



BAHIRDAR UNIVERSITY

BAHIRDAR UNIVERSITY INSTITUTE OF TECHNOLOGY

FACULTY OF COMPUTING

Industrial project on: Self-Improvement App.

Submitted to the faculty of computing in partial fulfillment of the requirements for the degree of Bachelor of Science in Software Engineering Department.

Group Members:

Name	ID Number:
1. Bethemariam Asrat Derib	1102834
2. Kidist Amsalu Birhanu	1102029
3. Debasu Tibebe Ambaye	1102837

Advisor: Mr. Samuel Ashagrie.

2015 E.C.

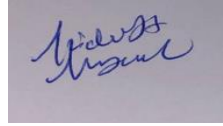
Bahir Dar University, Bahir Dar Institute of Technology

Declaration

The Project is our own and has not been presented for a degree in any other university and all the sources of material used for the project have been duly acknowledged.

Kidist Amsalu

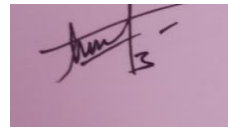
Name



Signature

Bethemariam Asrat

Name



Signature

Debasu Tibebu

Name



Signature

Faculty: Computing

Program: Software Engineering

Project Title: Self-improvement app

This is to certify that I have read this project and that in my supervision and the students' performance, it is fully adequate, in scope and quality, as a project for the degree of Bachelor of Science.

Samuel Ashagrie

Name of Advisor



Signature

Examining committee members**signature****Date**

- | | | | |
|---------------|-------|-------|-------|
| 1. Examiner 1 | | _____ | |
| 2. Examiner 2 | | _____ | |

It is approved that this project has been written in compliance with the formatting rules laid down by the faculty.

Roles and Responsibilities of the Group Members

List of Tasks	List members		
	Kidist Amsalu	Bethemariam Asrat	Debasu Tibebu
Feasibility Study	√	√	√
Requirement Gathering	√	√	√
System use case	√		
User Interface prototyping		√	
Activity Diagram		√	
Sequence Diagram	√	√	
Analysis class model	√		
Logic model	√	√	√
Architectural design	√		√
Component Diagram	√	√	
Deployment Diagram			√
Design Class model			√
User Interface Design		√	

Table 1: Roles and Responsibilities of the Group Members

Acknowledgment

We are overwhelmed in all humbleness and gratefulness to acknowledge our depth to all those who have helped us to do this project.

First, we would like to thank our Almighty God for giving us the strength till we've accomplished this project.

We would like to express our special thanks of gratitude to Mr.Samuel Ashagrie our advisor who helped us in doing this project by mentoring on how to do it.

Any attempt at any level can't be satisfactorily completed without the support and guidance of our parents and friends, so we would like to thank our parents, friends, and the internet.

List of acronyms

OOM: Object-Oriented Modeling

OOD: object-oriented design

OOA: Object-Oriented Analysis Activities

List of Figures

Figure 1: Use case diagram.....	131
Figure 2:user prototype	28
Figure 3: State char diagram for register	29
Figure 4: State char diagram for login	30
Figure 5: State char diagram for note	31
Figure 6: Activity diagram for notes.....	32
Figure 7: Activity diagram for diary	33
Figure 8: Activity diagram for calendar.....	34
Figure 9: Activity diagram for booklist	35
Figure 10: Activity diagram for to-do.....	36
Figure 11: Sequence diagram for login.....	Error! Bookmark not defined.
Figure 12: Sequence diagram for register	Error! Bookmark not defined.
Figure 13: Sequence diagram for note	38
Figure 14: Sequence diagram for pomodoro	39
Figure 15: Analysis class model	40
Figure 16: CRC analysis	45
Figure 17: Conceptual class modeling.....	46
Figure 18:home prototype.....	47
Figure 19:front page.....	48
Figure 20:note page.....	49
Figure 21:calorie prototype.....	50
Figure 22:todo prototype.....	50
Figure 23:pomodoro prototype	51
Figure 24: Three-tier architecture	52
Figure 25: Component modeling	53
Figure 26: Deployment diagram	54
Figure 27: Class model	55
Figure 28: Persistence model	56
Figure 29:home page.....	56
Figure 30:front page.....	57

Figure 31:note page.....	58
Figure 32:calorie page.....	59
Figure 33:todo page	59
Figure 34:pomodoro.....	60
Figure 35:interview questions.....	6064

List of tables

Table 1: Roles and Responsibilities of the Group Members	ii
Table 2: Time Feasibility	6
Table 3: note Use Case Description.....	18
Table 4:calendar Use Case Description	19
Table 5: calorie calculator Use Case Description	20
Table 6:To-do list Use Case Description	21
Table 7: Book list scription.....	23
Table 8: pomodoro Use Case Description	23
Table 9: Wikipedia Use Case Description	24
Table 10:Diary Use Case Description.....	26
Table 11: Logout Use Case Description	26

Contents

Declaration.....	i
Roles and Responsibilities of the Group Members.....	ii
Acknowledgment.....	iii
List of acronyms.....	iv
List of Figures.....	v
Abstract.....	xi
CHAPTER ONE	1
1. Introduction.....	1
1.1. Background	1
1.2. Statement of the problem	2
1.3. Objective	2
1.3.1. General objective	2
1.3.2. Specific objective.....	2
1.4. Methodology	3
1.4.1. Requirement gathering methods.	3
1.4.2. Analysis and design Methodology.....	4
1.4.3. Implementation Methodology.....	5
1.5. Feasibility.....	5
1.5.1. Economic feasibility	5
1.5.2. Technical feasibility.....	5
1.5.3. Time feasibility	5
1.6. Significance and Beneficiaries of the project	6
1.7. Limitation.....	6
1.8. Scope.....	7

1.9.	Organization of the project	7
CHAPTER TWO		9
2.	System features	9
2.1.	Existing System	9
2.2.	Proposed System.....	9
2.3.	Requirement Analysis	10
2.3.1.	Functional requirement	10
2.3.2.	System use case.....	12
2.3.2.1.	Use case diagram.....	12
2.3.2.2.	Use case documentation.	13
2.3.3.	Business Rule Documentation	27
2.3.4.	User interface prototype.....	27
2.3.5.	State chart diagram	28
2.3.6	Activity diagram.....	32
2.3.6.	Sequence diagram	37
2.3.7	Analysis Class Model	39
2.3.7.	Logic model	40
2.4.	Non-functional requirements	42
2.5.	System requirement	43
2.5.1.	Hardware requirements.....	44
2.6.	Key abstraction with CRC analysis	45
2.6.1.	Conceptual modeling: class modeling	46
2.6.2.	Identifying change cases	46
CHAPTER THREE		52
3.	System Design	52

3.1.	Architectural Design	52
3.1.1.	Component modeling.....	52
3.1.2.	Deployment Modeling	54
3.2.	Detail Design	54
3.2.1.	Design class model	54
3.2.2.	Persistent model	55
3.3.	User Interface Design	56
3.4.	Access control and security	60
	Reference	62
	Appendices.....	64

Abstract

“self-improvement app” is the project title we have been working on for the past couple of months. The purpose of using this app is, it can help us become the best version of ourselves. It can help us stay focused, organized, and motivated to reach our goals. These app allow us to keep track of our progress, so we can measure our improvement over time and stay on track. All in all this project has a calendar to schedule our events according to the day, a To-do list to organize our tasks, an app that perform a pomodoro technique for studying purpose, a diary to write our day to day life experience, a calorie calculator that calculates out calorie intake and suggests us what kind of exercise to perform and for how long to do such activities, a book library that contains all the book we want to be listed, a notes app for writing notes that we might need to remember, and a Wikipedia searching page for user to search without living this app. This project contains all this functionality in one place which makes it special because this by itself helps to avoid distraction and getting lost in the world full of distracting things. all things considered; this self-improvement app can help us become a better version of ourselves!

CHAPTER ONE

1. Introduction

1.1. Background

As a human being we often ask ourselves how to improve and why self-improvement is important. And we have been doing this for centuries.

Personal development helps us in just about every aspect of life, allowing us to improve our mental health, boost our self-esteem, and gain confidence. Similarly, if we focus on the physical benefits of self-improvement, we'll feel better than ever. But more practically, our new skills and habits will improve our relationships, help us develop new connections, elevate our career, and give us a real sense of direction. Who doesn't want that?

Of course, it doesn't happen overnight. we need strategies and techniques to get there. Trying to learn a new skill and completing the ones we started, are becoming hard in this world full of distraction. Of course, everyone has their own strategies that suit them best. But for those who really struggle in keeping their focus and obeying their schedule, should use whatever help they could find from the technologies out there, that could encourage and guide them through their personal growth journey.

Self-improvement can bring about many positive advantages for an individual, both in the short and long term. One of the most immediate advantages of taking part in self-improvement activities is the boost in confidence and self-esteem. When you take steps to improve yourself, whether it's through learning a new skill or taking on a new challenge, it can help you feel more capable and prouder of your accomplishments. Additionally, those who engage in self-improvement activities often find that they have more energy, better sleep, and improved mental clarity. Furthermore, with an improved sense of self-esteem and confidence, individuals may find more success in relationships, work, and other areas of life. Ultimately, self-improvement can help you lead a more fulfilling and successful life.

No matter what aspect of self-improvement we want to follow, there's an app designed to help us reach our goals. To give us a helping hand, we decided to build the very best apps around to guide you on your journey.

1.2. Statement of the problem

Personal growth is the ongoing process of understanding and developing yourself to achieve your fullest potential. Personal development is a vital part in a person's growth, maturity, success, and happiness.

However, according to studies, there are several reasons why people nowadays are easily distracted, and technology is one of them.

we are getting distracted with technology and are suffering from mental fatigue. We finish the workday exhausted while feeling we've accomplished nothing of any real value. Our distraction is eroding our relationships, time management abilities, success, and productivity.

As we all know "modern problems require modern solution." So, staying away from technology won't solve these problems instead using technologies to help us in advance is the only way.

1.3. Objective

1.3.1. General objective

- ✓ to develop self-improvement app that will change our life in small step at a time.
- ✓ to create an app that helps people achieve their goals and improve themselves. The app will provide users with helpful tips and tools to help them reach their goals and become their best selves.

1.3.2. Specific objective

- ✓ for building better habits or breaking bad ones
- ✓ for our health and fitness
- ✓ to track and save time.
- ✓ to develop self-discipline on the way to our journey
- ✓ to develop our reading habit

- ✓ to study with concentration by using pomodoro technique, scientific research has shown that people can only focus for around 25 minutes at a time without being distracted, the pomodoro technique has gained popularity in recent years. It enables us to concentrate for 25 minutes and take a 5-minute break and listening white noise or nature sound is another option while using this method.

1.4. Methodology

Software development methodology is a process or series of processes used in software development. For our project we chose to use the Agile methodology for development. Agile methodology is a way to manage a project by breaking it up into several phases. It involves constant collaboration with stakeholders and continuous improvement at every stage. Once the work begins, teams' cycle through a process of planning, executing, and evaluating. Continuous collaboration is vital, both with team members and project stakeholders.

1.4.1. Requirement gathering methods.

Requirement gathering methods are methods that are used for gathering information for our projects. We used various data collection methodologies to gather information. Some of these are listed below.

Primary Data Collection Method

- **Interview-** The project team spoke with many people face-to-face about their time management strategies and the apps they use to advance themselves.
- **Observation-** apart from interviewing, the project team gather information by observing how people use technologies to improve themselves. We used more of active observation than passive observation by asking question while observing.
- **brainstorming session-** we conducted a short discussion between where all participants such as the group members, friends, and families, are allowed to say whatever they feel is important to the topic of discussion. After that, the facilitators of our group organized and prioritized the results.

➤ Secondary Data Collection Method

Document analysis- the project team analyze research that were done before regarding on how people use self-improvement apps and the things they want, to be included in the app.

Language and framework used

- Frontend: HTML, CSS, Bootstrap, JavaScript.
- Backend: Python.
- Framework: Django.
- Database: SQLite3

1.4.2. Analysis and design Methodology

In analysis and design, we applied object-oriented analysis methodology to design and analyze our system.

We chose the object-oriented modeling (OOM) approach. OOM is a common approach to modeling applications and systems by using the object-oriented paradigm throughout the entire development life cycles. It is the main technique heavily used by both object-oriented design (OOD) and object-oriented analysis (OOA) activities in modern software engineering.

We chose OOM because of the reasons below:

- Better quality of code. Because of the stable abstraction the model provides the developer understands the system better from the analysis and design and code it right.
- It is a common approach to modeling applications and systems by using the object-oriented paradigm throughout the entire development life cycles.
- To organize requirements around objects, which integrate both behaviors (processes) and states (data) modeled after real world objects that the system interacts with.

In object-oriented design, a developer applies implementation constraints to the conceptual model produced in object-oriented analysis.

The tools to be used for design and analyze:

- ✓ Argo UML and Draw.io – for modeling our system.

- ✓ Figma-to design UX/UI
- ✓ Microsoft Word for documenting and writing the documentations.

1.4.3. Implementation Methodology

We use Django framework and python language for our implementation. And we will be using flutter for developing the mobile application because it is a platform independent framework. As a code editor we planned on using visual studio code.

1.5. Feasibility

A feasibility study analyzes the viability of a project to determine whether the project is likely to succeed.

1.5.1. Economic feasibility

It was difficult to find numerous productivity applications in one location, and to use various self-improvement apps; users had to download numerous apps, and for some applications, they might have to pay for the app; however, this new software is simple to download and doesn't cost anything to use.

1.5.2. Technical feasibility

The proposed system is technically possible to implement. The resources (hardware and software) that are used to develop the system are available and overall, the system is doable. The supplies that are needed include computers, phones, Wi-Fi service.

1.5.3. Time feasibility

We estimate the product to be complete in 2 months.

Task Name	duration
Project proposal	2 days
Requirement gathering	10 days

System analysis and design modeling	15 days
Implementation	1 month
Testing documentation	8 days

Table 2: Time Feasibility

1.6. Significance and Beneficiaries of the project

This self-improvement web-based app can help you stay consistent in achieving your goals and develop good habits while overcoming the bad ones. This also try to keep your morale high and increase your motivation.

The major benefit to be obtained from the deployment of this application is achieving personal growth, by providing all the necessary functionalities in one compartmentalized workspace.

It offers every productivity app in one location, making it different from other apps that are available right now.

Project beneficiaries are mostly students, who spend most of their time studying and trying to reduce their school load. Others are people with busy schedule, who needs to keep their tasks and obligations in order. To makes things more manageable and keep them mentally focused on the tasks at hand.

People who are concerned with their health and constantly investigate their diet are this project another beneficiary because it suggests exercise according to the calories they consume.

1.7. Limitation

- ✓ The system needs internet connections for apps calorie calculator because we integrate some APIs.

1.8. Scope

This project focuses on developing a web-based app that helps users to build a habit, anyone who is interested in self-development can use this app.

The scope of this web-based application covers the following,

- The system should allow users to SIGN UP
- The system should allow users to LOGIN.
- The system should let the users reset their password.
- The system should allow the user to create, update, and delete notes.
- The system should allow the user to create, update, and delete to-do.
- The system should allow the user to set pomodoro.
- The system should allow the user to search on Wikipedia.
- The system should allow the user to create, update, and delete diary.
- The system should allow the user to calculate calorie.
- The system should allow the user to create, update, and delete calendar.
- The system should allow the user to create and delete booklist.

1.9. Organization of the project

Our team structured this document using final year documentation guideline format that was provided by Bahir Dar University Computing Faculty. So, for this documentation we perform chapter one, two and three and the last two chapters will be done after we presented this documentation. which is implementation and testing of the project.

In first chapter we try to discuss the background of the project which describe how the old ways of self-improvement app works, statement of the problem that describe problems that exist in existing system, objective of our system that describe why our automated system is needed and how we can improve the current system, significance and target beneficiary of our system that describe the importance of our system to the community. The objectives section defined the

general and specific objectives of our system for individual user. The methodology we used in project requirement gathering like conduct a brainstorming session, interview and observation and document analysis and design methodology like object-oriented methodology and project implementation methodology like Django and others. Finally, feasibility of our system such as economic, time, and technical feasibility.

In second chapter we discuss mainly on requirement gathering, analysis and design which is, about introduction of existing system which describe, different applications that are used for self-managing purpose that would be beneficial to all users. We introduce or propose a system which can solve the existing problem. We discussed the features of the proposed system. The business rule of proposed system which describe the rule that followed by our system.

Functionality of proposed system which describes what our system performs while the nonfunctional requirement describes quality of our new system. The analysis and design part of the document contains use case diagram which shows relationships that exists between actors of our system and the functionalities of the system, the class diagram which include attributes, methods and relationship between classes, the sequence diagram, the activity diagram, state chart, CRC cards and a user interface prototype.

In third chapter we discuss the architectural design of our system which describe the overall architecture of the system, from development to deployment. In this section we discuss on the architecture of the system, component modeling, deployment diagram and the user interface design of our system.

