**Tutor Hunt Application**



**Team ID – BSEF19**

**Session: BS SE Fall 2019**

**Project Advisor: Dr. Shahid Farid**

**Submitted By**

|  |  |
| --- | --- |
| Mahnoor Shoukat | BSEF19A502 |
| Iqra Junaid | BSEF19A511 |
| Mirza Muhammad Hassan Zahid | BSEF19A513 |
| Asim Afzal | BSEF19A542 |

FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY

UNIVERSITY OF THE PUNJAB, LAHORE

**Abstract**

Finding a tutor is difficult job. But we have solution for it. Our project is to develop a mobile application in which tutors can register themselves and post their qualification and availability to teach. Similarly, the students can register themselves and search for tutor according to their requirements. If they find a tutor then they can communicate with him/her via this platform to discuss all the details. A student can also request for a free demo lecture online using this platform and there will be no need to communicate with them or take a demo lecture using any other app. After hiring a tutor, a student can also give a review and feedback about him/her.

**Acknowledgement**

I would like to express my sincere appreciation to everyone who has contributed to the completion of my final year project. My heartfelt thanks go to my project supervisor, **Dr. Shahid Farid, Assistant** **Professor** of **Punjab University College of Information Technology, University of the Punjab, Lahore** for their invaluable guidance and support. I am also grateful to the faculty members, classmates, and friends who have provided assistance and encouragement throughout this endeavor. Additionally, I extend my gratitude to the participants and my family for their contributions and unwavering support. Together, their collective efforts have made this project a success, and I am truly grateful for their contributions.

Date: June 16, 2022

**Proofreading Certificate**

I hereby certify that I have thoroughly reviewed the document with great attention to detail. After careful examination, I am confident that the project is free from any spelling, punctuation, or grammatical errors. The document exhibits a commendable level of organization, and I am convinced that its objectives have been accomplished successfully.

Ms. Samreen Shahid,

Business Communication and Technical Writing,

Lecturer, PUCIT

**Statement of Submission**

This is to certify that **Mahnoor Shoukat** **(BSEF19A502)**, **Iqra Junaid (BSEF19A511)**, **Mirza Muhammad** **Hassan Zahid (BSEF19A513)** and **Asim Afzal (BSEF19A542)** have successfully completed the final project named as: **Tutor Hunt Application**, at the Faculty of Computing and Information Technology, University of The Punjab, Lahore, to fulfill the partial requirement of the degree of **Bachelors of Science in Software Engineering.**

Project Office Supervisor

PUCIT, Lahore

|  |  |
| --- | --- |
| Project Primary Advisor  Name: Dr. Shahid Farid  Designation: Assistant Professor | Project Examiner  Name: Dr. Shahid Farid  Designation: Assistant Professor |

**Table of Contents**

[1 Introduction 1](#_Toc138530584)

[1.1 Project Title 1](#_Toc138530585)

[1.2 Project Overview Statement Template 1](#_Toc138530586)

[1.3 Project Goals & Objectives 2](#_Toc138530587)

[1.4 High Level System Component 3](#_Toc138530588)

[1.5 List of Optional Functional Units 3](#_Toc138530589)

[1.6 Exclusions 3](#_Toc138530590)

[1.7 Application Architecture 4](#_Toc138530591)

[1.8 Gantt Chart 5](#_Toc138530592)

[First Deliverable Guide 8](#_Toc138530593)

[1 Introduction 8](#_Toc138530594)

[1.1 Project Feasibility Report 8](#_Toc138530595)

[1.1.1 Technical Feasibility 8](#_Toc138530596)

[1.1.2 Operational Feasibility 8](#_Toc138530597)

[1.1.3 Economic Feasibility 8](#_Toc138530598)

[1.1.4 Schedule Feasibility 9](#_Toc138530599)

[1.1.5 Specification Feasibility 9](#_Toc138530600)

[1.1.6 Informational Feasibility 9](#_Toc138530601)

[1.1.7 Motivational Feasibility 9](#_Toc138530602)

[1.1.8 Legal and Ethical Feasibility 9](#_Toc138530603)

[1.2 Project/Product Scope 9](#_Toc138530604)

[1.3 Project/Product Costing 10](#_Toc138530605)

[1.3.1 Project Cost Estimation by Function Point Analysis 10](#_Toc138530606)

[1.3.2 Project Cost Estimation by using COCOMO’81 (Constructive Cost Model) 12](#_Toc138530607)

[1.4 CPM – Critical Path Method 16](#_Toc138530608)

[1.5 Gantt chart 18](#_Toc138530609)

[1.6 Introduction to Team members and their skill set 20](#_Toc138530610)

[1.7 Tools and Technology with reasoning 20](#_Toc138530611)

[1.8 Vision Document 21](#_Toc138530612)

[1.9 Risk List 22](#_Toc138530613)

[1.10 Requirement Engineering 22](#_Toc138530614)

[1.10.1 System Specifications 22](#_Toc138530615)

[1.10.2 Identifying External Entities 23](#_Toc138530616)

[1.10.3 Context Level Data Flow Diagram 24](#_Toc138530617)

[1.10.4 Capture "shall" Statements 24](#_Toc138530618)

[1.10.5 Allocate Requirements 25](#_Toc138530619)

[1.10.6 Prioritize Requirements 26](#_Toc138530620)

[1.10.7 Requirements Trace-ability Matrix 27](#_Toc138530621)

[1.10.8 High Level Use-case Diagram 28](#_Toc138530622)

[Second Deliverable Guide 29](#_Toc138530623)

[Introduction 29](#_Toc138530624)

[1.1 Use Case Description 29](#_Toc138530625)

[1.2 Use Case Diagram (Refined) 37](#_Toc138530626)

[1.3 Domain Model 38](#_Toc138530627)

[1.4 Sequence Diagram 39](#_Toc138530628)

[1.5 Collaboration Diagram 40](#_Toc138530629)

[1.6 Operation Contracts 41](#_Toc138530630)

[1.7 Design Class Diagram 45](#_Toc138530631)

[1.8 Data Model 46](#_Toc138530632)

[Prototypes 47](#_Toc138530633)

[1. Boarding Screens 47](#_Toc138530634)

[2. Sign-in Screen 48](#_Toc138530635)

[3. Sign-up Screen 49](#_Toc138530636)

[4. Forget Password Screen 50](#_Toc138530637)

[5. Tutor Dashboard 51](#_Toc138530638)

[6. Student Dashboard 52](#_Toc138530639)

[7. Edit Profile Screen 53](#_Toc138530640)

[8. Request Screen 54](#_Toc138530641)

[9. Chatting Screen 55](#_Toc138530642)

# 

# Introduction

# Project Title

ANDROID-BASED **TUTOR HUNT** APP

# 1.2 Project Overview Statement Template

|  |
| --- |
| **PROJECT TITLE:**  ANDROID BASED **TUTOR HUNT** APP |
| **GROUP LEADER:**  IQRA JUNAID |
| **PROJECT MEMBERS:**   |  |  |  | | --- | --- | --- | | NAME | ROLL NUMBER | EMAIL ADDRESS | | MAHNOOR SHOUKAT | BSEF19A502 | bsef19a502@gmail.com | | IQRA JUNAID | BSEF19A511 | bsef19a511@gmail.com | | HASSAN ZAHID | BSEF19A513 | bsef19a513@gmail.com | | ASIM AFZAL | BSEF19A542 | bsef19a542@gmail.com | |
| **PROJECT GOAL:**  The main goal of this project is to provide an application for users to find a tutor online. |
| **OBJECTIVES:**  The objective of our project is to make a user-friendly, time and cost- efficient application of an online tutor finder through which a user can find a tutor- check his qualification, contact them, hire them and give feedback about them. |
| **PROJECT SUCCESS CRITERIA:**  A user-friendly platform where a user will find a tutor according to his/her requirements. |
| **RISKS AND OBSTACLES:**   * Time constraint due to university work load * A risk to meet user’s satisfaction as users have different requirements that are not feasible to implement all in one. * Difficulty in learning new technologies. |
| **TYPE OF PROJECT:**  Development |
| **TARGET END USERS:**  Admin and Users (Tutors, Students and Parents) |

|  |
| --- |
| **DEVELOPMENT TECHNOLOGY:**   * HTML, CSS, JS and JSX * React Native for mobile app * MERN for web version |
| **PLATFORM:**  Mobile-based application |
| **PROJECT SUPERVISOR:**  Dr Shahid Farid |
| **APPROVED BY:** |
| **DATE:** |

# 1.3 Project Goals & Objectives

To develop a mobile and web based app in which tutors can register themselves and post their qualification and availability to teach. Similarly, the students can register themselves and search for tutor according to their requirements. If they find a tutor then they can communicate with him/her via this platform to discuss all the details. A student can also request for a free demo lecture online using this platform and there will be no need to communicate with them or take a demo lecture using any other app. After hiring a tutor, a student can also give a review and feedback about him/her.

# 1.4 High Level System Component

Following are the high-level system components of this application:

* Sign Up
* Login/Logout
* Register as a tutor
* Register as a student
* Search for available tutors
* View tutor details
* Choose a tutor
* Communicate with tutor
* Hire a tutor
* Give feedback and review about the hired tutor

# 1.5 List of Optional Functional Units

Following are the optional features that we might implement according to availability of time:

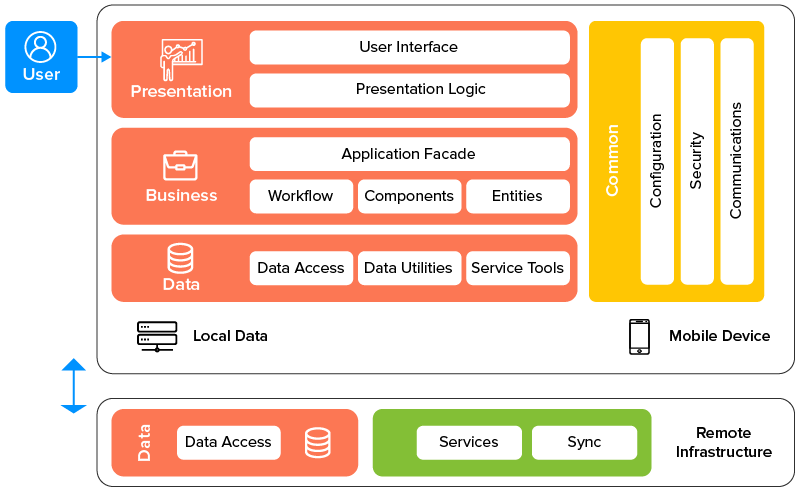
* Real Time Communication with tutor on voice call and video call
* Web based version for “Tutor Hunt” mobile app

