Abstract

24 Hours Tutors is a system that we have developed to allow students to have an easier time searching for tutors. The main goal of the system is to increase the efficiency of looking for a tutor and to help ease the difficulty of looking for a tutor.

As we all know, it is a quite a challenge for students to look for tutors all by themself. By using the designed program, students will have a better time in looking for tutors. Whereas for tutors, they will have a better chance of starting their own tuition because of the increase of students and the ease of grouping students together. Admins on the other hand will be in charge of the status of tutors. Tutors will have to be verified by admins so that they will be able to teach students. Admins will also provide assistance for students to help them in searching for tutors.

In this project documentation report, the development team will be covering the process of developing the system, including both problems and successes. Besides that, this report also covers the clarification of the programming language used to develop the system.

There is also screenshots of the completed system provided for users to have a closer look of the system before using it. The screenshots will be covering all the functions and features that is offered by the system.

With this project documentation and report, the development team hopes that it will help users to understand the developed system more before using it. The development team also hope that this documentation and report will help whomever is interested to create a similar system whilst letting people who are studying programming language to have a better understanding.

List of Tables

List of Figures

Contents

Chapter 1: Introduction

1.1 Overview: Current Technologies / Trends

In this current era, it is distinct that technology is an important aspect in human's daily life. It is used everywhere and all the time. After all we are in an age of continuous innovation and new inventions. Furthermore, the trend and development of technology is far more advanced compared to the past technologies. For example, how do we listen to music, how do we watch television and so on have changed immensely. In the previous days, watching television usually meant sitting down in the living room and share the television with your family. But as technology has become such an impact on life, people can watch television anywhere, for example streaming on mobile phones and computers. (Marissa Higgins, 2018)

We believe that it is difficult for one to find a home tutor or even a tuition class. It is very common that we see people distributing tuition brochures and even posters hanging around to attract the attention of students who are in search of a tutor. Through our research and interviews with students and even parents, it has proven that looking for a tutor is a troublesome task.

By replacing the current advertising method to a computerized system, it can help in easing the effort of students who are in search for a tutor. A computerized system will be able to help in minimising the time spend on searching and reduce the use of papers to advertise a tuition program. A systematic system can be developed by using suitable technologies and knowledgeable resources. With the implementation of advance technology, task of searching for a tutor can be done efficiently and fast with just a few clicks. Besides that, using a fully computerised system will also enhance the efficiency of storing data and records. The internet has unlimited of space which is useful to store massive load of data for a long period of time. It is far more efficient compared to the old method where tutors will have to make use of paper to save student records. Tutors will face risk of losing the records if they lost the paper and it requires space to store all the papers. (Margaret Rouse, 2017)

During the past where technology is not that advanced and was not an important aspect in daily life, people used to rely on the paper-based system on storing data and information. Using paper-based systems means that there are wasted man hours spent filing, searching, retrieving, and re-filing these documents. But technology had been rapidly improved as time pass. Naturally, there are also advantages and disadvantages of using a paper-based system. The paper-based system will have stronger data security compared to the modern technology as people can easily hack in and retrieve data. (Gregory Hamel, 2018). But, using modern technology, unlimited data can be

stored. Companies and individuals should try to keep up and learn to implement the latest technology into their daily life as it will help to ease their lifestyle and would be beneficial in certain times. As we can see in the present days, people use mobile phones and even computers to help in their tasks such as storing data and information.

It is obvious that the current technology and trends are changing rapidly compared to the past. We can see everyone is staring at their smartphones instead of looking around every time and everywhere. In the past time, there is technology like telegraphs, two-way radios that allow people like taxi drivers and emergency services to communicate and so on. As the time passes, we can see technology is slowly rising and the discovery of the smartphone is slowly found. It is hard to manage something if we choose not to upgrade our usage and knowledge of technology as it has been a very important aspect in the current society. (Christine, 2017)

1.2 Background of project

The entire team did some research about the difficulty of finding a tutor. The team had a discussion with lecturers in KDU University College and even interviewed parents to find out whether looking for a tutor today is difficult. The results are positive as the method used by the current society is not systematic enough and can be improved. Based on research, we found out that looking for a tutor is not an easy task to begin with. Therefore, technology comes into place to help parents or students who are looking for a tutor. It replaces the paper-based system used by tutors and helps to ease the effort to store every record of students. To store things physically is a problem as space is always limited. With technology implemented, storing student records is a problem solved for tutors. It also helps to ease the effort of parents who are in search of a tutor.

After several discussion, the team came out with an idea to solve the problems of parents and tutors. The team's decision is to come out with a system to help students to look for tutors nearby and to help tutors store their data and information effectively. The system will also suggest tutors following by the categories students choose. With this system, looking for a tutor is just a few clicks away and have never been this easy.

The team will be developing a web based system to ensure an easy use of the system. Nowadays most of the people would prefer a web based application over a software of a mobile phone because things are easier to look on webview compared to mobile view as things are smaller. The developed system will be able to help in finding a home tutor or a tuition class. It will be able to match students with a tutor according to the selection applied to the filter system. Moreover, students will be also able to have a

live chat and ask for information of the class if the tutor is available. The system developed will be able to let students select between home tuition and or group tuition. There is also a function for admin to log in so they can manage the database and update the system.

The system is developed by a group of 5. The task of each members are:

- Ong Wei Cheng - Programmer, Documentation, Tester
- Go Yik Yek - Main Programmer, Database Designer
- Bryan Lim - Web Designer
- Kenneth Chong - Documentation
- Yeap Chi Hang - Database Designer

1.3 Problem Statements

As we all know, in the 21st century technology has been a big impact in the current society that it has become a lifestyle of everyone today. People started to use technology in all fields of work. The reason people started using computerised system in their daily life is because it can help reduce the workload and boost the efficiency of the job being done. (Karehka Ramey, 2012). Moreover, certain people also uses computerised system to replace the old paper-based system as it has unlimited space compared to the paper-based system. Below are the problem statements that are going to be discussed.

During the process of developing the new system, the team have faced some systematic problems such as error when trying to store all the records in the database. Therefore, the admin must update the records manually into the database if there is any changes to the user's information or adding a new user into the system. Incidents can happen at all times such as hardware failure or human error may cost the loss of records. Without a proper system, the admin would be having a hard time keeping up with all the updated information and registration of a new user into the system.

In addition, making an announcement is easier by using a computerised system. Tutors will not be having a trouble in updating their students. Unlike those days when technology is not highly advanced, tutors have a hard time to updating their students about announcements. They have to make an announcement for class replacement or change of tuition time in class. It is a troublesome problem because there will be some students who are absent on the day of announcement. In order to fix the problem, we implemented a fully computerised announcement system for all tutors and admins to make an announcement easily. Then, by using a computerised announcement system, students will not make excuse about them not knowing about the latest update. Most of

all we also implemented a chat system for all user which means that all users are able to exchange information and students will be able to communicate with tutors easily.

Last but not least, one of the biggest problem we faced in developing this project is that our system would not guarantee the safety for users. Nowadays, anyone can enroll themself for a job such as uber, grab and jobstreet. Being a user in the system does not guarantee the safety of both parties because we might not know their intentions. As we can see on the news, there are a few dangerous cases such as sexual abuse, robbery and other risks that can happened during a uber ride. (Sara Ashley O'Brien, Nelli Black, Curt Devine and Drew Griffin, 2018). In order to enhance the security and safety for our users, the team discussed about a solution which is to have tutors who wants to enroll themself to go through an online test and will have to go through a face to face interview with the team before they can be officially registered and verified by the system. With this solution, users will at least have a peace in mind as the security and the safety of the web site is improved.

In conclusion, a systematic computerised system is a way to overcome the difficulties encountered by the current system.

1.4 Aims of the project

To complete something successfully, we must have an aim as a guideline for us to achieve our goals perfectly. The team's aim is to create a web based system to help students ease their effort of finding tutors for tuitions. The system developed will be able to help students to look for a tutor efficiently. With the help of a systematic system, data and information of each student and tutor could be stored safely. In addition, the system also benefits tutors as they will have a increasement of income as well as students.

1.5 Objectives of the project

Objectives can be also known as the desired results that you thrive for in doing the project. Objectives are the successful development of the project’s procedures of initiation, planning, execution, regulation and closure as well as the guidance of the project team’s operations towards achieving all the agreed upon goals within the set scope, time, quality and budget standards (Team Clarizen, 2018). A well written objective is crucial because it can affect every step of the project life cycle. By setting a goal or objective to complete something, it can lead the project to a successful path.

The main objective of the current project is to ensure that students can find a tutor for either home tuition or tuition in groups easily. The first objective of the project is to make the system flexible and can be accessed easily for students and tutor.

Furthermore, the team decided to design the system's UI into a simple and organised web platform so that students and tutors can easily navigate through the web page and look for tutors with ease.

The second objective of the project will be the database system. The team planned to establish a systematic database system to store data of students and tutors. The data kept in the database can be saved and organised. Hence it can be easily accessible for students and tutors who are looking for certain information.

1.6 Summary of each chapters

Chapter 1

This chapter acts as an introduction of the project and it discusses the main reason and ways on developing the system.

Chapter 2

This chapter contains the explanation of feasibility of the web-based 24hours tutor project to point out the opportunities and limitations of the system. At the same time, it also covers the process of creating the system

Chapter 3

In chapter 3, we discussed the requirements needed for the system such as methodologies and functions. We also listed out the functional and non-functional requirements. We also explained about the process of development. Furthermore, this chapter also discuss about the tools that are used to create the system.

Chapter 4

The chapter goes through diagrams to show the details during the planning and analysis stage. We included ER diagrams and Use Case diagrams as solution the show the improvement of the system. The diagram also plays a role in helping the team to get the full idea of what is needed to be in the system.

Chapter 5

The following chapter contains programming language that are used to develop the project. The developed system contains programming languages such as HTML, PHP, CSS, JavaScript, PDO and SQL. The details and explanation of the codes implemented are also stated in this chapter

.

Chapter 6

Chapter 6 goes through the testing phase of the developed project. Testing methods are shown when the team encounters bugs which was then solved.

Chapter 7

In the last chapter, it contains a conclusion for all chapters that are included. There are also recommendations on how to improve the software and enhance its strength to stand out in the future.

Chapter 2 : Primary Study/Literature Review

2.1 Feasibility Study

Studies regarding the feasibility of the proposed web-based finding tutors system had been done by the development team to identify the opportunities and limitations of the system. (Margaret Rouse, 2018) The following feasibility studies will show how the proposed system's functions and the possibilities that can be achieved.

2.1.1 Operational feasibility

Operational feasibility measures on the ability of a project or proposed website to solve existing problems and take advantage of opportunities that are presented during the course of the project. This section is about how practical the proposed website works and this website will be useful for users who are searching for available tutors. (John Spacey, 2017)

The proposed website 24hrs Tutors requires 3 types of users which are admin, student and tutors. The admin has the ability to manage user accounts, changing username and password and view past transactions and payments made. All information will be stored inside the database thus the admin do not have to worry about data being lost. The admin can generate a list of students for the tutors to let them know their students. This allows the lecturer to contact their students easily and being able to communicate with their students with different communication tools.

Students whom are looking for tutors for tuition in our website have to register an account. The requirements of registration is students must provide their name, age, gender, email, phone number, address and many more. These information will be stored inside the database. Besides that students are able to search any tutors in our database. Once the students have found their tutors, the payment is gone through by banking. For students who are paying their tutors through online, their parents will be the one paying the tuition fees unless the students must be above the age of 17 then they can start paying their own tuition fees. Students are able to rate tutors on how well their teaching has improved their studies.

Whilst on the tutor side, if user wants to register as a tutor in our website, the website will send them an email to the responsive user. The email contents contain about user must gone through a face to face interview with our staff and they must take an exam in our company before they can become a tutor in this program. Tutors are able to create their own timetable as well as adding syllabus. In addition tutors are able to generate a report about the students progress. All tutors in our program will have a rating system which is the higher the rating of the tutor, the more recommended he is in

the website. Furthermore tutors are able to contact their students through messaging them in the system. All tutor earn their allowance through online banking. Our staff will be in charge of transferring the users allowance through online banking.

In conclusion the team should be able to develop the proposed system. Our system has improve some management system and increase more functionality to the system compared to some similar system such as “tutorhunt” and “find your trainer”.

2.1.2 Technical feasibility

Technical feasibilities can be identified as the process of validating the system assumption, architecture and the design of the system itself. The IT group has been researching on improving the system itself which requires lots of technical solutions to improve the system.

At first, the group does not have any idea on what kind of system that we want to propose to our supervisor. Then, we started a group discussion for what kind of system that we wanted to propose. In the process of discussing what kind of of system that we want to build, the group got an idea of proposing a system which relates to find available tutors for tuition. In addition we decided that we should add in some elements of uber and grab into our system such elements are able to instantly hire someone now. Our group supervisor Mr. Ang Sau Loong approve with our proposed system. Then we started with gathering information for developing the system. Afterwards we started building the system with the information that we gathered for developing the system. At first we try to generate a sample of the database of the system, how the system works and how to generate our income during the first month.

Secondly, the team started to build the system with familiar technology development to implement the system. The team developed the system by using HTML, CSS, Javascript and PHP for programming the system whereas phpMyAdmin for the database. The reason why we only chose these programming language was we are familiar with HTML and CSS while PHP is completely new to the team and the team had to self study PHP along with the development process.

On top of that, we decided to focus on functions that are supported by MySQL when developing the system. Then we used the prototype of the system will be directly hosted in the college’s servers or on a local host. Furthermore there is a part where user can make payment through the system, keeping track of every month payment easily. We wanted to create a system that uses lesser paper, and where every piece of information can be easily organized. Organizing the information through handwritten will

result in many unwanted mistakes. Thus we decided to transform a tuition system into a slightly more advanced system.

In conclusion, the team should be able to develop the proposed system. The team will provided a user manual and the terms and condition to ensure the system features can be understandable and learned easily.

2.1.3 Economic feasibility

Economic feasibility defines as the cost of benefit analysis of the system. The development team has studied and investigated the development and operational cost, and the benefits of using 24hrs Tutors system.

As for the development cost, the team did not spend any money on developing the prototype 24hrs Tutors system. The team used phpMyadmin to act as our database which is free and suitable for demo version of the system and the development team is able to manage and upload system files into the database. Besides that, the team used Visual Studio Code and Notepad++ to write and develop the system code. In addition all pictures in the system are royalty free. Furthermore the team was able to use KDU University College servers to host our website for testing the system prototype during the development process. Apart from a live server we also used XAMPP to host our website locally.

The operational cost of the proposed system is none. Since KDU University College already has their own server for web hosting. We learnt in order to use KDU University College own server, we register our proposed system to Mr.Alvin Goh who is in charge of the Information Management Centre of KDU University College so that our proposed system can be used for web hosting. As for the user side, the user side has been differentiate into 2 users which are students and tutors. All users are able to register without any cost except for the tutors because there are some requirements for creating an account in our system. First of all, users who wanted to be a tutor in our system they will received an email which the content is about asking the user for a face to face interview and a trial exam. Once they have succeed in the interview and the trial exam, our staff members will record them into our system then they are officially tutor in our system. Users will only need a browser with internet connection to access and interact with the proposed system.

Nevertheless, the benefits of users using this system is all users are able to view their timetable in their own personal account and most importantly the system allowed admin to manage all user and application. The database stores all users information

while users are able to look through all users profile information. In addition the staff or admin are in charge of tracking and managing all user and application which shows a well-organized information can be seen as well. Secondly the benefits of users using this system is that users are able to communicate in the system. In order for students and tutors to communicate in the system, the students must book the tutor as their tutor for tuition then the tutor's contact will appear in the message box of their personal account which means that both students and tutors personal account their message box will appear the students or tutors name. In addition students hiring available tutors in the system will be recorded as well and the system will provide to user about recommended tutors in the system. Furthermore students and tutors are able to gain points through the system once users have reached certain amount of points, these points can be used to exchange items which list in the system.

In conclusion, the proposed system can be developed and operated by just using minimal cost and resources. The benefits delivered by the system will benefit all users.

2.2 Critical review of literature relevant to the project

This subchapter will explain about the review and study of literature that are relevant to the project. It is divided into three parts, the following below states the three parts of this subchapter:

1. Review on similar existing systems.
2. Recommendation system using on rating feature.
3. Recommendation rating feature algorithm.

2.2.1 Review on similar existing systems

This section will be reviewing the existing system that we used for referencing which the system is related to user matching.

a) JobStreet

JobStreet is a job searching system. It categorizes users into two types of users which are job seekers and employers. The main purpose of this system is to allow registered users to search and apply for jobs that are posted by companies. This system allows users to search specific job in the search bar in addition the system has another search function which is more specific than the search bar. That function works only when users fills specific details into each drop down list. This features gives user a more wider range of options where users can choose a more specific options about their work.

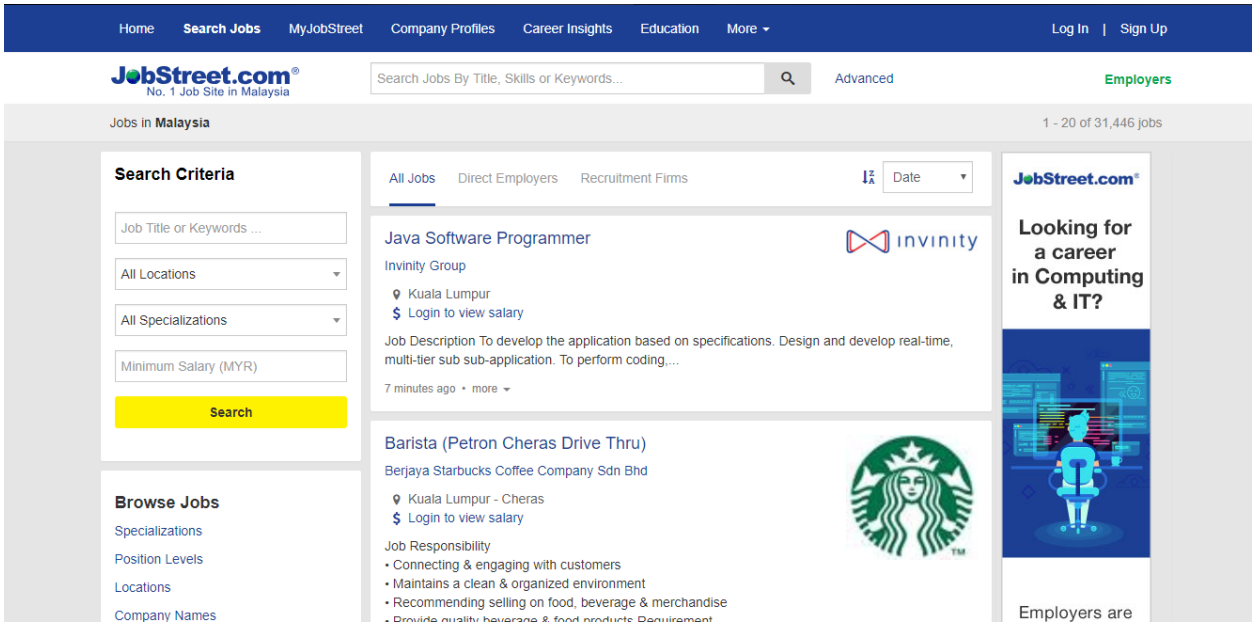


Figure 1: Homepage of JobStreet.com after accessing as a job seeker.

b) Tutor Hunt

Tutor Hunt is a tutor searching system. The user in this system has been split into two kinds of users which are students and tutors. The main purpose of the system is to allow students to find available tutors for tuition. There are two types of tuition for user to pick which are one to one tuition and online tuition. The difference between the both is that one to one tuition, users have to key in their postcode so that the tutor is able to find their student location whereas online user does not need to insert postcode which their tuition will be progress online. In addition there are three buttons for users which are find tutors, become a tutor and learn more. Each buttons are linked to their assigned page. Furthermore there is a button named view all subject that function allows users are able to browse the subject that they wanted for tuition.

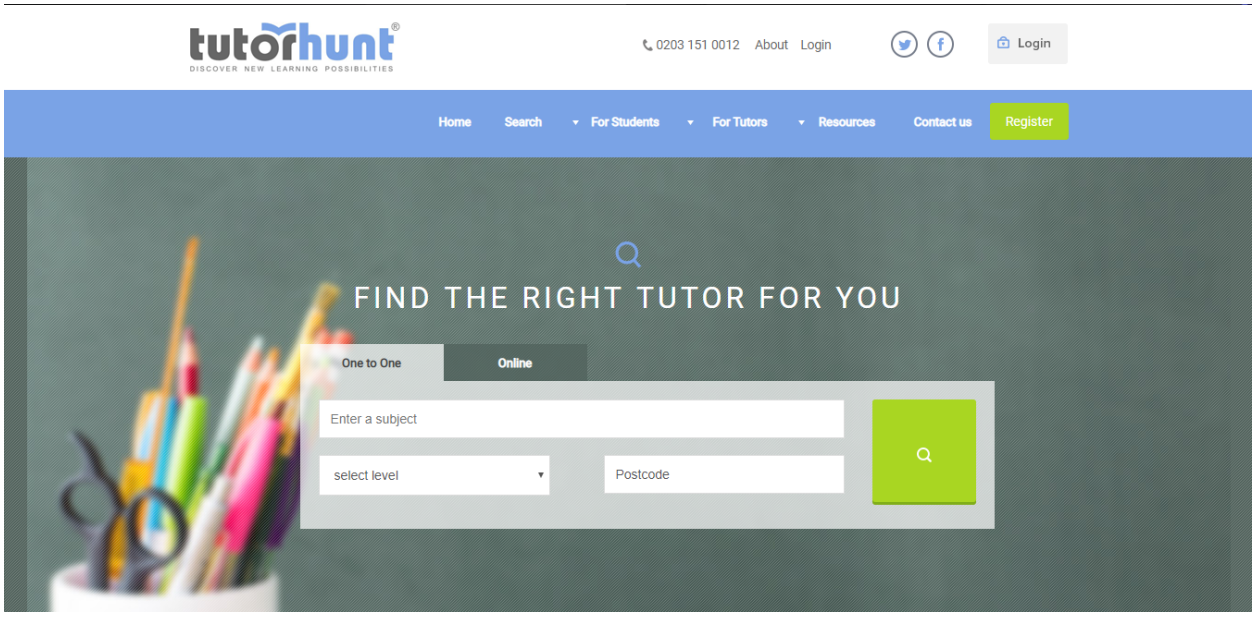


Figure 2: Homepage of tutorhunt.com (top)

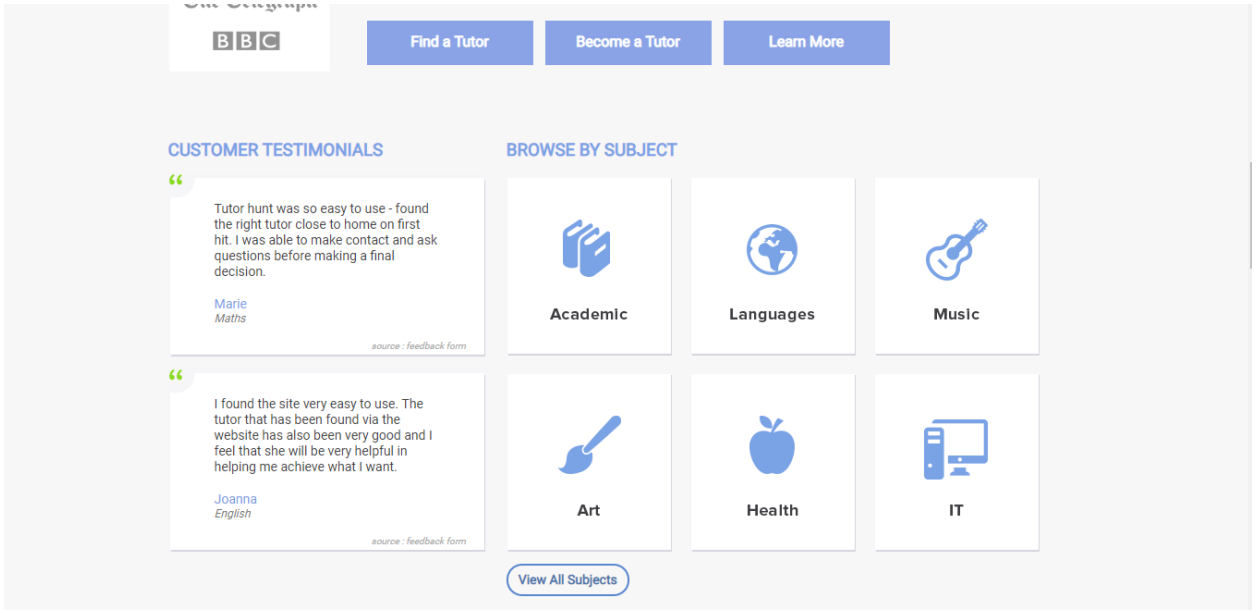


Figure 3: Homepage of tutorhunt.com (bottom)

c) Find Your Trainer

Find Your Trainer is a system which allows users to find a personal trainer or gym instructors. The main purpose of this system allows users to find their ideal personal trainer throughout the system. Their search bar in the homepage is to find the nearest area where user can find available trainers but the system searching is limited to be usable only in United State of America. The system allows users to train anywhere they

like for example health club or studio and in home training. In addition the system allows users to find their own personal trainer depending on the price ranges that the user has given. Furthermore the system also guarantee safety to all users which is their main priority in the system. Find Your Trainer partners with Smart Screen to ensure their trainers are qualified and undergo a background check and Smart Screen also regularly checks sex offender registries.