CHAPTER TWO

2. System features

2.1. Existing System

There are a lot of self-improvement apps that have been created and made available for the public. These apps are loved and continuously being used all over the world by people who are interested in making themselves a better human being. The only thing is these apps don't contain multiple functionalities in one environment, they are usually built for providing one or two self-improving tasks per app.

Most of the apps that currently exist need payment after the user completed the free trial, which is not considering all class of society and people don't continue using this apps after this time is up.

Generally, the reasons why we want to change the existing system are:

- Some of the apps are expensive.
- Couldn't find all the apps in one place.

Because of these reasons the existing system is not suitable for many people.

2.2. Proposed System

The proposed system aims to improve human live by providing productivity apps all together in one environment.

It focuses on reducing distraction which is the main problem nowadays.

We live in a world where technology has a significant impact on our daily lives, so the proposed system use technology to have a positive impact on our lives.

It contains various productivity tools including notetaking, journaling(diary), book list, calendar, calorie counting and suggest workouts for us to perform.

The system notifies us unfinished tasks in our to-do list.

Additionally, it includes the pomodoro technique, a time-management strategy that improves reading comprehension.

Pomodoro technique, scientific research has shown that people can only focus for around 25 minutes at a time without being distracted, this is why the pomodoro technique has gained its popularity in recent years. It enables us to concentrate for 25 minutes and take a 5-minute break. It helps us resist all those self-interruptions and re-train our brains to focus. Each pomodoro is dedicated to one task and each break is a chance to reset and bring your attention back to what you should be working on. listening white noise or nature sound is another scientifically proven method of high concentration, this is why we merge the two ways in one to boil down our thought and processes.

2.3. Requirement Analysis

In requirement analysis there are two main ideas defined or explained i.e., functional requirement and non-functional requirements.

2.3.1. Functional requirement

The functional requirements highlight the specific functions the system should be able to carry out. It explains and describes what things are performed by the system. Describe user tasks that the system needs to support. Generally, it's the interaction between the system and the users or functionality we are going to get from this system.

Functional requirements also capture the intended behavior of the system. This behavior may be expressed as services. The following are some functionalities of proposed system categorized by actors of the system:

Pertaining to the self-improvement app, it should have:

User

FRQ-1: The system should allow users to SIGN UP.

- ✓ The system should let the user put in the information needed to register in the provided input fields and press the signup button.

- ✓ Priority: High.

FRQ-2: The system should allow users to LOGIN.

- ✓ It shall enable Admin to login by username and password.

- ✓ It shall enable user to login by username and password.

- ✓ Priority: High.

FRQ-3: The system should let the users reset their password.

- ✓ If the user somehow forgot their password, they should be able to reset their passwords by pressing the forgot password button and by providing their email addresses and by answering their security questions.

- ✓ Priority: High.

FRQ-4: The system should allow the user to create, update, and delete notes.

- ✓ Priority: medium.

FRQ-5: The system should allow the user to create, update, and delete to-do.

- ✓ Priority: medium.

FRQ -6: The system should allow the user to set pomodoro.

- ✓ Priority: medium.

FRQ -7: The system should allow the user to search Wikipedia.

- ✓ Priority: medium.

FRQ -8: The system should allow the user to create, update, and delete diary.

- ✓ Priority: medium.

FRQ -9: The system should allow the user to calculate calorie.

- ✓ Priority: medium.

FRQ -10: The system should allow the user to create, update, and delete calendar.

- ✓ Priority: medium.

FRQ -11: The system should allow the user to create and delete booklist.

- ✓ Priority: medium.

2.3.2. System use case.

A use case diagram is a summary of the whole system like who uses the system and what they can do with it. It describes the relationship among the requirements, users, and the major components of the system.

2.3.2.1. Use case diagram.

use case for self-improvement web app

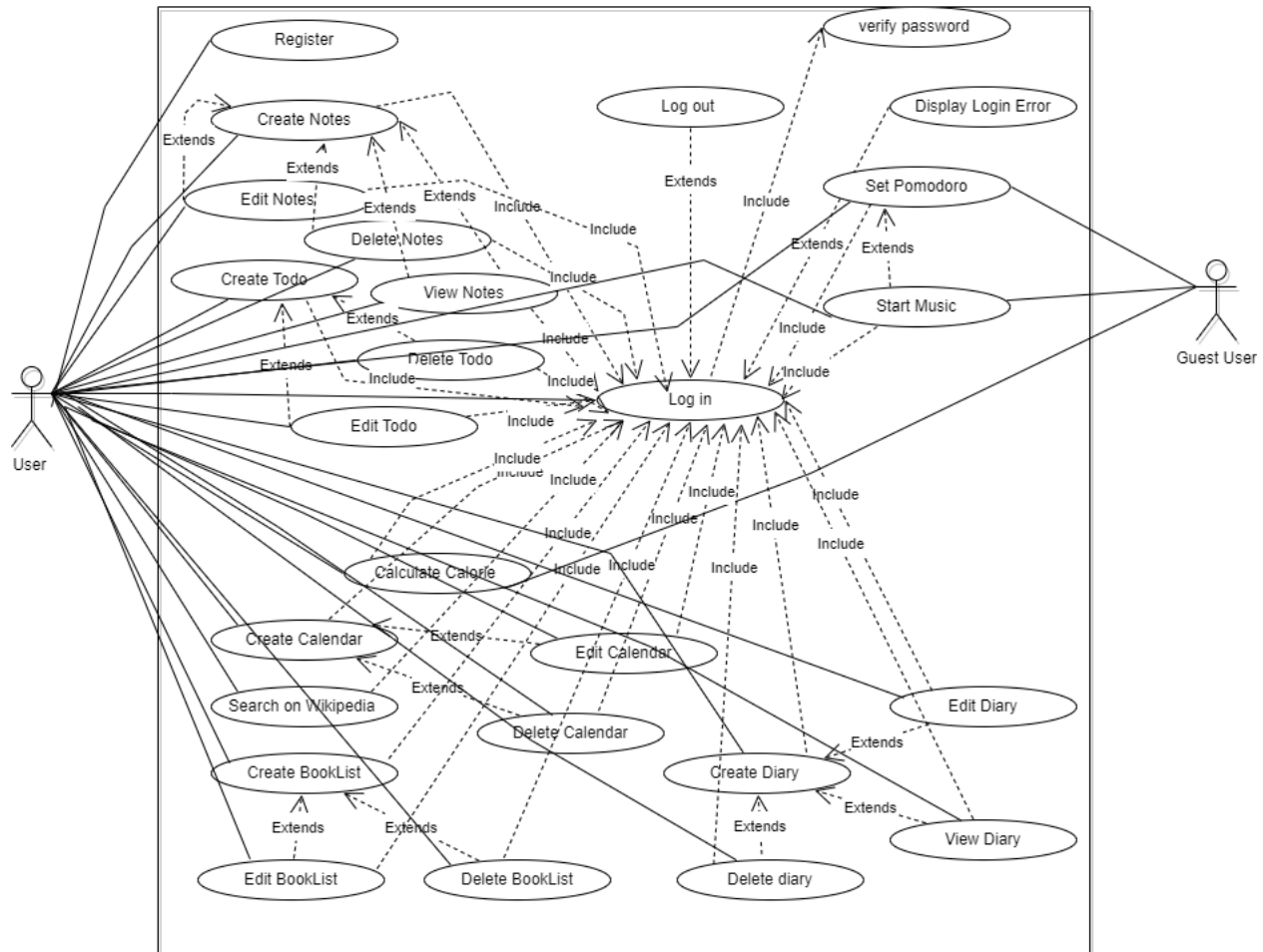


Figure 1: Use case diagram

2.3.2.2. Use case documentation.

Register use case description.

Use case ID	UC- 01
Use case name	Register
Actor	user
Description	This allows users to get registered.
Pre-condition	The user should have valid username and password

Post-condition	If the pre-condition is met, the user get registered successfully.	
Priority	High	
Basic course of action	<u>Actor Action</u>	<u>System Response</u>
	<u>Step-1</u> The user clicks on “Register” button on dashboard or “create an account” link in the login page. If they don’t already have an account. <u>Step-3</u> The user fills out username and password and clicks “sign up” button.	<u>Step-2</u> The system displays the form to register users. <u>Step-4</u> The account is created.
Alternate course of action	On cancel	If the user chooses to cancel, then the user will not be registered.
	Invalid information	If the actor entered invalid information the system will generate an error message to fill the information properly again.
	Blank information	If the actor left blank the text boxes that are mandatory to be filled by them then the

		system will generate error message.
Business rule	Valid textboxes should not be null and invalid.	

Table 1. user registration Usecase description

Login Use case description

Use case number	UC-02	
Use case name	Login	
Actors	user	
Description	Allows the actors to login.	
Pre-condition	Each actor must be registered with valid username and password.	
Post-condition	If the pre-condition is met the actors can login, otherwise the login page will keep showing error username and password.	
priority	High	
Basic course of action	<u>Actor action</u>	<u>System response</u>
	<u>Step-1</u> The actors click “Login” link. <u>Step-3</u> The actors provide username and password. <u>Step-5</u> If the actors selected “sign in”, the login information sub flow is executed.	<u>Step-2</u> The system displays the login form. <u>Step-4</u> Once the actors provide the requested information, the system first validates the information provided. <u>Step-6</u>

		The system logs the actor in to the account.
Alternate course of action	On cancel	If the actors choose to cancel before logged in, then the user can't access their account.
	Invalid information	If the actors entered invalid information the system will generate an error message to fill the information properly again.
	Blank information	If the actors left blank the text boxes that are mandatory to be filled by him then the system will generate error message.
Business rule	Valid username and password textboxes should not be null and invalid.	

Table 2: login use case description.

Notes Use case description.

Use case ID	UC-03
Use case name	Notes
Actor	User
Include	Login
Description	Allow users to create, view, edit and delete their notes.
Pre-condition	The user must be logged in.

Post-condition	If the pre-condition is met the user can create, edit, and delete notes.	
Priority	Medium	
Basic course of action	<u>Actor Action</u>	<u>System Response</u>
	<u>Step-1</u> The actor clicks the “Notes” app. <u>Step-3</u> The actor enters title and write description for the note the want to create. <u>Step-4</u> the actor clicks the “create” button. <u>Step-6</u> the actor clicks the title of the note.	<u>Step-2</u> The system displays the form for the user to write their own notes. <u>Step-5</u> The note is created and displayed. <u>Step-7</u> The system shows the description to the note.
Alternate course of action	On cancel	If the actor chooses to cancel before creating note there won’t be notes displayed.
	Blank information	the system would generate error message If the actor left the title and description space blank.

Business rule	Title and description should not be null.
---------------	---

Table 3: note Use Case Description

Calendar Use Case Description

Use case ID	UC-04	
Use case name	calendar	
Actor	Users	
Description	Allow the actor to schedule their events.	
Pre-condition	The user must be logged in..	
Post-condition	If the pre-condition is met, the user can write down all the events they needed to attend on the calendar.	
Priority	Medium	
Basic course of action	<u>Actor Action</u>	<u>System Response</u>
	<u>Step1</u> The actor clicks on “Calendar” app.	<u>Step2</u> The system displays calendar page.
	<u>Step3</u> If the actor clicks on “new event” link.	<u>Step4</u> The system displays a form with title, description, and date to be filled.
Alternate course of action	<u>Step5</u> The actor fills the information needed and click on “Submit” button.	<u>Step6</u> Once the actor provides the requested information and click “Submit” the system creates an event and register it on the calendar.
	On cancel	If the actor chooses to cancel, then event will not be created.

	Blank information	If the actor left blank the text boxes and the time, that are mandatory to be filled, the system will ask the user to fill them. .
Business rule	Valid textboxes should not be null.	

Table 4:calendar Use Case Description

calorie calculator Use Case Description

Use case ID	UC-05	
Use case name	Calorie calculator	
Actor	user	
Description	Allows users to calculate their calorie intake in their day-to-day diet.	
Pre-condition	The user must be logged in.	
Post-condition	If the pre-condition is met the user can calculate their calorie intake in their diet.	
Priority	Medium	
Basic course of action	<u>Actor Action</u>	<u>System Response</u>
	<u>Step1</u> The actor clicks on “calorie calculator” app. <u>Step3</u> After entering the food, the user clicks on “find calorie” button.	<u>Step2</u> The system displays the page for users to enter the food they consumed. <u>Step4</u> The system displays the calculated calorie intake using graphs and tables and shows what exercises and for

		how long they need to do these exercises.
Alternate course of action	On cancel	If the actor chooses to cancel, the user doesn't find out their calorie intake.
	Blank information	If the user clicks on "find calories" without entering the food name, the system asks to fill the blank text field.
Business rule	Valid textboxes should not be null and invalid.	

Table 5: calorie calculator Use Case Description

To-do list Use Case Description

Use case ID	UC-06	
Use case name	To-do list	
Actor	User	
Include	Login	
Description	Allow the users to create, edit and delete their to-do list.	
Pre-condition	The user must be logged in.	
Post-condition	If the pre-condition is met the user can create, edit, and delete their tasks using the to-do list.	
Priority	Medium	
Basic course of action	<u>Actor Action</u>	<u>System Response</u>
	<u>Step-1</u> The actor clicks the "to do" app on homepage or on the navigation bar.	<u>Step-2</u> The system displays the form to create a to-do list.

	<p><u>Step-3</u> The actor enters the title of the to-do list and press “create” button.</p> <p><u>Step-5</u> If the user finished the task, they change the status to finished.</p> <p><u>Step-6</u> If the user doesn’t no longer want the list, they can click the trash icon.</p>	<p><u>Step-4</u> The system creates the task and display “to-do list added from the user” message.</p> <p><u>Step-7</u> The system will remove the list.</p>
Alternate course of action	On cancel	If the actor chooses to cancel, tasks will not be created.
	Blank information	If the user didn’t fill the title the system will asks the user to fill the blank text field.
Business rule	textboxes should not be null.	

Table 6: To-do list Use Case Description

Book list Use Case Description:

Use case ID	UC-07
Use case name	Book list
Actor	User
Include	Login
Description	Allow users to create their own book library
Pre-condition	The user must be logged in.
Post-condition	If the pre-condition is met the user can create a library of the books they want.

Priority	Medium	
Basic course of action	<u>Actor Action</u>	<u>System Response</u>
	<u>Step-1</u> The actor clicks the “books.” <u>Step-3</u> <u>The user enters the title and attach the book.</u>	<u>Step-2</u> The system displays the form to create book list. <u>Step-4</u> <u>The system creates the book list and display book list added from the user” message.</u>
Alternate course of action	On cancel	If the actor chooses to cancel, then the
	Invalid information	If the actor entered invalid information the system will generate an error message to fill the information properly again.
	Blank information	If the actor left blank the text boxes that are mandatory to be filled by actor, then the system will generate error message.
Business rule	Valid information must be filled, textboxes should not be null and invalid.	

Table 7: Book list Description

Pomodoro Use Case Description:

Use case ID	UC-08	
Use case name	pomodoro	
Actor	User	
Description	Allow users to use Pomodoro techniques for studying.	
Pre-condition	The user must be logged in.	
Post-condition	If the pre-condition is met, the user can start the timer and the sound and keep on studying.	
Priority	Medium	
Basic course of action	<u>Actor Action</u>	<u>System Response</u>
	<u>Step1</u> The actor clicks on “pomodoro” button in homepage or the navigation bar.	<u>Step2</u> The system displays the timer.
	<u>Step3</u> The actor clicks on “start” button.	<u>Step4</u> The system starts the timer.
	<u>Step5</u> The Actor clicks the “paly” button.	<u>Step6</u> The system starts playing music’s that helps to focus on the background.
Alternate course of action	On cancel	If the actor chooses to cancel, they can’t be able to use this pomodoro technique.

Table 8: pomodoro Use Case Description

Wikipedia use case description:

Use case ID	UC-09	
Use case name	Wikipedia	
Actor	User	
Description	Allows users to search anything they needed to know.	
Pre-condition	The user must be logged in.	
Post-condition	If the pre-condition is met, the user can search on Wikipedia and get information.	
Priority	Medium	
Basic course of action	<u>Actor Action</u>	<u>System Response</u>
	<u>Step1</u> The actor clicks on “Wikipedia” button on homepage or the navigation bar. <u>Step3</u> The user writes on the search bar and clicks “search”.	<u>Step2</u> The system displays the search bar. <u>Step4</u> The system displays the result below the search bar.
Alternate course of action	On cancel	If the actor chooses to cancel, then this Use case can’t be executed.
	Blank information	If the actor left the search bar blank the system asks the user to fill it.
Business rule	The search textbox should not be null.	

Table 9: Wikipedia Use Case Description

Diary Use Case Description:

Use case ID	UC-10	
Use case name	Diary	
Actor	User	
Include	Login	
Description	Allows users to create, edit and delete the diary they created.	
Pre-condition	The user must be logged in.	
Post-condition	If the pre-condition is met the user can create, edit, and delete the diary they created.	
Priority	Medium	
Basic course of action	<u>Actor Action</u>	<u>System Response</u>
	<u>Step-1</u> The actor clicks on “Diary” on the homepage or the navigation bar. <u>Step-3</u> The user enters title and write description for the diary they want to create. <u>Step-4</u> the actor clicks the “create” button. <u>Step-6</u> If the user clicks the title of the diary	<u>Step-2</u> The system displays the form to write the diary. <u>Step-5</u> The system displays “diary created successfully.” Message. <u>Step-7</u> The system displays the detail information of the diary.