# 1.6 Exclusions

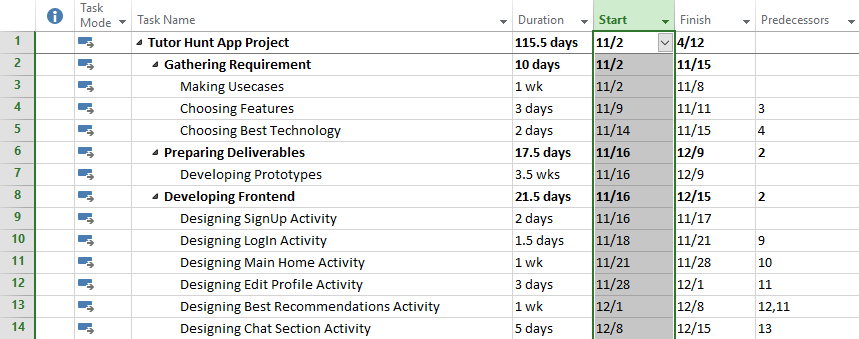
Exclusions are description of tasks that we are not including in our project scope. These exclusions are:

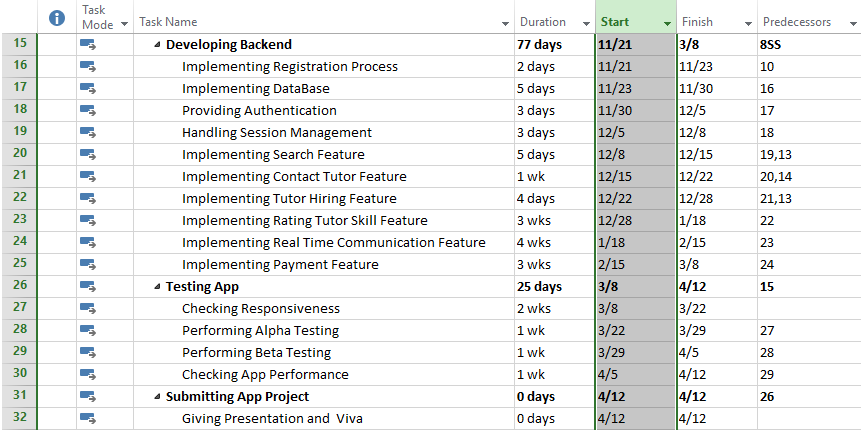
* Tutor payment/salary module will not be included because of time constraint, resources and real time issues.
* Visual representation of tutor’s address will also not be included.

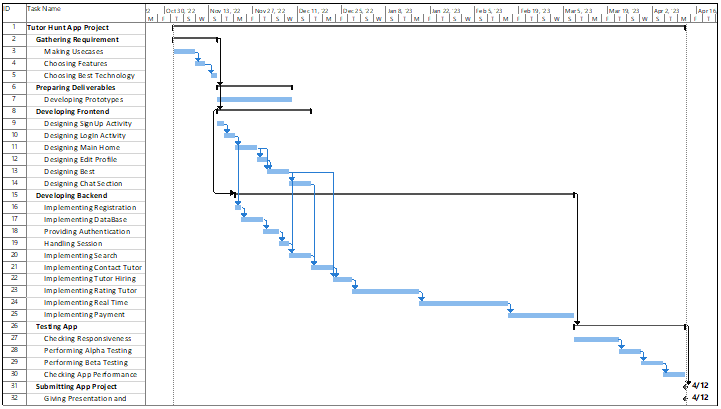
# 1.7 Application Architecture



# 1.8 Gantt Chart







**1.9 Hardware and Software Specification**

**Hardware**

* Laptop or desktop with medium specifications
* Android mobile phone with new generations

**Software**

* Android play store
* Android emulators
* Windows
* Web browser --- Chrome

**1.10 Tools and Technologies used with reasoning**

**Technologies**

* HTML / JSX for general UI purposes
* CSS for designing purposes
* JS for some dynamic operations
* React Native --- Main Tech Stack for our tutor mobile App
* MERN --- For its web version

**Tools**

* VS Code --- IDE for coding
* Browser inspection window tool for little design manipulations testing
* Npm --- Node JS
* MongoDB Compass --- Database purposes
* JWT authentication
* Heroku --- For deployment
* Android emulators for testing and running
* GitHub Desktop for Version control system

# First Deliverable Guide

# Introduction

## Project Feasibility Report

In this project, we are developing an online mobile application which provides the functionality for tutors to register themselves and provide their qualification details so they can be hired for teaching the students. On the other hand our application provides the functionality for students to search tutors, can take interview from tutors, hire them and give feedback about them.

Feasibilities features are given below:

* Technical Feasibility
* Operational Feasibility
* Economic Feasibility
* Schedule Feasibility
* Specification Feasibility
* Informational Feasibility
* Motivational Feasibility
* Legal and Ethical Feasibility

### Technical Feasibility

Tools and technologies that are required for this project is easily available and free of cost also. Therefore, there will be no issue for us regarding the required technology. Technologies that we’ll be using are HTML, CSS, JS, JSX, React Native and MERN and the main tool we’ll be using is VS Code. And with that we’ll use Npm for node JS, MongoDB Compass for database purposes, Heroku for development, JWT authentication, Android emulators and GitHub Desktop for version control system. All of these are easily available. However, there are few chances to face hardware issues when our application will be run on user’s mobile. If their mobile’s version is too old then they might face certain hardware limitations.

### Operational Feasibility

Our project will be developed in such a way so that it will be user friendly and anyone can easily use it. Sometimes, it is very difficult to develop a system or an application that is efficient and user friendly also, but in this project our main focus is to build a user friendly mobile application that will be operated easily.

### Economic Feasibility

Our main purpose while doing this project is learning, so for the time being, none of us is interested in any kind of financial benefit. And we don’t need to spend any money to build this project as we already have all the essential things that are required for the development of the project like hardware (laptop) and software (VS Code). However, in the future if we want our mobile application to be available globally then there will be a deployment cost.

### Schedule Feasibility

Time granted to us for this project is enough to complete this project. Although there is a work load of other subjects too but still each member of our team is completing his/her part of work by the given time and everyone is taking his/her part of responsibility properly. So under these circumstances, we will complete our project within the given time.

### Specification Feasibility

As we are developing our own project, so we are sure about the detailed requirements or specifications that are required in this project and we’ll surely fulfill them.

### Informational Feasibility

All information provided is feasible to complete this project smoothly. All information provided is reliable and all details help us for the better performance of this project. If any problem occurs during the completion of this project, we will definitely overcome it.

### Motivational Feasibility

Our team leader and all the team members are very supportive. They do their best and help each other in making the project. Hopefully our leader will motivates and backups all the members well. Motivational feasibility is and will available all the time for this project.

### Legal and Ethical Feasibility

The main aim of our project is to provide an online mobile application in which a tutor can register himself and a student can find the nearby tutor online. This project will follow all the legal and ethical rules. No such thing (such as any technology or interface or any type of thread) is included in our project which is unethical or illegal. There will be no negative impact on any community or user by this project. This project is feasible and helpful in all aspects. It will run smoothly and it will be beneficial for all of its users.

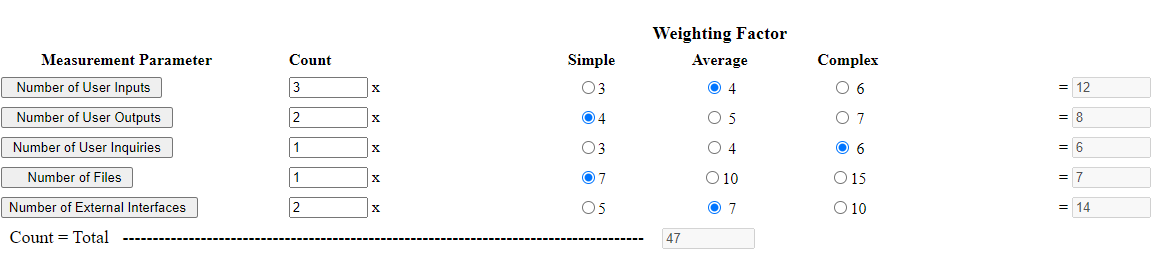
## Project/Product Scope

*Scope of projects just covers the following functional units:*

* Sign Up
* Login/Logout
* Register as a tutor
* Register as a student
* Search for available tutors
* View tutor details
* Choose a tutor
* Communicate with tutor
* Hire a tutor
* Give feedback and review about the hired tutor

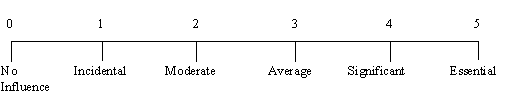
## Project/Product Costing

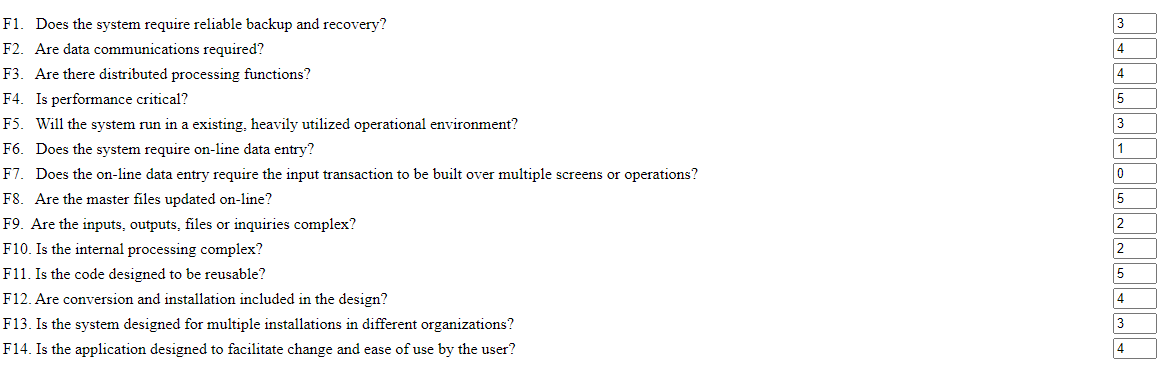
### Project Cost Estimation by Function Point Analysis



So, Count Total is **47**

Rate each factor (Fi, i=1 to 14) on a scale of 0 to 5





So, sum Fi is **45**

**FP est. = Count Total \* [0.65 + 0.01 \* (Sum Fi)]**

= 47 \* [0.65 + 0.01 \* (45)] = 47 \* [0.65 + 0.45]

= 47 \* [1.1]