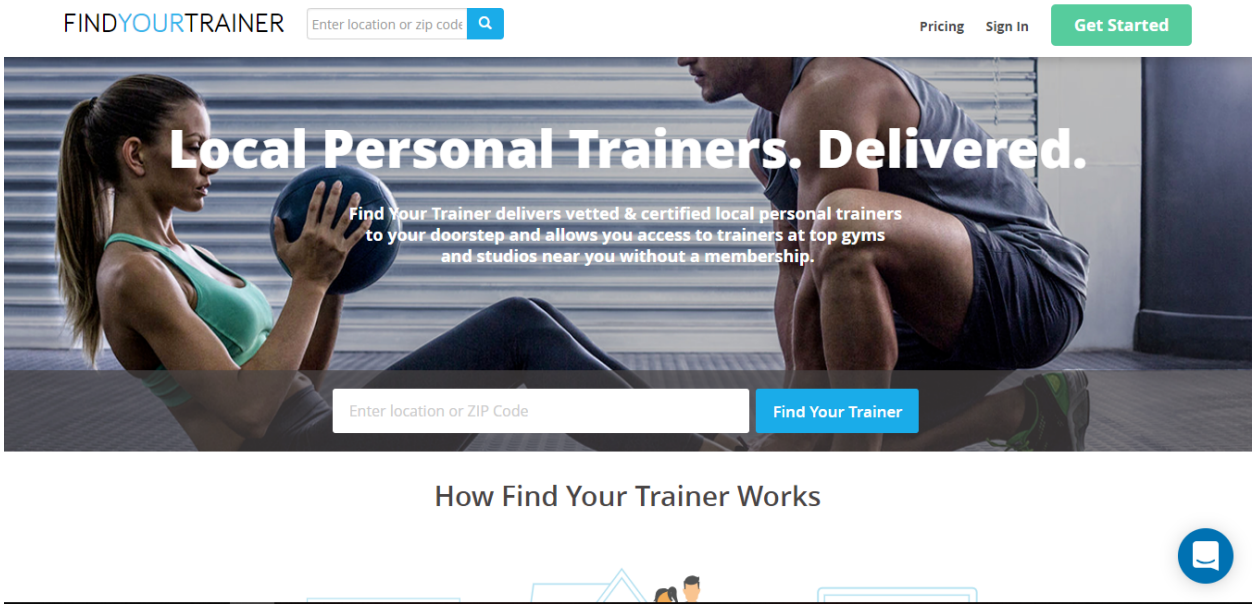


Figure 4 Homepage of findyourtrainer.com

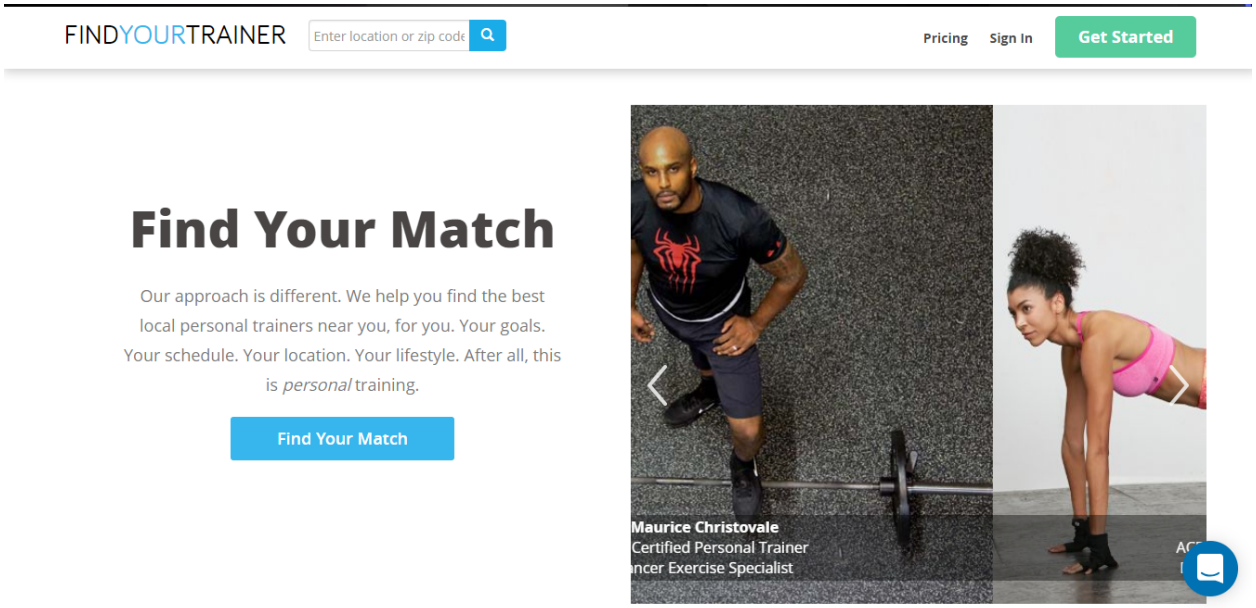


Figure 5 Homepage of findyourtrainer.com (Find Your Match)

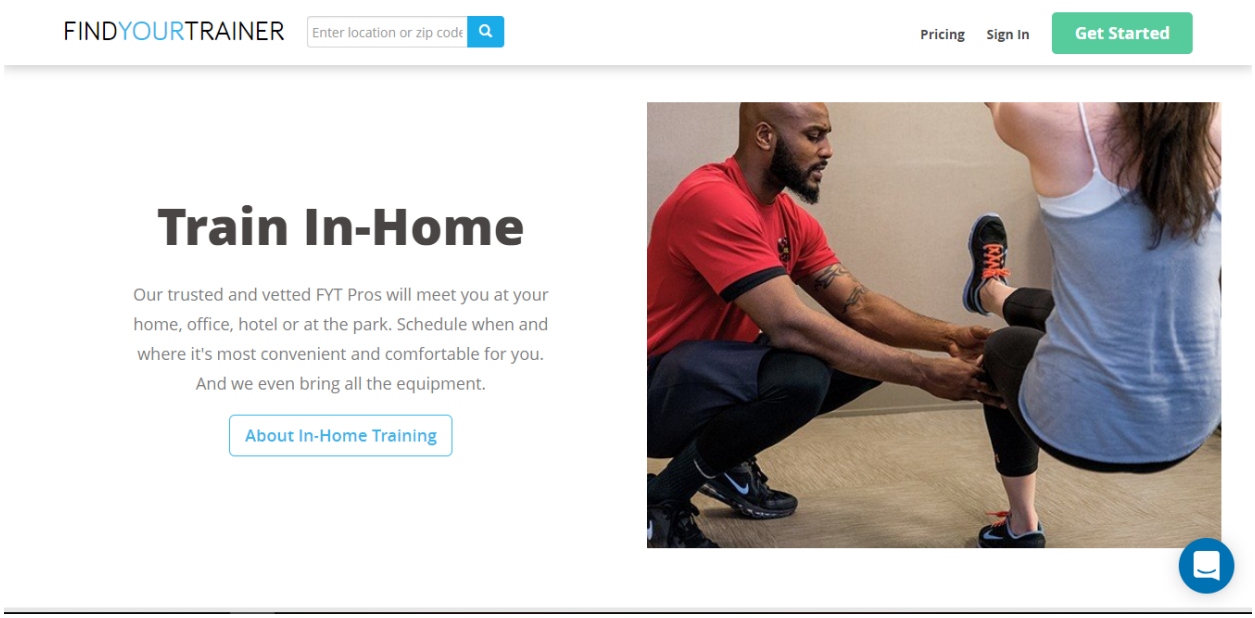


Figure 6 Homepage of findyourtrainer.com (Train in-Home)

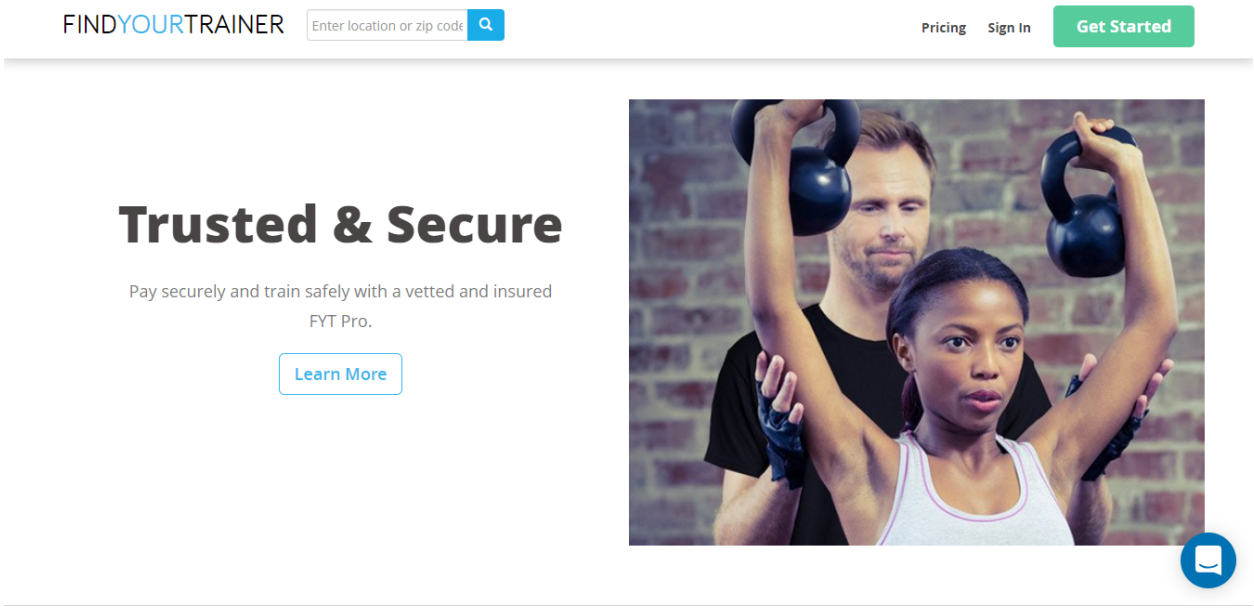


Figure 7 Homepage of findyourtrainer.com (Trusted & Secure)

2.2.2 Comparison between all existing system

Table 1: Show the comparison of all existing system with the proposed system.

Features	JobStreet.com	Tutorhunt.com	findyourtrainer.com	Proposed System
System Type	Online, Computerized	Online, Computerized	Online, Computerized	Online, Computerized
Search Tutor	No	Yes, by search feature	Yes, by search feature	Yes, by search feature
Tutor Registration	Yes	Yes	Yes	Yes
Browse Subject	No	Yes, by search feature	Yes, by drop down list	Yes, by search feature
Recommended Tutors	No	Yes, review on users account	Yes, review on users account	Yes, by search feature and review on user account
Safety Security to users	No	No	Yes, partnership with Smart Screen	Yes, check manually and approved by our staff members

Based on the comparison between existing system with the proposed system, online computerised system is more easier to use and it is usable anywhere around the world as long as the area have internet service. Users are able to search for specific tutors. Users are able to register as a tutor in the system by following the given instructions and fulfilling the requirements. The system also allows users to browse the subject that they wanted for tuition. Each of the system above, their browsing function works differently to each other, for example Tutorhunt.com browsing functions allowed users to search their desired subject through the search bar function whereas findyourtrainer.com used a drop down list to search the subject that the users wanted. In addition tutor recommendation is a useful feature to users that are looking for a good tutor whereas Jobstreet.com used the recommendation system for job recommendation unlike the other two systems Tutorhunt.com and findyourtrainer.com only these two implemented these recommendation system. Last but not least only findyourtrainer.com

apply this to there system but without the partnership with Smart Screen they would not be able to implement this user safety security system.

b) Recommendation System

According to an information from webopedia, Recommender System is an information filtering technology commonly used on e-commerce Web sites that uses collaborative filtering to present information on items and products that are likely to be of interested to the user. The recommendation feature comes into play when the users have no idea what they are looking for, the recommendation system will list out a list of results to the users depending on the users interested in. (Gaston, 2018)

2.2.3 Recommendation System using on Rating Feature

The definition of recommendation system is that an information filtering technology based on the users interest. This technology is widely used on e-commerce websites. There are two types of recommendation systems which are content-based filtering and collaborative filtering. Content-based filtering is also known as cognitive filtering. Content-based filtering recommends users items with similar content to the items that the user has shown an interest in. (Recommender-System, 2018) Whereas collaborative filtering finds the similar interest in the user activity to produce recommendations based on the users interest. (N Lakshmipathi Ananth, Bhanu Prakash Bhattula)

The rating feature allow users to provide feedback information about an object, which means users can list out the characteristics of the object such as the quality of the object, certifications and articular scores. This feature plays a huge role for data management strategy, ratings can be crucial in finding better products because consumer will look through the ratings about a product before they purchase the product. This will create a popularity on a product based on users preference. (Ellie Kutsevol, 2018)

The development team researched and decided that we could combine both features into one. In which our recommendation system will be based on the ratings of a tutor. This will easily differentiate the positive and negative reviews about the tutor. So our recommendation system will be using the rating features to produce a list of positive results about the tutor in our system to the users. The highest rating tutor will be on top of the list which is also the most recommended tutor in our system.

2.2.4 Conclusion

In conclusion, our development team will not be using content-based filtering and collaborative filtering but instead our development team will be integrating both the recommendation system and the rating features into one. The reason is that rating features are starting to become a trend for recommendation in every system due to it is easier to recommend user about how other users opinion about the interested asset then the user will decided that whether they wanted to purchase the interested asset. The highest rating tutor will be on top of the list due to more positive recommendation from different users.

Chapter 3: System Analysis/ Methodology

3.1 Requirement Specification

The following chapter resolves around the requirements that are needed by the system, both functional and non-functional requirements, an explanation on the system development life cycle used, and why the development team used the tools that they have used to create the system.

3.1.1 Functional Requirements

Functional requirements defines a function of a system operating within normal parameter where the function is meet user expectation of the system. Which means it is the desired operations of a program. Functional requirements are a part of requirements analysis which is an interdisciplinary field of engineering that concerns the design and maintenance of the systems (Margaret Rouse, 2018). The system is split into 3 different users which are Students, Tutors and Admin.

Function Requirements for the Whole System

W01	A user should be able to login to the system.
W02	A user should be able to logout from the system.
W03	A user should be able to sign up an account into the system.

Functional Requirements for the Student

S01	A student should be able to search specific types of tutors in the search function.
S02	A student should be able to look into tutors profile for further information.
S03	A student should be able to view their syllabus.
S04	A student should be able to view their timetable.
S05	A student should be able to pay their tuition fees through online.
S06	A student should be able to give rating and comments to all tutors.
S07	A student should be able to view and edit their own profile.
S08	A student should be able to change their password and user account.
S09	A student should be able to participate in a chatroom which consists of students and tutors.

Functional Requirements for the Tutors

T01	A tutor should be able to manage their teaching course.
T02	A tutor should be able to create their own timetable.
T03	A tutor should be able to add syllabus.
T04	A tutor should be able to participate in a chatroom which consists of students and tutors.
T05	A tutor should be able to view all the ratings and comments given by the students.
T06	A tutor should be able to receive their allowance through online banking.
T07	A tutor should be able to view and edit their own profile.
T08	A tutor should be able to change their password.
T09	A tutor should be able to view their timetable.

Functional Requirements for the Admin

A01	An admin should be able to manage user and full application.
A02	An admin should be able to manage all tutors account in the system.
A03	An admin should be able to receive the fees from students.
A04	An admin should be able to distribute the allowance to tutors.
A05	An admin should be able to check the account password and details of all users in the system.
A06	An admin should be able to manage all students account in the system.
A07	An admin should be able to add and delete subjects into the system.
A08	An admin should be able to check all booking records in the system.

3.1.2 Non-functional Requirements

- **Usability**
  - The system is compatible through a web browser on a computer device.
- **Accessibility**
  - There will be error messages for any input of data that are incorrect.
- **Confidentiality**
  - All students will be able to view the information that are set public by the tutor
  - All tutors will be able to view the information but not the address of the student
- **Availability**
  - The system is available 24 hours, 7 days a week for students and tutors.
- **Access Security**
  - A newly registered tutor is unable to gain access into the system unless the admin accepts their registration.
- **Redundancy**
  - For all users such as students and tutors except admin, all students and tutors would not be created if a username and email already existed in the system.
- **Reliability**
  - Students and tutors will be able to access the system as long as they have access to the internet.

3.2 System Development Process

The team applied the waterfall method as their methodology to develop the system. The waterfall methodology emphasizes a logical progression of steps to be taken throughout the software development life cycle. By using the waterfall methodology, the team will acquire a few advantages throughout the project development. By using the waterfall methodology, the team will be suited for milestone-focused development. With clear, concrete, and well understood stages, everyone on the team can understand and prepare for further progression. (Andrew Powell-Morse, 2016)

Firstly, during the planning phase, the development team started the development of the system, they started to plan out the functional and the non-functional specifications was required in the system. The team meet up with Mr Ang Sau Loong for further advice on creating the system. In the end the team decided on creating a system that aid students who are looking for tutors as their project.

Secondly, in analysis phase, the development team decided that they will be using HTML, CSS, PHP, PDO, SQL and even implemented some JavaScript to create the website proposed. In addition the team used phpMyAdmin as the database management tool for the system. The development team also stored the programmed files into the database so that the system could be accessed as long as the server is not down. The system is completed after a methodical discussion on what programming languages that is suitable for the system.

Next, in the design phase, the development team had illustrated and prepared diagrams to enable the developing of the system to be done smoothly. The team created a few diagrams such as Use Case diagrams, ERD diagram, Activity diagram and a Gantt chart to assist during the process.

Fourthly, the implementation stage. The development team advanced to the programming stage of the program. They started creating pages of the website such as home page, register and login page. After the pages were successfully created, the team progressed towards the login and register function of the website. Student's records as well as tutor's are safely stored in the database. After the basic functions are implemented, the website is currently progressing into a prototype stage.

After that, the development team commenced a test on the developed system to observe for bugs, errors and incomplete functions of the website that might be left out

during the implementation stage. After the encountering of bugs and errors, the team will focus on resolving it until the occurrence is minimised and the website is improved.

There are a few errors and bugs that the team had encounter. One of the errors encountered by the team is the duplication of data appeared while trying to create an account. Secondly, the looping of login page while users try to log-in. Besides that, the UI design did not change after implementing certain CSS code.

The system was then handed in to Mr Ang Sau Loong for further checking on desired requirements and expectations. The team will then proceed to improve the website and improvised it following the suggestions and feedback that are offered regarding the system. The process will be kept revolving until the system is created successfully.

3.3 Justification of Development Tools and Methodology

The development team created the proposed system by using HTML, CSS, PHP, JavaScript and PDO to build the proposed system. Some of the team members have some knowledge of PHP which they learnt it throughout their internship. As for the database our development team used SQL to build the database. We learnt SQL language from previous semester and most of all we are implementing the knowledge into our proposed system which was taught by our database lecturer.

The integrated development environment (IDE) that the development team used to program the system is Notepad++ and Visual Studio Code. Notepad++ is used for designing the website whereas Visual Studio Code is used for creating the function of the system. Both softwares are free to download and can be used without any payment. Notepad could have been used but Notepad++ is an advanced version of Notepad as it offers more option and is more convenient to use..

At early stages we decided to use XAMPP to set up a local host for our website. We used phpMyAdmin to manage our database. PhpMyAdmin is one of the easiest management tool available to manage a SQL database. It is also fairly easy to expand the database at our own pace. But during one of our discussion we discussed about using the server provided by KDU Penang University College to host our website. In the end most of us agree to migrate our website and system to the server which our college provided to us. In order for us to use the college server, we must register our proposed system including with all of our group members name and student id. This is to get a

confirmation from the admin to grant us the permission to gain access to their live server.

Last but not least, most of the diagrams were drawn by using lucidchart which is also free to design all the diagrams in our system. The disadvantages of using lucidchart is when user is designing diagrams the link between one table to another table is harder to link which takes time to link properly from one diagram to another diagram. We could not use other alternatives like Visual Paradigm as that may require the entire team to spend money for a license.

Chapter 4: System Design/Proposed Solution

4.1 Proposed System

The following chapter goes through the diagrams and designs our team created to get the idea of what is needed to create the idea system.

4.1.1 Activity Diagram

An activity diagram is a simple and intuitive illustration of what happens in a workflow, what activities can be done in parallel, and whether there are alternative paths through the workflow. (Maria Ericsson, 2004)

4.1.2 Use Case Diagram

A Use Case Diagram is a graphic depiction of the interactions among the elements of a system. A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. (Margaret Rouse, 2015)

4.1.3 Use Case Description

Use Case Name:	View Timetable
Actor:	Students, Tutors
Trigger Condition:	Students are able to check their schedule every week. Tutors are able to check their teaching schedule every week.
Summary :	This use case allows students and tutors to view their timetable in the system.
Basic course:	**Students** 1) Students must login to their personal account in order to view their personal timetable. [**Exception:** Students did not book any tutor for tuition in the timetable] 2) System will display their personal timetable. 3) The use case terminate. **Tutors** 1) Tutors must login to their personal account in order to view their personal timetable. [**Exception:** Tutors did not create their time table] 2) System will display their personal timetable. 3) The use case terminate.

Alternative Scenario:	
Exception Scenario:	**Students did not create a timetable** 1) Timetable will be blank and have no information 2) Students must book a tutor for tuition in the system. 3) The primary scenario continues from Students step 2. **Tutors did not create a timetable** 1) Timetable will be blank and have no information 2) Tutors creates their own timetable 3) The primary scenario continues from Tutors step 2.
Pre-Conditions:	**Students** 1) Students have to create an account. 2) Students have to head to schedule to manage their own timetable **Tutors** 1) Tutors have to create an account 2) Tutors have to set available time at their timetable
Post-Conditions:	1) New student and tutor accounts are created 2) Timetable is created
Assumption:	1. System is functioning. 2. Internet connection is available for database. connection.

Use Case Name:	Manage Tutor
Actor:	Admin, Tutors
Trigger Condition:	Admin are able to manage tutor accounts
Summary :	This use case lets admin of the system to modify the status of tutors and to view their profile information
Basic course:	**Tutor** 1) Tutor have to register for an account 2) Tutor have to fill in their information 3) Information will be updated to the database and admin will be informed 4) Tutor have to pass a test that will be provided by the admin [**Exception:** Tutors did not pass or take the test] 5) Tutor will be verified. 6) The use case is terminated. **Admin** 1) Admin selects newly registered tutor. 2) System shows information of the registered tutor. 3) Admin provide test to verify the tutor. 4) The use case is terminated.
Alternative Scenario:	
Exception Scenario:	**Tutors did not pass or take the test** 1) Tutor account will not be verified 2) Tutor will not be able to start teaching 3) Tutor take the test and pass 4) Primary scenario continues at Tutor step 5
Pre-Conditions:	1) Tutors must register an account as a tutor 2) Tutors must fill in their personal information
Post-Conditions:	1) Tutors will be verified 2) Tutors will be able to start teaching students
Assumption:	1) System is functioning. 2) Internet connection is available for database connection

Use Case Name:	Receive/Pay Fees
Actor:	Admin, Tutors, Students
Trigger Condition:	**Students** Students book tutor for tuition in the system. Then the payment will be transferred into the tutors personal bank account. **Tutors** Tutors are being book for tuition in the system. Then the fees will be received into their own personal bank account. **Admin** Admin is able to check the fees is being paid and received to each users. Most of all admin is able to generate a list of users booking records.
Summary:	This use case allows students and tutors to receive and pay fees through the system. In addition the admin is able to look through the booking records of each users.
Basic Course:	**Students** 1) Students have to register an account in the system. 2) Students have to insert their debit or credit card number for further booking with the tutors in the system. 3) The pricing of the tutor depends on the subject that students are willingly to learn. 4) The use case is terminated. **Tutors** 1) Tutors have to have an official account in the system. 2) Tutors have the rights to set the price ranges of the subject that they are teaching for tuition. 3) Once the students have book the tutors for tuition then it will provide a receipt to both students and tutors. 4) After the booking is recorded in the tutor side, the tutors have to confirm with their tuition success if the tutors confirm that the tuition is a success then the money will transfer into the tutors account. 5) The use case is terminated.

	Admin 1) Admin have the rights to check through all users booking records. 2) Admin have to manage all tutors price ranges of the subject that the tutors are teaching. The price ranges cannot be exceeding to a certain amount and must be following the company policy. 3) The admin will contact the staff to transfer the tuition fees to the tutors once they have a successfully finish the tuition. 4) The use case is terminated.
Alternative Scenario:	
Exception Scenario:	**Students** 1) Students did not have any money to pay for the tuition. 2) Students have the rights to cancel the tuition, once the tuition is cancelled by either the students or the tutors, the fees is refund back to the students. **Tutors** 1) Tutors are not able to book another tutor for tuition. 2) Tutors have the rights to cancel the tuition, once the tuition is cancelled by either the students or the tutors, the fees is refund back to the students.
Pre-Conditions:	**Students** 1) Students must register as a student in the system. 2) Students must have enough money to pay for the tuition in the system. **Tutors** 1) Tutors must register as a tutor in the system. 2) Tutors must have at least a subject to teach the students in the system. **Admin** 1) Admin must have at least a booking record in the database.
Post-Conditions:	**Students** 1) Students have officially registered into the system. 2) Students book tutors for tuition. 3) Students will received a receipt and it will be kept into their personal account.

	4) All the bookings will be stored into the history of users personal account. **<u>Tutors</u>** 1) Tutors have officially registered into the system. 2) Tutors will be able to start teaching students. 3) Tutors will received a receipt and it will be kept into their personal account. 4) All the bookings will be stored into the history of users personal account. **<u>Admin</u>** 1) Admin able to see a list of booking records of all users in the system. 2) Admin able to generate a list of booking records of all users in the system.
Assumption:	1) System is functioning. 2) Internet connection is available for database connection

Use Case Name:	Manage User and Full Application
Actor:	Admin
Trigger Condition:	The users information will be stored into the system which the admin will manage the user account and information without being leaked.
Summary:	This use case allows the admin to fully control the users information given when registered. The admin will verified the tutors that wanted to registered as a tutor in the system.
Basic Course:	1) The admin can generate a list of users from the database. 2) The admin can keep the information without being leaked. 3) The admin allows to edit all users information in the system.
Alternative Scenario:	
Exception Scenario:	1) Admin unable to manage non existing user information in the database. 2) Admin is not able to check non existing user information in the database.
Pre-Conditions:	1) Users must create an account in the database. 2) The database must be able to store users information. 3) Admin must be active to manage the database of the system.
Post-Conditions:	1) Admin is able to generate a list of users in the database. 2) Admin is able to edit the users information in the database. 3) Admin is able to add and delete users in the database.
Assumption:	1) System is functioning. 2) Internet connection is available for database connection.