Alternate course of action	On cancel	If the actor chooses to cancel, then the diary is not created.
	Blank information	If the actor left blank the text boxes that are mandatory to be filled by them then the system will generate error message.
Business rule	Valid textboxes should not be null and invalid.	

Table 10:Diary Use Case Description

Logout Use Case Description:

Use case ID	UC-11	
Use case name	Logout	
Actor	User	
Description	Allow the actor to logout.	
Pre-condition	To logout first the user must be logged in.	
Post-condition	If the pre-condition is met, can logout from the application.	
Priority	High	
Basic course of action	<u>Actor Action</u>	<u>System Response</u>
	<u>Step1</u> The actor clicks on “logout” link in the navigation bar.	<u>Step2</u> The system logs the user out of the application.
Alternate course of action	On cancel	The user will not be logged out.

Table 11: Logout Use Case Description

2.3.3. Business Rule Documentation

Business rules are directives that defines an organization's business activities. They are important because they clarify an organization's objectives and detail how processes will be performed.

BR- 1: User Registration / Sign up.

Anyone can register to the service if they have valid username, password, and email.

Username format should have at least six characters and there should not be no space between characters.

Password format should include at least six characters, No space, No non-standard keyboard keystroke, at least one digit, at least one letter.

BR-2: User Login / Sign in

Username should match the username from time of registration or database.

Password should match the password from time of registration or database.

User will be noticed if username or password did not match and is prompted to retry.

BR-3: One account is only accountable for one user. Each user must have an account.

2.3.4. User interface prototype

Prototypes are an essential part of designing user flows and interfaces. They allow designers to show their design through an interactive and engaging product, resulting in a better understanding of the design for everyone involved.

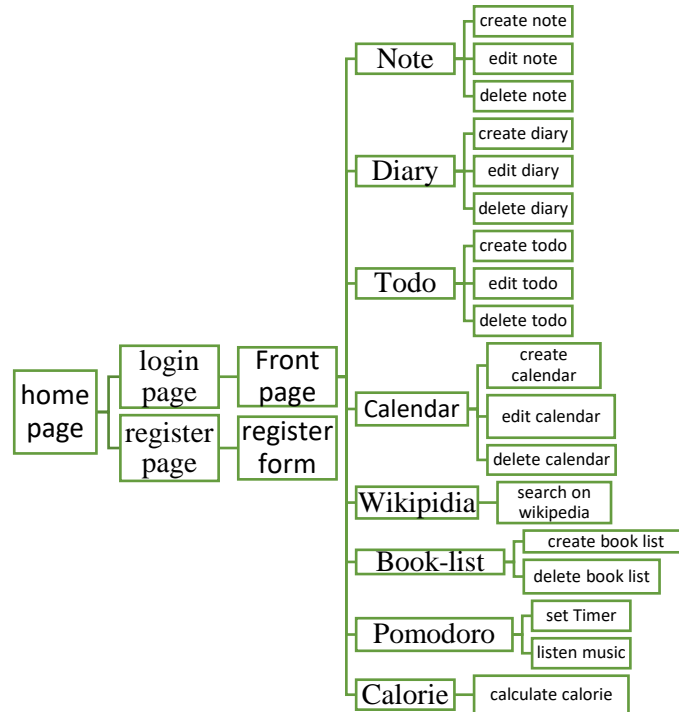


Figure 2:user prototype

2.3.5. State chart diagram

State chart diagram is one of the five UML diagrams used to model the dynamic nature of a system. They define different states of an object during its lifetime and these states are changed by events. State chart diagrams are useful to model the reactive systems.