= 51.7

Labor rate = Rs 2000/month, Productivity = 7FP/pm

**Cost/FP = Labor rate / productivity**

= 2000/7

= **Rs 285.7/FP**

**Total Project Cost = Estimated FP \* Cost/FP**

= 51.7 \* 285.7

= **Rs 14,770.7**

**Estimated Effort = Estimated FP / Productivity**

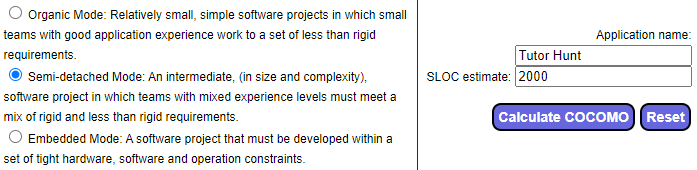
= 51.7/7

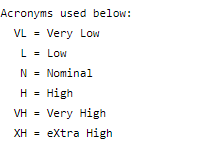
= **7PM**

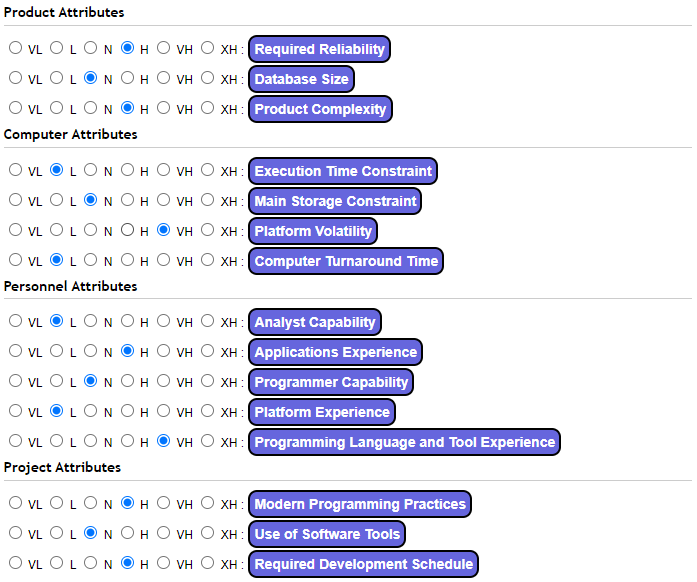
**Conclusion**

7 person per month working with the productivity of 7 FP per person per month is required to complete the above project with in duration of 130 days.

### Project Cost Estimation by using COCOMO’81 (Constructive Cost Model)







|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **Product Attributes** | | | Required Reliability | 1.15 (H ) | | Database Size | 1.00 (N ) | | Product Complexity | 1.15 (H ) | |  |  | | **Computer Attributes** | | |  |  | | Execution Time Constraint | 1.00 (L ) | | Main Storage Constraint | 1.00 (N ) | | Platform Volatility | 1.30 (VH) | | Computer Turnaround Time | 0.87 (L ) | |  |  | | **Personnel Attributes** | | |  |  | | Analyst Capability | 1.19 (L ) | | Applications Experience | 0.91 (H ) | | Programmer Capability | 1.00 (N ) | | Platform Experience | 1.10 (L ) | | Programming Language and Tool Experience | 0.95 (VH) | |  |  | | **Project Attributes** | | |  |  | | Modern Programming Practices | 0.91 (H ) | | Use of Software Tools | 1.00 (N ) | | Required Development Schedule | 1.04 (H ) | |

For semi-detached, initial effort is:

Ei = 3.0 (KLOC) 1.12

= 3.0 (2.0) 1.12

= 3.0 (2.1735)

= **6.5205**

After taking the average of the Adjustments factors

|  |  |
| --- | --- |
| **Product Attributes** | 1.1 |
| **Computer Attributes** | 1.0 |
| **Personal Attributes** | 1.03 |
| **Project Attributes** | 0.9 |

∑ EAF = 4.03

(Summation of Effort Adjustment Factor)

The final effort E is obtained as **E = ∑ EAF \* Ei**

Putting values

E = 4.03 \* 6.5205

**Effort = 26.27**

So for people required

Duration of project is set to = 130 days = 4.3 months

**People required = Effort / Duration**

= 26.27/4.3

= 6 PM

**Conclusion**

6 person per month is required to finish in 130 days.

## CPM – Critical Path Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Alias** | **Activity** | **Predecessors** | **Duration**  **(in days)** |
| **A** | **GATHERING\_REQUIREMENT** | - | 10 |
| **B** | **PREPARING\_DELIVERABLES** | A | 17 |
| **C** | **DEVELOPING\_FRONTEND** | A | 21 |
| **D** | **DEVELOPING\_BACKEND** | A, B, C | 70 |
| **E** | **TESTING\_APP** | C, D | 10 |
| **F** | **DEPLOYING\_APP** | D, E | 2 |

**Node Presentation**

|  |  |  |
| --- | --- | --- |
| **Early Start** | **Duration** | **Early Finish** |
| **Activity Alias** | | |
| **Late Start** | **Slack** | **Late Finish** |

**Activity on Node Diagram**

|  |  |  |
| --- | --- | --- |
| **10** | **17** | **27** |
| **B** | | |
| **14** | **14** | **31** |

|  |  |  |
| --- | --- | --- |
| **0** | **10** | **10** |
| **A** | | |
| **0** | **0** | **10** |

|  |  |  |
| --- | --- | --- |
| **31** | **70** | **101** |
| **D** | | |
| **31** | **0** | **101** |

|  |  |  |
| --- | --- | --- |
| **111** | **2** | **113** |
| **F** | | |
| **111** | **0** | **113** |

|  |  |  |
| --- | --- | --- |
| **10** | **21** | **31** |
| **C** | | |
| **10** | **0** | **31** |

|  |  |  |
| --- | --- | --- |
| **101** | **10** | **111** |
| **E** | | |
| **101** | **0** | **111** |

**Conclusion**

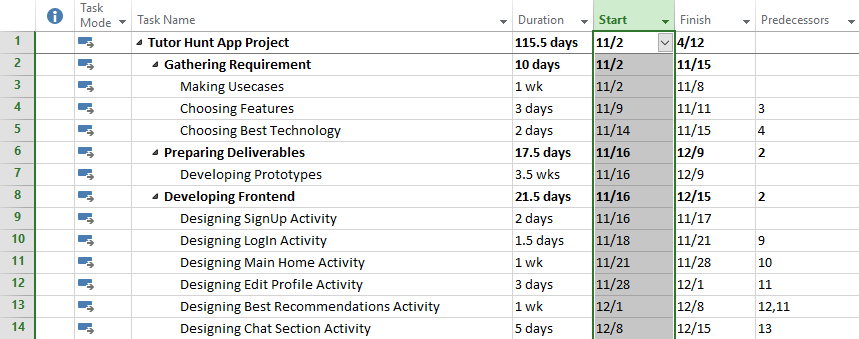
**A 🡪 C 🡪 D 🡪 F**

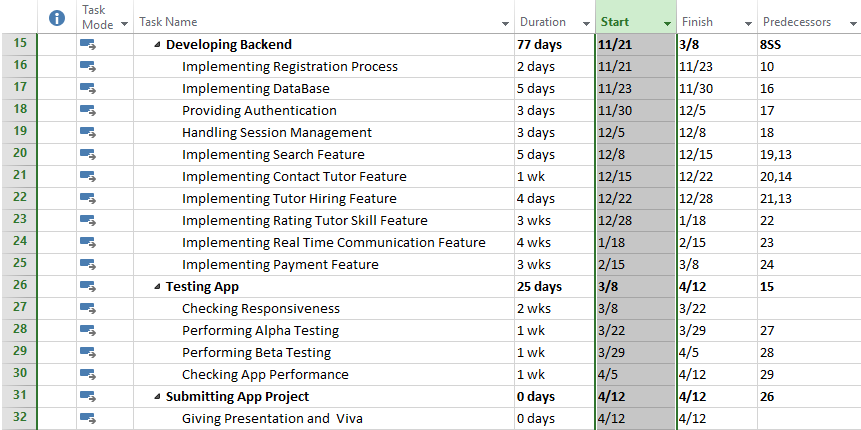
CRITICAL PATHS

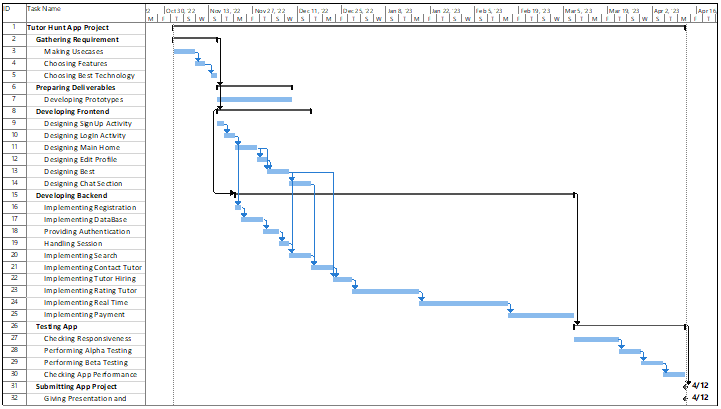
**A 🡪 D 🡪 F**

**A 🡪 C 🡪 E 🡪 F**

## Gantt chart







## Introduction to Team members and their skill set

|  |  |  |
| --- | --- | --- |
| **Team member Name** | **Roll Number** | **Skills** |
| **Mahnoor Shoukat** | **BSEF19A502** | Front-end development, Communication skills, Basic database skills |
| **Iqra Junaid** | **BSEF19A511** | Web + Mobile development skills, Time management, Basic database skills |
| **Hassan Zahid** | **BSEF19A513** | Mobile development, Web development, Database skills |
| **Asim Afzal** | **BSEF19A542** | Mobile dev, Web dev, Enterprise App dev, VCS |

## Tools and Technology with reasoning

**Technologies**

* HTML / JSX for general UI purposes
* CSS for designing purposes
* JS for some dynamic operations
* React Native --- Main Tech Stack for our tutor mobile App
* MERN --- For its web version

**Tools**

* VS Code --- IDE for coding
* Browser inspection window tool for little design manipulations testing
* Npm --- Node JS
* MongoDB Compass --- Database purposes
* JWT authentication
* Heroku --- For deployment
* Android emulators for testing and running
* GitHub Desktop for Version control system

## Vision Document

Main goal of our project is to provide an online application for students to find there nearby tutor online easily and for tutors to register themselves and make themselves easily accessible by the students. Our aim is to provide an application where a student can communicate online with a tutor. It will be user-friendly software for ease of everyone. This system will provide the facility of creating a student account and tutor account. Tutors will create their profile and add their qualification and other information in their profile. They can also edit their profile. Students can search their nearby tutor, they can take online interview from the tutor, they can also request an online demo lecture, they can hire the tutor and they can give feedback about the hired tutor. We will provide all the functionality of manual searching of a tutor in our online application, to make it fast and useful. Manually searching a tutor or student is a hectic and time-consuming task for both, tutor and students. Therefore, we are developing this application to overcome these issues.