Use Case Name:	Give/Received Feedback
Actor:	Admin, Students, Tutors
Trigger Condition:	**Students** Students provide ratings and feedbacks to about the tutor. **Tutors** Tutors are able to see the ratings and feedback that the students that have given them in their personal account. **Admin** Admin is able to see the ratings and feedback that the students given to their tutor.
Summary:	This use case allows students to provide rating and feedbacks to existing tutors in the system and they are allowed to see the ratings and feedback given in the tutor profile. Whereas Tutors and Admin are only allowed to see the ratings and feedback that the students have given to the tutors.
Basic Course:	**Students** 1) Students providing ratings and feedback to the tutors. 2) The use case is terminated. **Tutors** 1) Tutors are only allowed to see the ratings and feedback given from their students. 2) The use case is terminated. **Admin** 1) Admin is allowed to see the tutors ratings and feedback given from their students. 2) The use case is terminated.
Alternative Scenario:	
Exception Scenario:	**Students** 1) Students unable to provide ratings and feedback to non existing tutor in the system. **Tutors** 1) Tutors unable to check ratings and feedback from a non existing student in the system.

	Admin 1) Admin unable to look through all the ratings and feedback about the tutor from their students in the system.
Pre-Conditions:	**Students** 1) Students must register an account in the system. 2) Students must access into their personal account to rate the tutor in the tutors profile. **Tutors** 1) Tutors must register an account in the system. 2) Tutors must access into their personal account to check the ratings and feedback given by their students. **Admin** 1) The database must have at least an existing user in the system for the admin to check the ratings and feedbacks given by the users.
Post-Conditions:	**Students** 1) Students are able to see the given ratings and feedback in the tutors profile. **Tutors** 1) Tutors are able to see the given ratings and feedback from the students in the tutors personal profile. **Admin** 1) Admin are able to check all the students given ratings and feedback about their tutors in the database.
Assumption:	1) System is functioning. 2) Internet connection is available for database connection.

Use Case Name:	Add Subject
Actor:	Tutor
Trigger Condition:	Tutors are able to add in more subjects into their teaching list.
Summary:	This use case allows the tutor to add in more subjects into their teaching list.
Basic Course:	Tutors are allowed to add in more subjects into their teaching list, which allows the students to have options on picking which tutor to guide them in that particular subject.
Alternative Scenario:	Users who registered as a tutor in the system are allowed to request the admin to add in more subjects into the system which allows tutors to have more options on adding in more subjects into their teaching list.
Exception Scenario:	Users are not registered as a tutor in the system do not have the ability to add in more subjects into their teaching list.
Pre-Conditions:	1) Users must register as a tutor in the system. 2) Users who registered as a tutor in the system must access into their personal account to be able to add in more subjects into their teaching list. 3) Tutors are allowed to add in subjects from the system and before adding in more subjects into the tutors teaching list, tutors must set a price tag of the particular subject.
Post-Conditions:	1) All users are able to check the tutor of which subject they are able to teach for tuition in the system.
Assumption:	1) System is functioning. 2) Internet connection is available for database connection.

Use Case Name:	Create timetable
Actor:	Tutor
Trigger Condition:	Tutors are able to create their own timetable in the system.
Summary:	This use case allows tutors in the system to create their own timetable.
Basic Course:	Tutors are able to create their own timetable in the system this allows the users to plan their working days and their own off days.
Alternative Scenario:	
Exception Scenario:	Users are not registered as a tutor in the system are not able to create their own timetable.
Pre-Conditions:	1) Users must register as a tutor in the system. 2) Users who registered as a tutor in the system must access into their personal account in order to create their own timetable. 3) Tutors can set their available time in the timetable.
Post-Conditions:	1) All users are able to check when the tutor is available for tuition in the timetable given from the system.
Assumption:	1) System is functioning. 2) Internet connection is available for database connection.

4.1.4 Gantt Chart

A Gantt Chart is frequently used in project management, a Gantt chart provides a graphical illustration of a schedule that helps to plan, coordinate, and track specific tasks in a project. Gantt charts may be simple versions created on graph paper or more complex automated versions created. (Margaret Rouse, 2007)

The Gantt Chart was created before the start of development of the proposed system. This is done so that the development team can keep track of time on developing the proposed system. (see Appendix A.3 for the Gantt Chart)

4.2 Database Design

4.2.1 ER Diagram

An entity relationship diagram (ERD), also known as an entity relationship model, is a graphical representation of an information system that depicts relationships within that system. An ERD is a data modeling technique that can help define business processes and be used as the foundation for a relational database. Entity relationship diagrams provide a visual starting point for database design that can also be used to help determine information system requirements throughout an organization.

The development team came up with an idea of creating a system which students can search for tutors online. In order for the system to work internally and externally, we started creating ER Diagram of the system. During the first stage of designing the system, the development team already has figured out which entities are needed in our system which are student, tutor, subjects, subject offer, payment info, non available, favourite and rating. These 8 entities are the most important entities in the system the students and tutors play the role of holding user's personal information. Subject and subject offer relates with tutors entities because the subject entities stores all available subjects in the system whereas subject offer stores all information about which tutor has the ability to teach what kind of subject in the system and as well as storing each subject price tag in the system. In addition the development team has created the payment info entities because it is necessary for the system because our system is based on students booking tutors for tuition online in our system. Non available entity is for tutor creating their own timetable which means that tutors can take many off days as they want but they needed to create a timetable which shows their availability to all students in the system. Favourite entity is to store all information regarding students adding tutor as their favourite tutor in the system. This defines that students has their own prefered tutor for them to pick in the system. Whereas the rating function works as the measurement on how students judge the tutor based on his teachings in tuition and

his favorablility. Our system provides students the most recommended tutors in our system based on the ratings.

The remaining entities in the ER Diagram were created during at a discussion with the team. At first the discussion on the remaining entities was not like the exact one we have now but at that moment we decided to add in the chat function into our system so that students and tutors can communicate with each other in the system. The team came up with an idea that all chat must be recorded in our system so the team decided to add in an entity named chat message this entity is to record all chat message between users in the system. This is to backup all chat message into our database so that user would not lost all conversation information with other users.

Last but not least the testing card entity is created for online booking purpose. It does not link to any entity in the diagram. The role of this entity does not affect much in the system because our development team wanted to have an experiment with a permanent online purchasing.

In conclusion most of the diagram entities were already sought out on the early phase of whereas only a few entity were created during the process of creating the system. The development team did not face any trouble of creating the ER Diagram.

4.2.2 Data Dictionary

chat

Name	Type	Null	Default	Comments	Extra
chat_id	int(100)	No			AUTO_INCREMENT
stud_id	int(100)	No			
tutor_id	int(100)	No			

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	chat_id	0	A	No	
stud_id	BTREE	No	No	stud_id	0	A	No	
tutor_id	BTREE	No	No	tutor_id	0	A	No	

chatmessages

Name	Type	Null	Default	Comments	Extra
message_id	int(11)	No			AUTO_INCREMENT
sender_name	varchar(50)	No			
message	text	No			
m_timestamp	timestamp	No	CURRENT_TIMESTAMP		
chat_id	int(100)	No			
m_viewed	tinyint(1)	No			

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	message_id	3	A	No	
chat_id	BTREE	No	No	chat_id	3	A	No	

nonavailable

Name	Type	Null	Default	Comments	Extra
nonavailable_id	int(11)	No			AUTO_INCREMENT
nonavailable_date	date	No			
booked	tinyint(1)	No			
tutor_id	int(100)	No	CURRENT_TIMESTAMP		
start_time	time	No			
duration	int(10)	No			

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	nonavailable_id	7	A	No	
tutor_id	BTREE	No	No	tutor_id	2	A	No	

payment_info

Name	Type	Null	Default	Comments	Extra
payment_id	int(100)	No			AUTO_INCREMENT
paytime	timestamp	No	CURRENT_TIMESTAMP		
payamount	int(30)	No			
stud_id	int(100)	No			
tutor_id	int(100)	No			
sub_name	varchar(100)	No			
bookingdate	datetime	No			
appoint_hrs	int(11)	No			
status	enum('pending','cancel','success'	No			
cancel_reason	carchar(100)	Yes	NULL		

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	payment_id	9	A	No	
tutor_id	BTREE	No	No	tutor_id	2	A	No	
stud_id	BTREE	No	No	stud_id	4	A	No	

rating

Name	Type	Null	Default	Comments	Extra
rating_id	int(100)	No			AUTO_INCREMENT
stud_id	int(100)	No			
r_stud_firstname	varchar(50)	No			
r_message	text	No			
r_stars	int(1)	No			
r_datetime	datetime	No			
tutor_id	int(100)	No			

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	rating_id	0	A	No	
stud_id	BTREE	No	No	stud_id	0	A	No	
tutor_id	BTREE	No	No	tutor_id	0	A	No	

report_issues

Name	Type	Null	Default	Comments	Extra
report_id	int(11)	No			AUTO_INCREMENT
report_time	timestamp	No	CURRENT_TIMESTAMP		ON UPDATE CURRENT_TIMESTAMP
report_msg	varchar(100)	No			
studtutor_id	int(100)	No			
solvestatus	tinyint(1)	No			

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	report_id	0	A	No	
studtutor_id	BTREE	No	No	studtutor_id	0	A	No	

student

Name	Type	Null	Default	Comments	Extra
stud_id	int(100)	No			AUTO_INCREMENT
stud_fname	varchar(50)	No			
stud_lname	varchar(50)	No			
stud_email	varchar(100)	No			
stud_usrn	varchar(20)	No			

stud_pass	varchar(100)	No			
stud_hash	varchar(32)	No			
stud_active	tinyint(1)	No	0		
stud_travel	int(2)	No			
stud_zip	int(5)	No			
stud_addr	varchar(100)	No			
stud_phone	int(10)	No			
stud_profilepic	varchar(100)	No	default.png		
stud_gender	enum('M','F','O')	No			
stud_intro	text	No			
stud_birthdate	date	No			
stud_registerdate	datetime	No			
stud_lastin	timestamp	No	CURRENT_TIMESTAMP		

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	stud_id	3	A	No	

subject

Name	Type	Null	Default	Comments	Extra
subject_id	int(11)	No			AUTO_INCREMENT
subject_name	varchar(100)	No			
categories	varchar(100)	No			

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	subject_id	28	A	No	

subject_offer

Name	Type	Null	Default	Comments	Extra
offer_id	int(100)	No			AUTO_INCREMENT
subject_id	int(100)	No			
tutor_id	int(100)	No			
price_perhs	int(10)	No			

Key Name	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	offer_id	9	A	No	
tutor_id	BTREE	No	No	tutor_id	9	A	No	
subject_id	BTREE	No	No	subject_id	9	A	No	

testing_card

Name	Type	Null	Default	Comments	Extra
card_num	bigint(255)	No			
exp_d	int(4)	No			
cvc	int(3)	No			
amount	int(255)	No			

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
card_num	BTREE	Yes	No	card_num	1	A	No	

tutor

Name	Type	Null	Default	Comments	Extra
tutor_id	int(100)	No			AUTO_INCREMENT
t_fname	varchar(50)	No			
t_lname	varchar(50)	No			
t_email	varchar(100)	No			
t_usrn	varchar(20)	No			
t_pass	varchar(100)	No			
t_hash	varchar(32)	No			
t_active	tinyint(1)	No	0		
t_travel	int(2)	No			
t_zip	int(5)	No			
t_district	varchar(40)	No			

t_addr	varchar(100)	No			
t_phone	int(10)	No			
t_profilepic	varchar(100)	No	default.png		
t_gender	enum('M','F','O')	No			
t_intro	text	No			
t_job	varchar(50)	No			
t_birthdate	date	No			
t_qualifications	varchar(400)	No			
t_officalcheck	tinyint(1)	No			
t_registerdate	datetime	No			
t_lastin	timestamp	No	0000-00-00 00:00:00		

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	tutor_id	3	A	No	

4.3 User-Interface Design

4.3.1 Rough Wireframe of System

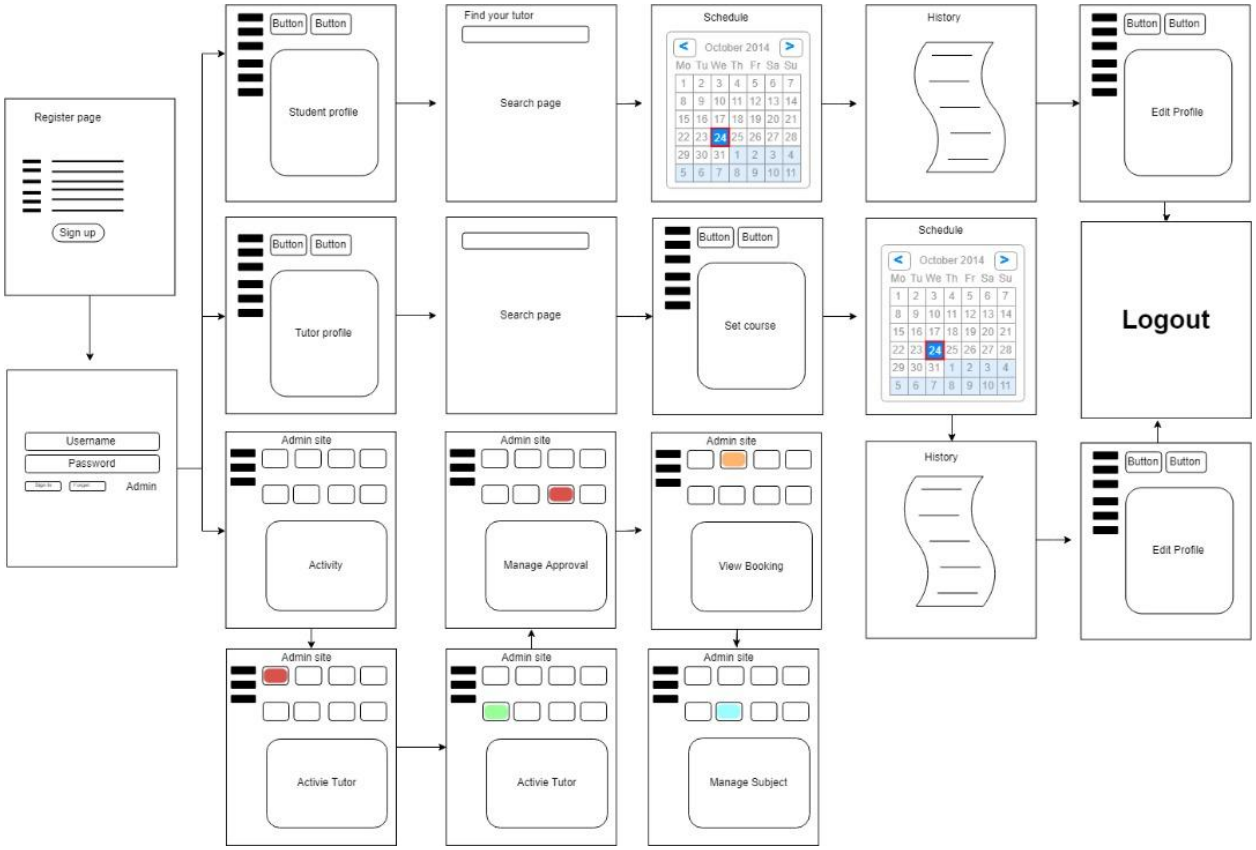


Figure 6 : Rough Wireframe of System

The figure above shows a rough wireframe of the proposed system, there are three different sites to be accessible for three different users. This figure will explain and show every function that can be accessible by three different types of users. There are more web pages to be access by the users. The wireframe figure only serves the purpose to show the main pages of the system.

4.3.2 Login Form

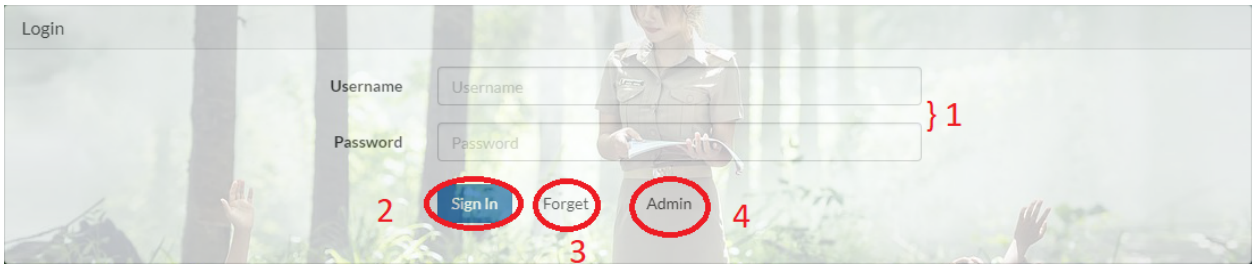


Figure 7: Login

This initial design of the login form has been planned by the development team

1. Users will enter their registered user ID and passwords in this section

2. After inserting credentials, login button must be clicked to continue. The system will then go through records of registered account in the database and match the ID and password entered. If the ID and password is matched, the system will then direct users to the home page

3. If users forgot their password, they can choose to reset their password by clicking the 'Forget' button. The page will automatically redirect users to the reset password page where users will have to enter their email.

4. This is the admin button where only admin are allowed to access but for now our development team did not apply any restriction to the Admin security.

4.3.3 Navigation Bar

Figure 8: Navigation bar of non-registered users

Figure 9: Navigation bar of registered users

This will be the navigation bar that is used for the web page. There will be some differences according to different users.

1. By clicking on the 24hrs Tutor logo, it will redirect users to the home page.

2. The search function is included in the navigation bar. Users will be able to search for tutors by clicking on the search button. But users will have to login to their accounts before they can search.

3. Users will be able to login or register from the selection provided on the navigation bar.

4. This allows user to reset their password in the system. But user has to submit their email address then the system will provide a reset password into your personal email.

5. If users clicked into the search button, it will redirect the user to the search page.

6. The main menu button is linked back to the user's main page.

4.3.4 Search form

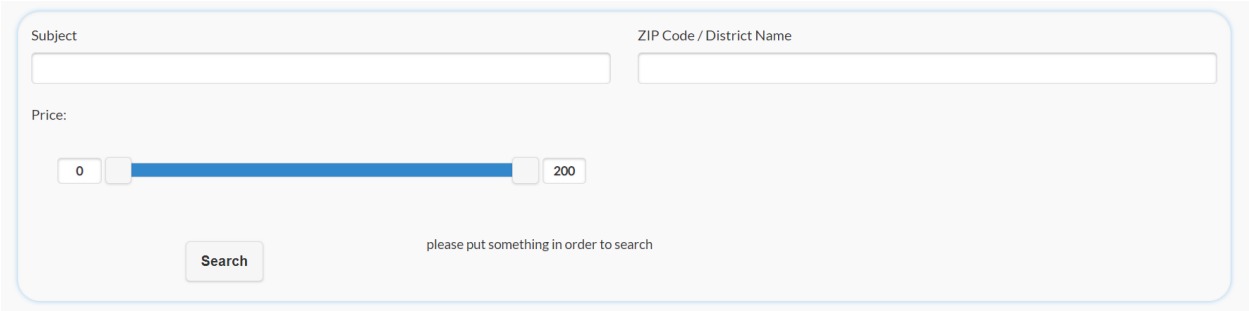


Figure 10: Searching form

The figure above shows the searching form of the website. The search form serves the purpose to help students who are in search of tutors. The search form will be filtering tutors based on the selected preferences by the student.

4.4 Chapter conclusion

With the help of the diagrams, the development team is able to come out with the main ideas for the system . The development team started to implement ideas that are presented throughout Chapter 3 and 4 to complete the system. The implementation stage can be viewed in the following chapter.

Chapter 5: Implementation

This chapter contains two major sections, Coding and Integration of the developed system and the Installation of the system. The coding section will go through the programming language that are used to develop the system. And as for the integration section, it includes the usage to technical method and functions on different important parts of the project. Next, the installation section will be guiding users who wanted to use the proposed system the process of installing the system and database to a server.

Coding/Integration

5.1 Selection of programming language

The proposed system will be a web based application designed for the convenience of users. A web application is any computer program that performs a specific function by using a web browser as its client. (Daniel Nations, 2018). It is important to utilize the right technology in order to develop a workable web application. Technologies such as programming languages, databases, hosting servers and frameworks are an important aspect and must be taken into detail consideration in order to come out with a good web application.

The development team had several discussions and had land foot on which programming languages that are suitable for the desired system. HTML, CSS, JavaScript, PHP, PDO and SQL are implemented into the system. The team had also taken two different browsers such as Chrome and Safari into consideration. The programming languages chosen are both supported by the two browsers and that makes the system a multi-platform web application whereby users can have access no matter the browsers they are using. Of course, users will need to have an internet connection in order to connect to the system. Moreover, the team had decided to make use of Visual Studio Code as the framework to manage codings and files. As for the database, MySQL is chosen by the team and the hosting server will be provided by KDU University College Penang. Hence, pHpMyAdmin was used to manage database on the server while FileZilla is used to manage files on the server.

5.1.1 HTML (Hypertext Markup Language), CSS (Cascading Style Sheets), JavaScript

As we all know, HTML CSS and JavaScript are the basic essentials to build a web based application. HTML consists of a series of short codes typed into a text-file and was then translated the text into a visible form so that your Internet browser may display them as they are intended to look. (Ross Shannon, 2012). HTML codes can be enhanced and modified by using CSS and JavaScript. The implementation of CSS codes will be used to beautify and enhance the beauty of the interface. Meanwhile,

JavaScript will be used to control the behaviour of different elements such as animations, pop-up messages and so on.

5.1.2 PHP (Hypertext Preprocessor)

PHP is a script language and a simple yet powerful tool for making interactive web pages. As we know, PHP is widely used around the globe to create stunning websites.

Furthermore, PHP codes can be embedded into HTML codes and it is helpful when displaying the function of the codes. PHP also works well in managing data in the database. Most importantly, PHP is a programming language that is easy to use and understandable by the development team.

5.1.3 SQL (Structured Query Language)

Storage is a definite requirement to keep data and information, therefore a database is needed. SQL plays an important role in managing database table and index structures. Therefore, SQL is a mandatory choice to store important data and to manage them easily.

5.2 Integration methods

This section will cover the integration methods that are used in the important parts of the project.

