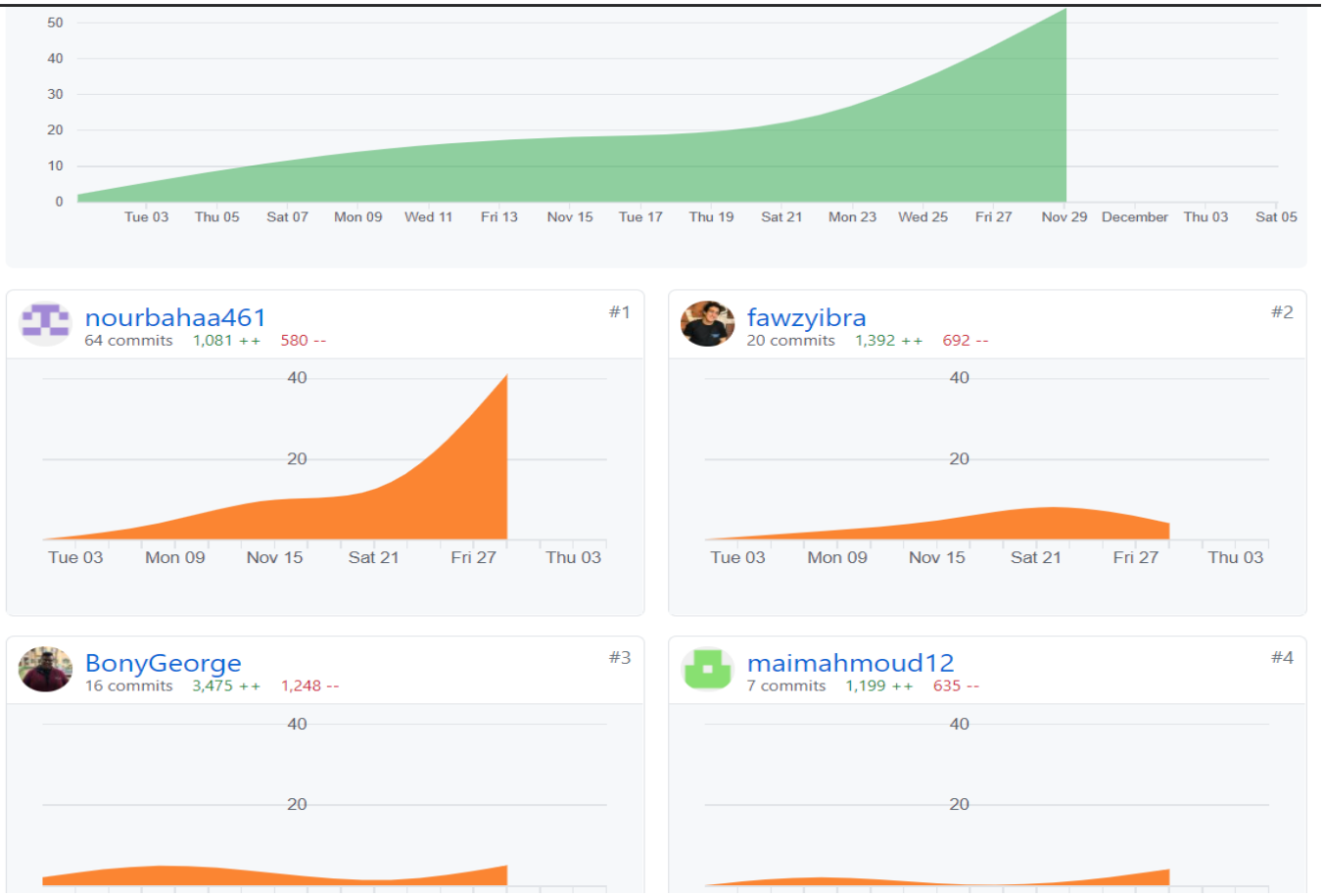


Figure 16: Contributors

13 Time Plan

Table 7: Pomodoro Project time plan

Id	Task	Start Date	Team Member
1	Basic UI Components	18/10/2020	Abanoub
2	Landing Screen	21/10/2020	Abanoub
3	Login Screen	21/10/2020	Nour
4	View/Edit Profile	21/10/2020	Mai
5	Signup Screen	23/10/2020	Nour
6	Timer Screen	23/10/2020	Ibrahim
7	Add/View Task Screens	24/10/2020	Abanoub
8	Reset Password Screen	25/10/2020	Nour
9	Break Screen	26/10/2020	Ibrahim
10	Add/View/Edit Project Screens	29/10/2020	Ibrahim & Mai

References

[1] P. Aminov, N. Bola, D. Shiralkar, and M. Yoganarasimha. Cloud based algorithm for task management. In *2019 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*, pages 249–253, 2019.

[2] Francesco Cirillo. The pomodoro technique (the pomodoro). *Agile Processes in Software Engineering and*, 54(2):35, 2006.

[3] Francesco Cirillo. *The pomodoro technique*. Lulu. com, 2009.

[4] S. F. dos Reis Alves, A. J. Uribe-Quevedo, I. Nunes da Silva, and H. F. Filho. Pomodoro, a mobile robot platform for hand motion exercising. In *5th IEEE RAS/EMBS International Conference on Biomedical Robotics and Biomechatronics*, pages 970–974, 2014.

[5] Federico Gobbo and Matteo Vaccari. The pomodoro technique for sustainable pace in extreme programming teams. In *International Conference on Agile Processes and Extreme Programming in Software Engineering*, pages 180–184. Springer, 2008.

[6] M. Ruensuk. An implementation to reduce internal/external interruptions in agile software development using pomodoro technique. In *2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)*, pages 1–4, 2016.