*In our project, stakeholders are admin, user (Tutor, Student), designer, developer and tester.*

* Admin needs an interface to perform the following tasks:
* Admin is one, who will approve/reject the user’s request to register as a tutor by verifying his/her qualification.
* User needs an interface to perform the following tasks:
* User (tutor, student) can request to create/delete account, login/log-out from existing account, register as a student or a tutor, tutors can post their availability to teach and students can search and hire a tutor.
* Designer has too map /design all this requirements using some UML diagrams and design patterns so that all the admin and user (actors) requirements (scenarios) are covered in that.
* Developer has to provide functioning code that will be working exactly the same which is required by the admin and the user. Proper functioning should be tested by the tester.
* Tester has to test all these functional requirements to be tested using requirement traceability matrix.

**Technical Requirements:**

* For developing the project and going through all the phases , we should have the developers that are expert in MERN Stack and react native , designer who is expert to cover the required features graphically and design the project so decently so that no complex regarding understand the project remains there.
* Apache tomcat server installed and configured on the system where we are going to develop the project.
* STARUML should be installed and developers should have grip to use it for designing purposes.

**Scope of the project:**

*Scope of projects just covers the following functional units:*

* Sign Up
* Login/Logout
* Register as a tutor
* Register as a student
* Search for available tutors
* View tutor details
* Choose a tutor
* Communicate with tutor
* Hire a tutor
* Give feedback and review about the hired tutor

## Risk List

Risks that can be occur during this software development are:

* Time constraint due to university work load
* A risk to meet user’s satisfaction as users have different requirements that are not feasible to implement all in one.
* Difficulty in learning new technologies.

## Requirement Engineering

Requirements engineering process provides the appropriate mechanism for understanding what the customer wants, analyzing need, assessing feasibility, negotiating a reasonable solution, specifying the solution unambiguously, validating the specification and managing the requirements as they are transformed into an operational system. The task of capturing, structuring, and accurately representing the user's requirements. Requirements specification would lead to the following steps:

### System Specifications

Main purpose of this application is to provide an ease to users to find tutor online and gain knowledge more easily. It will be a user friendly application that can be used by everyone easily. Its major advantage is that user/student can find tutor from their home, they don’t need to find tutors manually. It is a mobile based application so users/students don’t need any special hardware devices because now a days everyone has mobile phone.

**Existing Systems**

Existing techniques and tools for telling everyone about tutors are publishing any ad or article in newspapers or posters. And to find tutor there is no any simple way. This application will serve a lot of people and help them to find tutors.

**Organizational Chart**

Organizational chart will be very much supportive to get a better overview of the organization’s business areas and their decomposition into different departments/authorities.

Student Tutor



Student Group

Student

Student

Student

Student

Student

### Identifying External Entities

External Entities are those entities which interact with and use the system from outside. Following are external entities which interact with this system:

1. **Over Specify Entities from Abstract**

|  |  |
| --- | --- |
| * Parents | * Students |
| * Tutors | * School Management |

1. **Perform Refinement**

* Tutors
* Students

### Context Level Data Flow Diagram

Context level data flow diagram contains only one process, representing the entire system. The process is given the number zero and all external entities are shown on the context diagram as well as major data flow to and from them. The diagram does not contain any data stores.

**Tutors**

**Students**



### Capture "shall" Statements

“Shall” statements are all the functional requirements, which are:

|  |  |  |
| --- | --- | --- |
| **Para#** | **External Entity** | **Initial Requirements** |
| 1.0 | Tutor | A tutor “shall” first create an account. |
| 1.0 | Tutor | A tutor “shall” login if he/she wants to access his/her account. |
| 1.0 | Tutor | A tutor “shall” enter appropriate data for his/her profile details. |
| 1.0 | Tutor | A tutor “shall” view profile details. |
| 1.0 | Tutor | A tutor “shall” edit his/her profile. |
| 1.0 | Tutor | A tutor “shall” contact with students. |
| 1.0 | Tutor | A tutor “shall” video call students. |
| 1.0 | Tutor | A tutor “shall” logout. |
| 2.0 | Student | Student “shall” login to his/her account. |
| 2.0 | Student | Student “shall” search for tutor. |
| 2.0 | Student | Student “shall” view tutor’s profile details. |
| 2.0 | Student | Student “shall” his/her profile details. |
| 2.0 | Student | Student “shall” contact to tutor through messages. |
| 2.0 | Student | Student “shall” contact to tutor through video call. |
| 2.0 | Student | Student “shall” rate tutor. |
| 2.0 | Student | Student “shall” logout. |

### Allocate Requirements

Allocation of the requirements in the use-cases are:

|  |  |  |  |
| --- | --- | --- | --- |
| **Para#** | **External Entity** | **Initial Requirements** | **Use Case Name** |
| 1.0 | Tutor | A tutor “shall” first create an account. | UC\_create\_account |
| 1.0 | Tutor | A tutor “shall” login if he/she wants to access his/her account. | UC\_login\_T |
| 1.0 | Tutor | A tutor “shall” enter appropriate data for his/her profile details. | UC\_enter\_data |
| 1.0 | Tutor | A tutor “shall” view profile details. | UC\_view\_profile |
| 1.0 | Tutor | A tutor “shall” edit his/her profile. | UC\_edit\_profile\_T |
| 1.0 | Tutor | A tutor “shall” contact with students. | UC\_contact\_student |
| 1.0 | Tutor | A tutor “shall” video call students. | UC\_video\_call |
| 1.0 | Tutor | A tutor “shall” logout. | UC\_logout\_T |
| 2.0 | Student | Student “shall” login to his/her account. | UC\_login\_S |
| 2.0 | Student | Student “shall” search for tutor. | UC\_search\_tutor |
| 2.0 | Student | Student “shall” view tutor’s profile details. | UC\_view\_tutor\_profile |
| 2.0 | Student | Student “shall” edit his/her profile details. | UC\_edit\_profile\_S |
| 2.0 | Student | Student “shall” contact to tutor through messages. | UC\_message\_tutor |
| 2.0 | Student | Student “shall” contact to tutor through video call. | UC\_video\_call |
| 2.0 | Student | Student “shall” rate tutor. | UC\_rate\_tutor |
| 2.0 | Student | Student “shall” logout. | UC\_logout\_S |

### Prioritize Requirements

Requirements prioritization helps in achieving tasks easily. Ranks are “highest, medium and lowest”.

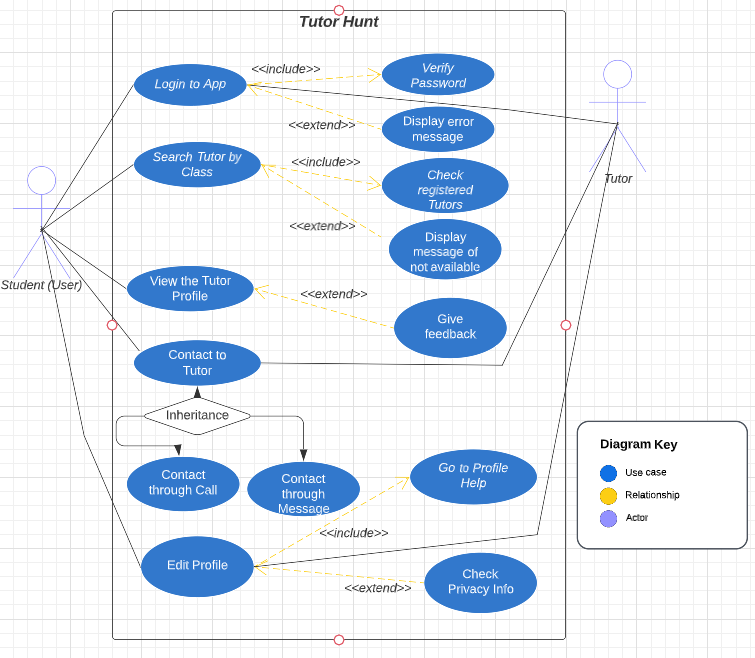
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Para#** | **Rank** | **Initial Requirements** | **Use Case ID** | **Use Case Name** |
| 1.0 | Highest | A tutor “shall” first create an account. | UC\_01 | UC\_create\_account |
| 1.0 | Highest | A tutor “shall” login if he/she wants to access his/her account. | UC\_02 | UC\_login\_T |
| 1.0 | Medium | A tutor “shall” enter appropriate data for his/her profile details. | UC\_03 | UC\_enter\_data |
| 1.0 | Lowest | A tutor “shall” view profile details. | UC\_04 | UC\_view\_profile |
| 1.0 | Lowest | A tutor “shall” edit his/her profile. | UC\_05 | UC\_edit\_profile\_T |
| 1.0 | Medium | A tutor “shall” contact with students. | UC\_06 | UC\_contact\_student |
| 1.0 | Lowest | A tutor “shall” video call students. | UC\_07 | UC\_video\_call |
| 1.0 | Highest | A tutor “shall” logout. | UC\_08 | UC\_logout\_T |
| 2.0 | Highest | Student “shall” login to his/her account. | UC\_09 | UC\_login\_S |
| 2.0 | Highest | Student “shall” search for tutor. | UC\_10 | UC\_search\_tutor |
| 2.0 | Highest | Student “shall” view tutor’s profile details. | UC\_11 | UC\_view\_tutor\_profile |
| 2.0 | Lowest | Student “shall” edit his/her profile details. | UC\_12 | UC\_edit\_profile\_S |
| 2.0 | Medium | Student “shall” contact to tutor through messages. | UC\_13 | UC\_message\_tutor |
| 2.0 | Lowest | Student “shall” contact to tutor through video call. | UC\_14 | UC\_video\_call |
| 2.0 | Lowest | Student “shall” rate tutor. | UC\_15 | UC\_rate\_tutor |
| 2.0 | Highest | Student “shall” logout. | UC\_16 | UC\_logout\_S |