PHP

Login

```php
<?php
  //start a session
  session_start();
  require_once("csrf.php");
  if(isset($_SESSION["username1"]) && isset($_SESSION["identity"]) && $_SESSION["actype"])
  {
    header("location: login_success.php");
  }else{
  }
?>
```

Figure 11 Login

Figure 11 includes codings where users will have to key in their username and password. If the input has the same username and password from the database, the user is login to their perspective account or else the user cannot access into their perspective account.

```php
<?php

session_start();
require_once 'db.php';
require_once("csrf.php");
$old_sessionid = session_id();
session_regenerate_id();
$new_sessionid = session_regenerate_id(true);

if (isset($_POST['login'])) {
    if (hash_equals($csrf, $_POST['csrf'])) {

    $username = ($_POST['username1']);
    $password = ($_POST['password1']);

$stmt1 = $conn->prepare("SELECT * FROM tutor WHERE t_usrn=:username AND t_active ='1'");
$stmt1->bindParam(":username",$username);
$stmt1->execute();

$result = $stmt1->fetch(PDO::FETCH_ASSOC);


$stmt2 = $conn->prepare("SELECT * FROM student WHERE stud_usrn=:username AND stud_active ='1'");
$stmt2->bindParam(":username",$username);
$stmt2->execute();

$result2 = $stmt2->fetch(PDO::FETCH_ASSOC);

if ($result >0 ||$result2 >0){
   //echo "horray";
        $exit ='1';
        if(password_verify($password,$result['t_pass'])){
            echo"1";
            $_SESSION["username1"] = $result['t_usrn'];
            $_SESSION["identity"] = $result['tutor_id'];
            $_SESSION["firstname"] = $result['t_fname'];
            $_SESSION["lastname"] = $result['t_lname'];
            $_SESSION["actype"] = "tutor";
            $_SESSION['expire'] = $_SESSION['start'] + (30 * 60);
            $stmt3 = $conn->prepare("UPDATE `tutor` SET `t_lastin`=CURRENT_TIMESTAMP WHERE `tutor_id` =:id");
            $stmt3->bindParam(":id",$_SESSION["identity"]);
            $stmt3->execute();
            header("location:login_success.php");
            $exit ='0';
        }

        if(password_verify($password,$result2['stud_pass'])){
            echo"1";
            $_SESSION["username1"] = $result2['stud_usrn'];
            $_SESSION["identity"] = $result2['stud_id'];
            $_SESSION["firstname"] = $result2['stud_fname'];
            $_SESSION["lastname"] = $result2['stud_lname'];
            $_SESSION["actype"] = "student";
            $_SESSION['expire'] = $_SESSION['start'] + (30 * 60);
            $stmt4 = $conn->prepare("UPDATE `student` SET `stud_lastin`=CURRENT_TIMESTAMP WHERE `stud_id` =:id");
            $stmt4->bindParam(":id",$_SESSION["identity"]);
            $stmt4->execute();

            header("location:login_success.php");
            $exit ='0';
        }
        if ($exit =='1'){
            session_destroy();
            header('location: loging.php?fail=1');
        }

}else if($result <0 || $result2 <0){
    session_destroy();
    header('location: loging.php?fail=1');            // Tells Unity it wasn't successful, so to try again.
}else{
    session_destroy();
    header('location: loging.php?fail=1');
}
    }else{
    session_destroy();
    header('location: loging.php?fail=1');
    }
}
```

Figure 12 Matching the username and password in the database

The codings above serves the purpose to match the username and password that users have enter with the information in database. If the information given is matched with the information from the database then it will grant the user the permission to access into their account in the system.

```php
if (isset($_POST["submit"])){
  if (hash_equals($csrf, $_POST['csrf'])) {
  //DEBUG   echo "running";
    try{
    if ($_SESSION["actype"] ==="tutor"){
        $stmt_tutor = $conn->prepare("UPDATE `tutor` SET `t_zip`=:zip,`t_addr`=:addr ,`t_phone`=:pho , `t_birthdate`=:bd , `t_gender`=:
        `t_travel`=:tv , `t_intro`=:intro , `t_job`=:job , `t_qualifications`=:ql , `t_district`=:dst WHERE `tutor_id` =:id");
        $stmt_tutor->bindParam(':zip', $_POST["zip"] , PDO::PARAM_STR);
        $stmt_tutor->bindParam(':addr', $_POST["address"], PDO::PARAM_STR);
        $stmt_tutor->bindParam(':pho', $_POST["phone"], PDO::PARAM_STR);
        $stmt_tutor->bindParam(':bd', $_POST["birthd"] , PDO::PARAM_STR);
        $stmt_tutor->bindParam(':gd', $_POST["gender"] , PDO::PARAM_STR);
        $stmt_tutor->bindParam(':tv', $_POST["travel"] , PDO::PARAM_STR);
        $stmt_tutor->bindParam(':intro', $_POST["intro"] , PDO::PARAM_STR);
        $stmt_tutor->bindParam(':job', $_POST["job"] , PDO::PARAM_STR);
        $stmt_tutor->bindParam(':ql', $_POST["qualify"] , PDO::PARAM_STR);
        $stmt_tutor->bindParam(':dst', $_POST["dist"] , PDO::PARAM_STR);
        $stmt_tutor->bindParam(':id', $_SESSION["identity"] , PDO::PARAM_STR);

        $stmt_tutor->execute();
    }
```

Figure 13 Login success tutor account

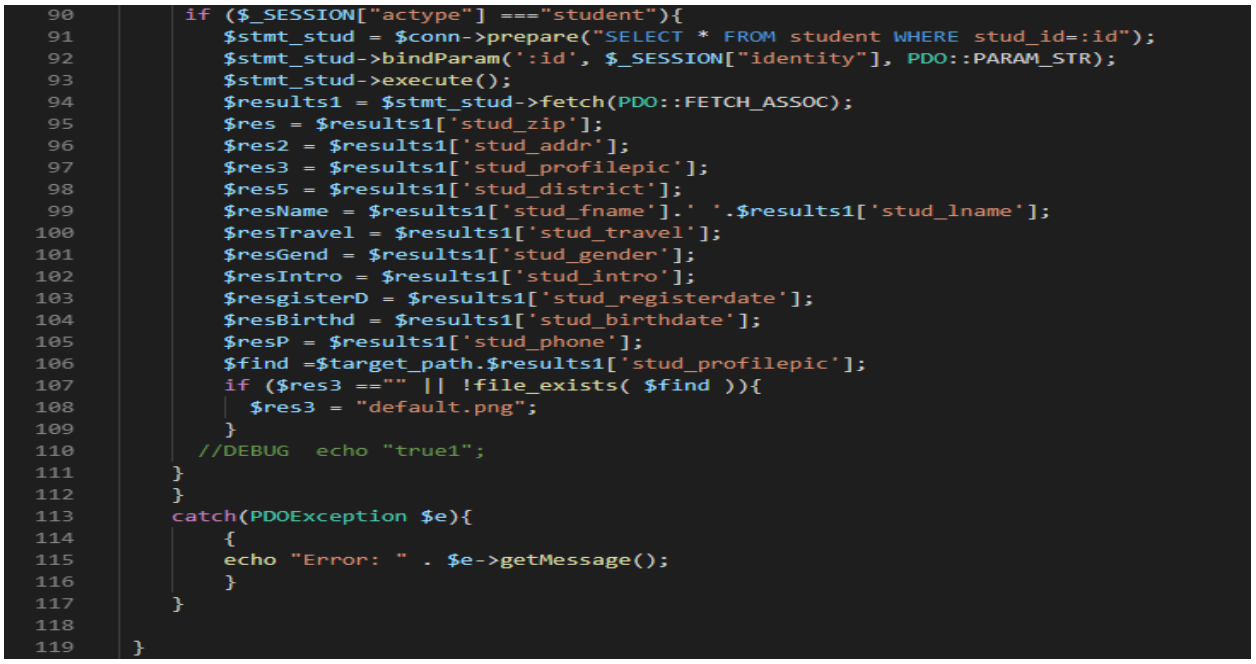
```php
    if ($_SESSION["actype"] ==="student"){
        $stmt_stud = $conn->prepare("UPDATE `student` SET `stud_zip`=:zip , `stud_addr`=:addr ,`stud_phone`=:pho , `stud_birthdate`=:bd ,
        `stud_gender`=:gd , `stud_travel`=:tv , `stud_intro`=:intro , `stud_district`=:dst WHERE `stud_id`=:id");
        $stmt_stud->bindParam(':zip',  $_POST["zip"], PDO::PARAM_STR);
        $stmt_stud->bindParam(':addr',  $_POST["address"], PDO::PARAM_STR);
        $stmt_stud->bindParam(':pho', $_POST["phone"], PDO::PARAM_STR);
        $stmt_stud->bindParam(':bd',  $_POST["birthd"], PDO::PARAM_STR);
        $stmt_stud->bindParam(':gd',  $_POST["gender"], PDO::PARAM_STR);
        $stmt_stud->bindParam(':tv', $_POST["travel"] , PDO::PARAM_STR);
        $stmt_stud->bindParam(':intro', $_POST["intro"] , PDO::PARAM_STR);
        $stmt_stud->bindParam(':dst', $_POST["dist"] , PDO::PARAM_STR);
        $stmt_stud->bindParam(':id',  $_SESSION["identity"], PDO::PARAM_STR);

        $stmt_stud->execute();
        }
        }

        catch(PDOException $e){
        {
        echo "Error: " . $e->getMessage();
        }
    }
  } else{
  echo 'CSRF Token Failed!';
}
}
```

Figure 14 Login Success student account

Figure 13 and Figure 14 are about once the user has submit their given information to the system then the system will login into their personal account, the code is based on the user submit their given information to the system and the system will justify what type of user they are in the system. In our system there is only three types of account of user which are Students, Tutors and Admin but in Figure 13 and Figure 14 Admin side is not included together with this part of coding.

```php
if(isset($_SESSION["username1"]) && isset($_SESSION["identity"]) && $_SESSION["actype"])
{

   try {
      $zip ="";
    if ($_SESSION["actype"] ==="tutor"){
      $stmt_tutor = $conn->prepare("SELECT * FROM tutor WHERE tutor_id=:id");
      $stmt_tutor->bindParam(':id', $_SESSION["identity"], PDO::PARAM_STR);
      $stmt_tutor->execute();
      $results1 = $stmt_tutor->fetch(PDO::FETCH_ASSOC);
      $res = $results1['t_zip'];
      $res2 = $results1['t_addr'];
      $res3 = $results1['t_profilepic'];
      $res4 = $results1['t_qualifications'];
      $res5 = $results1['t_district'];
      $resName = $results1['t_fname'].' '.$results1['t_lname'];
      $resTravel = $results1['t_travel'];
      $resGend = $results1['t_gender'];
      $resIntro = $results1['t_intro'];
      $resBirthd = $results1['t_birthdate'];
      $resgisterD = $results1['t_registerdate'];
      $resJob = $results1['t_job'];
      $approval = $results1['t_officialcheck'];
      $find =$target_path.$results1['t_profilepic'];
      $resP = $results1['t_phone'];
      if ($res3 =="" || !file_exists( $find )){
        $res3 = "default.png";
      }
     //DEBUG  echo "true<br>";
    }
```

Figure 15 Login success part 1



```php
    if ($_SESSION["actype"] ==="student"){
       $stmt_stud = $conn->prepare("SELECT * FROM student WHERE stud_id=:id");
       $stmt_stud->bindParam(':id', $_SESSION["identity"], PDO::PARAM_STR);
       $stmt_stud->execute();
       $results1 = $stmt_stud->fetch(PDO::FETCH_ASSOC);
       $res = $results1['stud_zip'];
       $res2 = $results1['stud_addr'];
       $res3 = $results1['stud_profilepic'];
       $res5 = $results1['stud_district'];
       $resName = $results1['stud_fname'].' '.$results1['stud_lname'];
       $resTravel = $results1['stud_travel'];
       $resGend = $results1['stud_gender'];
       $resIntro = $results1['stud_intro'];
       $resgisterD = $results1['stud_registerdate'];
       $resBirthd = $results1['stud_birthdate'];
       $resP = $results1['stud_phone'];
       $find =$target_path.$results1['stud_profilepic'];
       if ($res3 =="" || !file_exists( $find )){
         $res3 = "default.png";
       }
      //DEBUG  echo "true1";
    }
    }
    catch(PDOException $e){
        {
        echo "Error: " . $e->getMessage();
        }
    }

}
```

Figure 16 Login Success part 2

The two figures above shows the codings after the matching of id has been processed, users will be redirected to their profile page according to their id entered. Once the

given information is submitted to the system, the system will bring the user to their correspond ID. For example if you submit your student user ID and password into the system, the system will examine the given information is match exactly the same information from the system then the system will grant the user the permission to access into their perspective account.

Searching

```php
if(isset($_SESSION["username1"]) && isset($_SESSION["identity"]) && $_SESSION["actype"])
{
 if (isset($_POST["submit"])){
      $val1 = $_POST["subject"];
      $val2 = $_POST["zip"];
      $val3 = $_POST["price-min"];
      $val4 = $_POST["price-max"];

     if (ctype_digit($val2) && preg_match('#[0-9]{5}#', $val2)){
      $query1 = "SELECT d.tutor_id ,`t_fname` ,`t_lname` , `t_profilepic`, `t_travel` , `t_job`, `t_intro`
      FROM subject f ,subject_offer e, tutor d
      WHERE e.subject_id =f.subject_id
      AND f.subject_name = :subject
      AND d.t_zip =:zip
      AND d.tutor_id = e.tutor_id
      AND e.price_perhrs BETWEEN :min AND :max";
      //$zipstate = "d.t_zip";

      $search=$conn->prepare($query1);
      $search->bindParam(':subject', $val1, PDO::PARAM_STR);
      $search->bindParam(':zip', $val2, PDO::PARAM_INT);
      $search->bindParam(':min', $val3, PDO::PARAM_INT);
      $search->bindParam(':max', $val4, PDO::PARAM_INT);
      $search->execute();
      $total = $search->rowCount();
     }else{
      $query1 = "SELECT d.tutor_id ,`t_fname` ,`t_lname` , `t_profilepic`, `t_travel` , `t_job`, `t_intro`
      FROM subject f ,subject_offer e, tutor d
      WHERE e.subject_id =f.subject_id
```

Figure 17 Searching part 1

```php
            AND f.subject_name = :subject
            AND d.t_district =:zip
            AND d.tutor_id = e.tutor_id
            AND e.price_perhrs BETWEEN :min AND :max";
            //$zipstate = "d.t_zip";
            $search=$conn->prepare($query1);
            $search->bindParam(':subject', $val1, PDO::PARAM_STR);
            $search->bindParam(':zip', $val2, PDO::PARAM_STR);
            $search->bindParam(':min', $val3, PDO::PARAM_INT);
            $search->bindParam(':max', $val4, PDO::PARAM_INT);
            $search->execute();
            $total = $search->rowCount();
           }

     }

     }
    else
    {
        session_destroy();
        header("location:loging.php");
    }
    $old_sessionid = session_id();
    session_regenerate_id();
    $new_sessionid = session_regenerate_id(true);
    ?>
```

Figure 18 Searching part 2

The codings above shows the searching function of the system. Students will be able to search for their idea tutor. Users have to fill up the required information so that the system will be able to search through the database based on the information provided by the user then it will return a result based on the information provided. Line 8 to line 12 is about if user submit the given the information then it will proceed to the next line which is line 14.

Register

SQL

```
5   $_SESSION['email'] = $_POST["username1"];
6   $_SESSION['first_name'] = $_POST["firstname1"];
7   $_SESSION['last_name'] =  $_POST["lastname1"];
8
9   // Value that use to process
10  $firstname = $_POST["firstname1"];
11  $lastname = $_POST["lastname1"];
12  $email = $_POST["email1"];
13  $usrn = $_POST["username1"];
14  $pass =  $_POST["password1"];
15  $str = $_POST["radio"];
16  //Hash the password as we do NOT want to store our passwords in plain text.
17  $passwordHash = password_hash($pass, PASSWORD_BCRYPT);
18  $hash = md5( rand(0,1000) ) ;
19  $msg ="";
```

Figure 19 Declare

In Figure 19 this part is mostly declaring the names in SQL and setting the password unable to send plain text to the database. Line 17- 19 is for preventing user to set empty password into our database. This forces the user to create a password instead of just submitting a plain text to the database.

```php
if (isset($_POST['signup1'])) {
if (hash_equals($csrf, $_POST['csrf'])) {
try {

    $stmt_semail = $conn->prepare("SELECT * FROM student WHERE stud_email=:email");
    $stmt_temail = $conn->prepare("SELECT * FROM tutor WHERE t_email=:email");
    $stmt_susrn = $conn->prepare("SELECT * FROM student WHERE stud_usrn=:usrn");
    $stmt_tusrn = $conn->prepare("SELECT * FROM tutor WHERE t_usrn=:usrn");

    $stmt_semail->bindParam(':email', $email, PDO::PARAM_STR);
    $stmt_temail->bindParam(':email', $email, PDO::PARAM_STR);
    $stmt_susrn->bindParam(':usrn', $usrn, PDO::PARAM_STR);
    $stmt_tusrn->bindParam(':usrn', $usrn, PDO::PARAM_STR);
    $stmt_semail->execute();
    $stmt_temail->execute();
    $stmt_susrn->execute();
    $stmt_tusrn->execute();
    $results1 = $stmt_semail-> fetch();
    $results2 = $stmt_temail-> fetch();
    $results3 = $stmt_susrn-> fetch();
    $results4 = $stmt_tusrn-> fetch();
}
catch(PDOException $e){
    {
    echo "Error: " . $e->getMessage();
    }
}
```

Figure 20 User input

In this section, users will be typing the input of their username and email. If their input is invalid, the code will show an “Error” message. Line 24 - line 27 select the student or tutor table about their username and email. Line 29 - line 40 is to store the parameter into the correspond student or tutor table email and username. Then it will execute the result and bring the data to the front row into the correspond table.

```php
48  if ($results1 >0 || $results2>0 || $results3 >0 || $results4 >0){
49
50      header('location: registering.php?fail=3');
51      if ($results1 >0 ||$results2 >0){
52          header('location: registering.php?fail=1');
53      }
54      if ($results3 >0 || $results4 >0){
55          header('location: registering.php?fail=2');
56      }
57      if($results1 >0 && $results3 >0 || $results2 >0 &&$results4 >0){
58          header('location: registering.php?fail=3');
59      }
60
61  }else {
```

Figure 21 Register failed and link to other page for output result

In this section, when the user failed to register an account in the system,the system will direct link to Figure 24 where it will display the output result of the failed registered account.

```php
61  }else {
62      if($str=="student"){
63          try {
64
65              // prepare sql and bind parameters
66          $stmt = $conn->prepare("INSERT INTO student (stud_fname, stud_lname, stud_email, stud_usrn, stud_pass, stud_hash,
67                                  stud_registerdate)
68          VALUES (:firstname, :lastname, :stud_email, :stud_usrn, :stud_pass, :stud_hash , NOW())");
69          $stmt->bindParam(':firstname', $firstname,PDO::PARAM_STR);
70          $stmt->bindParam(':lastname', $lastname,PDO::PARAM_STR);
71          $stmt->bindParam(':stud_email', $email,PDO::PARAM_STR);
72          $stmt->bindParam(':stud_usrn', $usrn,PDO::PARAM_STR);
73          $stmt->bindParam(':stud_pass', $passwordHash,PDO::PARAM_STR);
74          $stmt->bindParam(':stud_hash', $hash,PDO::PARAM_STR);
75          // insert a row
76          $stmt->execute();
77          sendActiveMail($email,$firstname,$hash,$str);
78          header('location: registering.php?success=1');
79          }
80      catch(PDOException $e)
81          {
82          echo "Error: " . $e->getMessage();
83          }
84          $conn = null;
85      }
```

Figure 22 Student register successfully

After users entered the desired field correctly and have submited their information to the database, the system will send an email to verify the account. This code will be carried out if users choose student account in the selection. Once the user have chosen student account as their preferable account and the given information is appropriate then in line 77 it will lead user to Figure 25 where it will display the output of the outcome of the registration.

```php
    if($str=="tutors"){
        try {
            // prepare sql and bind parameters
        $stmt = $conn->prepare("INSERT INTO tutor (t_fname, t_lname, t_email, t_usrn, t_pass, t_hash , t_registerdate)
        VALUES (:firstname, :lastname, :t_email, :t_usrn, :t_pass, :t_hash , NOW())");
        $stmt->bindParam(':firstname', $firstname,PDO::PARAM_STR);
        $stmt->bindParam(':lastname', $lastname,PDO::PARAM_STR);
        $stmt->bindParam(':t_email', $email,PDO::PARAM_STR);
        $stmt->bindParam(':t_usrn', $usrn,PDO::PARAM_STR);
        $stmt->bindParam(':t_pass', $passwordHash,PDO::PARAM_STR);
        $stmt->bindParam(':t_hash', $hash,PDO::PARAM_STR);

        $stmt->execute();
        sendActiveMail($email,$firstname,$hash,$str);
        header('location: registering.php?success=1');
        }
    catch(PDOException $e)
        {
        echo "Error: " . $e->getMessage();
        }
        $conn = null;
    }
}
}else{
    session_destroy();
    header('location: registering.php?fail=1');
}
}
```

Figure 23 Tutor register successfully

After users entered the desired field correctly and submit their information, the system will send an email to verify the account. This code will be carried out if users choose tutor account in the selection. Once the user chosen student account as their preferable account and the given information is all appropriate then in line 77 it will lead user to Figure 25 where it will display the output of the outcome of the registration.

```php
function sendActiveMail($email,$firstname,$hash,$str){
    $_SESSION['active'] = 0; //0 until user activates their account with verify.php
    $_SESSION['logged_in'] = true; // So we know the user has logged in
    $_SESSION['message'] =

                "Confirmation link has been sent to $email, please verify
                your account by clicking on the link in the message!";


        // Send registration confirmation link (verify.php)
        $to      = $email;
        $subject = 'Account Verification ( 24hrs Tutor.com )';
        $message_body = '
        Hello '.$firstname.',

        Thank you for signing up!

        Please click this link to activate your account:

        http://localhost:8080/24t/verify.php?email='.$email.'&hash='.$hash.'&type='.$str;

        mail( $to, $subject, $message_body );

        //header("location: profile.php");
}
?>
```

Figure 24 Verify account in email

Figure 24 code explains the account verification feature. The system will automatically send an email to users after they submit their information to the database and successfully registered their account in the database. In order user to use their account in the system, they have to verify their account from their personal email.

HTML/CSS

```php
<?php
if ( isset($_GET['success']) && $_GET['success'] == 1 )
{
    $msg = 'Your account was created successfully, please go active by Email before login';
    echo "<br>Success<br>  <h3>$msg.<h3>";
}
if ( isset($_GET['fail']) && $_GET['fail'] == 1 )
{
    $msg = 'Your account not created, please user other email address';
    echo "<br>Sorry  <br>  <h3>$msg.<h3>";
}
if ( isset($_GET['fail']) && $_GET['fail'] == 2 )
{
    $msg = 'Your account not created, please user other username';
    echo "<br>Sorry  <br>  <h3>$msg.<h3>";
}
if ( isset($_GET['fail']) && $_GET['fail'] == 3 )
{
    $msg = 'Your account not created, please user other username and email address';
    echo "<br>Sorry  <br>  <h3>$msg.<h3>";
}

?>
```

Figure 25 Output of the result

This part of the code shows different results of the registering part. Figure 23 only works when user submit their given information to create an account in the system and the system will display an output depending on the registration information provided.

Javascript

```
<script type="text/javascript">
        $.validator.setDefaults( {
            submitHandler: function () {
                $('#signupForm1').submit();
            }
        } );

        $( document ).ready( function () {
            $( "#signupForm" ).validate( {
                rules: {
                    firstname: "required",
                    lastname: "required",
                    username: {
                        required: true,
                        minlength: 5
                    },
                    password: {
                        required: true,
                        minlength: 5
                    },
                    confirm_password: {
                        required: true,
                        minlength: 5,
                        equalTo: "#password"
                    },
                    email: {
                        required: true,
                        email: true
                    },
                    agree1: {required: true},
                    radio:{ required:true }
                },
                messages: {
```

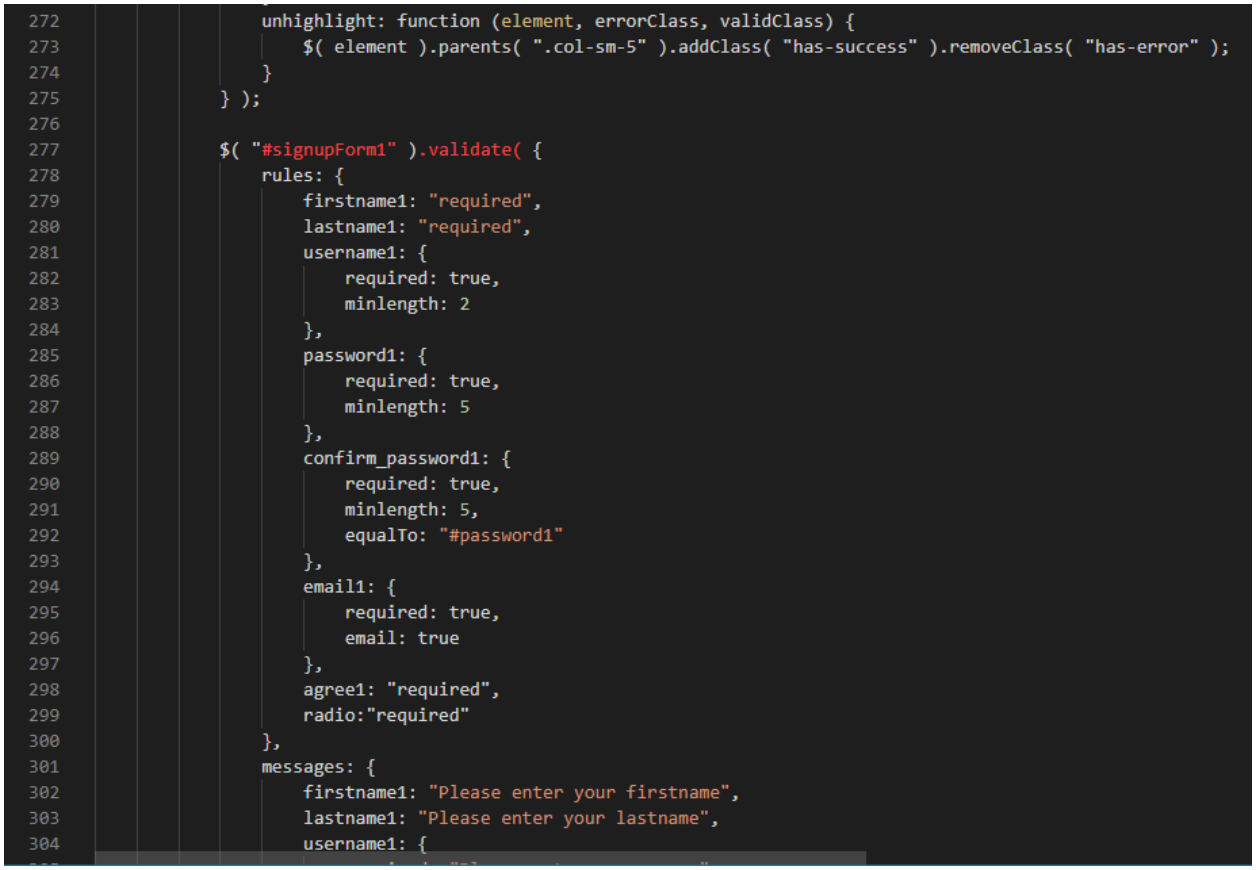
Figure 26 Checking Textbox settings for user input

The coding above shows which field is required to fill in by the user in the register page. Some textbox have input requirements such as user have to insert at least 5 characters in their firstname, lastname and username. The system will read the information given whether that the information has met its requirement for certain textbox, once the system acknowledge the given information, it will store into the database.

```
239                 messages: {
240                     firstname: "Please enter your firstname",
241                     lastname: "Please enter your lastname",
242                     username: {
243                         required: "Please enter a username",
244                         minlength: "Your username must consist of at least 2 characters"
245                     },
246                     password: {
247                         required: "Please provide a password",
248                         minlength: "Your password must be at least 5 characters long"
249                     },
250                     confirm_password: {
251                         required: "Please provide a password",
252                         minlength: "Your password must be at least 5 characters long",
253                         equalTo: "Please enter the same password as above"
254                     },
255                     email: "Please enter a valid email address",
256                     agree1: "Please accept our policy"
257                 },
258                 errorElement: "em",
259                 errorPlacement: function ( error, element ) {
260                     // Add the `help-block` class to the error element
261                     error.addClass( "help-block" );
262 
263                     if ( element.prop( "type" ) === "checkbox" ) {
264                         error.insertAfter( element.parent( "label" ) );
265                     } else {
266                         error.insertAfter( element );
267                     }
268                 },
269                 highlight: function ( element, errorClass, validClass ) {
270                     $( element ).parents( ".col-sm-5" ).addClass( "has-error" ).removeClass( "has-success" );
271                 },
272                 unhighlight: function (element, errorClass, validClass) {
```