For Register

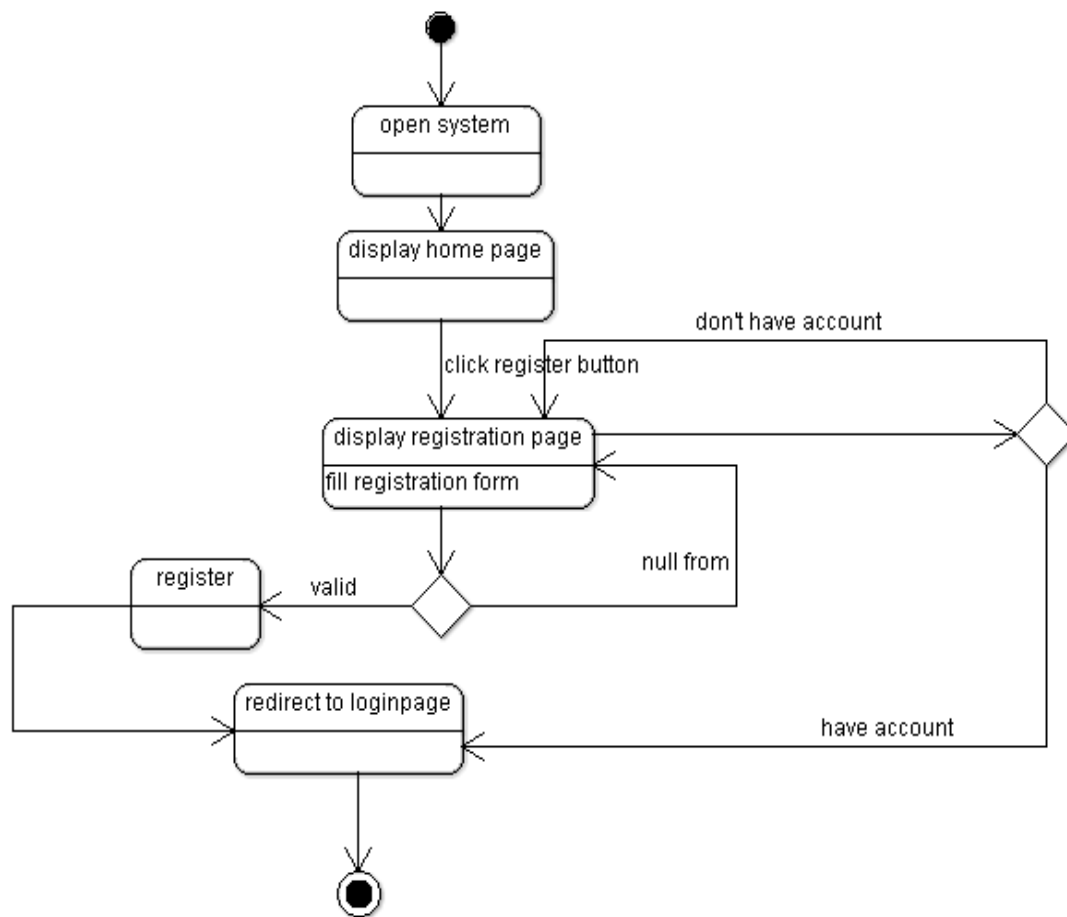


Figure 3: State chart diagram for register

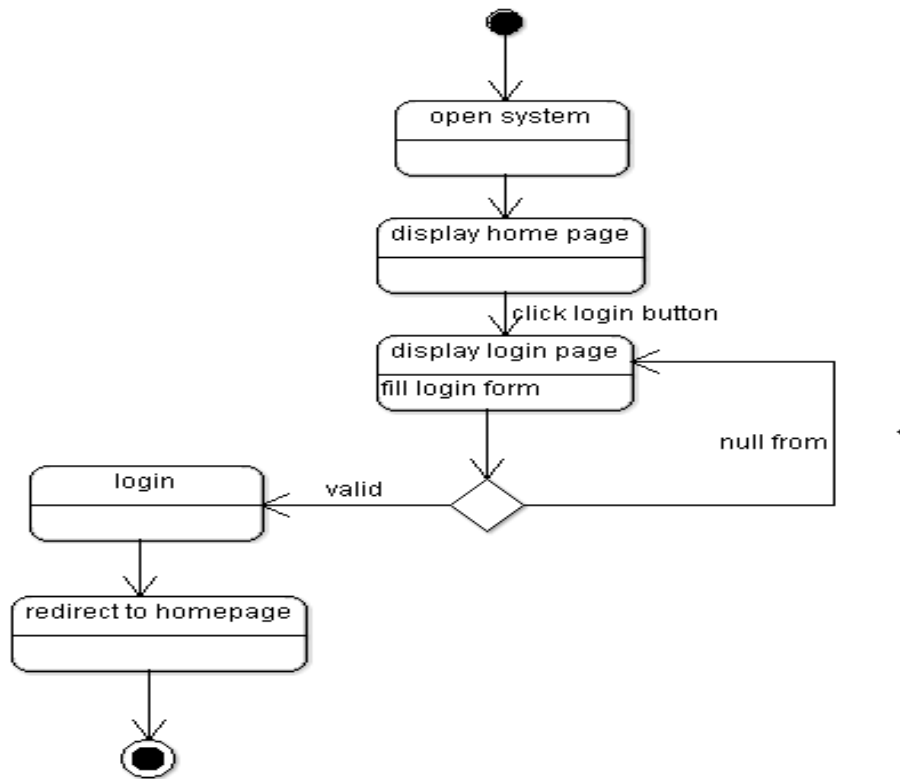
For Login

Figure 4: State chart diagram for login

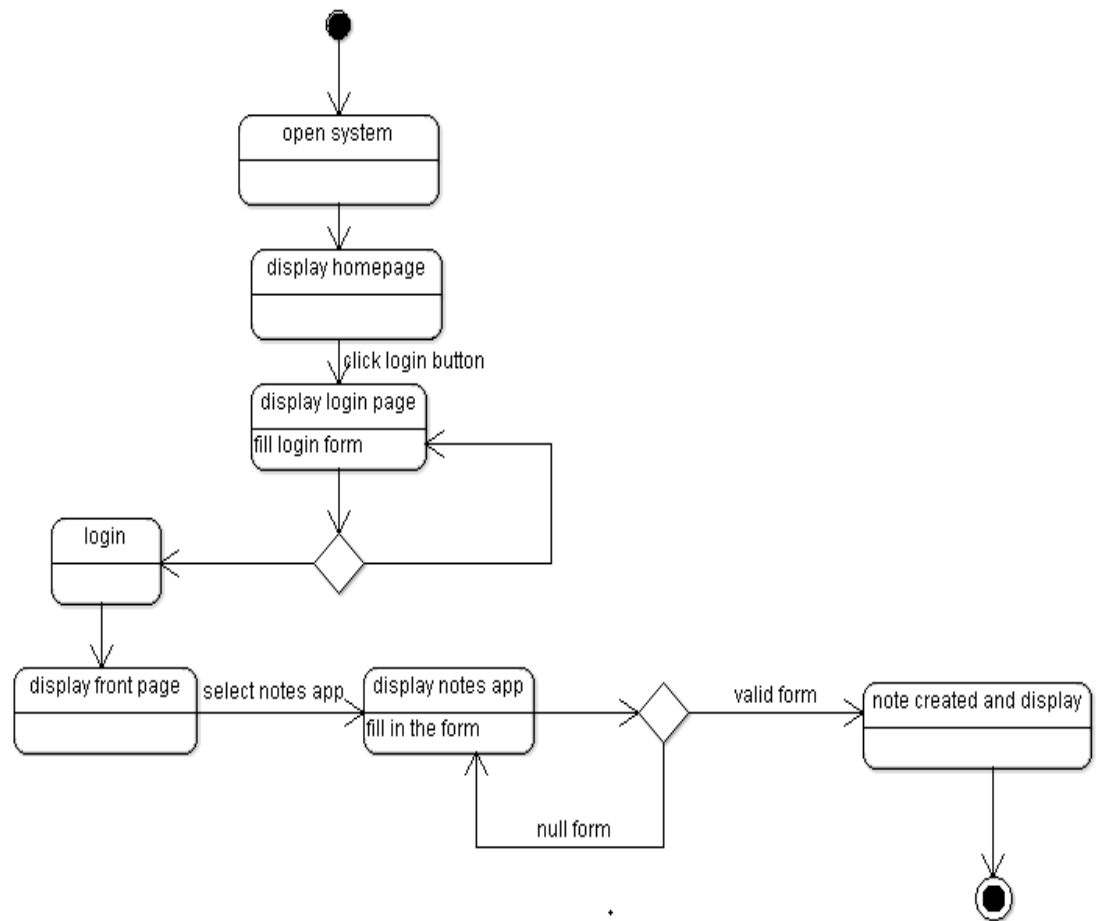
For Note

Figure 5: State chart diagram for note

2.3.6 Activity diagram

For Notes

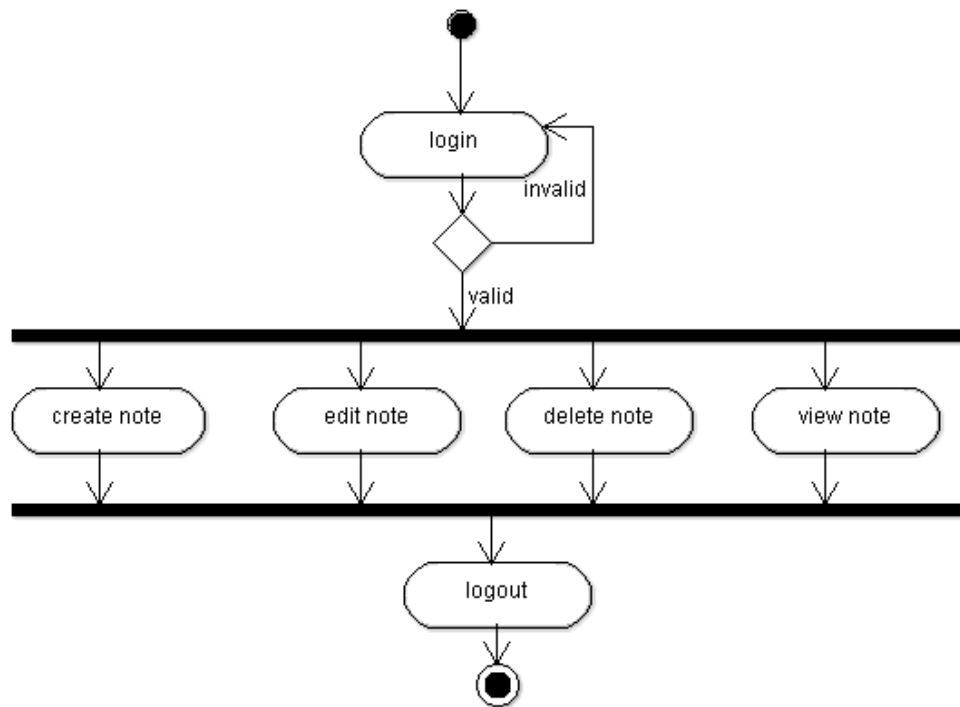


Figure 6: Activity diagram for notes

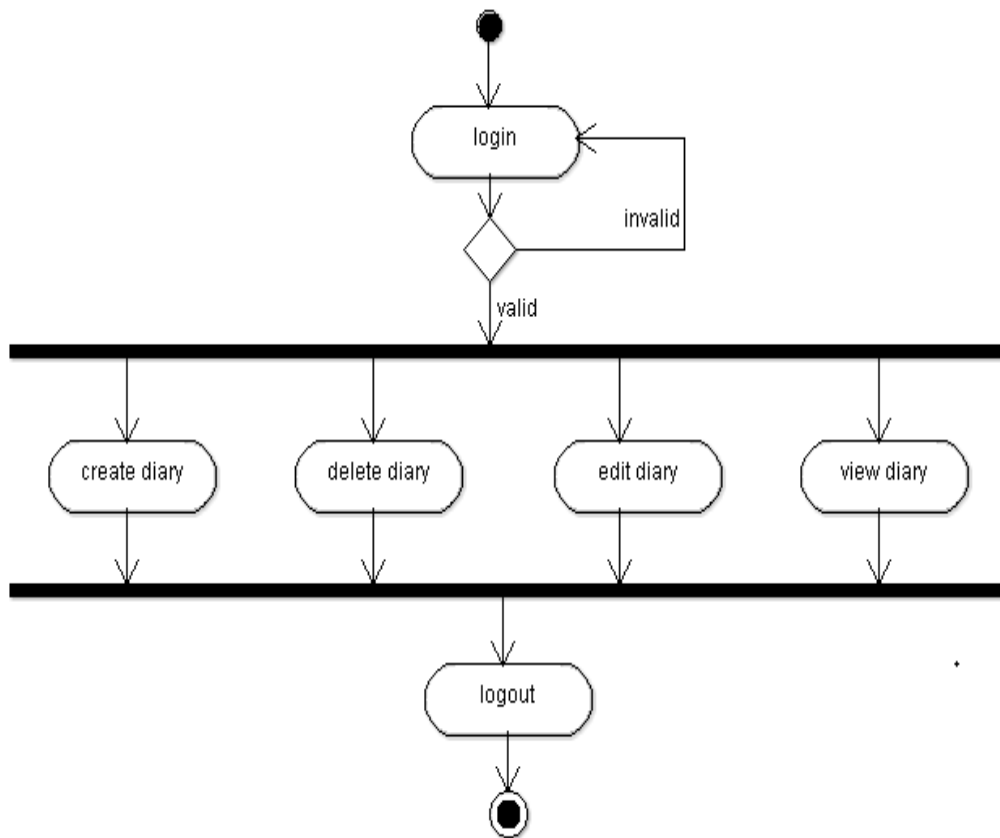
For Diary

Figure 7: Activity diagram for diary

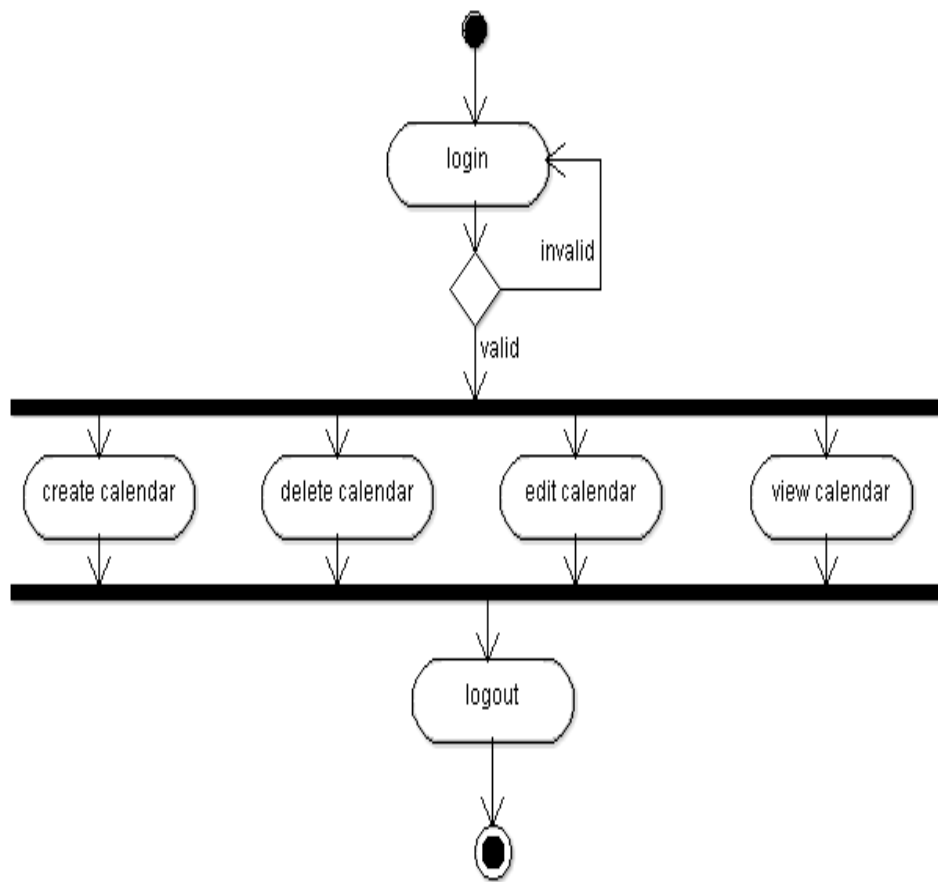
For calendar

Figure 8: Activity diagram for calendar

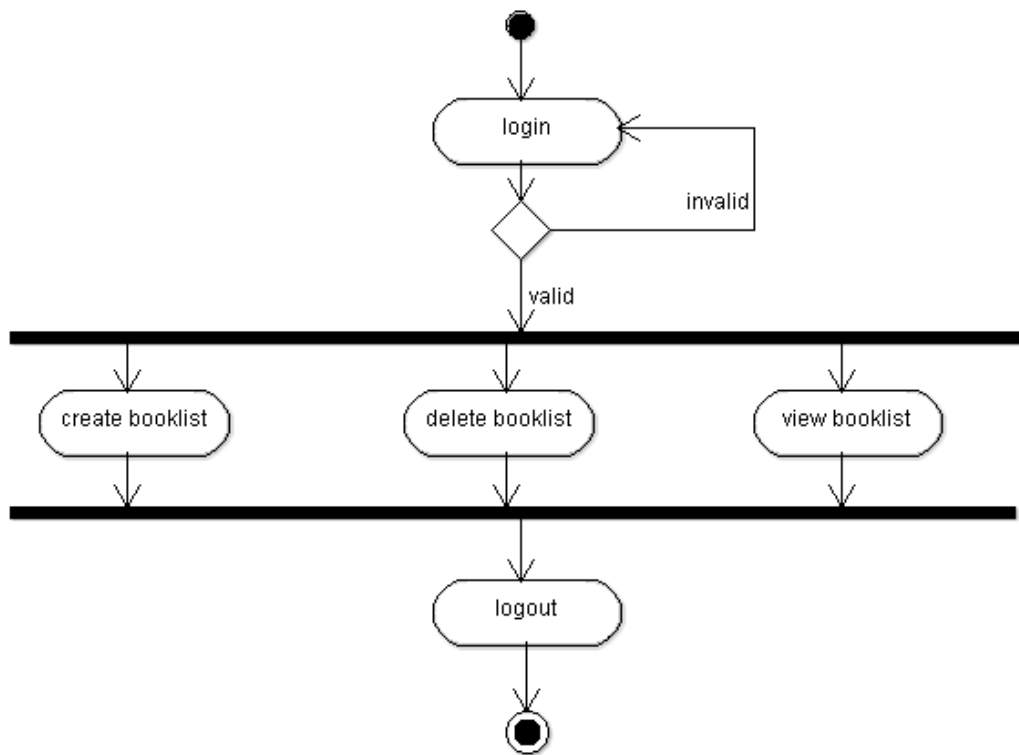
For Book-list

Figure 9: Activity diagram for booklist

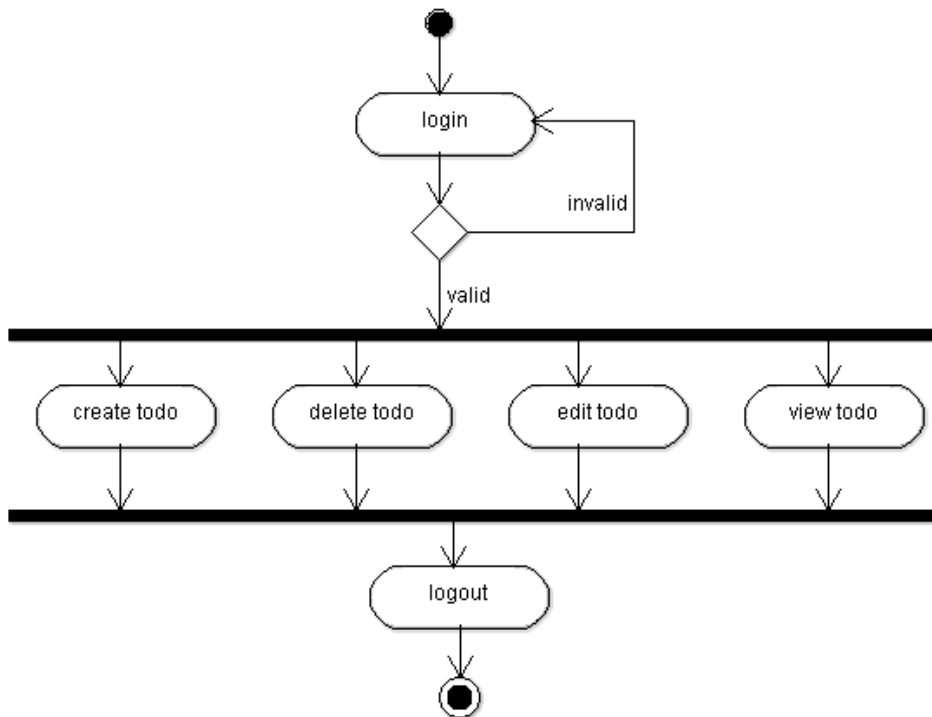
For To-do

Figure 10: Activity diagram for to-do

2.3.6. Sequence diagram

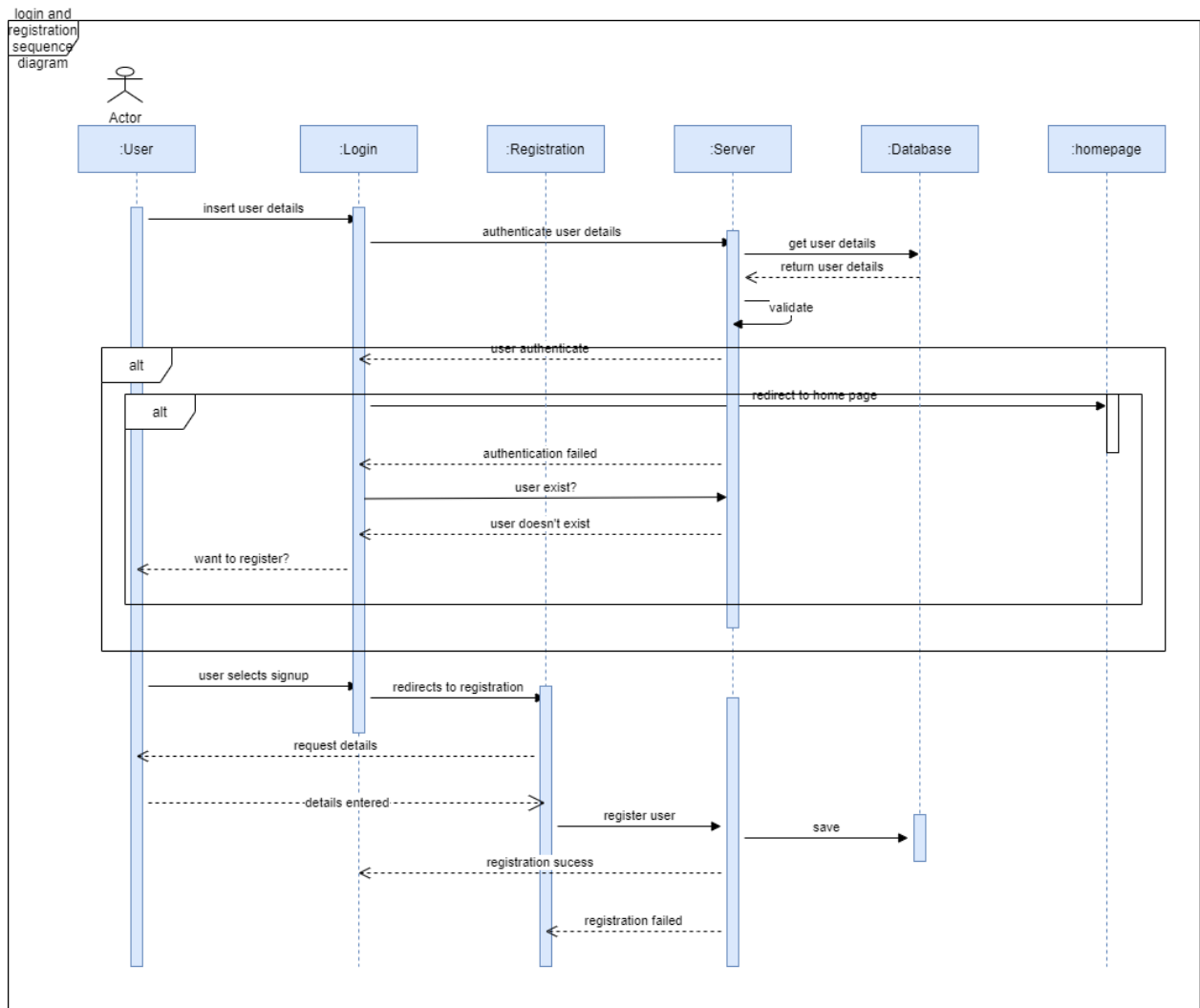


Figure 11:login and registration sequence diagram

For Note

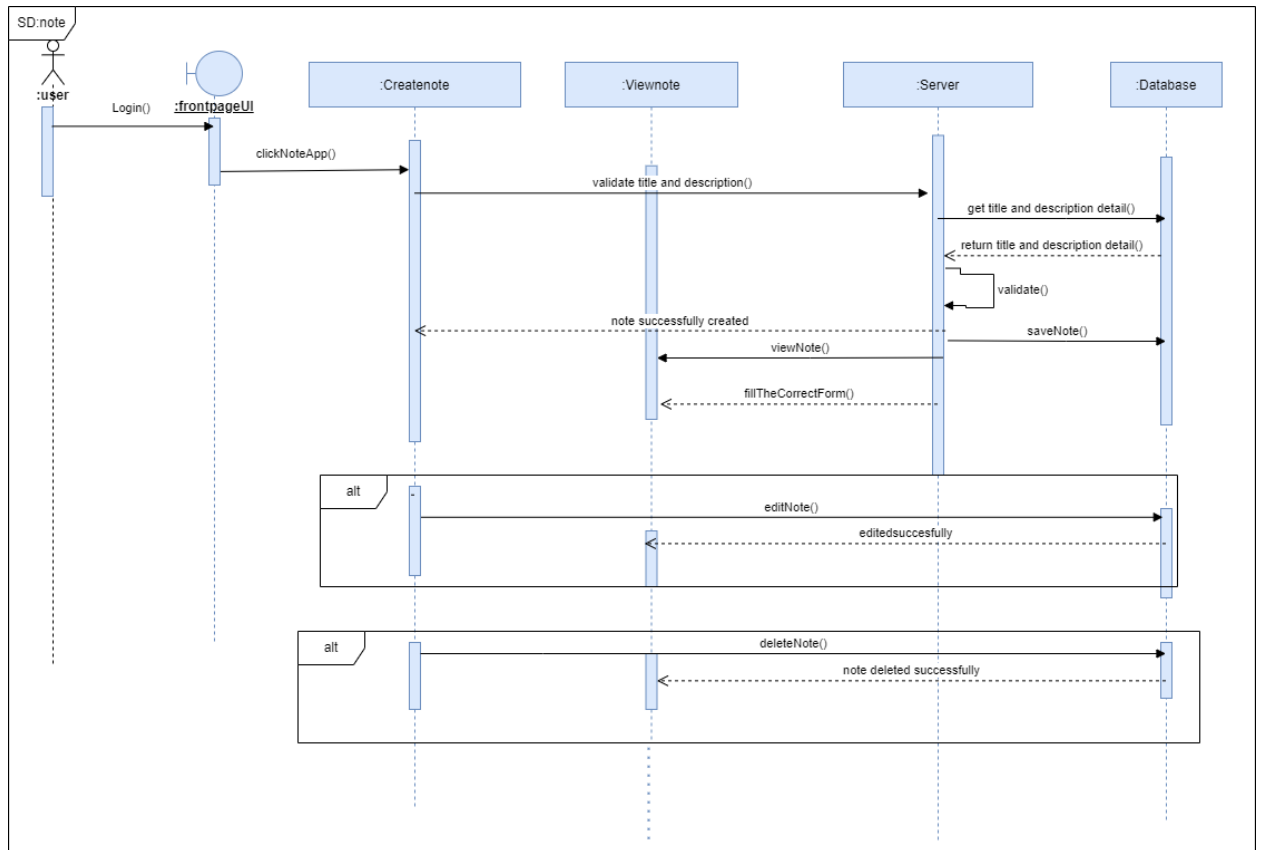


Figure 12: Sequence diagram for note

For Pomodoro

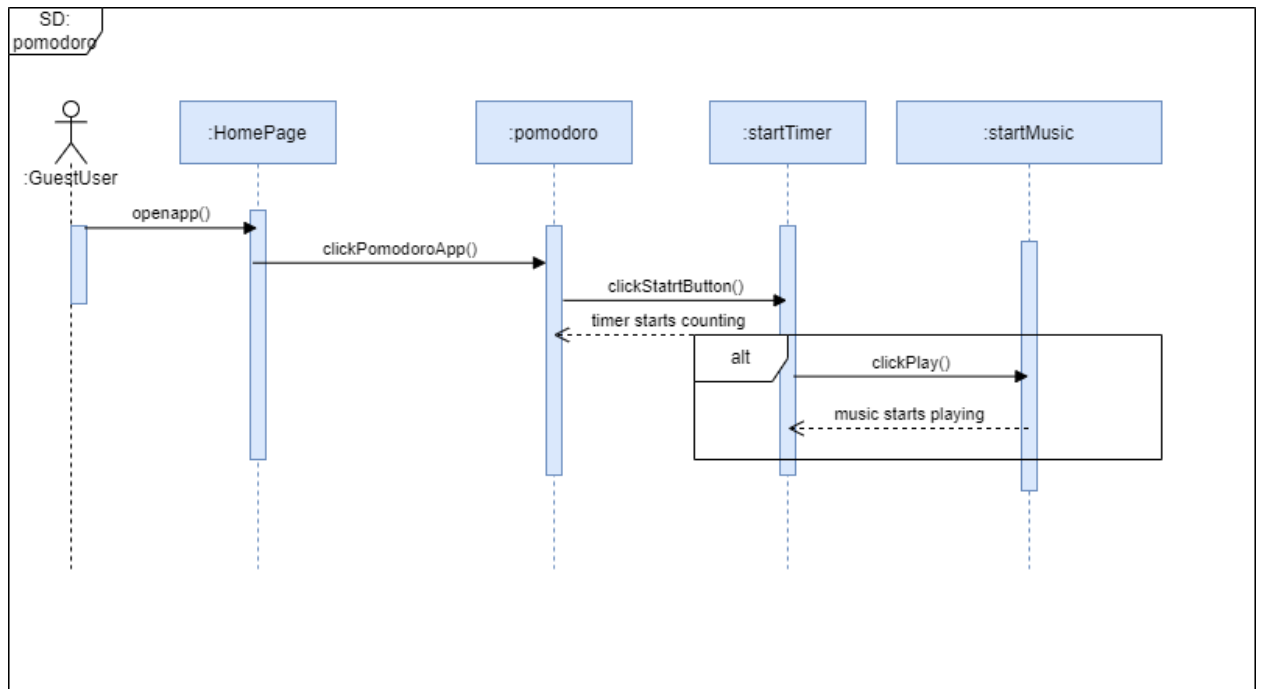


Figure 13: Sequence diagram for pomodoro

2.3.7 Analysis Class Model

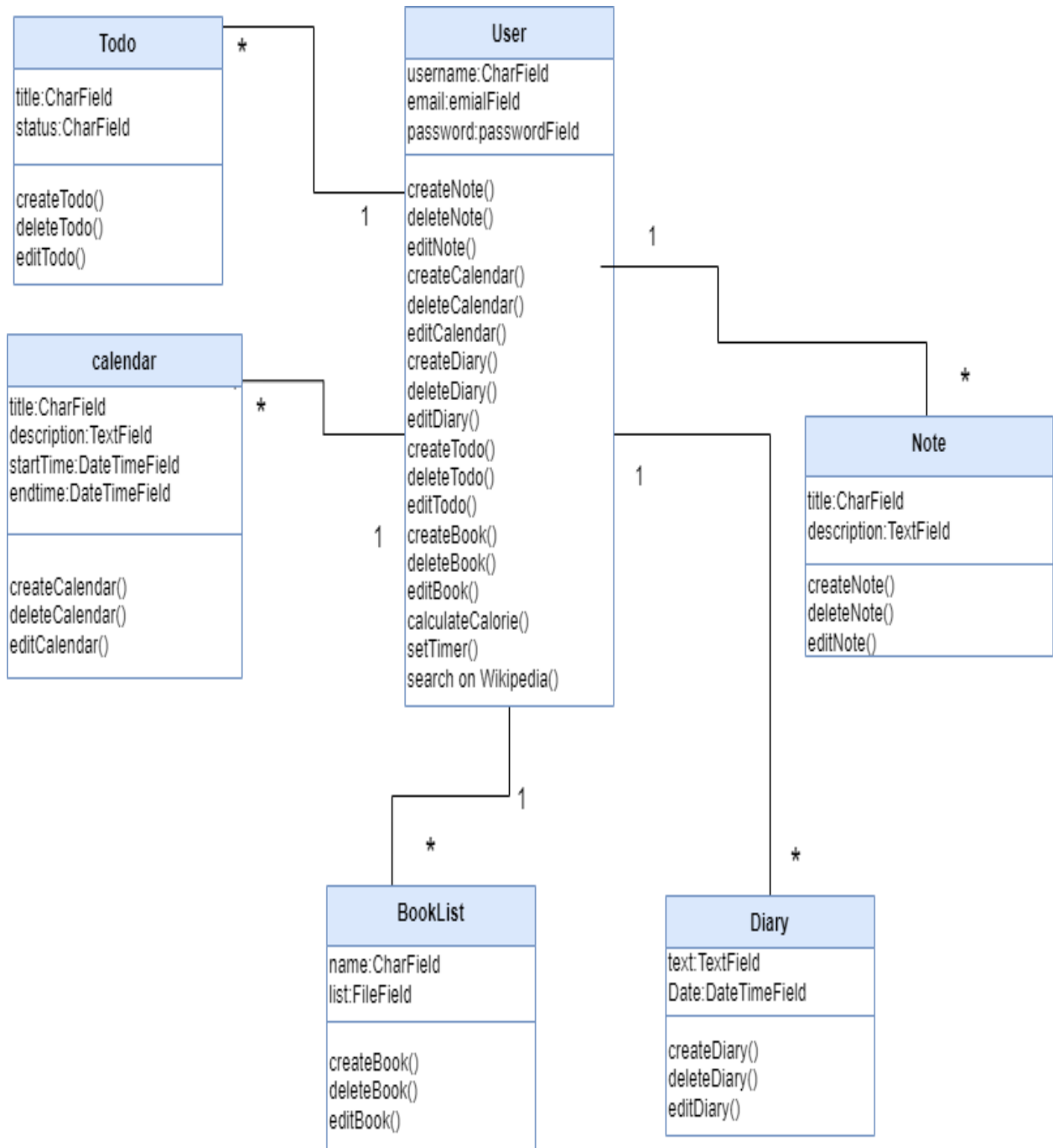


Figure 14: Analysis class model

2.3.7. Logic model

#Pseudocode Example: A Signup Form

Char field: input type = text, name= “username”

Password field: input type = password, placeholder: “Password”

Password confirmation field: input type = password, placeholder: “Password again”

Signup submit value: submit, name: “Sign Up”, default state, disabled.

###On leave focus of username

If username is blank

Error message: “Please fill this field.”

###On password

if password is not sufficiently strong

Error message: “Please replace with a stronger password.”

###On password confirmation

if password confirmation does not match password

Error message: “password doesn’t match.”

###On username, password, or password confirmation

IF username AND password AND password confirmation all contain valid values

Enable Signup

on username already existing

if username already is in use

Error message: “This username is already existed. Did you want to [Sign In] instead?”

#Pseudocode Example: A Login Form

###on password

If password is wrong

Error message: “Please enter a correct username and password.”

#Pseudocode Example: A note app

#Create Note

If title AND description are blank

Error message: “Please fill this field.”

#Edit Note

If title AND description are blank

Error message: “Please fill this field.”

#Pseudocode Example: A calendar app

#Create calendar

###on title and description

If title AND description are blank

Error message: “Please fill this field.”

###on start time and end time

If start time AND end time are blank

Error message: “Please fill this field.”

2.4. Non-functional requirements

Non-functional place constraints on how the system will do so.

Some of the non-functional requirements are:

NFRQ-1: Performance: - the system is available 24/7 except some features that needs internet connection.

NFRQ-2: Usability: - The system is designed to have user friendly interfaces and easy navigation which enhances user's efficiency of usage.

NFRQ-3: Security and access permissions: - one objective of security and access permissions is to control information's about specific user to reduce errors. When information is entered into the system, it should be correct depending on the information stored in the database. it stores specific information about specific user, validations of password, range checks, encrypted password mechanisms recovery of passwords is considered in our proposed system.

NFRQ-4: Error handling: - The system will support prevention of wrong data entry by notifying the user about the possible error.

NFRQ-5: Efficiency: - the response time is very small i.e., not more than five seconds.

NFRQ-6: Reliability: - is the probability that a system will perform its intended function satisfactorily. So, the system works continues when some interrupts faced to it.

NFRQ-7: User interface: -the user interface of the system is user friendly and easy enough to work with.

NFRQ-8: Availability: - describe the time that our system is ready for use. Availability of our system overview that the system is, operable, or usable to perform its designated or required function. It's the aggregate of the resource's accessibility, reliability, maintainability, serviceability, and security.

2.5. System requirement

System requirements are the configuration that a system must have for a hardware or software application to run smoothly and efficiently. Failure to meet these requirements can result in installation problems or performance problems.

2.5.1. Hardware requirements

- ✓ Any desktop or PC computer capable of connecting to internet (256GB)
- ✓ Writing tools (pen, paper): for writing all necessary information associated with the project during interview or time of data collection.
- ✓ Notebook: to take notes during requirements gathering and for making plan for the project

2.5.1.1. Software requirements

- Operating System: Windows 10
- Graphical User Interface: Html, CSS
- Application Logic: Python
- IDE/Workbench: - Django
- Database: MySQL 3
- Source Code Editor: Visual Studio Code
- For designing UML diagrams: Argo UML modeling tool, draw.io
- For preparing documentation: Microsoft word 2016
- To prepare presentation: Microsoft PowerPoint 2016

2.6. Key abstraction with CRC analysis

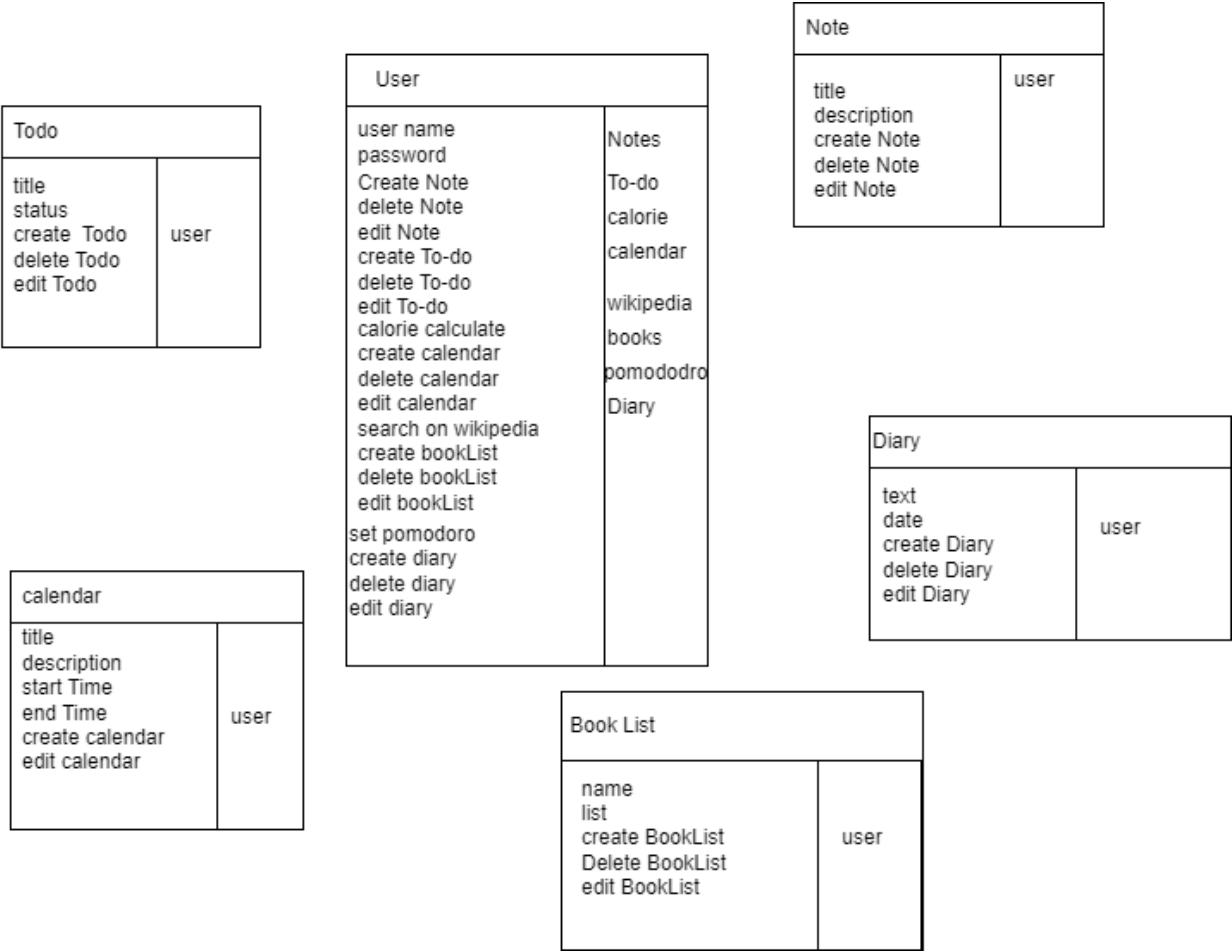


Figure 15: CRC analysis

2.6.1. Conceptual modeling: class modeling

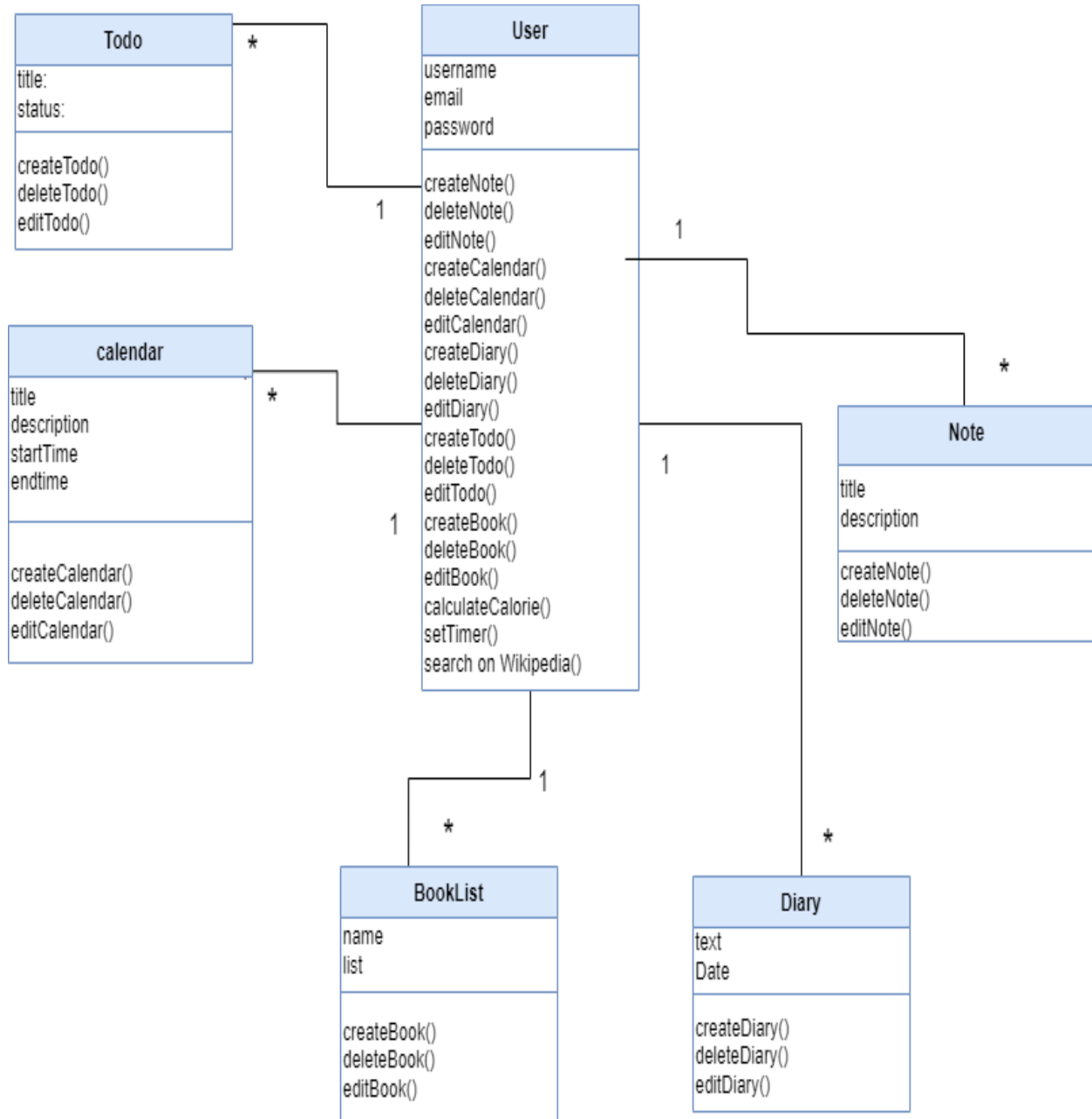


Figure 16: Conceptual class modeling

2.6.2. Identifying change cases

- 1) The app can be used on mobile platforms as well.

2) The software will interact with smartphone sensors to track users' mobility and make exercise recommendations to help them stay in shape.