### Requirements Trace-ability Matrix

The requirements trace-ability matrix is a table used to trace project life cycle activities and work products to the project requirements. The matrix establishes a thread that traces requirements from identification through implementation.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sr#** | **Para#** | **System Specification Text** | **Build** | **Use Case Name** | **Category** |
| 1 | 1.0 | A tutor “shall” first create an account. | B1 | UC\_create\_account | Business |
| 2 | 1.0 | A tutor “shall” login if he/she wants to access his/her account. | B2 | UC\_login\_T | Business |
| 3 | 1.0 | A tutor “shall” enter appropriate data for his/her profile details. | B3 | UC\_enter\_data | Business |
| 4 | 1.0 | A tutor “shall” view profile details. | B4 | UC\_view\_profile | Business |
| 5 | 1.0 | A tutor “shall” edit his/her profile. | B5 | UC\_edit\_profile\_T | Business |
| 6 | 1.0 | A tutor “shall” contact with students. | B6 | UC\_contact\_student | Business |
| 7 | 1.0 | A tutor “shall” video call students. | B7 | UC\_video\_call | Business |
| 8 | 1.0 | A tutor “shall” logout. | B8 | UC\_logout\_T | Business |
| 9 | 2.0 | Student “shall” login to his/her account. | B9 | UC\_login\_S | Business |
| 10 | 2.0 | Student “shall” search for tutor. | B10 | UC\_search\_tutor | Business |
| 11 | 2.0 | Student “shall” view tutor’s profile details. | B11 | UC\_view\_tutor\_profile | Business |
| 12 | 2.0 | Student “shall” edit his/her profile details. | B12 | UC\_edit\_profile\_S | Business |
| 13 | 2.0 | Student “shall” contact to tutor through messages. | B13 | UC\_message\_tutor | Business |
| 14 | 2.0 | Student “shall” contact to tutor through video call. | B14 | UC\_video\_call | Business |
| 15 | 2.0 | Student “shall” rate tutor. | B15 | UC\_rate\_tutor | Business |
| 16 | 2.0 | Student “shall” logout. | B16 | UC\_logout\_S | Business |

### High Level Use-case Diagram



# Second Deliverable Guide

# Introduction

Third deliverable is all about the use case modeling and software design. In the previous deliverables, analysis of the system is completed. So we understand the current situation of the problem domain. Now we are ready to strive for a solution for the problem domain by using object-oriented approach. Following artifacts must be included in this deliverable.

1. Use case description
2. Use case diagram refined
3. Domain Model
4. Sequence Diagram
5. Collaboration Diagram
6. Operation Contracts
7. Design Class Diagram
8. Data Model

Now we discuss these artifacts one by one as follows:

## Use Case Description

**Use case # 1**

|  |  |
| --- | --- |
| **Use Case Name** | UC\_Login |
| **Actors** | Student, Tutor |
| **Brief Description** | Users can login |
| **Pre-Condition** | User must be registered |
| **Post-Condition** | Access should be granted |

**Typical course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| User enter credentials | Redirect to main page |

**Alternate course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| User enter credentials | All fields of credentials are not filled |

**Use case # 2**

|  |  |
| --- | --- |
| **Use Case Name** | UC\_Search\_Tutor |
| **Actors** | Students |
| **Brief Description** | Students can search tutors |
| **Pre-Condition** | Register as a student |
| **Post-Condition** | View or choose tutor |

**Typical course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| Student enter his/her class | Tutors of corresponding class appear |

**Alternate course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| Student enter his/her class | No Tutor found |

**Use case # 3**

|  |  |
| --- | --- |
| **Use Case Name** | UC\_View\_Tutor\_Profile |
| **Actors** | Students, Tutor |
| **Brief Description** | Display complete information of tutor’s profile |
| **Pre-Condition** | Search Tutor |
| **Post-Condition** | See tutor’s information |

**Typical course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| Click on tutor’s profile | Tutor’s profile info displayed |

**Alternate course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| Click on tutor’s profile | Tutor’s profile info not loaded |

**Use case # 4**

|  |  |
| --- | --- |
| **Use Case Name** | UC\_Contact\_Tutor |
| **Actors** | Students |
| **Brief Description** | Students chat with tutor |
| **Pre-Condition** | Click on chat button |
| **Post-Condition** | Choose mode of contact i.e. call or text |

**Typical course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| Student click on chat button | Chat box opened |

**Alternate course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| Student click on chat button | Chat disabled |

**Use case # 5**

|  |  |
| --- | --- |
| **Use Case Name** | UC\_Call\_Tutor |
| **Actors** | Students |
| **Brief Description** | Contact with tutor through call |
| **Pre-Condition** | Click on contact tutor |
| **Post-Condition** | Talk with tutor |

**Typical course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| Student called tutor | Call connected |

**Alternate course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| Student called tutor | Call rejected |

**Use case # 6**

|  |  |
| --- | --- |
| **Use Case Name** | UC\_Message\_Tutor |
| **Actors** | Student |
| **Brief Description** | Contact with tutor through message |
| **Pre-Condition** | Click on contact tutor |
| **Post-Condition** | Chat with tutor |

**Typical course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| Student messaged Tutor | Message delivered |

**Alternate course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| Student messaged Tutor | Message sending failed |

**Use case # 7**

|  |  |
| --- | --- |
| **Use Case Name** | UC\_Edit\_Profile |
| **Actors** | Student, Tutor |
| **Brief Description** | Students and Tutors can edit their profiles |
| **Pre-Condition** | Register as a student or Tutor |
| **Post-Condition** | Review and update the profile |

**Typical course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| User click on edit profile | Prompt user to edit profile information |

**Alternate course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| User click on edit profile | User haven’t registered as a student or tutor |

**Use case # 8**

|  |  |
| --- | --- |
| **Use Case Name** | UC\_Check\_Privacy |
| **Actors** | Student, Tutor |
| **Brief Description** | Users can set who can see their profile info |
| **Pre-Condition** | Register as a student or Tutor |
| **Post-Condition** | Set and save the updated privacy |

**Typical course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| User click on privacy | Profile privacy displayed |

**Alternate course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| User click on privacy | Privacy not set displayed |

**Use case # 9**

|  |  |
| --- | --- |
| **Use Case Name** | UC\_Verify\_Password |
| **Actors** | Users (Student, Tutor) |
| **Brief Description** | Verify that the password is correct or not |
| **Pre-Condition** | Enter Password |
| **Post-Condition** | Login successfully |

**Typical course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| User enters password | Password verified and login successfully |

**Alternate course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| User enters password | Password unverified and login failed |

**Use case # 10**

|  |  |
| --- | --- |
| **Use Case Name** | UC\_Error\_Message |
| **Actors** | Users (Student, Tutor) |
| **Brief Description** | Wrong Credentials error appear |
| **Pre-Condition** | Enter credentials |
| **Post-Condition** | Re-enter correct credentials |

**Typical course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| User Enter Wrong Login Credentials | Username not found error displayed |

**Alternate course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| User Enter Wrong Login Credentials | Incorrect password error displayed |

**Use case # 11**

|  |  |
| --- | --- |
| **Use Case Name** | UC\_Check\_Registered\_Tutors |
| **Actors** | Student |
| **Brief Description** | Check all the registered tutors |
| **Pre-Condition** | Login as student |
| **Post-Condition** | Choose the desired tutor |

**Typical course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| User search for tutors | List of registered tutors appear |

**Alternate course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| User search for tutors | Tutor of specific class not found |

**Use case # 12**

|  |  |
| --- | --- |
| **Use Case Name** | UC\_Tutor\_Unavailable |
| **Actors** | Student |
| **Brief Description** | Display a message if tutor is unavailable |
| **Pre-Condition** | Login as student |
| **Post-Condition** | Try to search with different keywords |

**Typical course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| User search for tutors | Tutor not available message appears |

**Alternate course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| User search for tutors | List of available tutors appear |

**Use case # 13**

|  |  |
| --- | --- |
| **Use Case Name** | UC\_Give\_Feedback |
| **Actors** | Student |
| **Brief Description** | Give feedback to hired tutor |
| **Pre-Condition** | Hire a tutor |
| **Post-Condition** | Everyone can view the feedback |

**Typical course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| Student click on feedback | Text area to give feedback appear |

**Alternate course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| Tutor click on feedback | List of Tutor’s feedback appear |

**Use case # 14**

|  |  |
| --- | --- |
| **Use Case Name** | UC\_Profile\_Help |
| **Actors** | Student, Tutor |
| **Brief Description** | New user can seek help to use the app |
| **Pre-Condition** | User must be registered |
| **Post-Condition** | User achieved the desired help |

**Typical course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| User asks for help | User successfully did his task with the help |

**Alternate course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| User asks for help | User didn’t get the help |

**Use case # 15**

|  |  |
| --- | --- |
| **Use Case Name** | UC\_Make\_Payment |
| **Actors** | Student, Tutor, Bank |
| **Brief Description** | Make payment online |
| **Pre-Condition** | Have bank account, Connect bank account with app |
| **Post-Condition** | Verify transferred payment |

**Typical course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| Student click on make payment | Payment info taken and make payment successfully |

**Alternate course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| Student click on make payment | Payment info taken but payment failed due to insufficient balance |

**Use case # 16**

|  |  |
| --- | --- |
| **Use Case Name** | UC\_Checking\_Account\_Payment |
| **Actors** | Student, Tutor, Bank |
| **Brief Description** | Make payment through checking account |
| **Pre-Condition** | Connect checking bank account with app |
| **Post-Condition** | Verify payments |

**Typical course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| Student select checking account payment method | Payment info taken and make payment successfully |

**Alternate course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| Student select checking account payment method | Payment info taken but payment failed due to insufficient balance |

**Use case # 17**

|  |  |
| --- | --- |
| **Use Case Name** | UC\_Saving\_Account\_Payment |
| **Actors** | Student, Tutor, Bank |
| **Brief Description** | Make payment through saving account |
| **Pre-Condition** | Connect saving bank account with app |
| **Post-Condition** | Verify payments |

**Typical course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| Student select checking account payment method | Payment info taken and make payment successfully |

**Alternate course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| Student select checking account payment method | Payment info taken but payment failed due to insufficient balance |

**Use case # 18**

|  |  |
| --- | --- |
| **Use Case Name** | UC\_Check\_Balance |
| **Actors** | Student, Tutor, Bank |
| **Brief Description** | Check users account balance |
| **Pre-Condition** | Integrate bank account with app |
| **Post-Condition** | Logout from bank account for security |

**Typical course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| User click on check balance | User’s current bank balance displayed |

**Alternate course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| User click on check balance | Insufficient balance displayed |

**Use case # 19**

|  |  |
| --- | --- |
| **Use Case Name** | UC\_Insufficient\_Balance |
| **Actors** | Student, Tutor, Bank |
| **Brief Description** | Display insufficient balance message |
| **Pre-Condition** | Integrate bank account with app |
| **Post-Condition** | Add balance to make payments |