Figure 27 Checking Textbox settings for user input

The codings above shows the messages that will be shown in the register page. Line239 to Line 253 is the system will display an outcome if the user did not reach the textbox requirement. For example if the user did not provide at least 5 characters long of password it will display a message to the user that user must have to fulfill the requirements before they can proceed. Line 258 onwards is checking whether the checkbox has been checked by the user once the checking is done it will display the output of if the result.



```
272             unhighlight: function (element, errorClass, validClass) {
273                 $( element ).parents( ".col-sm-5" ).addClass( "has-success" ).removeClass( "has-error" );
274             }
275         } );
276 
277         $( "#signupForm1" ).validate( {
278             rules: {
279                 firstname1: "required",
280                 lastname1: "required",
281                 username1: {
282                     required: true,
283                     minlength: 2
284                 },
285                 password1: {
286                     required: true,
287                     minlength: 5
288                 },
289                 confirm_password1: {
290                     required: true,
291                     minlength: 5,
292                     equalTo: "#password1"
293                 },
294                 email1: {
295                     required: true,
296                     email: true
297                 },
298                 agree1: "required",
299                 radio:"required"
300             },
301             messages: {
302                 firstname1: "Please enter your firstname",
303                 lastname1: "Please enter your lastname",
304                 username1: {
```

Figure 28 Display error for user input (Register Page)

The figure above shows that the coding works when Figure 26 to Figure 27 done checking the textbox input is matched to the textbox setting requirement. If the input is not matched with the requirement it will display error message to user that the user must key in another value into the textbox until it is matched with the requirement given.

```
303                     lastname1: "Please enter your lastname",
304                     username1: {
305                         required: "Please enter a username",
306                         minlength: "Your username must consist of at least 2 characters"
307                     },
308                     password1: {
309                         required: "Please provide a password",
310                         minlength: "Your password must be at least 5 characters long"
311                     },
312                     confirm_password1: {
313                         required: "Please provide a password",
314                         minlength: "Your password must be at least 5 characters long",
315                         equalTo: "Please enter the same password as above"
316                     },
317                     email1: "Please enter a valid email address",
318                     radio:"Choose one account type"
319                 },
320                 errorElement: "em",
321                 errorPlacement: function ( error, element ) {
322                     // Add the `help-block` class to the error element
323                     error.addClass( "help-block" );
324 
325                     // Add `has-feedback` class to the parent div.form-group
326                     // in order to add icons to inputs
327                     element.parents( ".col-sm-5" ).addClass( "has-feedback" );
328 
329                     if ( element.prop( "type" ) === "checkbox" ) {
330                         error.insertAfter( element.parent( "label" ) );
331                     } else {
332                         error.insertAfter( element );
333                     }
334 
335                     // Add the span element, if doesn't exists, and apply the icon classes to it.
```

Figure 29 *Display error for user input (Register Page)*

```
332                         error.insertAfter( element );
333                     }
334 
335                     // Add the span element, if doesn't exists, and apply the icon classes to it.
336                     if ( !element.next( "span" )[ 0 ] ) {
337                         $( "<span class='glyphicon glyphicon-remove form-control-feedback'></span>" ).insertAfter
338                         ( element );
339                     }
340                 },
341                 success: function ( label, element ) {
342                     // Add the span element, if doesn't exists, and apply the icon classes to it.
343                     if ( !$( element ).next( "span" )[ 0 ] ) {
344                         $( "<span class='glyphicon glyphicon-ok form-control-feedback'></span>" ).insertAfter
345                         ( $( element ) );
346                     }
347                 },
348                 highlight: function ( element, errorClass, validClass ) {
349                     $( element ).parents( ".col-sm-5" ).addClass( "has-error" ).removeClass( "has-success" );
350                     $( element ).next( "span" ).addClass( "glyphicon-remove" ).removeClass( "glyphicon-ok" );
351                 },
352                 unhighlight: function ( element, errorClass, validClass ) {
353                     $( element ).parents( ".col-sm-5" ).addClass( "has-success" ).removeClass( "has-error" );
354                     $( element ).next( "span" ).addClass( "glyphicon-ok" ).removeClass( "glyphicon-remove" );
355                 }
356             } );
357         } );
358     </script>
```

Figure 30 *Display error for user input (Register Page)*

Booking/Payment

SQL

```
17  if (isset($_POST["submit"]) && isset($_POST["subject_p"]) && isset($_POST["subject_name"])  && isset($_POST["tutor_n"])
18   && isset($_POST["hour"]) && isset($_POST["appoint_date"]) && isset($_POST["tutor_id"]) && $_POST["appoint_date"] ){
19      echo "passpost";
20     if (hash_equals($csrf, $_POST['csrf'])) {
21         echo "pass csrf";
22         if (ctype_digit($_POST["cardNumber"])  && ctype_digit($_POST["cardExpiry"] ) && ctype_digit($_POST["cardCVC"] ) ){
23             $val1 = $_POST["cardNumber"];
24             $val2 = $_POST["cardExpiry"];
25             $val3 = $_POST["cardCVC"];
26
27                 $stmt_stud = $conn->prepare("SELECT * FROM `testing_card` WHERE `card_num`= :cnum and
28                 `exp_d`=:cexp AND `cvc`=:cvc");
29                 $stmt_stud->bindParam(':cnum', $val1, PDO::PARAM_STR);
30                 $stmt_stud->bindParam(':cexp', $val2, PDO::PARAM_STR);
31                 $stmt_stud->bindParam(':cvc', $val3, PDO::PARAM_STR);
32                 $stmt_stud->execute();
33                 $results = $stmt_stud->fetch(PDO::FETCH_ASSOC);
34                  print_r($results);
35         } //{closing for }if (ctype_digit($_POST["cardNumber"])
36         else{ echo "card invalid";}
37
38             if ($_POST["subject_p"] > $results["amount"]){
39                 echo "insuficcient balance";
40                 echo $_POST["subject_p"];
41                 echo "amount_bank".$results["amount"];
42             }else if ($_POST["subject_p"] < $results["amount"]){
43                 $transfer =$conn->prepare ("UPDATE testing_card SET amount = amount - :minus WHERE card_num = :cnum");
44                 $transfer->bindParam(':cnum', $_POST["cardNumber"], PDO::PARAM_STR);
45                 $transfer->bindParam(':minus', $_POST["subject_p"], PDO::PARAM_STR);
46                 $transfer->bindParam(':cnum', $_POST["cardNumber"], PDO::PARAM_STR);
47                 echo "transfer amount";
48
```

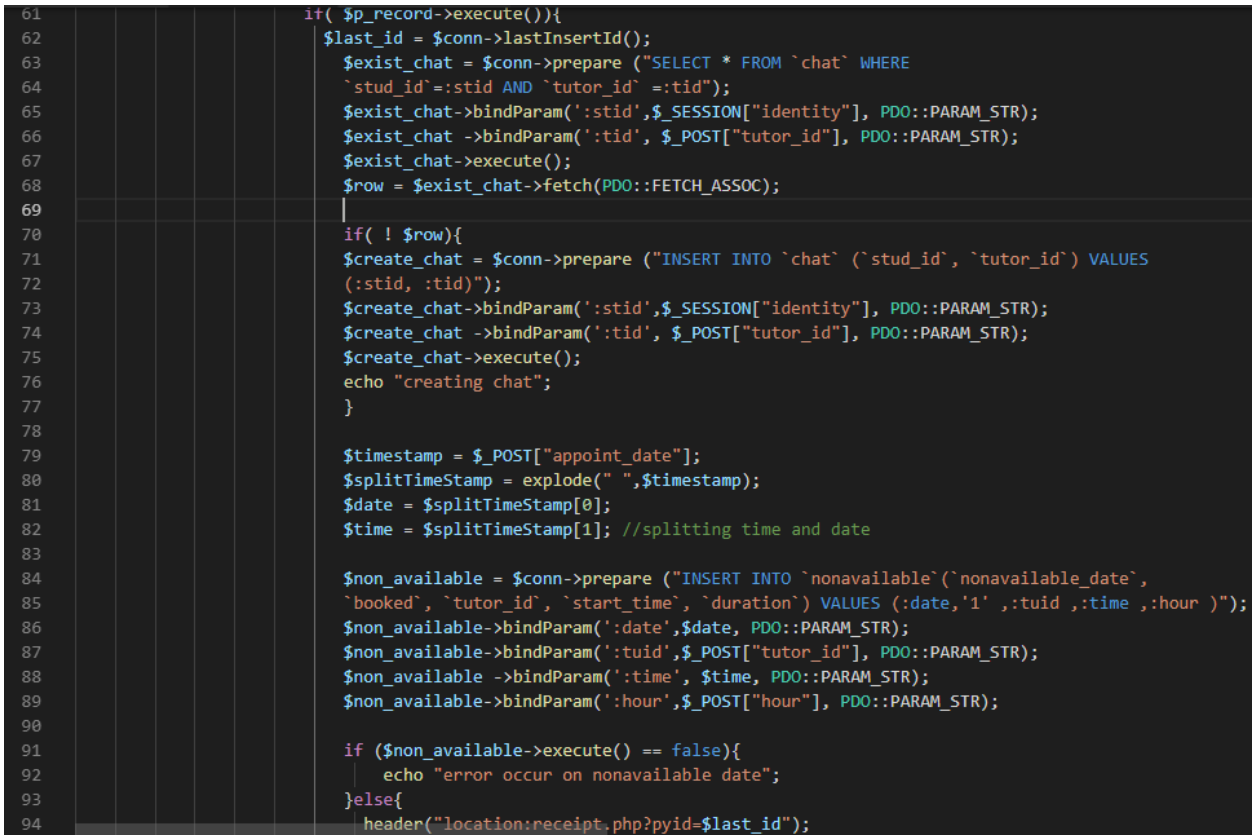
Figure 31 Payment

This is the code that will execute when students want to book a tutor. Students have to fill up the information required from the system because Line 17 is when Students insert their card number, card expiry and CV code. If those information were given correctly then the system will proceed to line 27 for getting the money from the tesing card entity. If not it will display “csrf invalid” which is in Figure 34. Afterwards it will print out the result and transfer it to the receipt page. Line 36 is when the card money is insufficient then it will display card invalid to the user. Line 42 states that if the subject price is lower than the card balance then it will print out “transfer amount” if not then it will print out “transfer and record made fail” which will be in Figure 34.

```
48                 if($transfer->execute()){
49                     $val6 = "pending";
50                         $p_record =$conn->prepare ("INSERT INTO `payment_info` ( `paytime`, `payamount`, `stud_id`,
51                          `tutor_id`, `sub_name`, `bookingdate`, `appoint_hrs`, `status`, `cancel_reason`)
52                           VALUES ( CURRENT_TIMESTAMP, :pyam, :stid, :tid, :sub, :bkd, :app_hr, :state, NULL)");
53                         $p_record ->bindParam(':pyam', $_POST["subject_p"], PDO::PARAM_STR);
54                         $p_record ->bindParam(':stid',$_SESSION["identity"], PDO::PARAM_STR);
55                         $p_record ->bindParam(':tid', $_POST["tutor_id"], PDO::PARAM_STR);
56                         $p_record ->bindParam(':sub', $_POST["subject_name"], PDO::PARAM_STR);
57                         $p_record ->bindParam(':bkd', $_POST["appoint_date"], PDO::PARAM_STR);
58                         $p_record ->bindParam(':app_hr', $_POST["hour"], PDO::PARAM_STR);
59                         $p_record ->bindParam(':state', $val6, PDO::PARAM_STR);
60                         echo "making record";
```

Figure 32 Creating record

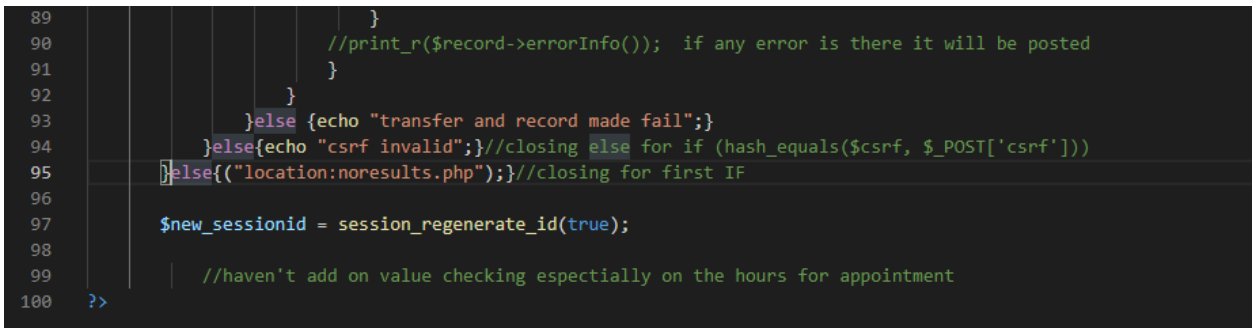
Figure 32 shows that the coding works when the money is transferring then it will start to insert information into the payment info database. Then it will create a record of the payment.



```
61                      if( $p_record->execute()){
62                        $last_id = $conn->lastInsertId();
63                          $exist_chat = $conn->prepare ("SELECT * FROM `chat` WHERE
64                          `stud_id`=:stid AND `tutor_id` =:tid");
65                          $exist_chat->bindParam(':stid',$_SESSION["identity"], PDO::PARAM_STR);
66                          $exist_chat ->bindParam(':tid', $_POST["tutor_id"], PDO::PARAM_STR);
67                          $exist_chat->execute();
68                          $row = $exist_chat->fetch(PDO::FETCH_ASSOC);
69                          
70                          if( ! $row){
71                          $create_chat = $conn->prepare ("INSERT INTO `chat` (`stud_id`, `tutor_id`) VALUES
72                          (:stid, :tid)");
73                          $create_chat->bindParam(':stid',$_SESSION["identity"], PDO::PARAM_STR);
74                          $create_chat ->bindParam(':tid', $_POST["tutor_id"], PDO::PARAM_STR);
75                          $create_chat->execute();
76                          echo "creating chat";
77                          }
78
79                          $timestamp = $_POST["appoint_date"];
80                          $splitTimeStamp = explode(" ",$timestamp);
81                          $date = $splitTimeStamp[0];
82                          $time = $splitTimeStamp[1]; //splitting time and date
83
84                          $non_available = $conn->prepare ("INSERT INTO `nonavailable`(`nonavailable_date`,
85                          `booked`, `tutor_id`, `start_time`, `duration`) VALUES (:date,'1' ,:tuid ,:time ,:hour )");
86                          $non_available->bindParam(':date',$date, PDO::PARAM_STR);
87                          $non_available->bindParam(':tuid',$_POST["tutor_id"], PDO::PARAM_STR);
88                          $non_available ->bindParam(':time', $time, PDO::PARAM_STR);
89                          $non_available->bindParam(':hour',$_POST["hour"], PDO::PARAM_STR);
90
91                          if ($non_available->execute() == false){
92                              echo "error occur on nonavailable date";
93                          }else{
94                            header("location:receipt.php?pyid=$last_id");
```

Figure 33 After recorded into the database

The figure above shows that once the information is recorded into the database it will create a chat function for both students and tutors who are involved with each other. Line 79 shows that the database will record the appointment date into the calendar. Line 84 is about inserting the appointment date into the non available date in the calendar then the code will proceed to Line 91 to check if there is any clash with non available dates in the database if yes then it will display “error occur on non available date”, if did not it will print out a receipt.



```
89                          }
90                        //print_r($record->errorInfo());  if any error is there it will be posted
91                        }
92                    }
93                }else {echo "transfer and record made fail";}
94            }else{echo "csrf invalid";}//closing else for if (hash_equals($csrf, $_POST['csrf']))
95        }else{("location:noresults.php");}//closing for first IF
96
97        $new_sessionid = session_regenerate_id(true);
98
99            //haven't add on value checking espectially on the hours for appointment
100   ?>
```

Figure 34 Error message of the payment

```php
if(isset($_SESSION["username1"]) && isset($_SESSION["identity"]) && isset($_SESSION["actype"])  )
{

    if ( isset($_GET['pyid']) && !$_GET['pyid'] == "" && ctype_digit($_GET['pyid']))
    {

        if ($_SESSION["actype"] ==="student"){
            $Q = $conn->prepare('SELECT t.t_fname, t.t_lname, t.t_profilepic, c.sub_name, c.payment_id , c.payamount ,
            c.tutor_id , c.bookingdate, c.appoint_hrs FROM payment_info c, tutor t
            WHERE `stud_id` = :id AND c.tutor_id = t.tutor_id  AND c.payment_id=:pyid' );
            $Q->bindParam(':id', $_SESSION["identity"], PDO::PARAM_INT);
            $Q->bindParam(':pyid', $_GET['pyid'], PDO::PARAM_INT);
            $Q->execute();
            $count = $Q->rowCount();
            $result = $Q->fetchAll();
            print_r($result);
        }
    }
}
```

Figure 35 Receipt

The figure above is about bringing in payment data from payment page and print it in the receipt page. The coding can only retrieve the latest payment from the database and print it into the Receipt Page.

Set Course

```php
$res = "0";
if(isset($_SESSION["username1"]) && isset($_SESSION["identity"]) && $_SESSION["actype"] ==="tutor")
{
   if(isset($_GET['del'])){
     $delval = $_GET['del'];
     $delQ = $conn->prepare('DELETE FROM `subject_offer` WHERE `subject_id`=:dl AND `tutor_id` = :id ');
     $delQ->bindParam(':dl', $delval, PDO::PARAM_INT);
     $delQ->bindParam(':id', $_SESSION["identity"], PDO::PARAM_INT);
     $delQ->execute();
   }
   $query = $conn->prepare('SELECT * FROM subject ORDER BY subject_name');
   $query->execute();

   $query2 = $conn->prepare('SELECT c.subject_name , d.subject_id , d.offer_id , d.price_perhrs FROM subject c ,
   subject_offer d WHERE d.tutor_id =:id AND c.subject_id = d.subject_id  ORDER BY c.subject_name');
   $query2->bindParam(':id', $_SESSION["identity"], PDO::PARAM_INT);
   $query2->execute();                                                    //query to show user's subjects
 } else
 {
      session_destroy();
      header("location:loging.php");
 }

```

Figure 36 Set course (Delete function)

This coding is related to the set course feature of the system. Tutors will only be able to use this function to set the course they will be teaching. Line 12 is when tutor press the delete function, the code will delete the subject id and tutor id from the subject offer table from the database.

```
31      if(isset($_POST['submit'])&& isset($_POST['category']) && $_POST['category'] !==''){
32        $val1 = $_POST['category'];  $val3 = $_POST['price'];  $set ='';
33        try {               // Use to add or update subject price
34        foreach ($query2 as $row2) {
35          if ($row2['subject_id']==$val1){
36            $set = '1';
37          }else if ($row2['subject_id']!==$val1){
38            $set = '0';
39          }else{
40            header("location:setcourse.php");
41          }
42        }
```

Figure 37 Set Course (Set Price Function)

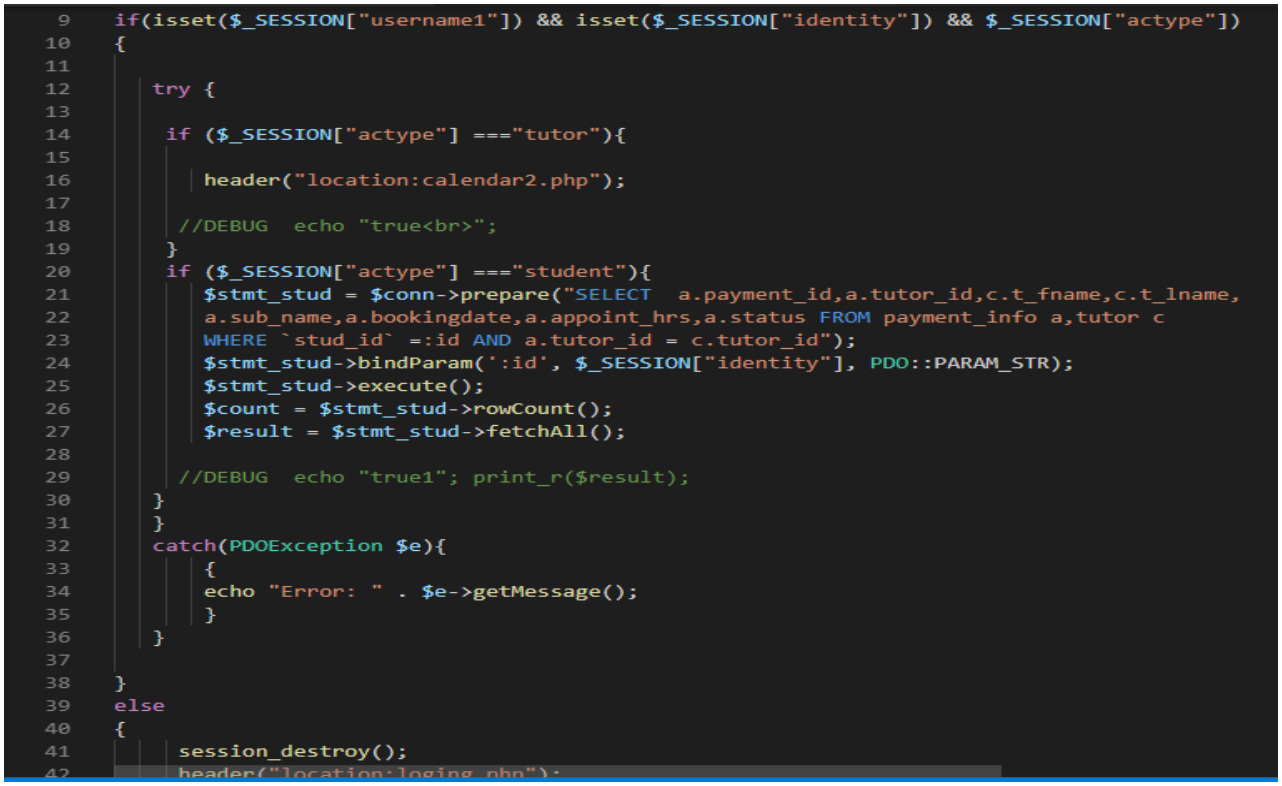
The figure above shows that tutor wants to add subject into their teaching list. Line 31 is is when the tutor has set their price of the subject and press the submit button then the it will add in the value given and store it into the database.

```
43        if ($set==1){
44        $addsubject = $conn->prepare("UPDATE `subject_offer` SET `price_perhrs`=:v3 WHERE `subject_id`=:v1
45        AND `tutor_id` =:v2");
46        }else if ($set==0){
47          $delQ2 = $conn->prepare('DELETE FROM `subject_offer` WHERE `subject_id`=:ct AND `tutor_id` = :id ');
48          $delQ2->bindParam(':ct', $_POST['category'], PDO::PARAM_INT);
49          $delQ2->bindParam(':id', $_SESSION["identity"], PDO::PARAM_INT);
50          $delQ2->execute();
51          $addsubject = $conn->prepare("INSERT INTO `subject_offer` (`subject_id`, `tutor_id`, `price_perhrs`) VALUES
52          ( :v1 , :v2 , :v3)");
53        }
54        $addsubject->bindParam(':v1', $val1, PDO::PARAM_INT);
55        $addsubject->bindParam(':v2', $_SESSION["identity"], PDO::PARAM_INT);
56        $addsubject->bindParam(':v3', $val3, PDO::PARAM_INT);
57        $addsubject->execute();
58        header("location:setcourse.php");
59      }
60    catch(PDOException $e)
61        {
62        echo "Error: " . $e->getMessage();
63        }
64        $conn = null;
65    }
```

Figure 38 Set Course (Add and Update Function)

The figure above shows the codings for subject adding. The code will be carried out if tutors decide to add or update subject into the database. Line 43 stands for when the database already has the stored valued, it will proceed to line 44 to update the subject information in the database. Whereas the value is not in the database it will proceed to line 41 to carry out the adding function. Line 51 is about insert subject into

Calendar



```
9   if(isset($_SESSION["username1"]) && isset($_SESSION["identity"]) && $_SESSION["actype"])
10  {
11
12      try {
13
14       if ($_SESSION["actype"] ==="tutor"){
15
16          header("location:calendar2.php");
17
18        //DEBUG  echo "true<br>";
19       }
20       if ($_SESSION["actype"] ==="student"){
21          $stmt_stud = $conn->prepare("SELECT  a.payment_id,a.tutor_id,c.t_fname,c.t_lname,
22          a.sub_name,a.bookingdate,a.appoint_hrs,a.status FROM payment_info a,tutor c
23          WHERE `stud_id` =:id AND a.tutor_id = c.tutor_id");
24          $stmt_stud->bindParam(':id', $_SESSION["identity"], PDO::PARAM_STR);
25          $stmt_stud->execute();
26          $count = $stmt_stud->rowCount();
27          $result = $stmt_stud->fetchAll();
28
29        //DEBUG  echo "true1"; print_r($result);
30      }
31      }
32      catch(PDOException $e){
33          {
34          echo "Error: " . $e->getMessage();
35          }
36      }
37
38  }
39  else
40  {
41      session_destroy();
42      header("location:loging.php");
```

Figure 39 Student side calendar

The figure above shows the codings of the calendar for students. As for the user who registered as student in the system can only check their timetable in the calendar. Line 20 onwards is if the account type is student then it will print out the information about your tuition information such as time of tuition, tutor name and the name of the subject.

```
9   if(isset($_SESSION["username1"]) && isset($_SESSION["identity"]) && $_SESSION["actype"])
10  {
11
12      try {
13
14       if ($_SESSION["actype"] ==="tutor"){
15          $stmt_tutor = $conn->prepare("SELECT  a.payment_id,a.stud_id,c.stud_fname,c.stud_lname,
16          a.sub_name,a.bookingdate,a.appoint_hrs,a.status FROM payment_info a,student c WHERE
17          `tutor_id` =:id AND a.stud_id = c.stud_id");
18          $stmt_tutor->bindParam(':id', $_SESSION["identity"], PDO::PARAM_STR);
19        $stmt_tutor->execute();
20           print_r($stmt_tutor);
21           echo "hi";
22           $count = $stmt_tutor->rowCount();
23           $result = $stmt_tutor->fetchAll();
24
25
26           $stmt_tutor2 = $conn->prepare("SELECT * FROM `nonavailable` WHERE `tutor_id`=:id AND `booked`=0");
27           $stmt_tutor2->bindParam(':id', $_SESSION["identity"], PDO::PARAM_STR);
28           $stmt_tutor2->execute();
29           //fetch from non available table
30            $result2 = $stmt_tutor2->fetchAll();
31
32        //DEBUG  echo "true<br>";
33       }
```

Figure 40 Tutor side calendar

The figure above shows the codings of the calendar for tutors. Line 14 states that only tutor account type can access into the tutor calendar. The tutor calendar also has the same similar function as the student calendar but the only difference is that the tutor calendar allows the tutor to pick their non available date in the calendar which means that they can take as many off days as they want in the database. Line 26 states the system will select the nonavailable table from the database and fill in the given value into the nonavailable table. Afterwards the system will proceed to execute the given value into the tutor calendar which means now personal user and all students are able to check the tutor non available date in the calendar. Students are able to check the tutor non available date in the booking page.