2.6.3 User interface prototyping

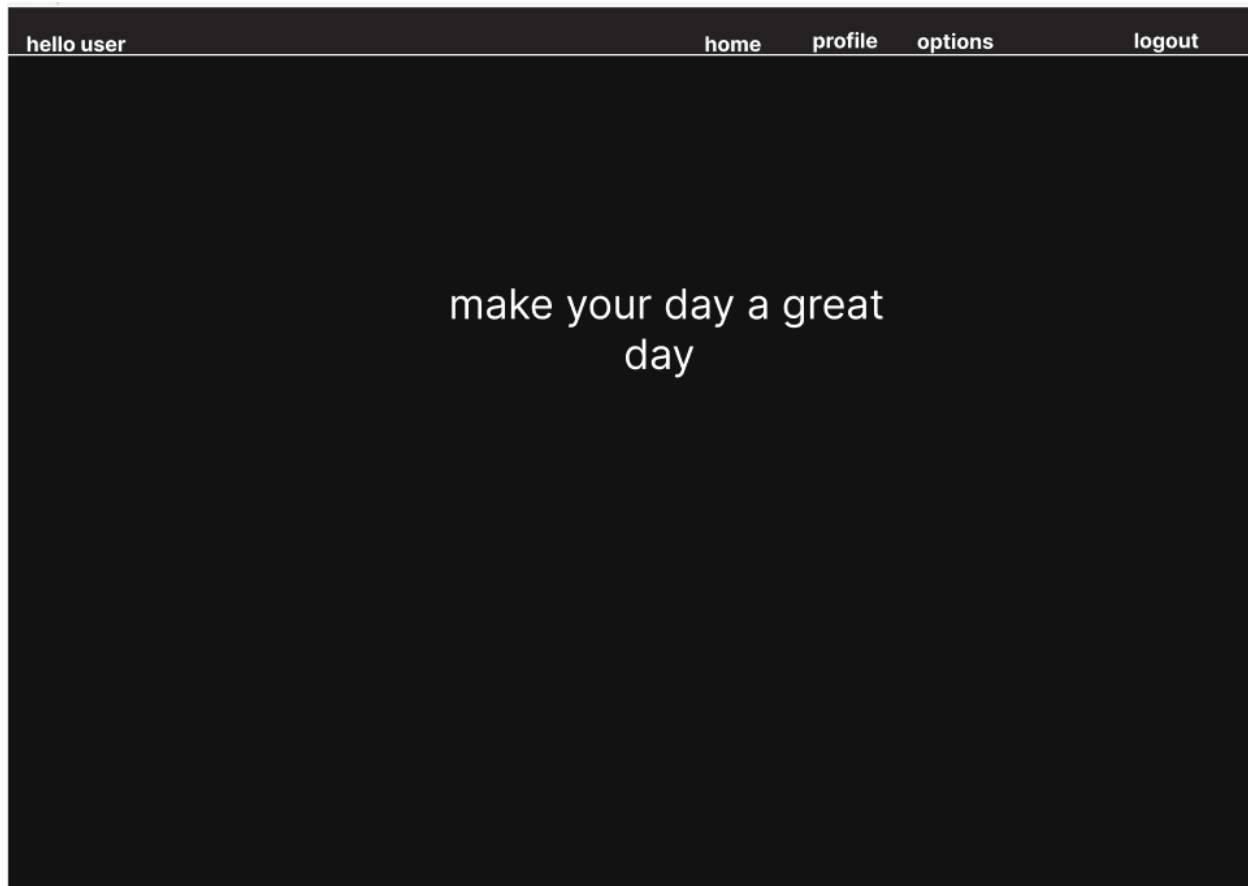


Figure 17:home prototype

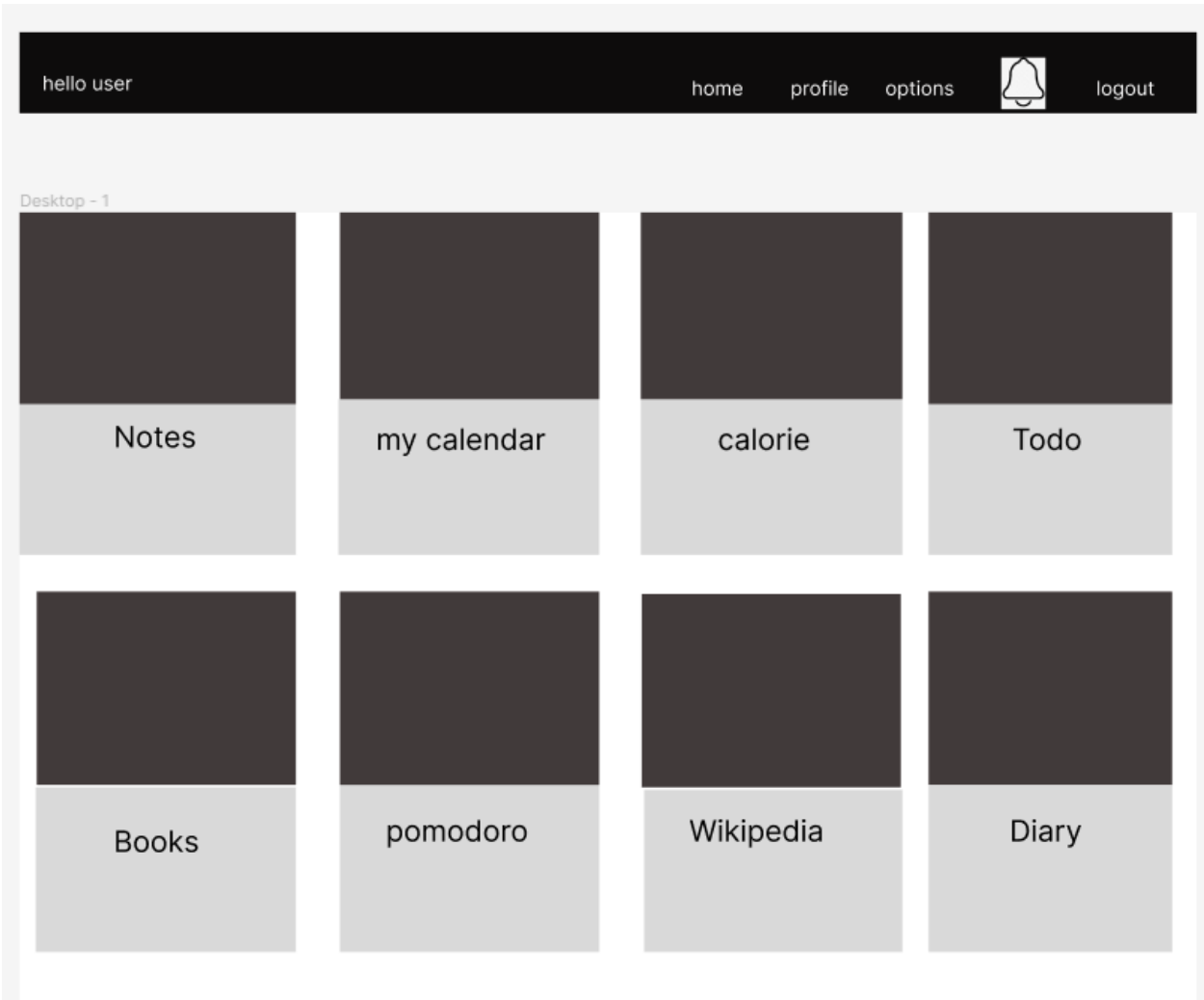


Figure 18:front page



Figure 19:note page

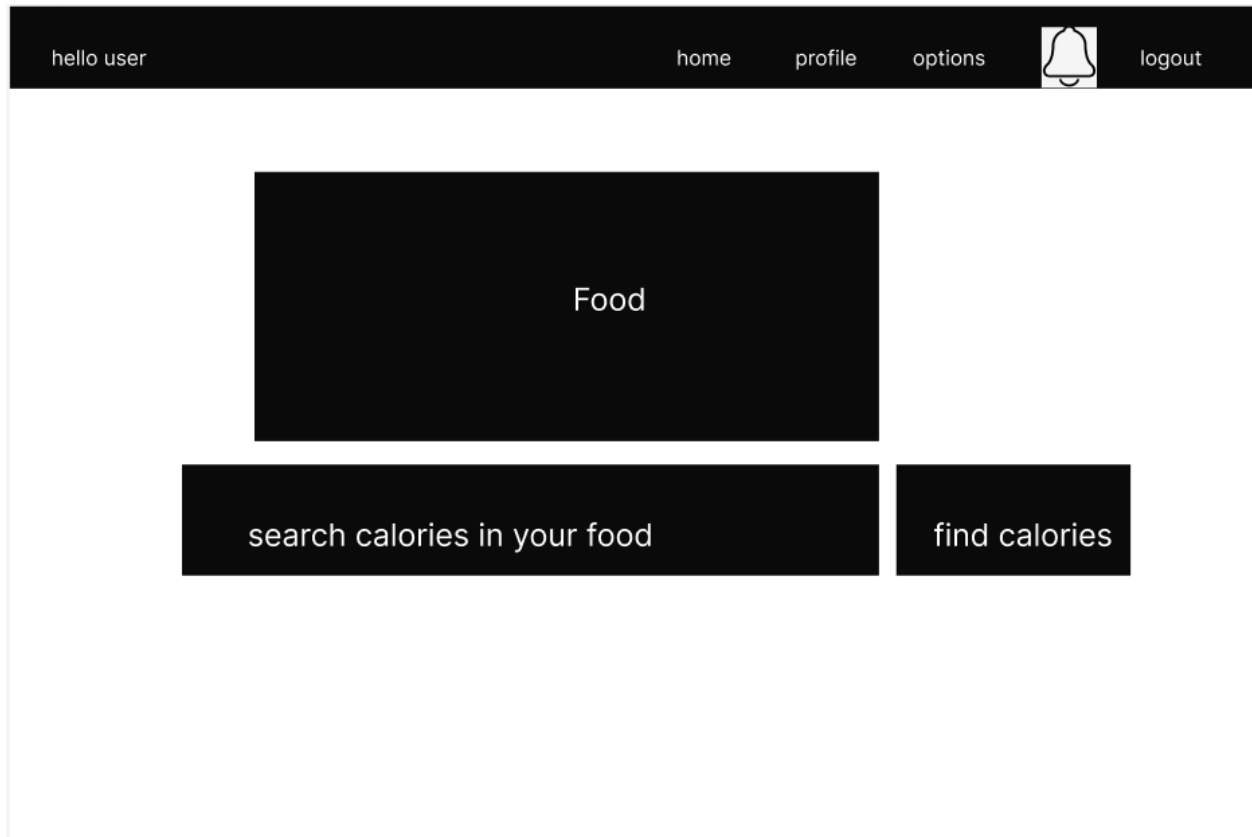


Figure 20:calorie prototype

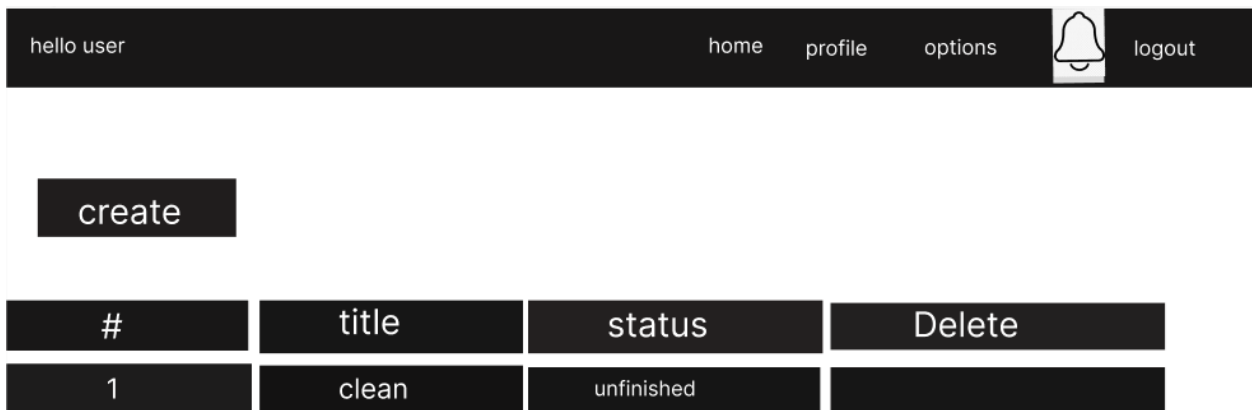


Figure 21:todo prototype



Figure 22:pomodoro prototype

CHAPTER THREE

3. System Design

3.1. Architectural Design

In this project we are using the Three-tier architecture, which separates applications into three logical and physical computing tiers: the presentation tier, or user interface; the application tier, where data is processed; and the data tier, where the data associated with the application is stored and managed.

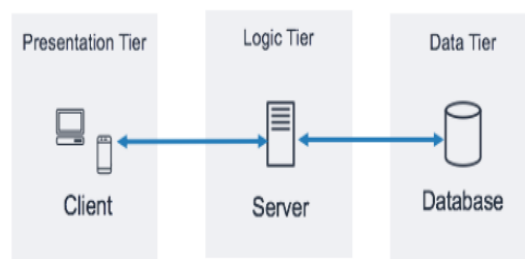


Figure 23: Three-tier architecture

We are using this architecture because each tier can run on a separate operating system and server platform, and this gives us benefits like:

- **Faster development:** Because each tier can be developed simultaneously by different team members.
- **Improved scalability:** Any tier can be scaled independently of the others as needed.
- **Improved reliability:** An outage in one tier is less likely to impact the availability or performance of the other tiers.
- **Improved security:** Because the presentation tier and data tier can't communicate directly, a well-designed application tier can function as a sort of internal firewall.