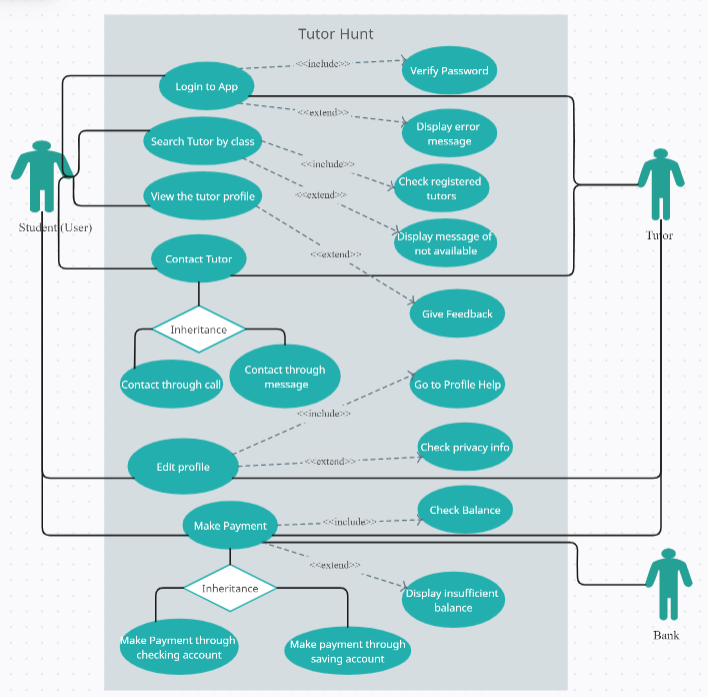
**Typical course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| User click on make online payment | Insufficient balance displayed |

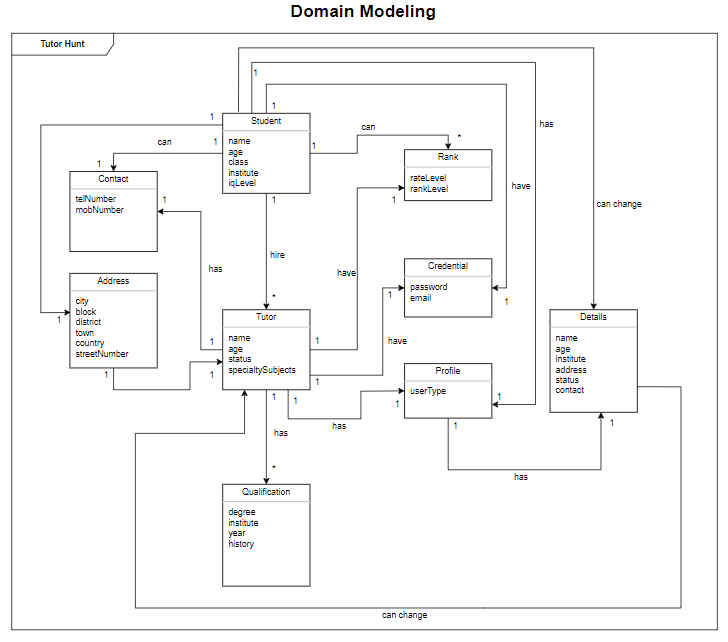
**Alternate course of Events**

|  |  |
| --- | --- |
| **Actor Actions** | **System Response** |
| User click on make online payment | Payment made successfully |

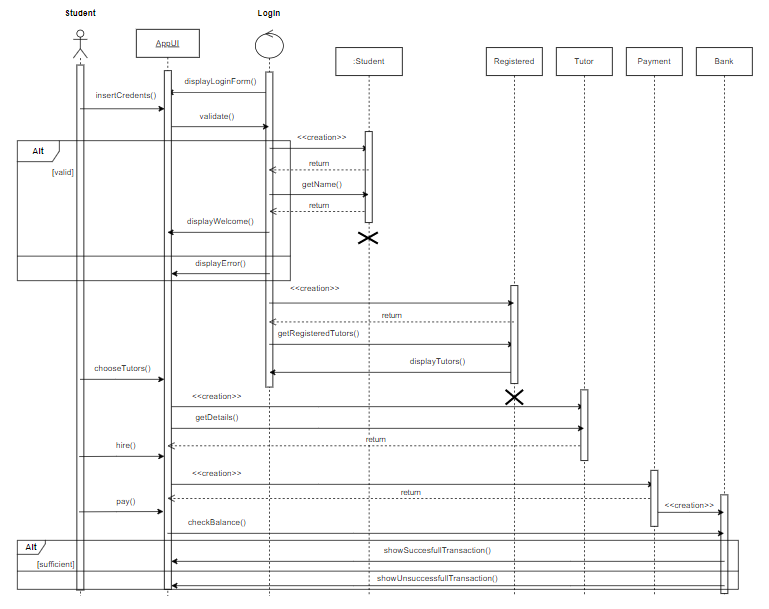
## Use Case Diagram (Refined)



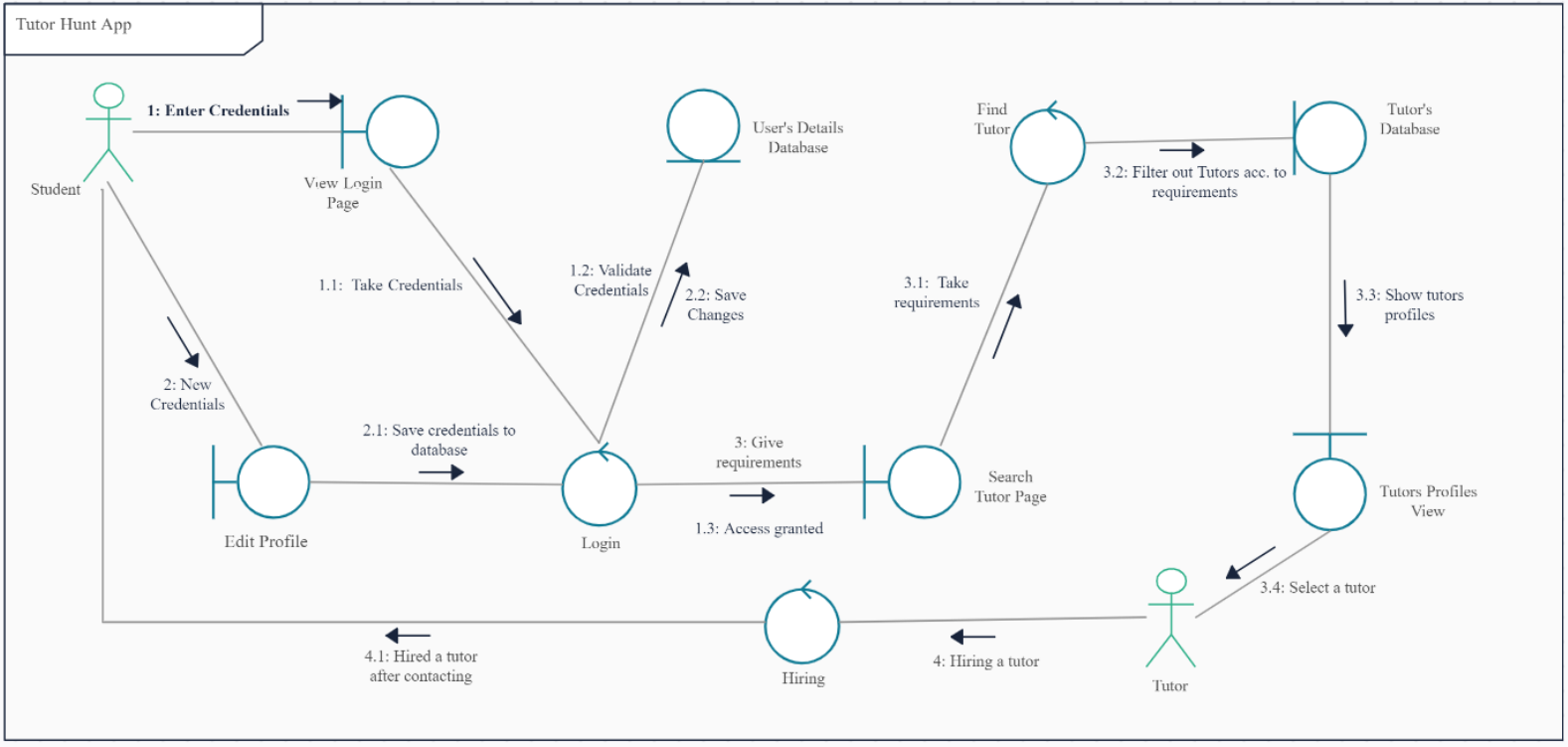
## Domain Model



## Sequence Diagram



## Collaboration Diagram



## Operation Contracts

**Contract OC 1: LOGIN**

|  |  |
| --- | --- |
| **Name** | login() |
| **Responsibilities** | User will login to the application |
| **Cross Reference** | Use Case: Login |
| **Exceptions** | User is not signed in |
| **Pre-Condition** | User should be signed in |
| **Post Condition** | User is granted access to the application |

**Contract OC 2: SEARCH\_TUTOR**

|  |  |
| --- | --- |
| **Name** | searchTutor() |
| **Responsibilities** | User will search the tutors |
| **Cross Reference** | Use Case: Search\_Tutor |
| **Exceptions** | 1: User may not be registered as a student  2: User may not find any tutor |
| **Pre-Condition** | User must be registered as a student |
| **Post Condition** | View or choose a tutor |

**Contract OC 3: VIEW\_TUTOR\_PROFILE**

|  |  |
| --- | --- |
| **Name** | viewTutorProfile() |
| **Responsibilities** | User will view the complete information of a tutor |
| **Cross Reference** | Use Case: View\_Tutor\_Profile |
| **Exceptions** | 1: User may not be registered as a student  2: User may not find any tutor |
| **Pre-Condition** | Searched a tutor |
| **Post Condition** | See the tutor’s information |

**Contract OC 4: CONTACT\_TUTOR**

|  |  |
| --- | --- |
| **Name** | contactTutor() |
| **Responsibilities** | User will contact the tutor |
| **Cross Reference** | Use Case: Contact\_Tutor |
| **Exceptions** | 1: User may not be registered as a student  2: User may not find any tutor  3: No contact of tutor is available |
| **Pre-Condition** | Clicked on contact button |
| **Post Condition** | Choose mode of contact i.e. call or text |

**Contract OC 5: CALL\_TUTOR**

|  |  |
| --- | --- |
| **Name** | callTutor() |
| **Responsibilities** | User will call the tutor |
| **Cross Reference** | Use Case: Call\_Tutor |
| **Exceptions** | Tutor will not attend the call |
| **Pre-Condition** | Clicked on contact tutor |
| **Post Condition** | Talk with tutor |