Matching list

```
9   if(isset($_SESSION["username1"]) && isset($_SESSION["identity"]) && isset($_SESSION["actype"]))
10  {
11
12      if ($_SESSION["actype"] ==="tutor"){
13          $Q = $conn->prepare('SELECT t.stud_fname, t.stud_lname, t.stud_profilepic, c.payment_id , c.stud_id,c.sub_name
14          ,c.bookingdate,DATEDIFF(c.bookingdate, CURDATE()) AS Date,c.appoint_hrs,c.status FROM payment_info c, student t
15          WHERE `bookingdate`>=  CURDATE() AND `tutor_id` = :id AND c.stud_id = t.stud_id AND `status` ="pending"
16          ORDER BY `bookingdate`');
17          $Q->bindParam(':id', $_SESSION["identity"], PDO::PARAM_INT);
18          $Q->execute();
19          $count = $Q->rowCount();
20          $result = $Q->fetchAll();
21      }
22      if ($_SESSION["actype"] ==="student"){
23          $Q = $conn->prepare('SELECT t.t_fname, t.t_lname, t.t_profilepic, c.payment_id , c.tutor_id,c.sub_name,
24          c.bookingdate,DATEDIFF(c.bookingdate, CURDATE()) AS Date,c.appoint_hrs,c.status FROM payment_info c, tutor t
25          WHERE `bookingdate`>= CURDATE() AND `stud_id` = :id AND c.tutor_id = t.tutor_id AND `status` ="pending"
26          ORDER BY `bookingdate`');
27          $Q->bindParam(':id', $_SESSION["identity"], PDO::PARAM_INT);
28          $Q->execute();
29          $count = $Q->rowCount();
30          $result = $Q->fetchAll();
31      }
32          //print_r($result);
33          //echo $count;
34  } else
35  {
36      session_destroy();
37      header("location:loging.php");
38  }
```

Figure 41 Matching list

The figure above shows the feature of matching tutors. The matching function of tutor and student site are shown in the figure above. For line 12 and line 22 state only student and tutor account type are able to access into the matching page. The matching page allows user to know all incoming tuition or previous tuition information line 13 and line 23 the system selects their perspective table from the database the it will proceed to to bind the identity of the user to the specific variable name which is named as $Q. Then it will print out all the information regarding the tuition information from the database.

Chapter 6: Testing

This chapter will be covering the testing session of the system from the development team. There are three different process that are picked and was tested by the development team to ensure the system's function can be carry out smoothly.

The development team decided to carry out testings on different complex and most used functions of the website to ensure that users will not be facing any problems in the future.

6.1 Login Process

Firstly, the login function was chosen because it is a basic need for a web based application. The team had carried out a few tests including a fake username and password to ensure that there is no mistake on the login function. Besides that, the team had also tested the login functions of different registered users to ensure that they will be redirected to a correct page after they have entered the correct credentials. (see Appendix B.1 for the White Box and Black Box Testing of the Stated Process)

6.2 Search Process

The search process is considered as one of the most important process in the system. The search features plays a role where users have to search available subjects in the database. As our testing team carries out a role to ensure that the search feature can be used and at the same time they have to ensure that the user input must be appropriate. (see Appendix B.2 for the White Box and Black Box Testing of the Stated Process)

6.3 Register Process

In addition, the development team carried out a test on the register function of the website as it has a lot of attributes to test when users decided to create an account. The development team will also have to ensure that the records of registered account is saved into the database for example the login ID and password. (see Appendix B.3 for the White Box and Black Box Testing of the Stated Process)

6.4 Set Course Process

Furthermore, the set course function is only allowed for tutor to add in more subjects into their teaching list. The development team will have to ensure that the records of subjects and price can be saved into the database. (see Appendix B.4 for the White Box and Black Box Testing of the Stated Process)

Chapter 7: Conclusion and future recommendation

In this chapter, we are going to make a conclusion based on every topic and suggest future recommendations to help enhance the system.

7.1 Conclusion

In total for chapter 1, there are a list of problem statements stated that are faced by the development team during the process of creating the system. Problems such as duplication of registered records and login failure are faced during the implementation of the system's feature. Of course, the problems stated are then solved by the development team. Besides that, chapter 1 also includes analysis and requirements that are gathered by the team for the project as well as the explanation of the project's background as well as the aims and objectives of the team for the system.

In chapter 2, studies regarding the feasibility of the proposed web-based system to identify the opportunities and limitations of the system. Hence, reviews and comparison on the system is included.

Nextly, in chapter 3, listings of the functions that are used in the system are included, both functional and non-functional. The importance of each element implemented is also explained.

For chapter 4, diagrams such as ER diagrams, use case diagrams, activity diagrams are included to show the functions of the system and to plan out a strategic way to proceed with the system. Moreover, gantt chart is also included to show the progress of the team. These documents are certainly important in guiding the team while developing the proposed system.

Chapter 5 includes the explanation of the codings that are implemented to create the system. It includes the programming language that are used to create the system and the purpose of using it. It provides a better understanding to the marker.

Furthermore, chapter 6 contains the testing that have been done by the development team. The team tested the system to look for bugs and glitches and solve them afterwards. Testing is an important sequence as it ensures the perfectness of the system. With a tested system, there will be less occurrence of bugs and glitches as it has been discovered and solved.

Last but not least, chapter 7. In this chapter, discussion about summary of each chapter and future recommendations are included to improvise the system to be better.

7.2 Future recommendations

In the future, the development team hope to achieve world wide recognition for the system in order to help more students who are having problems looking for tutors. The development team also crave to enhance the web-based performance to satisfy every user. Moreover, the development team will be affiliating with tuition centers to provide more choices of selection for students instead. This will affect the price of education and will be able to help students who are low on budget.

Furthermore, the development team will continue to work on maintaining the system's database and to always test for bugs or glitches to ensure users get the full experience of the software smoothly.

Besides that, the team are planning to enhance the system's security to let students have a peace in mind and would not be afraid of fraud tutors. With the system security enhanced, hackers will be having a problem getting into the system and retrieve information, this will reduce the risk of information leaks and misuse of information for unsave matters.

In addition the team will be making improvements on the UI design of the system to make it user friendly. This is to let users have comfortable feeling by using the website. Without complicated and messy UI design, users will have no problem in searching desired functions. The team will construct the UI design based of feedbacks and or comments given by users to satisfy the needs of users.

Lastly, the team is also planning to add new features to the system to enhance the capability of the system. Besides adding new features, the team will also strengthen the current available features so that users can use them with ease. With additional helpful features, the system will be more approachable and less time consuming for various users to use.

Appendix A: System Design Diagrams

A.1 Activity Diagrams

A.1.1 Student side activity diagram

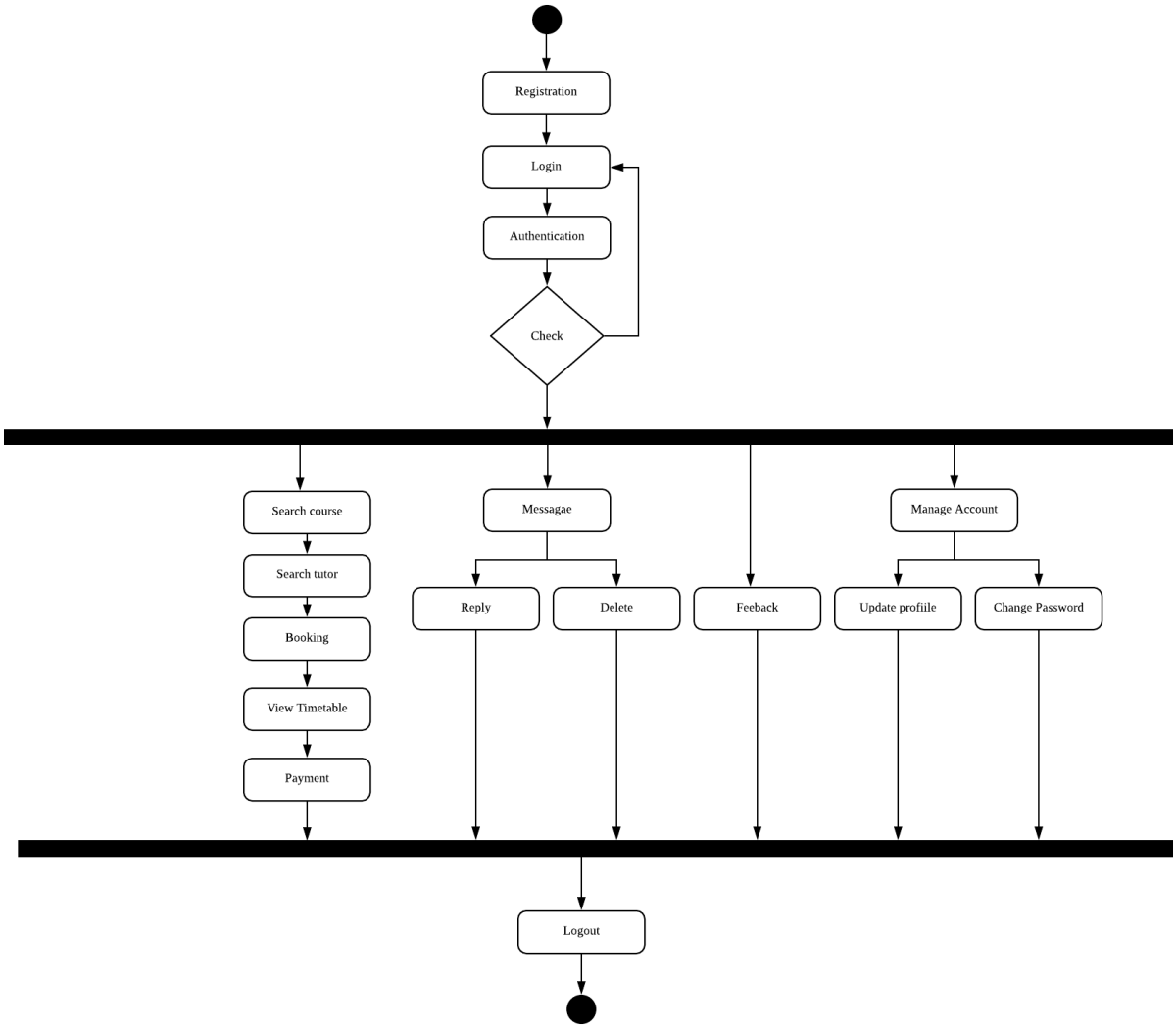


Figure 42: Activity Diagram of the Student-Side of the Proposed System

A.1.2 Tutor side activity diagram

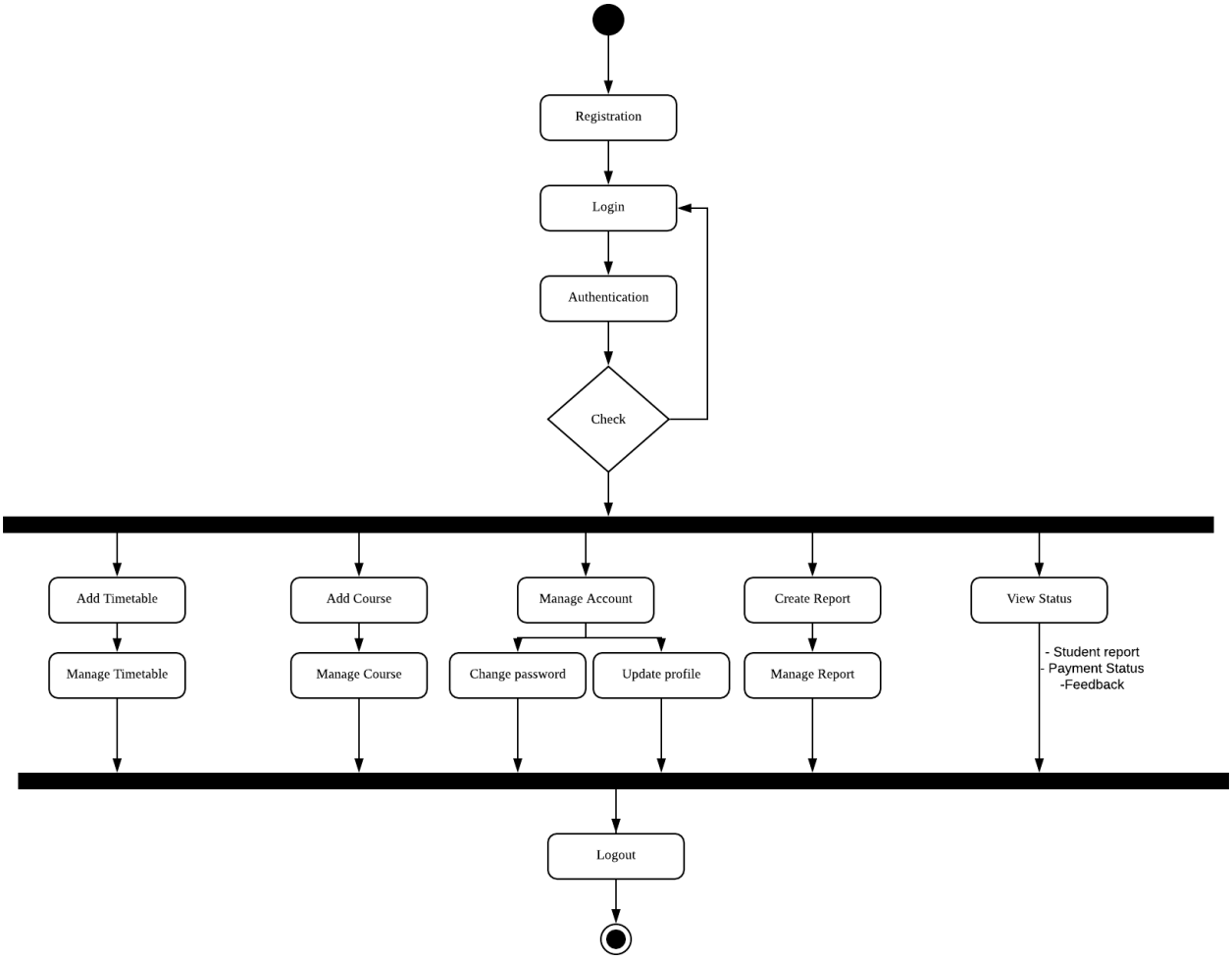


Figure 43: Activity Diagram of the Tutor-Side of the Proposed System

A.1.3 Admin side activity diagram

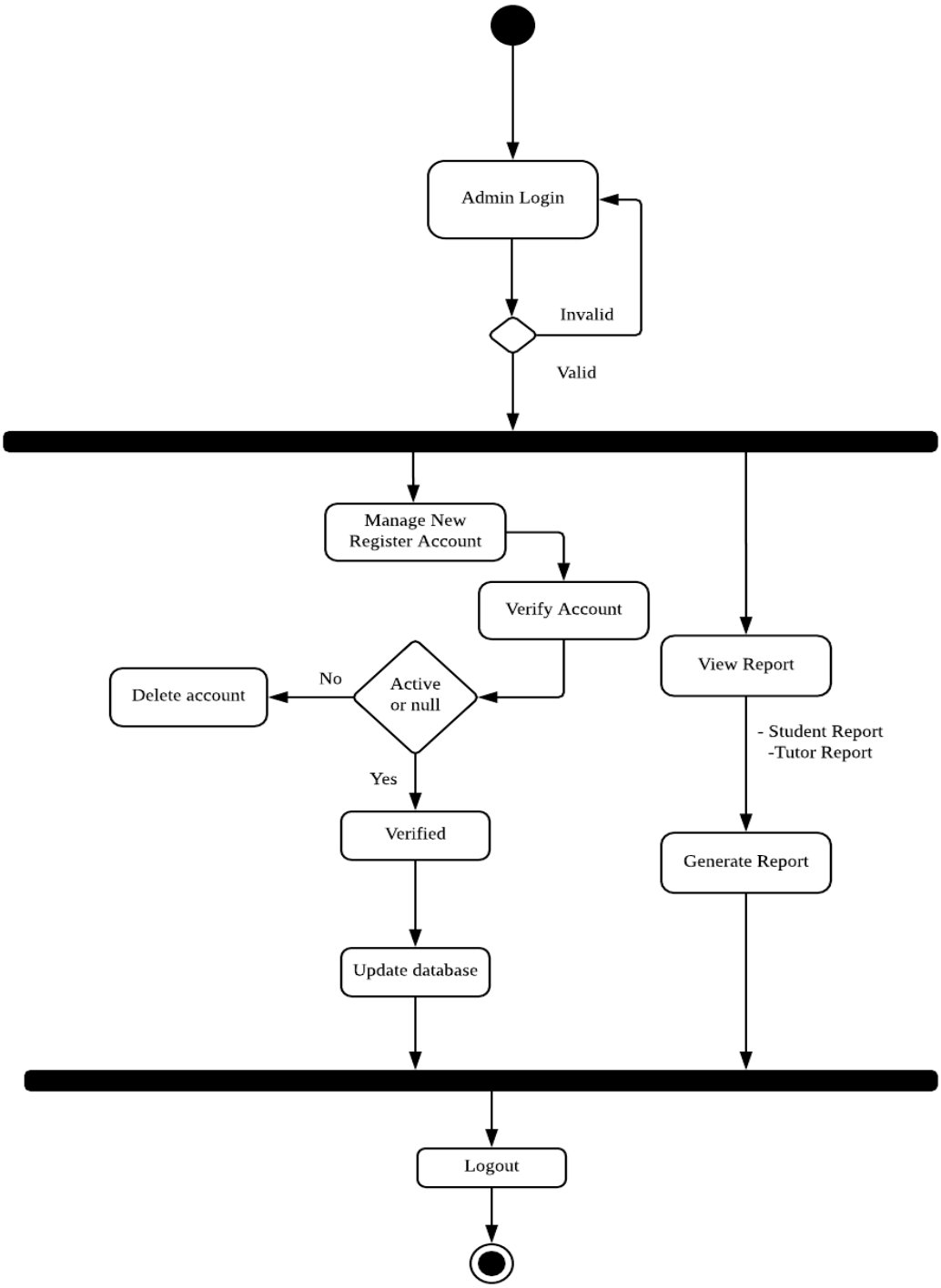


Figure 44: Activity Diagram of the Admin-side of the Proposed System

A.2 Use Case Diagram

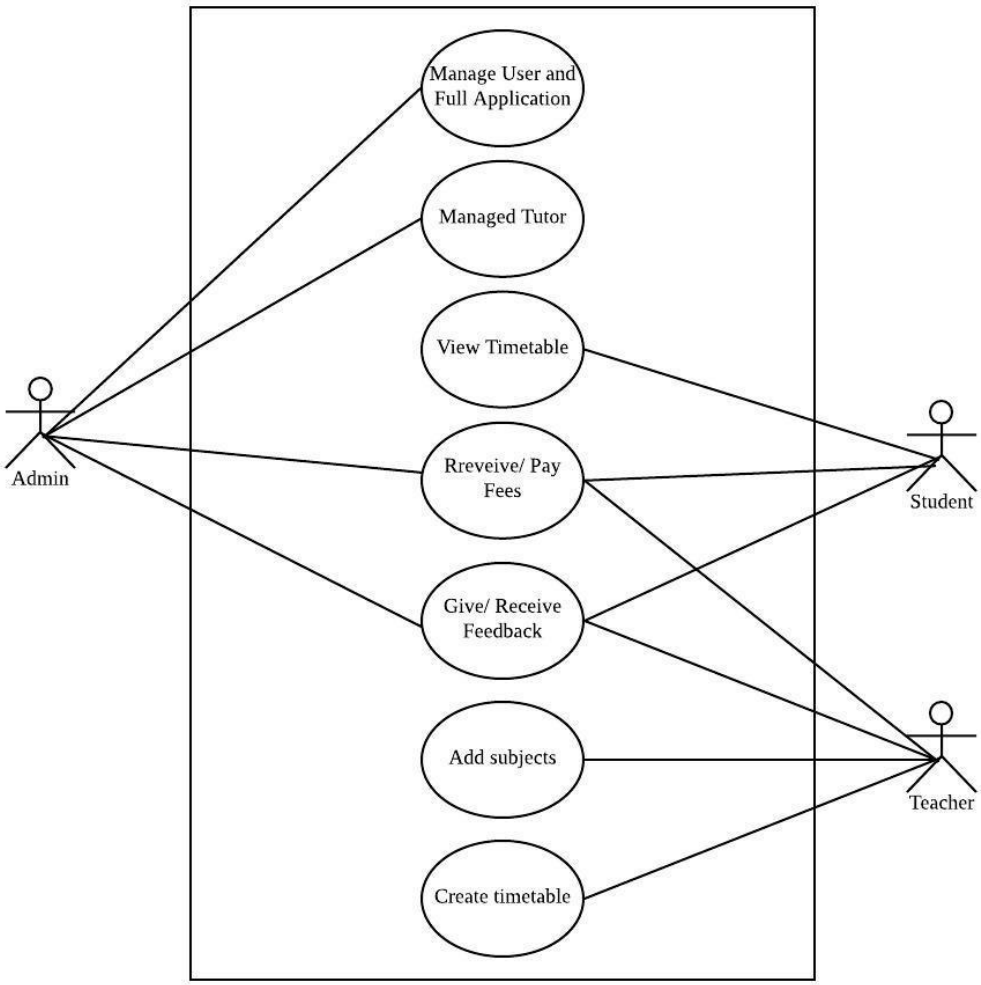


Figure 45: Use Case Diagram of the Proposed System

A.3 Gantt Chart

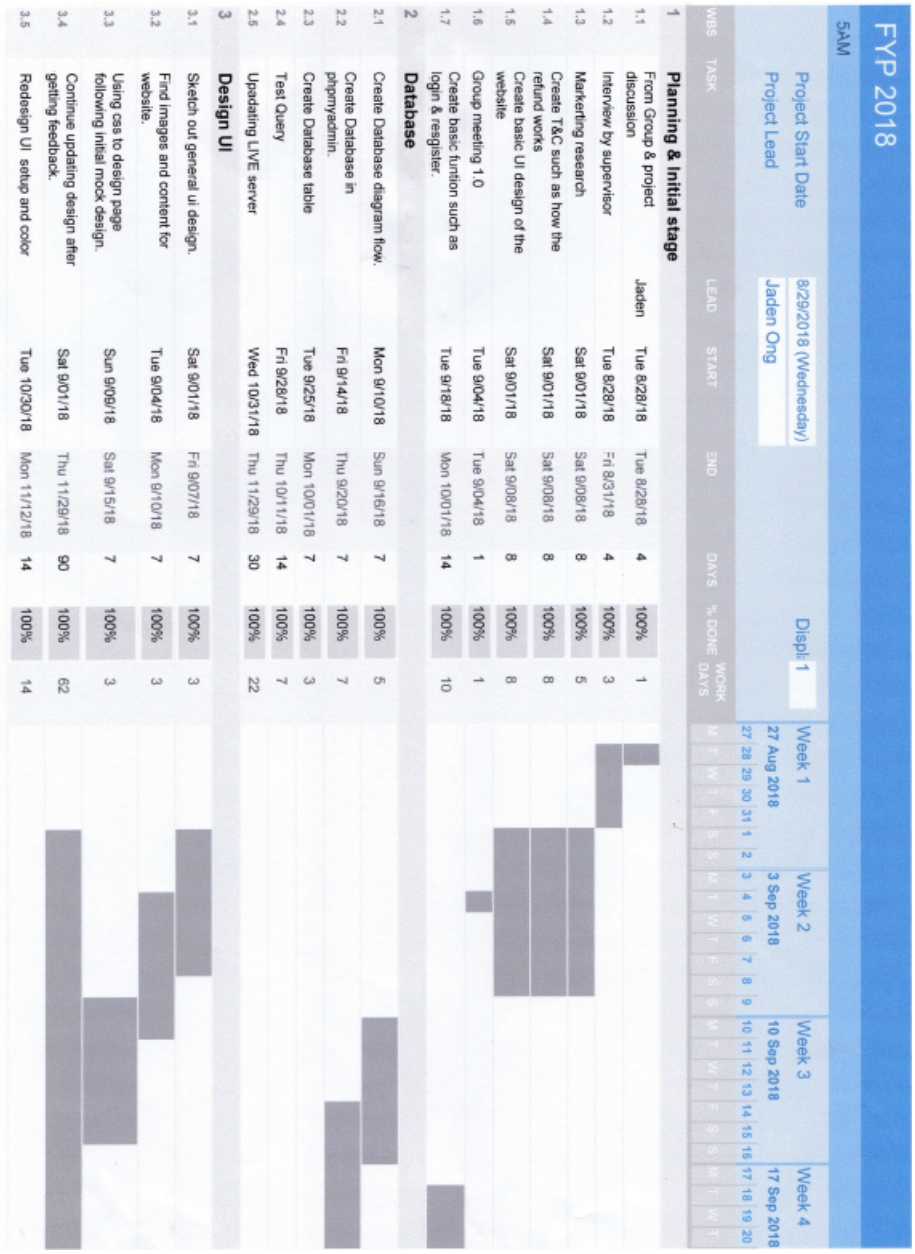


Figure 46 Gantt Chart (Top #1)