3.1.1. Component modeling

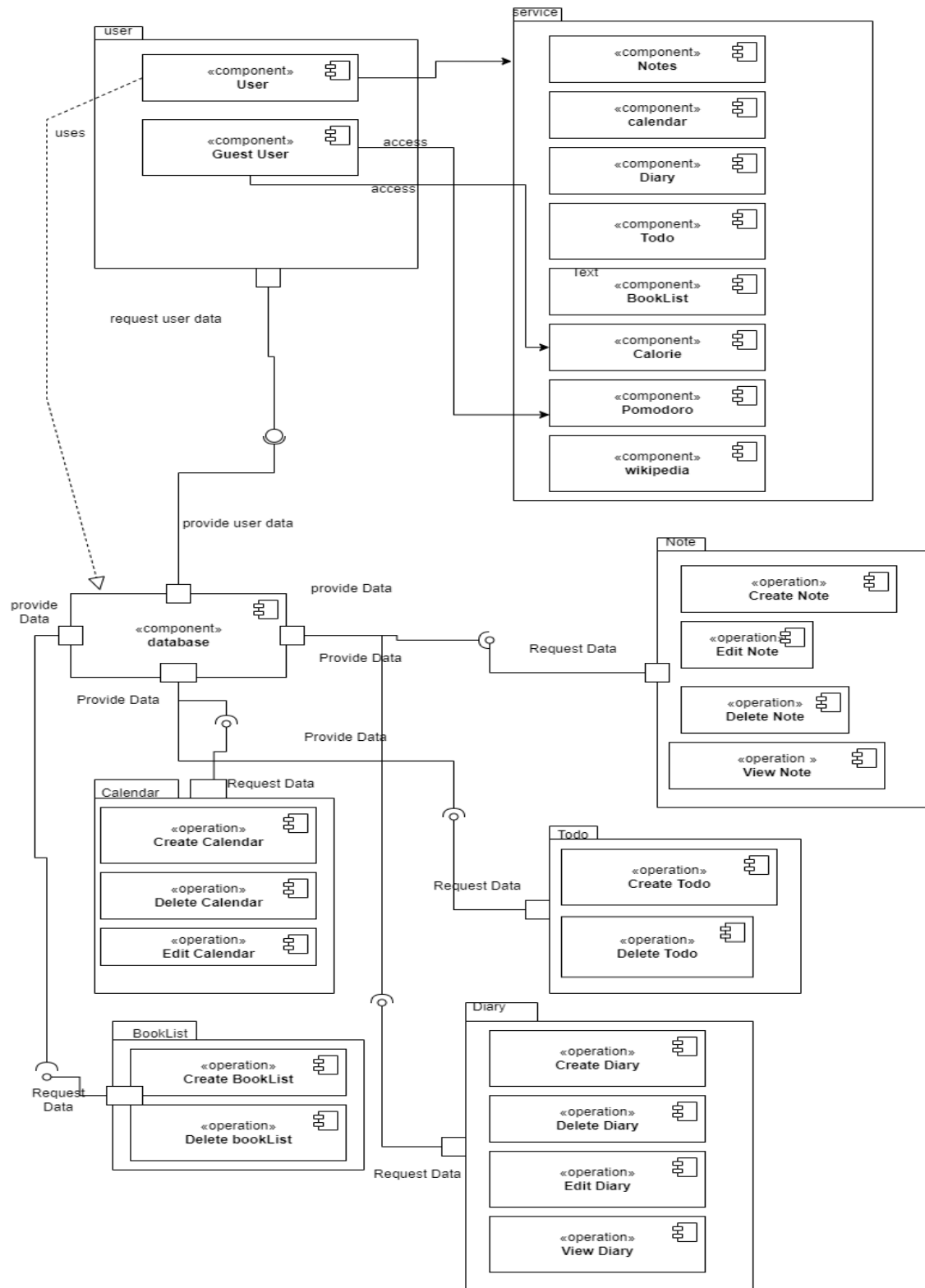


Figure 24: Component modeling

3.1.2. Deployment Modeling

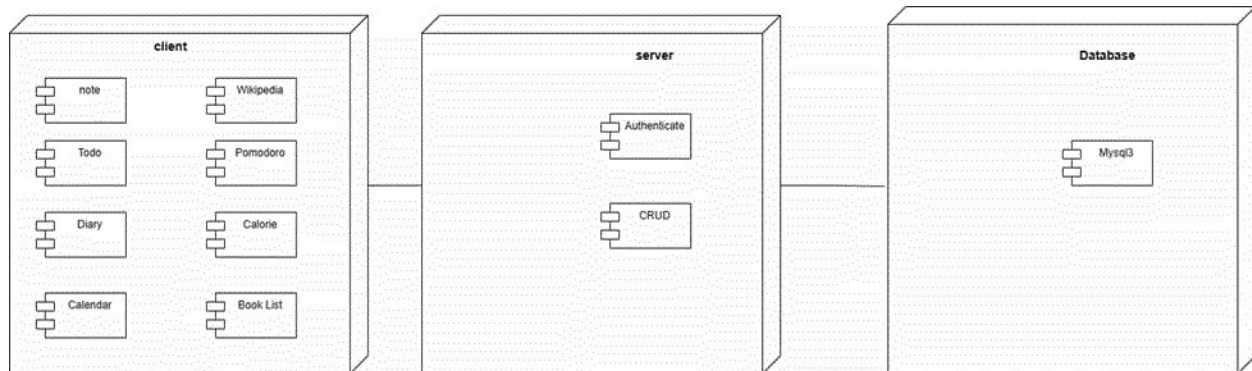


Figure 25: Deployment diagram

3.2. Detail Design

3.2.1. Design class model

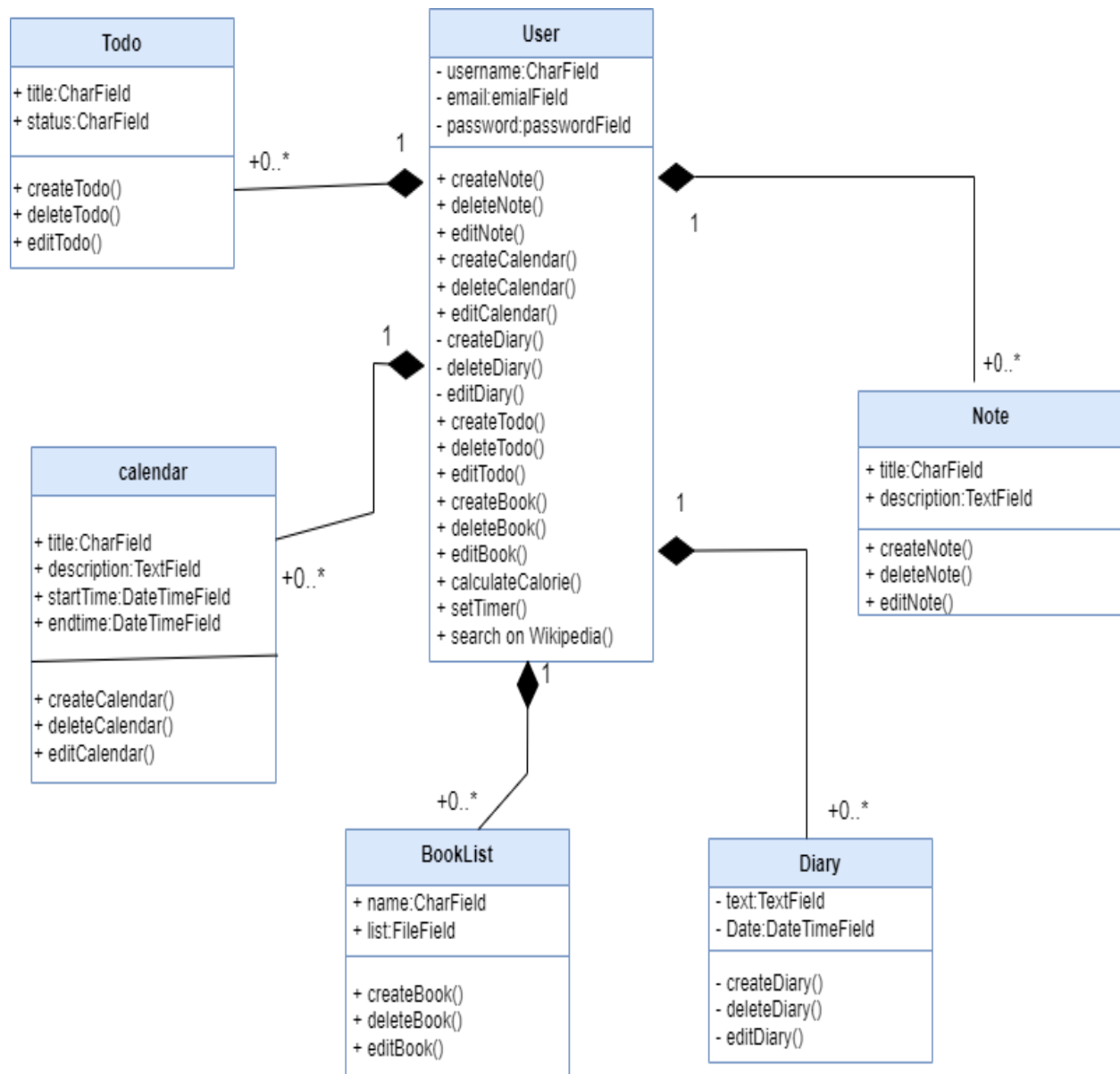


Figure 26: Class model

3.2.2. Persistent model

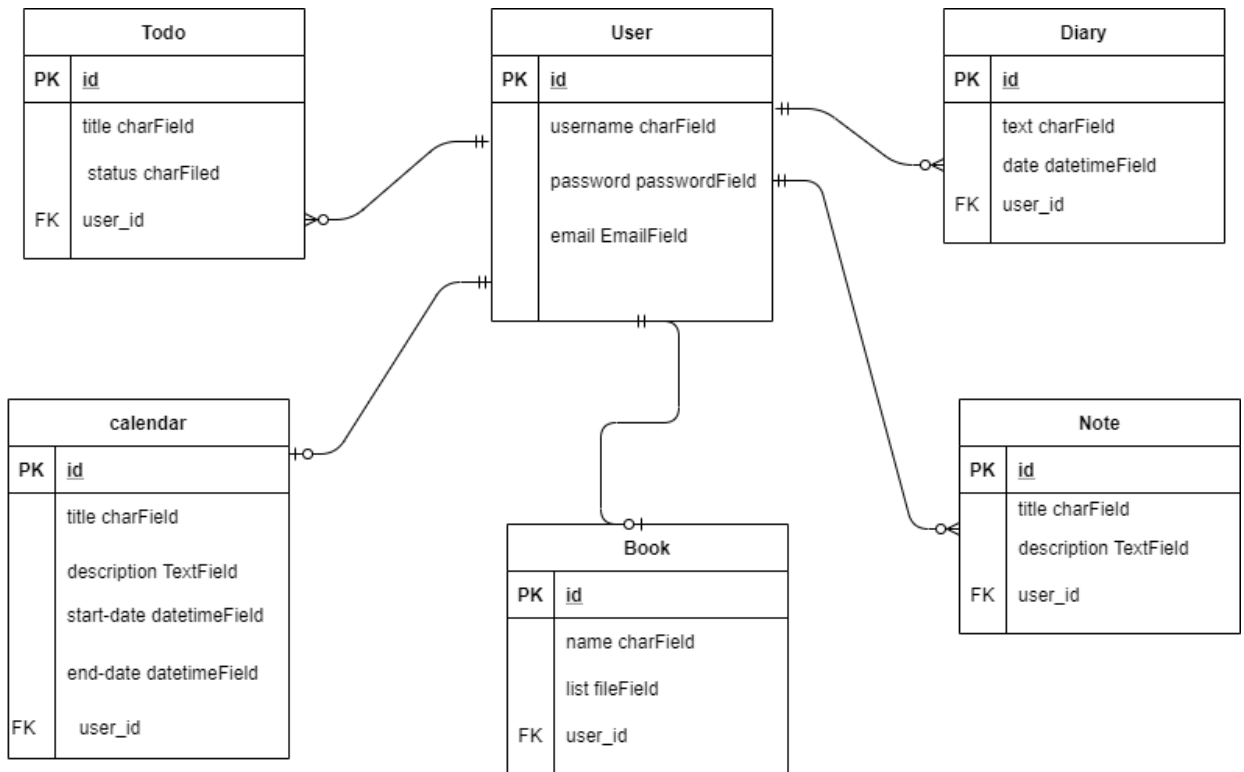


Figure 27: Persistence model

3.3. User Interface Design

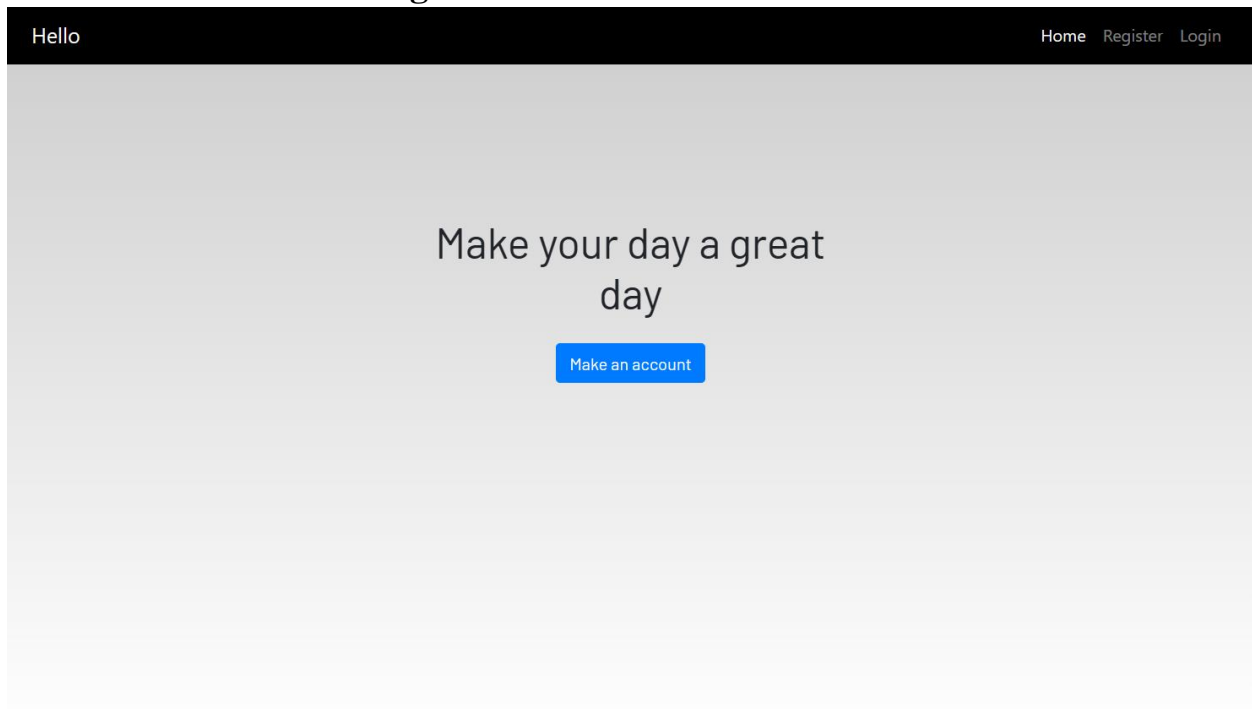


Figure 28:home page

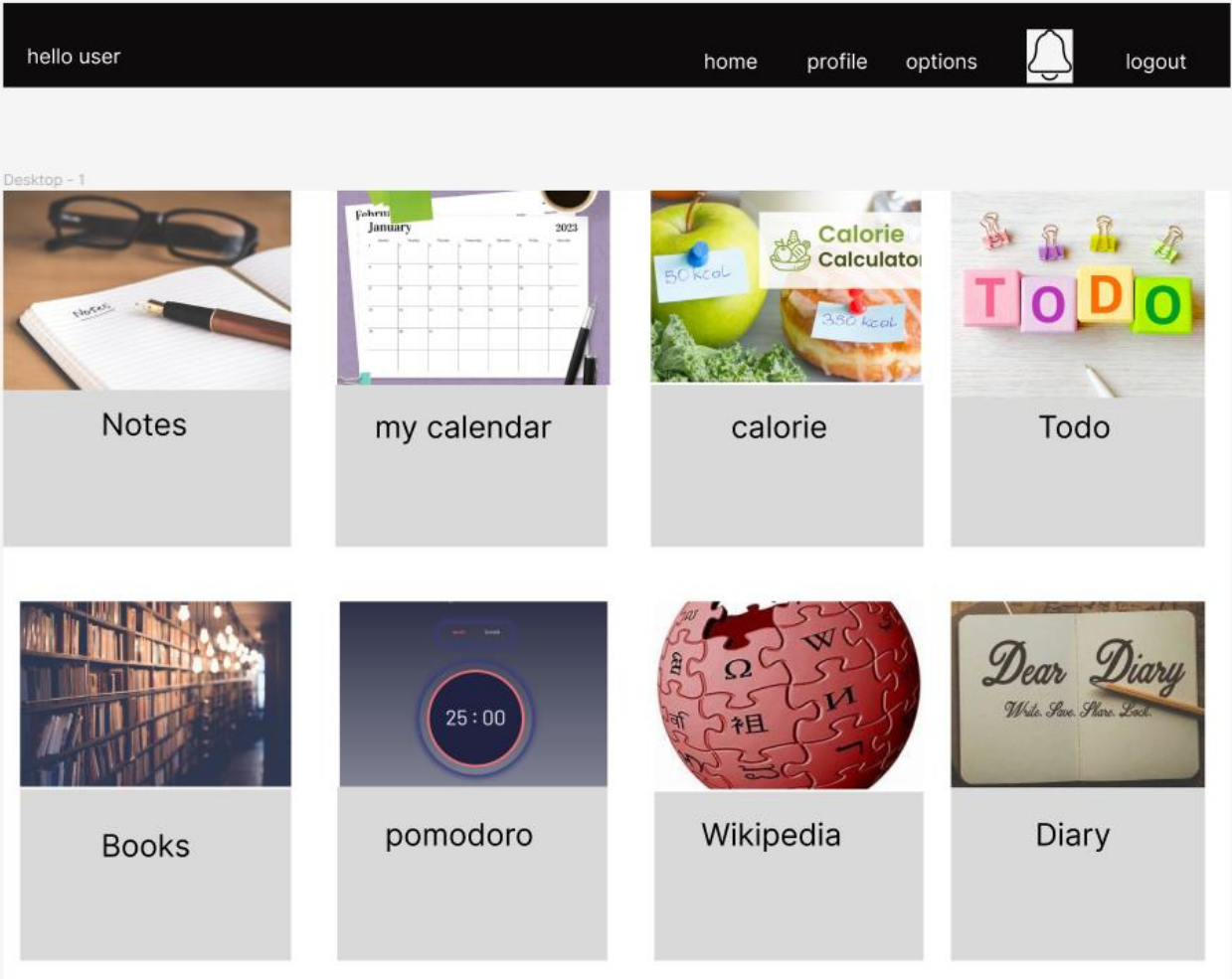


Figure 29:front page

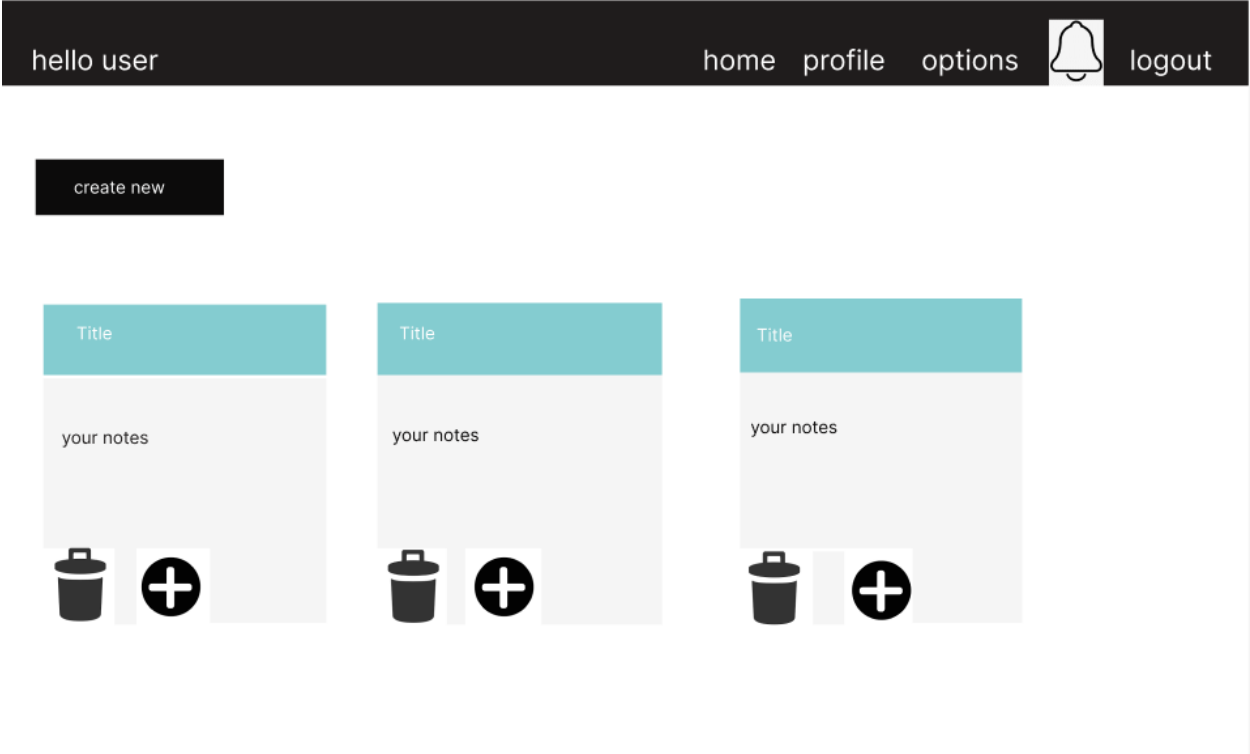


Figure 30:note page

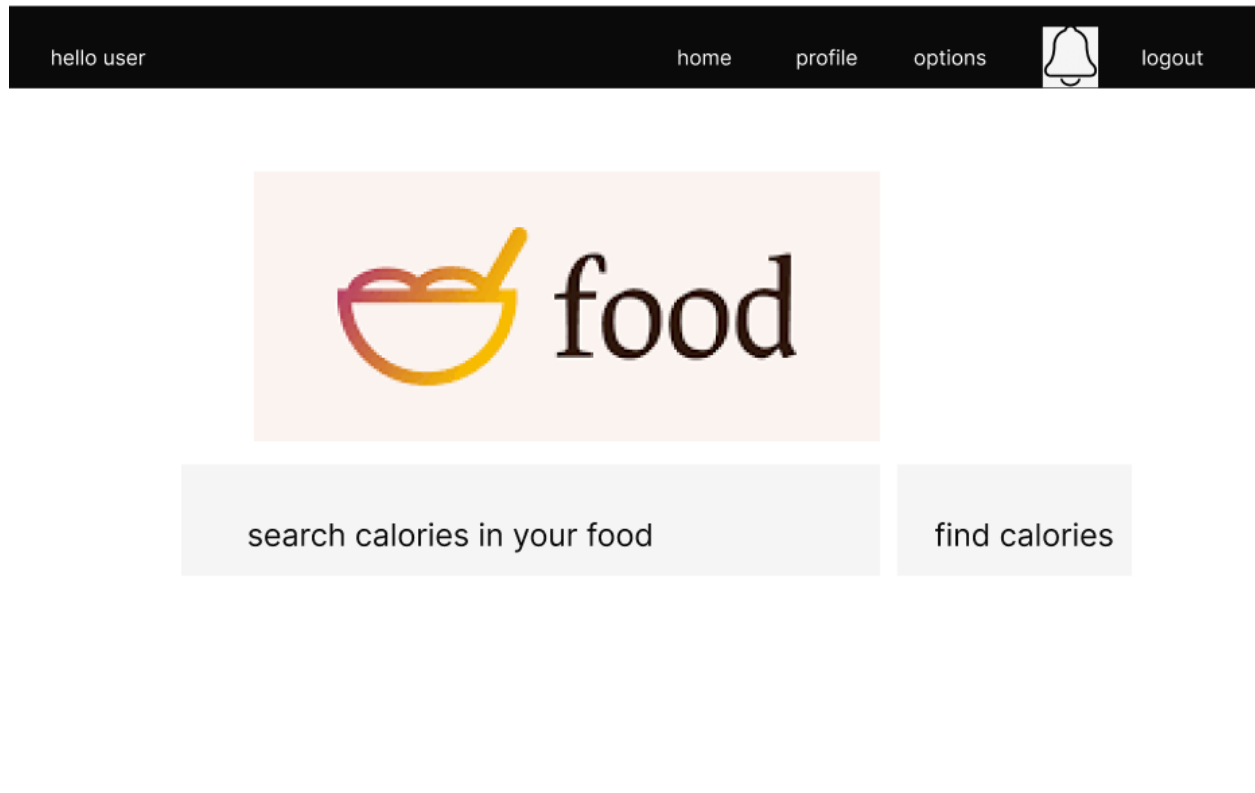


Figure 31:calorie page

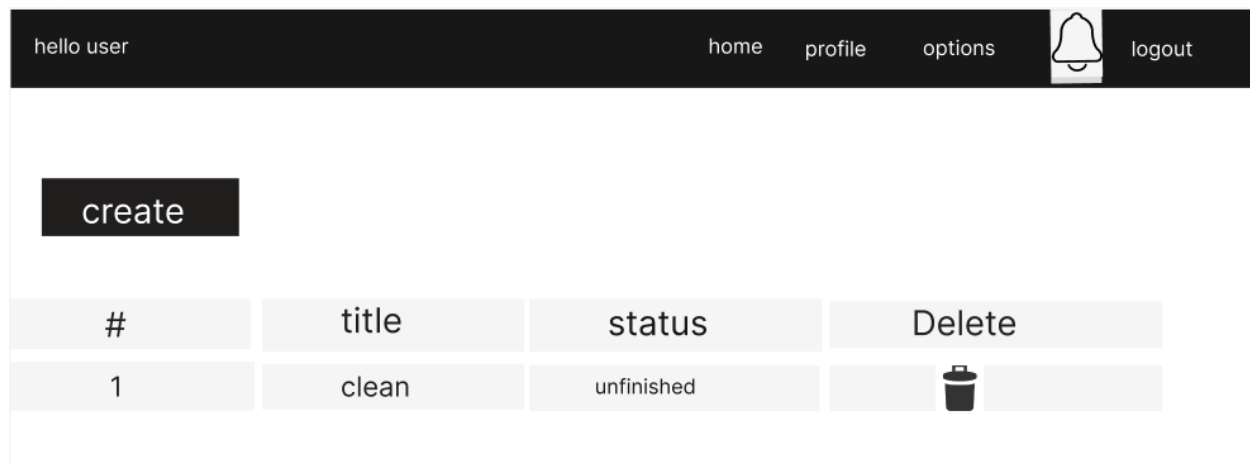


Figure 32:todo page

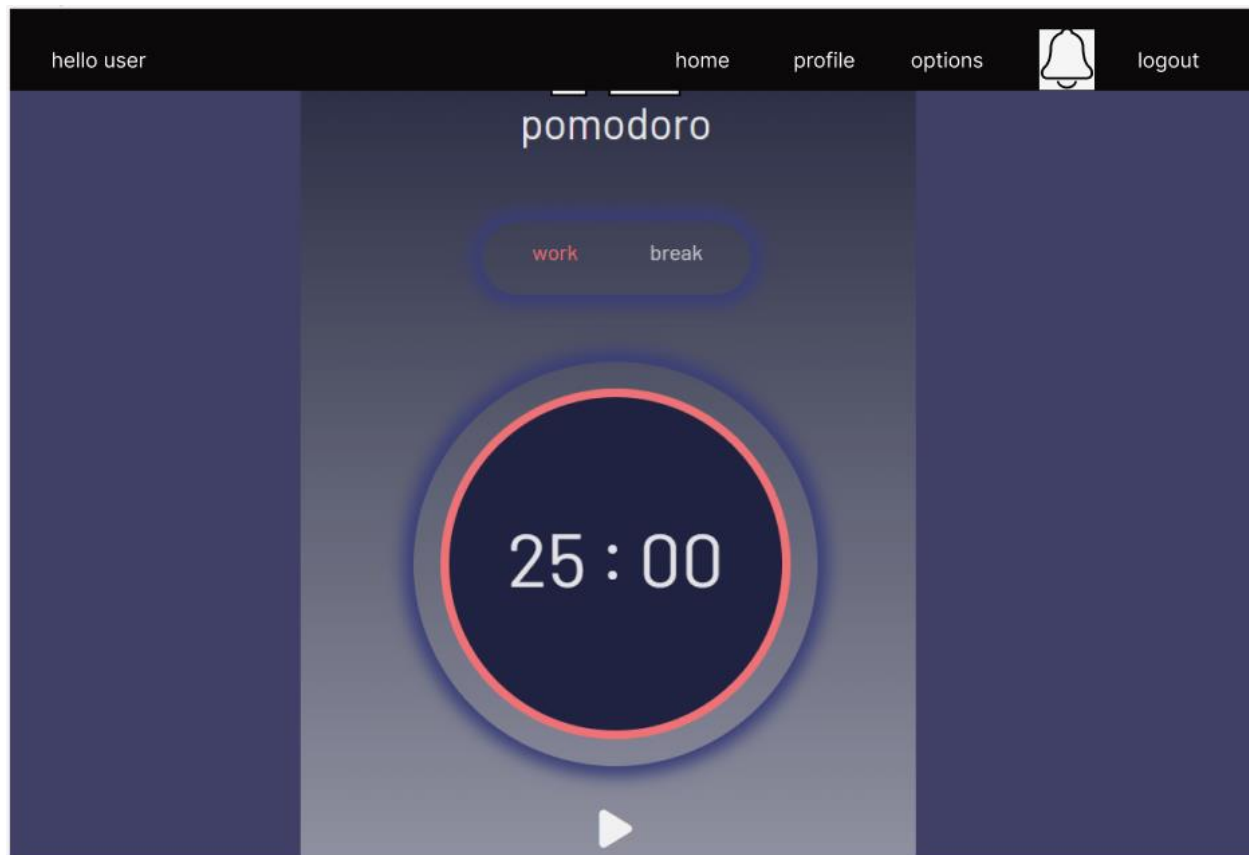


Figure 33: pomodoro

3.4. Access control and security

Access control is a security technique that regulates who or what can view or use resources in a system. The system uses authentication and authorization to restrict users.

Users are restricted to access the system if they are not authenticated users.

User- is someone who has access to the system if he/she has an account.

The user must register first to access the system and must login after register.

If the user is an authenticated user, he/she can,

- ✓ Create, delete, edit, and view Notes.
- ✓ Create, delete, edit, and view Diary.
- ✓ Create, delete, edit, and view calendar.

- ✓ Create, delete, and view Booklist.
- ✓ Create, edit, and view to-do.
- ✓ Set timer.
- ✓ Search on Wikipedia
- ✓ Calculate calorie.

Guest user – is someone who can only access pomodoro and calorie calculator without logging into the system.

Limitation of Guest user

- ✓ Guest user can only access pomodoro and calorie, for extra apps he/she must have an account.

Reference

- [1] Millington, B. (2014). Smartphone apps and the mobile privatization of health and fitness. *Critical Studies in Media Communication*, 31(5), 479-493.
- [2] "medium.com," medium.com, 1995.[online]. Available: <https://medium.com/@mehulshah1995/a-usability-study-on-self-improvement-and-productivity-apps/>[Accessed 10 2022]
- [3] "Wikilpedia," Wikipedia.org, 1998. [Online]. Available: <https://en.wikipedia.org/wiki/self-imrovemnt>. [Accessed 11 2022].
- [4] "geeksforgeeks," geeksforgeeks.org, [Online]. Available: <https://www.geeksforgeeks.org/todo-app-in-c-language/>
- [5] Esposito, M., Rocq, P. L., Novy, E., Remen, T., Losser, M. R., & Guerci, P. (2020). Smartphone to-do list application to improve workflow in an intensive care unit: A superiority quasi-experimental study. *International Journal of Medical Informatics*, 136, 104085.