**Contract OC 6: MESSAGE\_TUTOR**

|  |  |
| --- | --- |
| **Name** | messageTutor() |
| **Responsibilities** | User will chat with the tutor |
| **Cross Reference** | Use Case: Message\_Tutor |
| **Exceptions** | Tutor will not respond to user’s message |
| **Pre-Condition** | Clicked on contact tutor |
| **Post Condition** | Chat with tutor |

**Contract OC 7: EDIT\_PROFIE**

|  |  |
| --- | --- |
| **Name** | editProfile() |
| **Responsibilities** | Users will edit their profiles |
| **Cross Reference** | Use Case: Edit\_Profile |
| **Exceptions** | User is not logged in |
| **Pre-Condition** | Registered as a Student or Tutor |
| **Post Condition** | Review and Update the Profile |

**Contract OC 8: CHECK\_PRIVACY**

|  |  |
| --- | --- |
| **Name** | checkPrivacy() |
| **Responsibilities** | User will set who can see their profile info |
| **Cross Reference** | Use Case: Check\_Privacy |
| **Exceptions** | User is not logged in |
| **Pre-Condition** | Registered as a Student or a Tutor |
| **Post Condition** | Set and Save the updated privacy |

**Contract OC 9: VERIFY\_PASSWORD**

|  |  |
| --- | --- |
| **Name** | verifyPassword() |
| **Responsibilities** | User’s password will be verified either it is correct or not |
| **Cross Reference** | Use Case: Verify\_Password |
| **Exceptions** | Password entered is wrong |
| **Pre-Condition** | Entered the Password |
| **Post Condition** | Login Successfully |

**Contract OC 10: ERROR\_MESSAGE**

|  |  |
| --- | --- |
| **Name** | errorMessage() |
| **Responsibilities** | User will face the wrong credential error message |
| **Cross Reference** | Use Case: Error\_Message |
| **Exceptions** | User may not be signed in |
| **Pre-Condition** | Entered the Credentials |
| **Post Condition** | Re-enter Correct Credentials |

**Contract OC 11: CHECK\_REGISTERED\_TUTORS**

|  |  |
| --- | --- |
| **Name** | checkRegisteredTutors() |
| **Responsibilities** | User will check all registered tutors |
| **Cross Reference** | User Case: Check\_Registered\_Tutors |
| **Exceptions** | 1: User is not logged in as student  2: User may not found any tutor  3: Tutors are not registered |
| **Pre-Condition** | Login as student |
| **Post Condition** | Choose the desired tutor |

**Contract OC 12: TUTOR\_UNAVAILABLE**

|  |  |
| --- | --- |
| **Name** | unavailableTutor() |
| **Responsibilities** | User will get a message if tutor is unavaiable |
| **Cross Reference** | User Case: Tutor\_Unavailable |
| **Exceptions** | User is logged in as student |
| **Pre-Condition** | Login as a student |
| **Post Condition** | Try to search with different keywords |

**Contract OC 13: GIVE\_FEEDBACK**

|  |  |
| --- | --- |
| **Name** | giveFeedback() |
| **Responsibilities** | User will give a feedback about the hired tutor |
| **Cross Reference** | Use Case: Give\_Feedback |
| **Exceptions** | User didn’t hire a tutor |
| **Pre-Condition** | User must hired a tutor |
| **Post Condition** | Everyone can view the feedback |

**Contract OC 14: PROFILE\_HELP**

|  |  |
| --- | --- |
| **Name** | profileHelp() |
| **Responsibilities** | New user will seek help to use the application |
| **Cross Reference** | Us Case: Profile\_Help |
| **Exceptions** | User is not logged in |
| **Pre-Condition** | User must be registered as a student or tutor |
| **Post Condition** | User achieved the desired help |

**Contract OC 15: MAKE\_PAYMENT**

|  |  |
| --- | --- |
| **Name** | makePayment() |
| **Responsibilities** | User can make payment online |
| **Cross Reference** | Use Case: Make\_Payment |
| **Exceptions** | User is not logged in |
| **Pre-Condition** | User must have a bank account connected with the app |
| **Post Condition** | Verify the transferred payment |

**Contract OC 16: CHECKING\_ACCOUNT\_PAYMENT**

|  |  |
| --- | --- |
| **Name** | checkAccountPayment() |
| **Responsibilities** | Users will check their account payments |
| **Cross Reference** | Use Case: Checking\_Account\_Payments |
| **Exceptions** | User didn’t do any transaction |
| **Pre-Condition** | User must have a bank account connected with the app |
| **Post Condition** | Verify the transferred payment |

**Contract OC 17: SAVING\_ACCOUNT\_PAYMENT**

|  |  |
| --- | --- |
| **Name** | saveAccountPayment() |
| **Responsibilities** | Users will make payments through their saving account |
| **Cross Reference** | Use Case: Saving\_Account\_Payment |
| **Exceptions** | User may not have any saving bank account |
| **Pre-Condition** | User must have a saving bank account connected with the app |
| **Post Condition** | Verify the transferred payment |

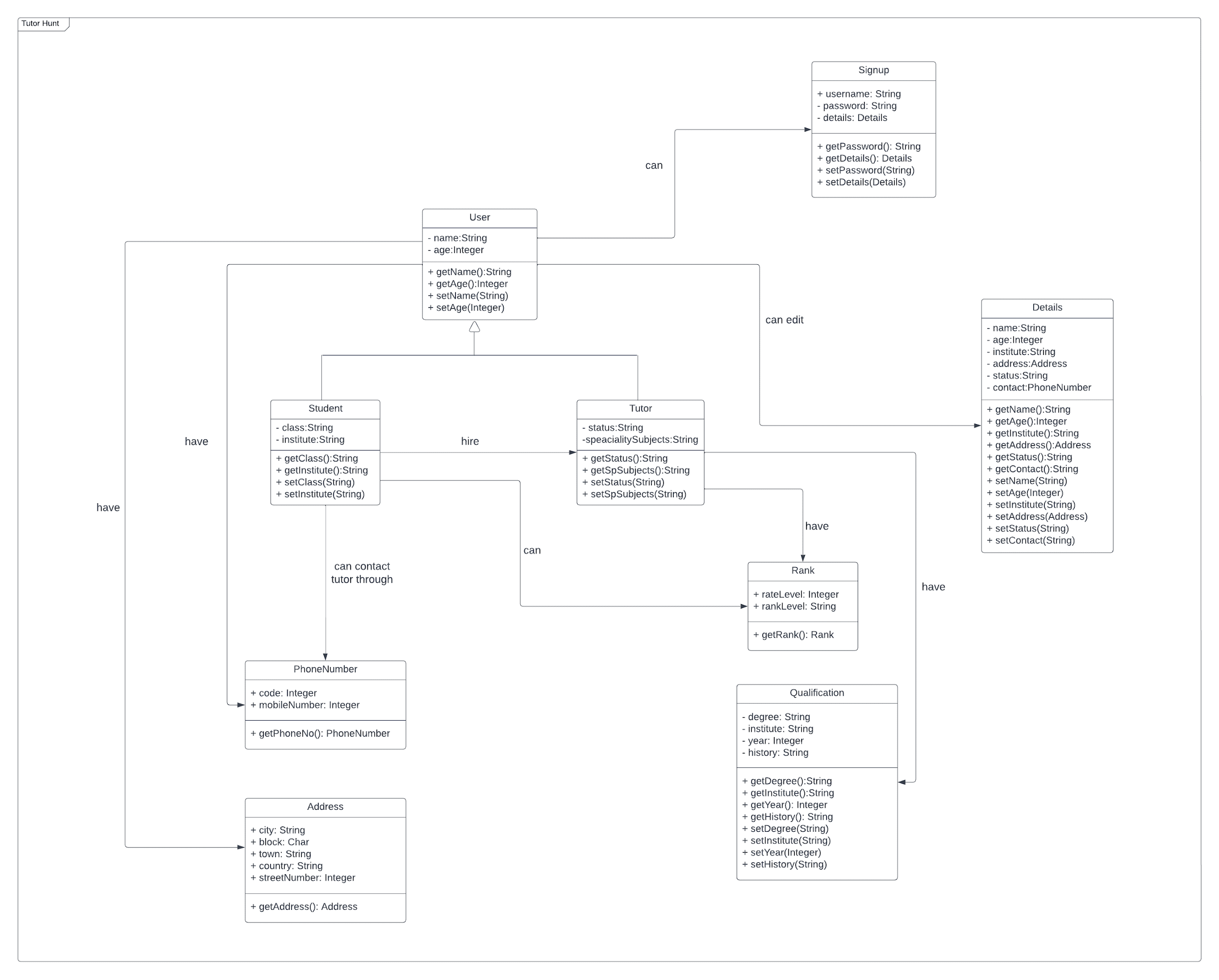
**Contract OC 18: CHECK\_BALANCE**

|  |  |
| --- | --- |
| **Name** | checkBalance() |
| **Responsibilities** | Users will check their account balance |
| **Cross Reference** | Use Case: Check\_Balance |
| **Exceptions** | User may not have a bank account |
| **Pre-Condition** | Integrate bank account with app |
| **Post Condition** | Logout from bank account for security |

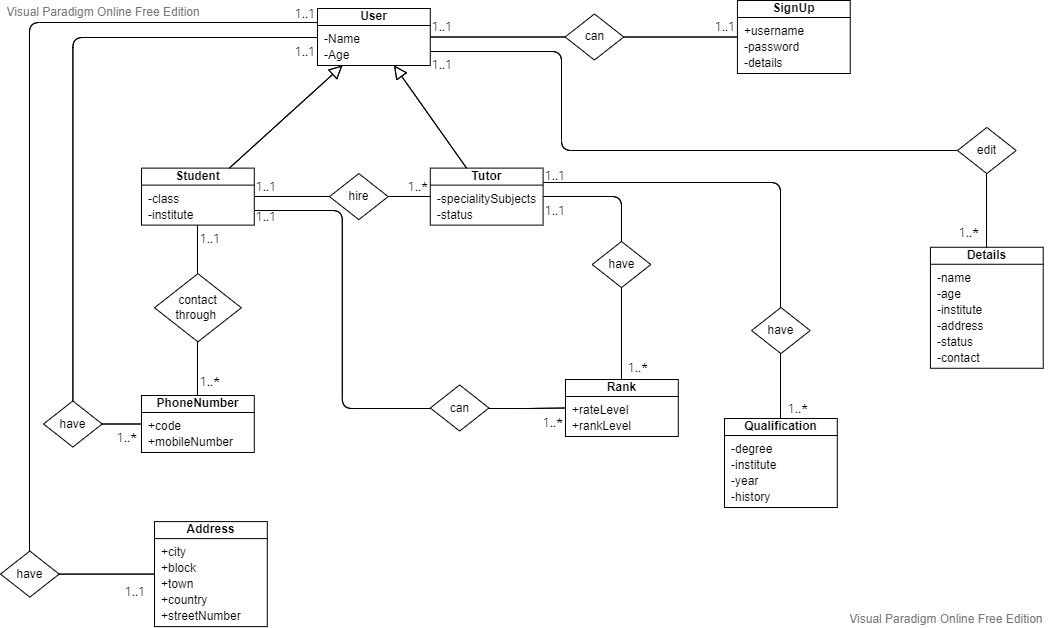
**Contract OC 19: INSUFFICIENT\_BALANCE**

|  |  |
| --- | --- |
| **Name** | insufficientBalance() |
| **Responsibilities** | User will get an insufficient balance message |
| **Cross Reference** | Use Case: Insufficient\_Balance |
| **Exceptions** | User may not have any bank account |
| **Pre-Condition** | Integrate bank account with the app |
| **Post Condition** | Add balance to make payments |