Figure 47 Gantt Chart (Top #2)

	Project Start Date	8/29/2018 (Wednesday)			Displ 1		
	Project Lead	Jaden Ong					

WBS	TASK	LEAD	START	END	DAYS	% DONE	WORK DAYS
4	**Coding (PhP)**						
4.1	Create SEARCH and MATCH function		Mon 9/17/18	Sun 9/23/18	7	100%	7
4.2	Create ADD_Subject function		Sat 9/01/18	Fri 9/07/18	7	100%	5
4.3	Create Calendar		Mon 9/10/18	Sun 9/16/18	7	100%	5
4.4	Create BOOKING function		Sat 9/22/18	Fri 9/28/18	7	100%	3
4.5	Create view HISTORY function		Mon 10/01/18	Sun 10/07/18	7	100%	3
4.6	Create CHAT_ROOM		Thu 10/18/18	Wed 10/31/18	14	100%	3
4.7	Create PAYMENT function		Fri 10/12/18	Thu 10/25/18	14	100%	3
4.8	Create DELETE_Subject function		Sat 10/20/18	Fri 10/26/18	7	100%	3
4.9	Create ADMIN _APPROVE function		Wed 11/07/18	Tue 11/20/18	14	100%	7
4.10	Create ADMIN site		Wed 11/07/18	Tue 11/13/18	7	100%	3
4.11	Create RATING function		Sun 11/11/18	Sat 12/01/18	21	100%	7
4.12	Create RECEIPT function		Mon 11/05/18	Sun 11/25/18	21	100%	7
5	**Report's progress**						
5.1	Chapter 1: Introduction		Thu 9/20/18	Wed 10/03/18	14	100%	7
5.2	Chapter 2 : Primary Study/Literatyre Review		Thu 10/04/18	Wed 10/17/18	14	100%	7
5.3	Chapter 3: System Analysis/ Methodology		Thu 10/18/18	Wed 10/31/18	14	100%	5
5.4	Chapter 4: System Design/Proposed Solution		Thu 11/01/18	Wed 11/14/18	14	100%	3
5.5	Chapter 5: Implementation		Wed 11/07/18	Tue 11/20/18	14	100%	3
5.6	Chapter 6: Testing		Wed 11/14/18	Tue 11/27/18	14	100%	3
5.7	Chapter 7: Conclusion and future recommendation		Fri 11/16/18	Thu 11/29/18	14	100%	3
5.8	Appendix A: System Design Diagrams		Fri 11/30/18	Thu 12/06/18	7	100%	3
5.9	Appendix B: Testing the Proposed System		Sat 12/01/18	Fri 12/07/18	7	100%	7

Figure 48 Gantt Chart (Lower #1)

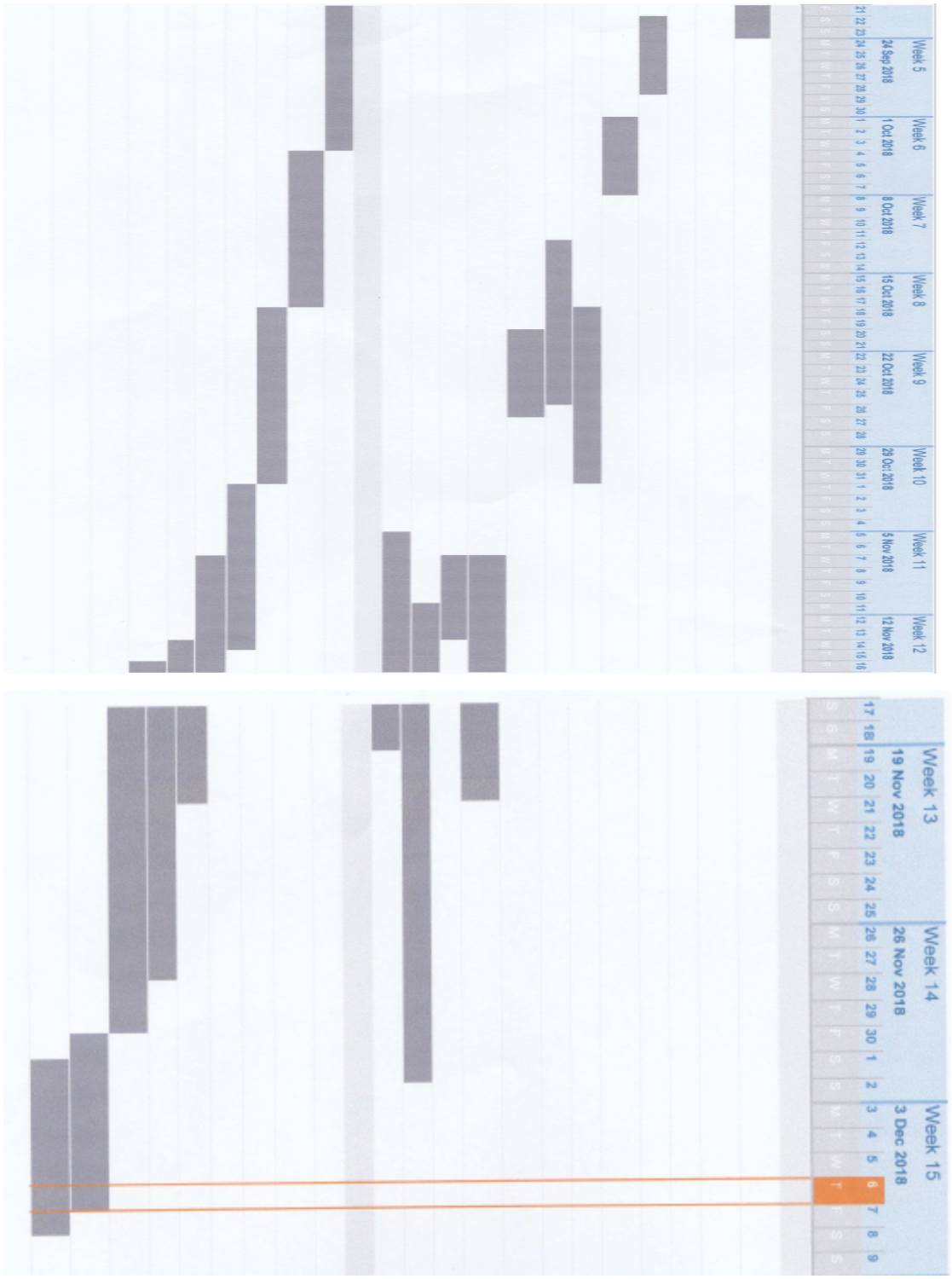


Figure 49 Gantt Chart (Lower #2)

A.4 ER Diagram

A.4.1 Detailed ER diagram

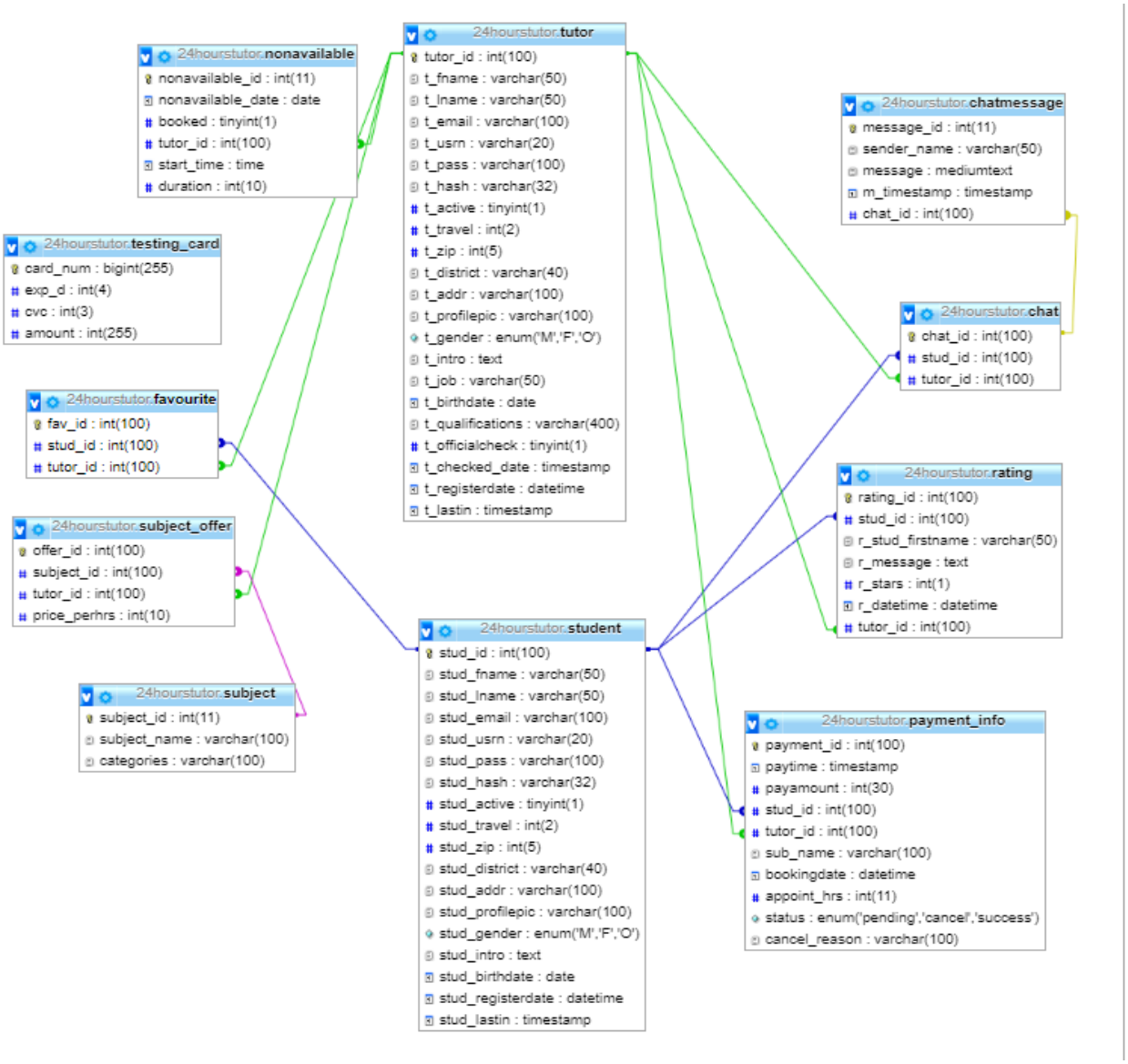
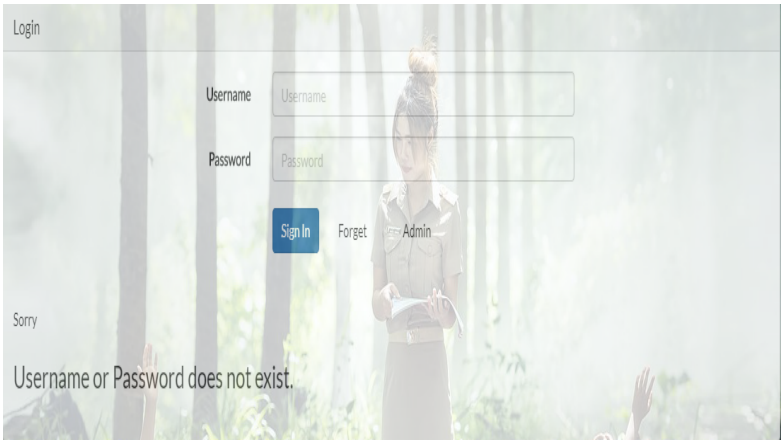


Figure 50: Detailed ER Diagram of the Proposed System

Appendix B: Testing of the Proposed System

B.1 Login Process

Test Number	1
Objectives	To login into the system.
Test Data	Username, Password
Expected Output	If login is successful, user will enter the system home page based on the user type. If login fails, error message will be prompted and user have to enter the correct username and password to retry the login progress.
Screenshot (Actual Result)	
Conclusion	In order to have a successful login, username and password has to be correct or else the error will continuously prompt.

White Box Testing : Multiple Condition Coverage Sample

Inputs	<input type="text" class="form-control" id="username1" name="username1" placeholder="Username" /> <input type="password" class="form-control" id="password1" name="password1" placeholder="Password"/>
Code	if (isset($_POST['login'])) { if (hash_equals($csrf, $_POST['csrf'])) { $username = ($_POST['username1']); $password = ($_POST['password1']); $stmt1 = $conn->prepare("SELECT * FROM tutor WHERE t_usrn=:username AND t_active ='1'"); $stmt1->bindParam(":username",$username); $stmt1->execute(); $result = $stmt1->fetch(PDO::FETCH_ASSOC); $stmt2 = $conn->prepare("SELECT * FROM student WHERE stud_usrn=:username AND stud_active ='1'"); $stmt2->bindParam(":username",$username); $stmt2->execute(); $result2 = $stmt2->fetch(PDO::FETCH_ASSOC); if ($result >0 ||$result2 >0){ //echo "horray"; $exit ='1'; if(password_verify($password,$result['t_pass'])){ echo"1"; $_SESSION["username1"] = $result['t_usrn']; $_SESSION["identity"] = $result['tutor_id']; $_SESSION["firstname"] = $result['t_fname']; $_SESSION["lastname"] = $result['t_lname']; $_SESSION["actype"] = "tutor"; $_SESSION['expire'] = $_SESSION['start'] + (30 * 60); $stmt3 = $conn->prepare("UPDATE `tutor` SET `t_lastin`=CURRENT_TIMESTAMP WHERE `tutor_id` =:id"); $stmt3->bindParam(":id",$_SESSION["identity"]); $stmt3->execute(); header("location:login_success.php");

```
                        $exit ='0';
        }

        if(password_verify($password,$result2['stud_pass'])){
            echo"1";
                        $_SESSION["username1"] = $result2['stud_usrn'];
                        $_SESSION["identity"] = $result2['stud_id'];
                        $_SESSION["firstname"] = $result2['stud_fname'];
                        $_SESSION["lastname"] = $result2['stud_lname'];
                        $_SESSION["actype"] = "student";
                        $_SESSION['expire'] = $_SESSION['start'] + (30 * 60);
                        $stmt4 = $conn->prepare("UPDATE `student` SET
`stud_lastin`=CURRENT_TIMESTAMP WHERE `stud_id` =:id");
                        $stmt4->bindParam(":id",$_SESSION["identity"]);
                        $stmt4->execute();

                        header("location:login_success.php");
                        $exit ='0';
                }
                if ($exit =='1'){
                        session_destroy();
                        header('location: loging.php?fail=1');
                }

}else if($result <0 || $result2 <0){
            session_destroy();
    header('location: loging.php?fail=1');              // Tells Unity it wasn't successful,
so to try again.
}else{
            session_destroy();
            header('location: loging.php?fail=1');
}
            }else{
            session_destroy();
            header('location: loging.php?fail=1');
            }
}
```

	($result >0 ||$result2 >0)	($result <0 || $result2 <0)
1	T (Tutor account)	F
2	T (Student account)	F

Black Box Testing

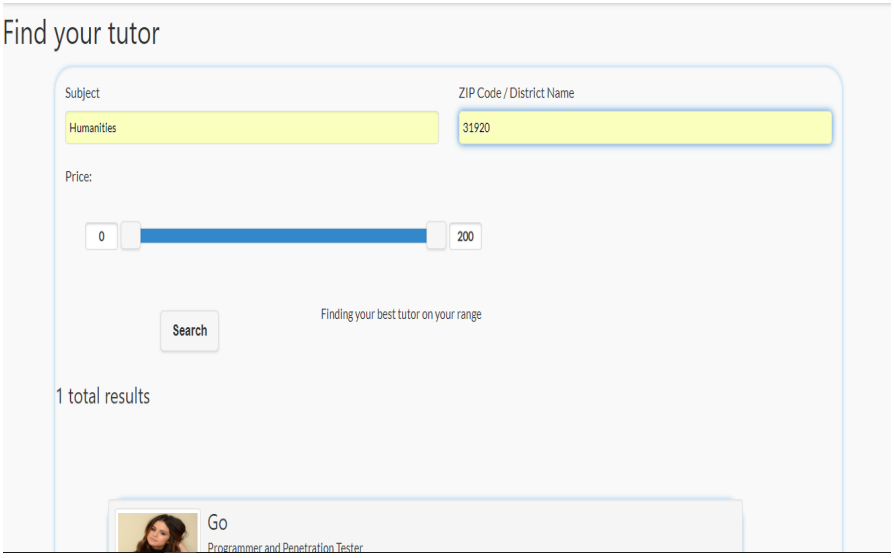
The input of the program consists of a username textbox which can be up to 20 characters in length and a password textbox which can be inserted up to 100 characters in length.

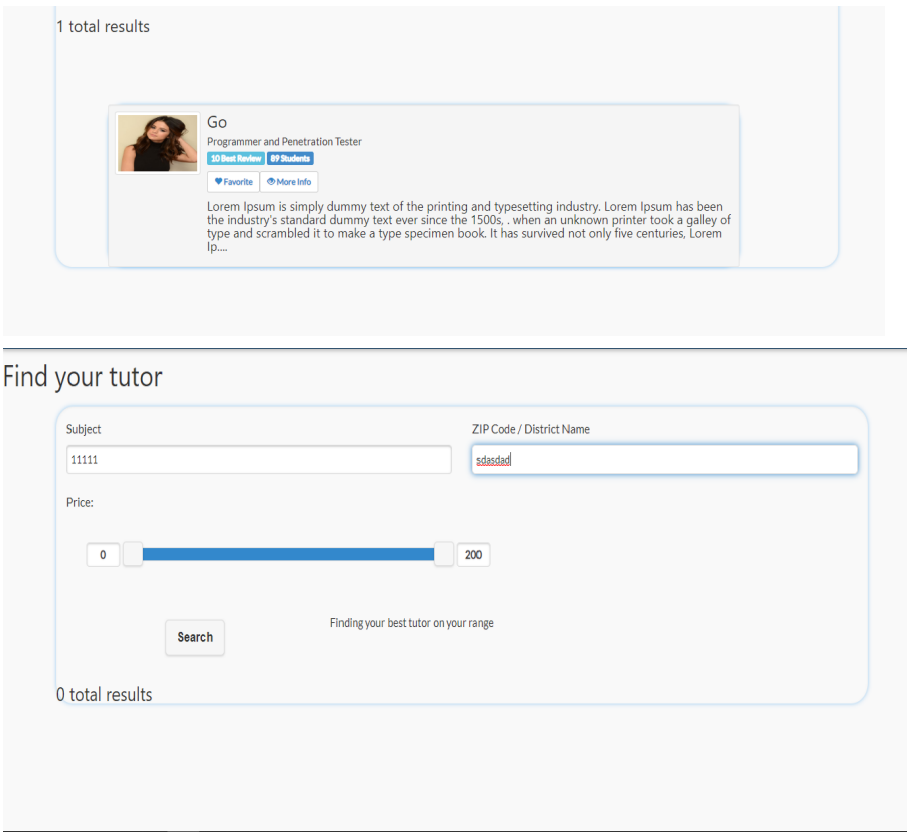
Input Condition	**Valid Equivalence Class**	**Valid Boundaries**	**Invalid Equivalence Class**	**Invalid Boundaries**
Username				
Content	All characters are available but the characters must be existed in the database		Not characters(3)	
Password				
Size	6 - 100	6(1) - 100(2)	<6, >100	1(5) , 101(6)
Type	Characters		Not characters(4)	

Valid Test Cases	**Boundaries Covered**
1. Test123 2. Test_123 3. ABCDEFGHIJKLMNOPQRSTUVWXYZ_qwertyuiopasdfghjklzxcvbnm12345678900.-0.9562iomnbvcxzxcasddsadsaddda	(1),(2)

Invalid Test Case	Boundaries Covered
1. !@#$%^&*()+={}[]:;”’<>,?/|\~`	(3),(4)
1. Q 2. User cannot input more than 101st letters because 100 is the maximum amount.	(5) (6)

B.2 Searching Page

Test Number	2
Objectives	Search tutor
Test Data	Subject, Zip code and price tag
Expected Output	If the searching is successful, the search result will list out a list of tutor for users to view their profile and add them as favourite tutor in the system. If the searching is not successful, the search result will not list out any tutors.
Screenshot (Actual Result)	

	
Conclusion	In order to make the searching successful, the searching requirement must be able to match its character in the database.

White Box Testing : Multiple Condition Coverage Sample

Inputs	<input type="text" class="form-control" id="subject" name="subject"> <input type="text" class="form-control" id="zip" name="zip"> <input type="range" name="price-min" id="price-min" value="0" min="0" max="200"> <input type="range" name="price-max" id="price-max" value="200" min="0" max="200">
Code	if(isset($_SESSION["username1"]) && isset($_SESSION["identity"]) && $_SESSION["actype"]) { if (isset($_POST["submit"])){

```
      $val1 = $_POST["subject"];
      $val2 = $_POST["zip"];
      $val3 = $_POST["price-min"];
      $val4 = $_POST["price-max"];

    if ($val2 != ""){
    if (ctype_digit($val2) && preg_match('#[0-9]{5}#', $val2)){
      $query1 = "SELECT d.tutor_id ,`t_fname` ,`t_lname` , `t_profilepic`, `t_travel` ,
`t_job`, `t_intro`
      FROM subject f ,subject_offer e, tutor d
      WHERE e.subject_id =f.subject_id
      AND f.subject_name = :subject
      AND d.t_zip =:zip
      AND d.tutor_id = e.tutor_id
      AND d.t_officialcheck = '1'
      AND e.price_perhrs BETWEEN :min AND :max";
      // searching by zip code= "d.t_zip";

      $search=$conn->prepare($query1);
      $search->bindParam(':subject', $val1, PDO::PARAM_STR);
      $search->bindParam(':zip', $val2, PDO::PARAM_INT);
      $search->bindParam(':min', $val3, PDO::PARAM_INT);
      $search->bindParam(':max', $val4, PDO::PARAM_INT);
      $search->execute();
      $total = $search->rowCount();
    }else{
      $query1 = "SELECT d.tutor_id ,`t_fname` ,`t_lname` , `t_profilepic`, `t_travel` ,
`t_job`, `t_intro`
      FROM subject f ,subject_offer e, tutor d
      WHERE e.subject_id =f.subject_id
      AND f.subject_name = :subject
      AND d.t_district =:dis
      AND d.tutor_id = e.tutor_id
      AND d.t_officialcheck = '1'
      AND e.price_perhrs BETWEEN :min AND :max";
      // searching by district = "d.district";
      $search=$conn->prepare($query1);
      $search->bindParam(':subject', $val1, PDO::PARAM_STR);
      $search->bindParam(':dis', $val2, PDO::PARAM_STR);
      $search->bindParam(':min', $val3, PDO::PARAM_INT);
      $search->bindParam(':max', $val4, PDO::PARAM_INT);
      $search->execute();
      $total = $search->rowCount();
    }
  }else{
    $query1 = "SELECT d.tutor_id ,`t_fname` ,`t_lname` , `t_profilepic`, `t_travel` ,
`t_job`, `t_intro`
```

```
    FROM subject f ,subject_offer e, tutor d
    WHERE e.subject_id =f.subject_id
    AND f.subject_name = :subject
    AND d.tutor_id = e.tutor_id
    AND d.t_officialcheck = '1'
    AND e.price_perhrs BETWEEN :min AND :max";
    // searching by district = "d.district";
    $search=$conn->prepare($query1);
    $search->bindParam(':subject', $val1, PDO::PARAM_STR);
    $search->bindParam(':min', $val3, PDO::PARAM_INT);
    $search->bindParam(':max', $val4, PDO::PARAM_INT);
    $search->execute();
    $total = $search->rowCount();
  }

 }

 }
else
{
    session_destroy();
    header("location:loging.php");
}
```

	(isset($_POST["submit"]))
1	T
2	F

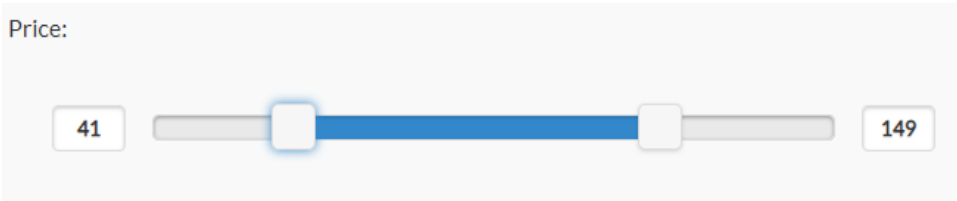
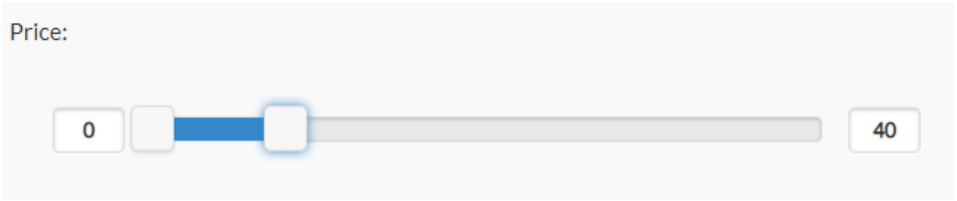
	(ctype_digit($val2) && preg_match('#[0-9]{5}#', $val2))
1	T
2	F

Subject	Zip Code	Comment
Programming	-	If Zip Code is empty, the result will be still listed out.
-	31920	If Subject entry is empty, the result will not be list out.
-	-	If both the subject and Zip Code is not filled up, the searching will failed because empty information is not existed in the database.
Programming	31920	If all searching criteria is filled up, the search engine will search existing information in the database related to the searching criteria given.