- [6] Ryu, N., Kawahawa, Y., & Asami, T. (2008, June). A calorie count application for a mobile phone based on METS value. In *2008 5th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks* (pp. 583-584). IEEE.
- [7] Payne, S. J. (1993). Understanding calendar use. *Human-Computer Interaction*, 8(2), 83-100.
- [8] Nelson, R. J., Denlinger, D. L., & Somers, D. E. (Eds.). (2010). *Photoperiodism: the biological calendar*. Oxford University Press.
- [9] Singer, P., Lemmerich, F., West, R., Zia, L., Wulczyn, E., Strohmaier, M., & Leskovec, J. (2017, April). Why we read Wikipedia. In *Proceedings of the 26th international conference on world wide web* (pp. 1591-1600).
- [10] "creately," 2008. [Online]. Available: <https://creately.com/blog/diagrams/class-diagram-tutorial/>.
- [11] "wikipedia," 1998. [Online]. Available: https://en.wikipedia.org/wiki/Component-based_software_engineering
- [12] "wikipedia," 1998. [Online]. Available: https://en.wikipedia.org/wiki/Class_diagram/

[13] "creately," 2008. [Online]. Available: <https://creately.com/blog/diagrams/deployment-diagram-tutorial/>. [Accessed 07 12 2022].

[14] "creately," 2008. [Online]. Available: [UML Diagram Objects and Their Usage \(creately.com\)](https://creately.com/blog/diagrams/uml-diagram-objects-and-their-usage/) [Accessed 07 12 2022].

[15] "creately," creately.com, [Online]. Available: <https://creately.com/blog/diagrams/class-diagram-tutorial/>. [Accessed 03 12 2022].

[16] "upGuard," upguard.com, [Online]. Available: <https://www.upguard.com/blog/access-control/>.

Appendices

Appendix I

- Diagram: a visual representation of a subset of features of a UML Mode.
- Action: is the fundamental unit of behavior specification and represents some transformation or processing in the modeled system, such as invoking a method of a class or a sub activity.
- Actor - a role that a user takes when invoking a use case.
- Class: the primary declarative construct of Object-Oriented Programming; a cohesive unit of Attributes and Operations; a compile-time template for an Object.
- Class diagram: a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, and the relationships between the classes.
- Activity: carrying out behavior in a state machine diagram.
- Activity-diagram: a diagram that describes procedural logic, business process and workflow supporting parallelism.
- Sequence diagram: describes the Messages sent between several participating Objects in a Scenario.
- Deployment diagram: diagrams are special diagrams used to focus on software components and hardware components.

Appendix II

Interview questions.

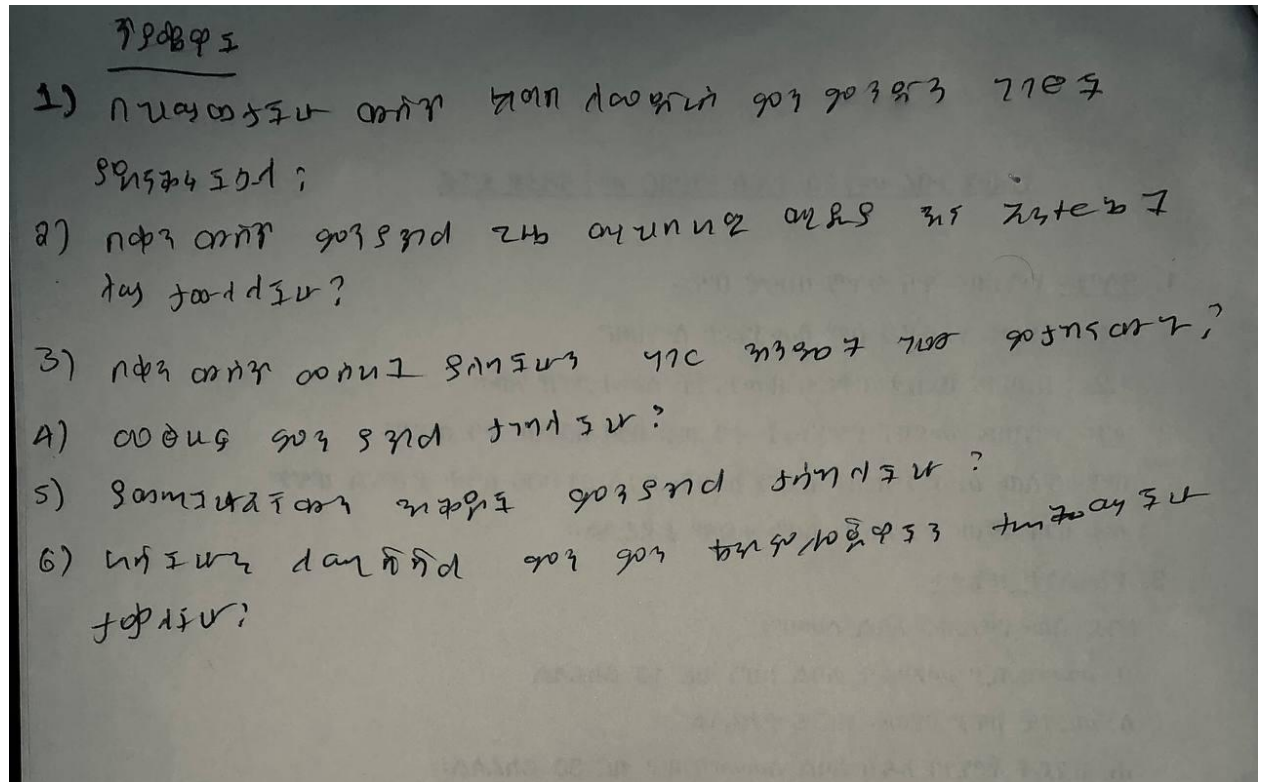


Figure 35: interview questions