## Design Class Diagram



## Data Model

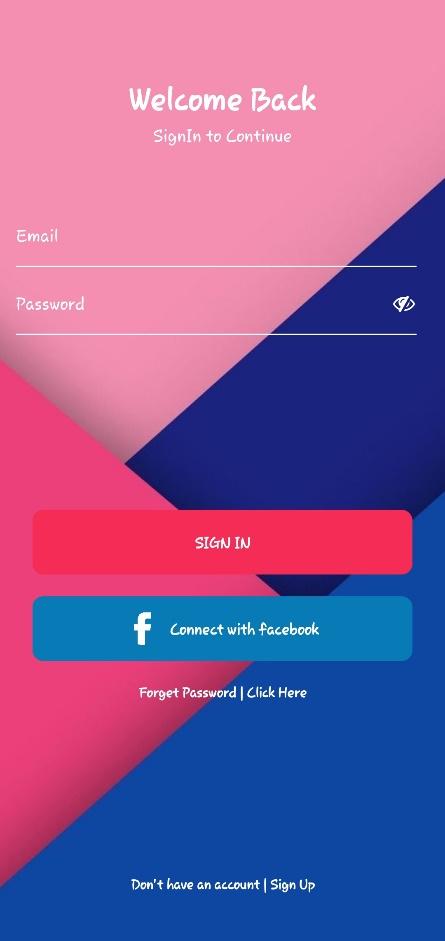


# Prototypes

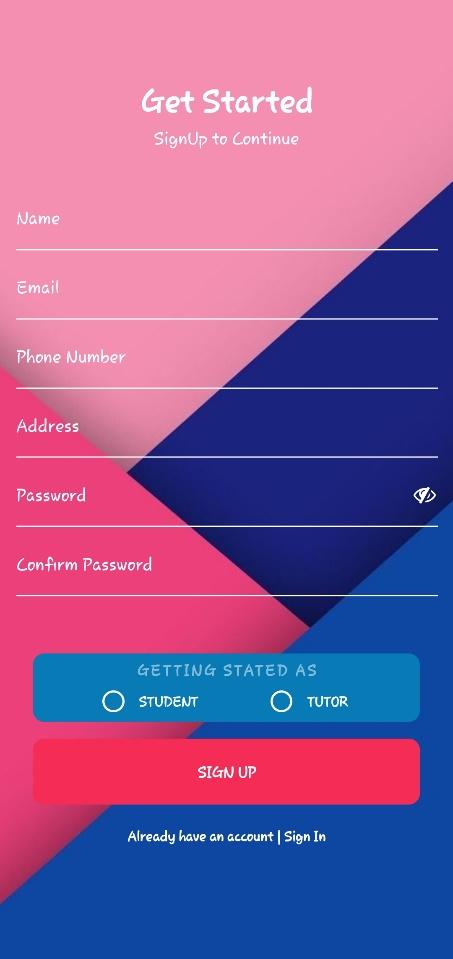
## Boarding Screens

|  |
| --- |
|  |

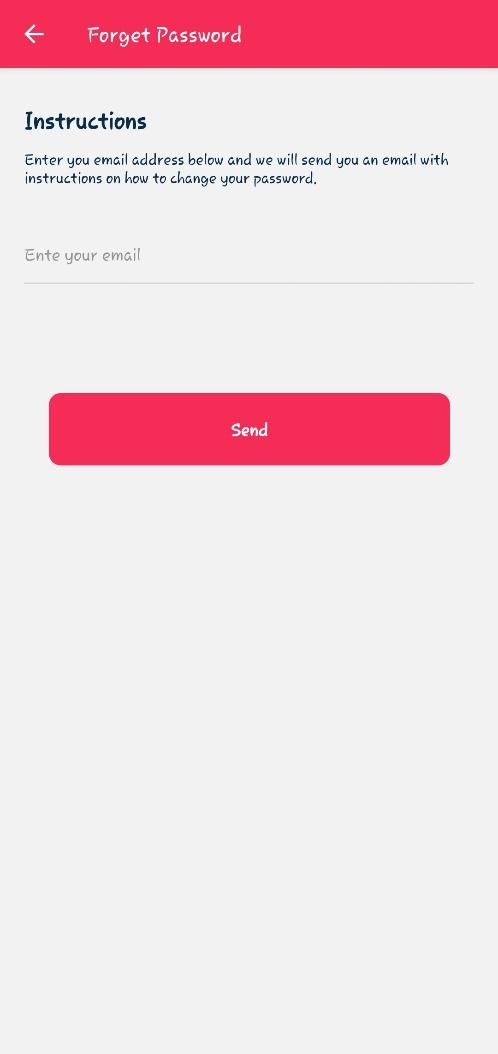
## Sign-in Screen



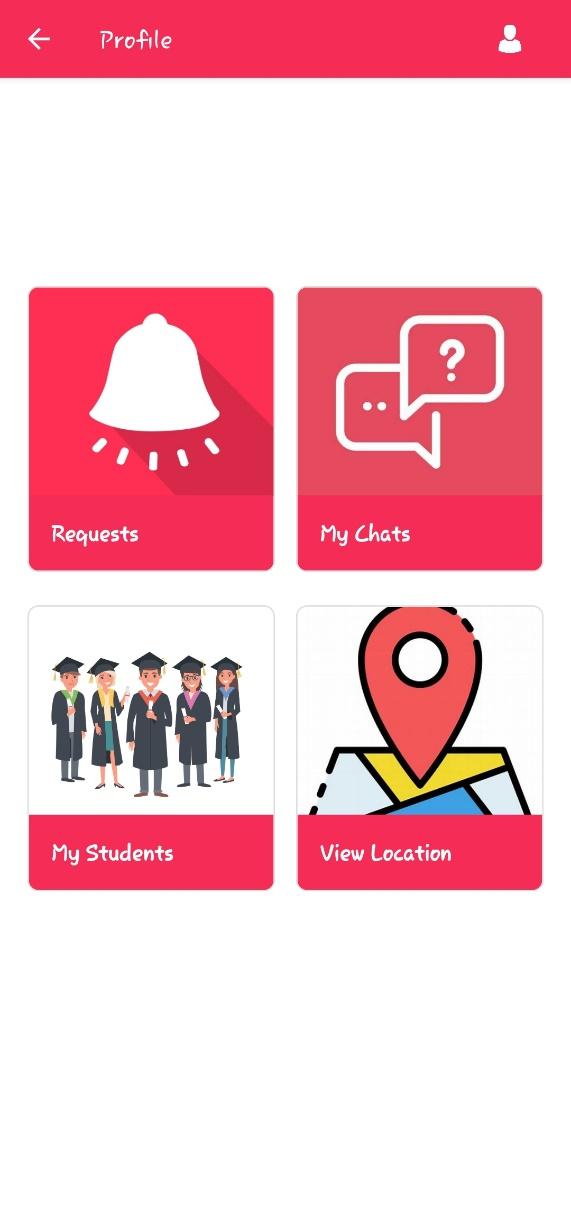
## Sign-up Screen



## Forget Password Screen



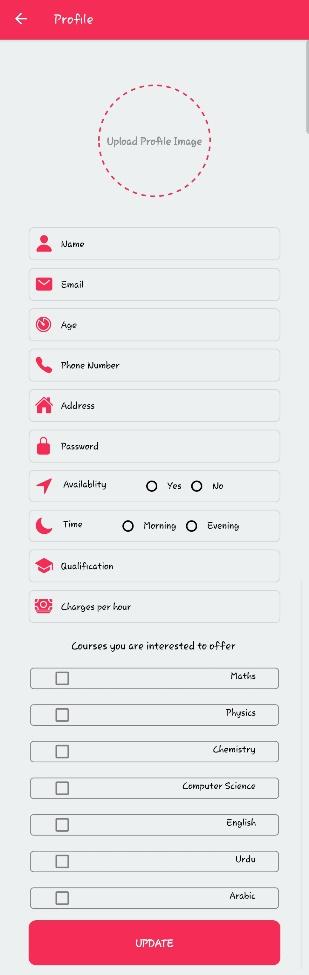
## Tutor Dashboard



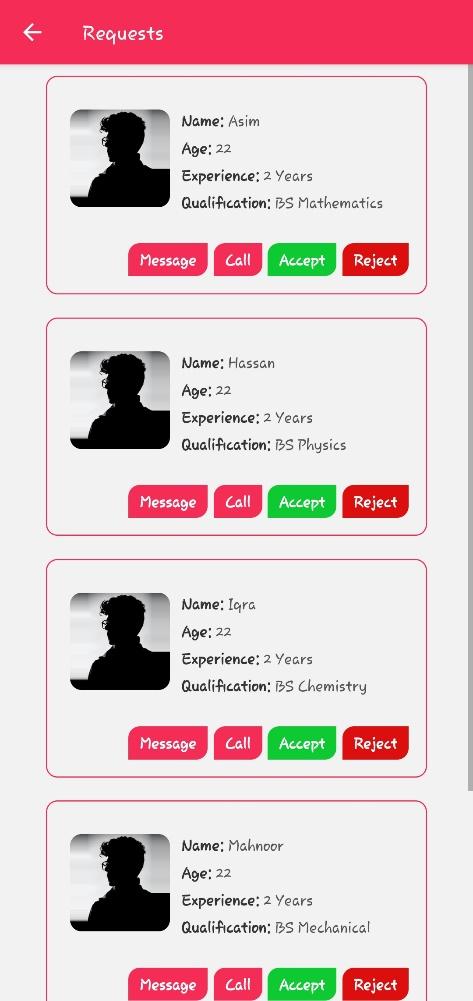
## Student Dashboard



## Edit Profile Screen



## Request Screen



## Chatting Screen