Black Box Testing

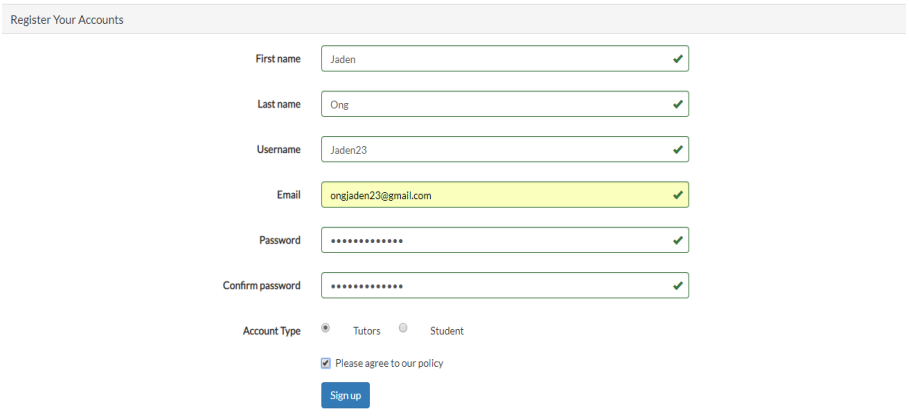
The input of the program consists of a subject's textbox which may be insert up to 100 letters in length, whereas the Zip Code can only insert up to 5 numbers.

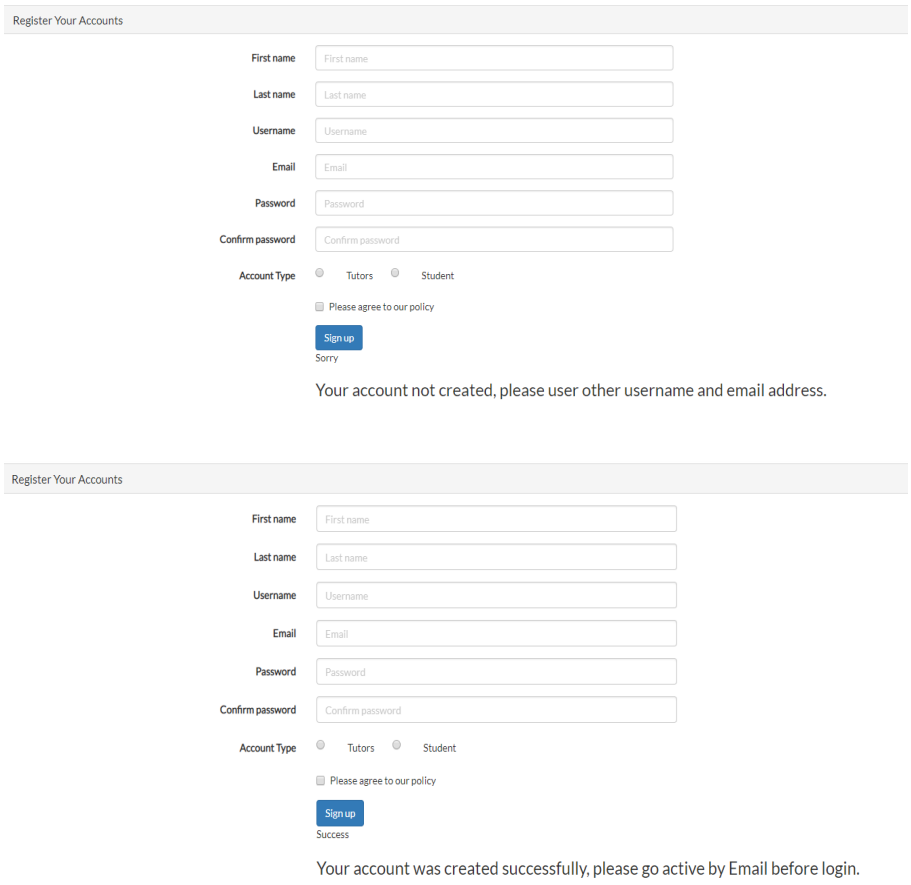
Input Condition	**Valid Equivalence Class**	**Valid Boundaries**	**Invalid Equivalence Class**	**Invalid Boundaries**
Subject				
Content	Subject names	All characters except symbols(1)	Not characters(4)	
Zip Code				
Size	5	5(2)	>5, >6, <1	6(6), 10(7), 0(8)
Type	Numbers		Not numbers(5)	
Prize				
Content	Range Slider	Adjust the range slider in any order(3)		

Valid Test Cases	Boundaries Covered
1. Chinese 2. English 3. Mathematics 4. Programming	(1)
1. 31920 2. 14000	(2)
  	(3)

Invalid Test Case	Boundaries Covered
1. !@#$%^&*()+={}[]:;”’<>,?/|\~`	(4)
1. !@#$% 2. Abcde 3. …..	(5)
1. 178900 2. 9999999999 3.	(6) (7) (8)

B.3 User Register Page

Test Number	3
Objectives	To register an account
Test Data	First name, Last name, Username, Email, Password, Confirm Password, Account Type
Expected Output	If the registration is successful, the user must access into their personal email to activate their account. But as for user who registered as a tutor will not be able to activate their account, they will received a notice in the email that they have to go for an interview before they are approvable. If the registration fails, a message error will be prompt.
Screenshot (Actual Result)	

	
Conclusion	In order to to register successfully, all information entered by user must not match exactly the same information in the database.

White Box Testing : Multiple Condition Coverage Sample

Inputs	<input type="text" class="form-control" id="firstname1" name="firstname1" placeholder="First name" /> <input type="text" class="form-control" id="lastname1" name="lastname1" placeholder="Last name" /> <input type="text" class="form-control" id="username1" name="username1" placeholder="Username" />

	<input type="text" class="form-control" id="email1" name="email1" placeholder="Email" /> <input type="password" class="form-control" id="password1" name="password1" placeholder="Password" /> <input type="password" class="form-control" id="confirm_password1" name="confirm_password1" placeholder="Confirm password" /> <input type="radio" class="radio-inline" id="tutors" name="radio" value="tutors" /> <input type="radio" class="radio-inline" id="student" name="radio" value="student" /> <input type="checkbox" id="agree1" name="agree1" value="agree" />Please agree to our policy
Code	if (isset($_POST['signup1'])) { if (hash_equals($csrf, $_POST['csrf'])) { try { $stmt_semail = $conn->prepare("SELECT * FROM student WHERE stud_email=:email"); $stmt_temail = $conn->prepare("SELECT * FROM tutor WHERE t_email=:email"); $stmt_susrn = $conn->prepare("SELECT * FROM student WHERE stud_usrn=:usrn"); $stmt_tusrn = $conn->prepare("SELECT * FROM tutor WHERE t_usrn=:usrn"); $stmt_semail->bindParam(':email', $email, PDO::PARAM_STR); $stmt_temail->bindParam(':email', $email, PDO::PARAM_STR); $stmt_susrn->bindParam(':usrn', $usrn, PDO::PARAM_STR); $stmt_tusrn->bindParam(':usrn', $usrn, PDO::PARAM_STR); $stmt_semail->execute(); $stmt_temail->execute(); $stmt_susrn->execute(); $stmt_tusrn->execute(); $results1 = $stmt_semail-> fetch(); $results2 = $stmt_temail-> fetch(); $results3 = $stmt_susrn-> fetch(); $results4 = $stmt_tusrn-> fetch(); } catch(PDOException $e){ { echo "Error: " . $e->getMessage(); } } if ($results1 >0 || $results2>0 || $results3 >0 || $results4 >0){

```
    header('location: registering.php?fail=3');
    if ($results1 >0 ||$results2 >0){
        header('location: registering.php?fail=1');
    }
    if ($results3 >0 || $results4 >0){
        header('location: registering.php?fail=2');
    }
    if($results1 >0 && $results3 >0 || $results2 >0 &&$results4 >0){
        header('location: registering.php?fail=3');
    }

}else {
    if($str=="student"){
        try {

            // prepare sql and bind parameters
        $stmt = $conn->prepare("INSERT INTO student (stud_fname, stud_lname,
stud_email, stud_usrn, stud_pass, stud_hash , stud_registerdate)
        VALUES (:firstname, :lastname, :stud_email, :stud_usrn, :stud_pass, :stud_hash
, NOW())");
        $stmt->bindParam(':firstname', $firstname,PDO::PARAM_STR);
        $stmt->bindParam(':lastname', $lastname,PDO::PARAM_STR);
        $stmt->bindParam(':stud_email', $email,PDO::PARAM_STR);
        $stmt->bindParam(':stud_usrn', $usrn,PDO::PARAM_STR);
        $stmt->bindParam(':stud_pass', $passwordHash,PDO::PARAM_STR);
        $stmt->bindParam(':stud_hash', $hash,PDO::PARAM_STR);
        // insert a row
        $stmt->execute();
        sendActiveMail($email,$firstname,$hash,$str);
        header('location: registering.php?success=1');
        }
    catch(PDOException $e)
        {
        echo "Error: " . $e->getMessage();
        }
        $conn = null;
    }

    if($str=="tutors"){
        try {
            // prepare sql and bind parameters
        $stmt = $conn->prepare("INSERT INTO tutor (t_fname, t_lname, t_email, t_usrn,
t_pass, t_hash , t_registerdate)
        VALUES (:firstname, :lastname, :t_email, :t_usrn, :t_pass, :t_hash , NOW())");
        $stmt->bindParam(':firstname', $firstname,PDO::PARAM_STR);
        $stmt->bindParam(':lastname', $lastname,PDO::PARAM_STR);
        $stmt->bindParam(':t_email', $email,PDO::PARAM_STR);
```

```
        $stmt->bindParam(':t_usrn', $usrn,PDO::PARAM_STR);
        $stmt->bindParam(':t_pass', $passwordHash,PDO::PARAM_STR);
        $stmt->bindParam(':t_hash', $hash,PDO::PARAM_STR);

        $stmt->execute();
        sendActiveMail($email,$firstname,$hash,$str);
        header('location: registering.php?success=1');
        }
    catch(PDOException $e)
        {
        echo "Error: " . $e->getMessage();
        }
        $conn = null;
    }
}
}else{
            session_destroy();
            header('location: registering.php?fail=1');
}
}
function sendActiveMail($email,$firstname,$hash,$str){
    $_SESSION['active'] = 0; //0 until user activates their account with verify.php
    $_SESSION['logged_in'] = true; // So we know the user has logged in
    $_SESSION['message'] =

                "Confirmation link has been sent to $email, please verify
                your account by clicking on the link in the message!";

        // Send registration confirmation link (verify.php)
        $to      = $email;
        $subject = 'Account Verification ( 24hrs Tutor.com )';
        $message_body = '
        Hello '.$firstname.',

        Thank you for signing up!

        Please click this link to activate your account:

        http://localhost:8080/24t/verify.php?email='.$email.'&hash='.$hash.'&type='.$str;

        mail( $to, $subject, $message_body );

        //header("location: profile.php");
}
```

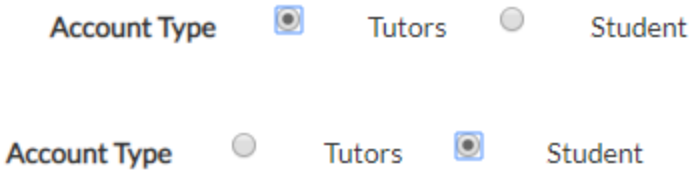
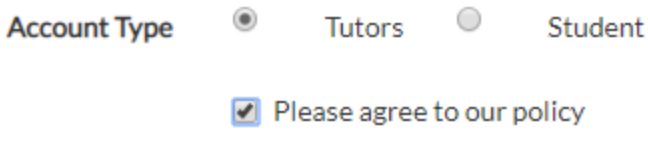
	($results1 >0 || $results2>0 || $results3 >0 || $results4 >0)
1	T
2	F (student email, tutor email, student username, tutor username)

Black Box Testing

The input of the program consists of a email's textbox which may insert up to 100 characters in length. The password textbox which the user must insert at least 6 characters and up to 100 characters in length, in addition the confirmation password textbox must be the same type and length of password as the password in password textbox. Account type contains two radio buttons.

Input Condition	**Valid Equivalence Class**	**Valid Boundaries**	**Invalid Equivalence Class**	**Invalid Boundaries**
Email				
Content	**Mail(1)**	**'@'(2) '.com'(3) 23(4)**	**non mail(5)**	-
Password				
Size	6-100	6(6) 100(7)	<6, >100	5(9) 101(10)
Type	Characters		Not Characters(8)	
Confirmation Password				
Size	**Same length as Password(11)**		**Different length then Password(13)**	

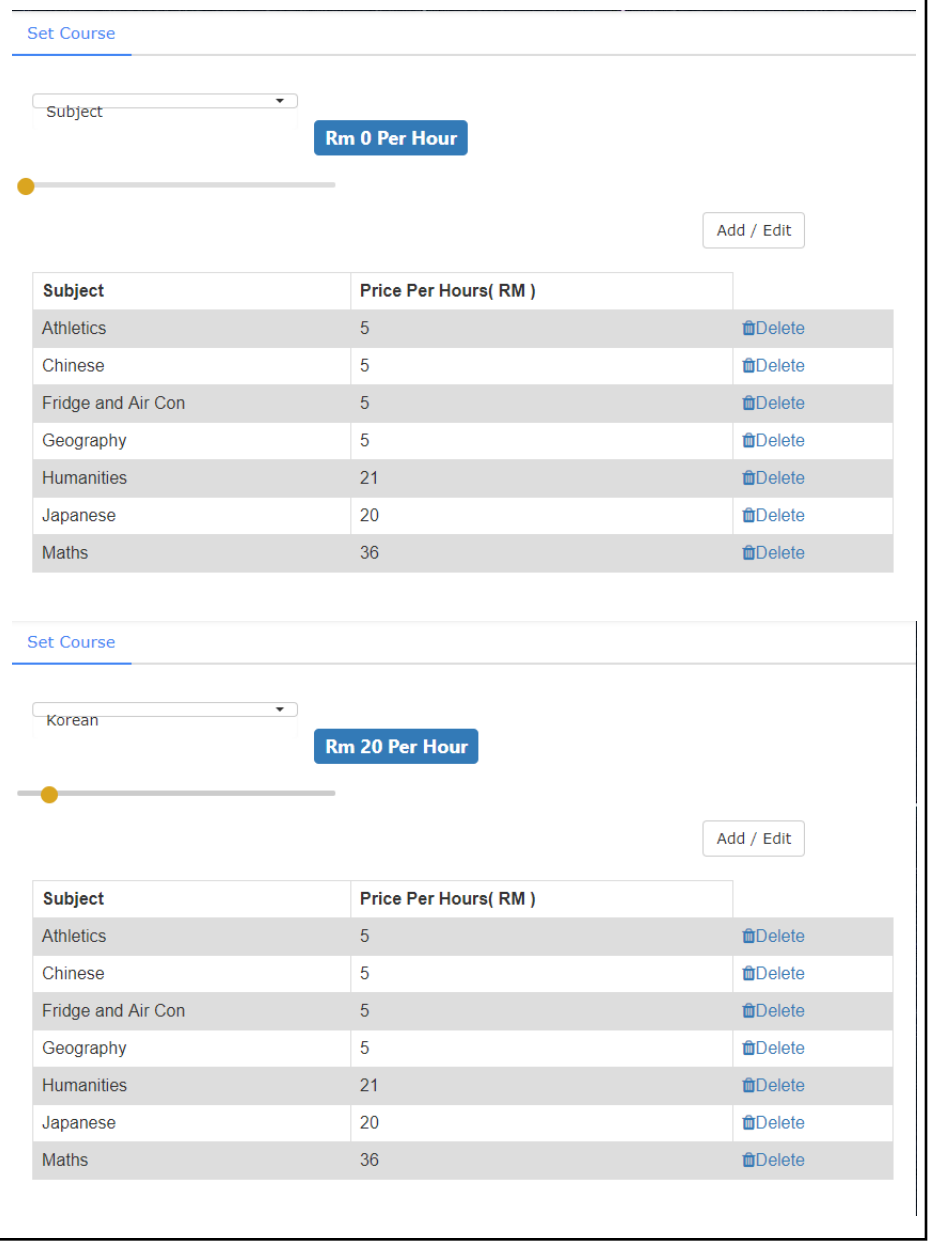
Type	Same type as Password[12]		Different type with Password[14]	
Account Type				
Content	**Radio Button**	**Select the radio button[15]**		**Did not select any radio button[16]**
Agree Policy				
Content	**Checkbox**	**Tick the checkbox[17]**		**Did not tick checkbox[18]**

Valid Test Cases	Boundaries Covered
1. example23@gmail.com 2. abcdefghijklmnopqrstuvwxyz_1234567890@gmail.com	(1),(2),(3),(4)
1. example23@gmail.com 2. abcdefghijklmnopqrstuvwxyz_1234567890@gmail.com 3. Tutor123 4. Tutor_123 5. tutor.123	(6), (7), (11),(12)
	(15)
	(17)

Invalid Test Cases	Boundaries Covered
1. Haha 2. Q 3. Example.com 4. &*#!!@gmail.com	(5)
1. !@#$%^&*()+={}[]:;"'<>,?/|\~` 2. TestA 3. Users cannot insert 101st characters because the maximum amount of character that the user can insert is 100th.	(8),(14) (9),(13) (10)
Account Type ○ Tutors ○ Student	(16)
☐ Please agree to our policy	(18)

B.4 Set Course Page

Test Number	4
Objectives	Add in more subjects for tutor.
Test Data	Add subject, Delete subject, Price Range
Expected Output	If adding new subject is successful, the container will list out the name of the subject and the price. If adding is failed, the system will not list out anything to the user.

Screenshot (Actual Result)	

Set Course

Subject

Rm 0 Per Hour

Add / Edit

Subject	Price Per Hours(RM)	
Athletics	5	Delete
Chinese	5	Delete
Fridge and Air Con	5	Delete
Geography	5	Delete
Humanities	21	Delete
Japanese	20	Delete
Maths	36	Delete

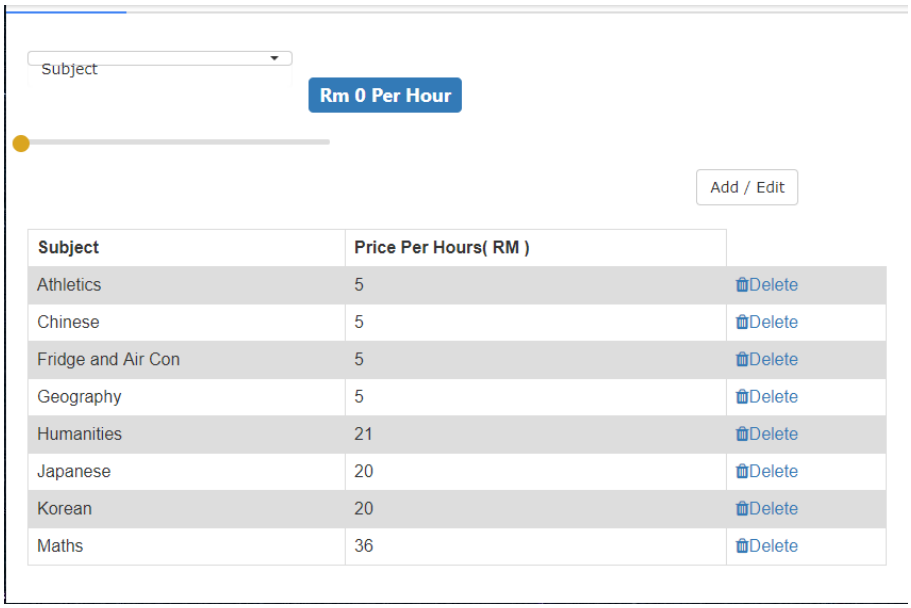
Set Course

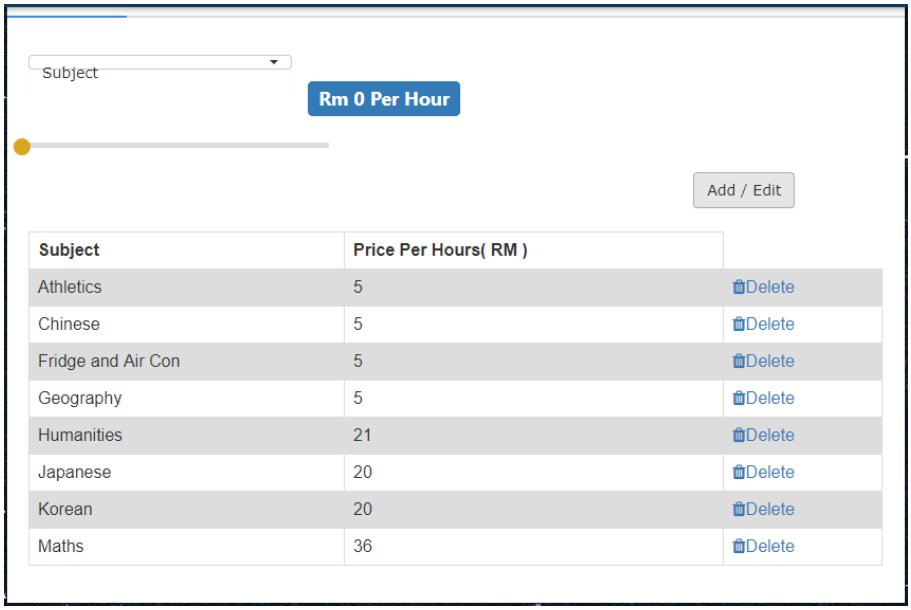
Korean

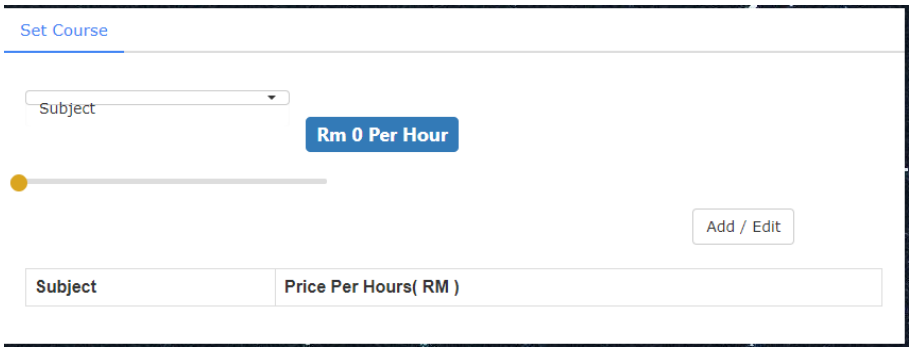
Rm 20 Per Hour

Add / Edit

Subject	Price Per Hours(RM)	
Athletics	5	Delete
Chinese	5	Delete
Fridge and Air Con	5	Delete
Geography	5	Delete
Humanities	21	Delete
Japanese	20	Delete
Maths	36	Delete


Subject
Rm 0 Per Hour
Add / Edit
Subject
Price Per Hours(RM)
Athletics 5 Delete
Chinese 5 Delete
Fridge and Air Con 5 Delete
Geography 5 Delete
Humanities 21 Delete
Japanese 20 Delete
Korean 20 Delete
Maths 36 Delete


Subject
Rm 0 Per Hour
Add / Edit
Subject
Price Per Hours(RM)
Athletics 5 Delete
Chinese 5 Delete
Fridge and Air Con 5 Delete
Geography 5 Delete
Humanities 21 Delete
Japanese 20 Delete
Korean 20 Delete
Maths 36 Delete


Set Course
Subject
Rm 0 Per Hour
Add / Edit
Subject
Price Per Hours(RM)

Conclusion	In order to add subject successfully, the user has to pick a subject from the drop down list.

White Box Testing : Multiple Condition Coverage Sample

Inputs	<select id="multipleSelect1" class="selectpicker show-menu-arrow form-control" single name='category'> <input type='range' min='5' max='200' margin-left: 215px; value='<?php echo $res;?>' step='1' name='price' onChange='sliderChange(this.value)' /> <?php foreach ($query2 as $row) { echo "<tr><td>".$row['subject_name']."</td> <td>".$row['price_perhrs']."</td><td> <i class='fa fa-trash'></i>Delete</td> </tr>"; } ?>
Code	if(isset($_SESSION["username1"]) && isset($_SESSION["identity"]) && $_SESSION["actype"] ==="tutor") { if(isset($_GET['del'])){ $delval = $_GET['del']; $delQ = $conn->prepare('DELETE FROM `subject_offer` WHERE `subject_id`=:dl AND `tutor_id` = :id '); $delQ->bindParam(':dl', $delval, PDO::PARAM_INT); $delQ->bindParam(':id', $_SESSION["identity"], PDO::PARAM_INT); $delQ->execute(); } $query = $conn->prepare('SELECT * FROM subject ORDER BY subject_name'); $query->execute(); $query2 = $conn->prepare('SELECT c.subject_name , d.subject_id , d.offer_id , d.price_perhrs FROM subject c , subject_offer d WHERE d.tutor_id =:id AND c.subject_id = d.subject_id ORDER BY c.subject_name'); $query2->bindParam(':id', $_SESSION["identity"], PDO::PARAM_INT); $query2->execute(); //query to show user's subjects

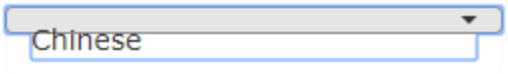
```
  } else
  {
      session_destroy();
      header("location:loging.php");
  }

    if(isset($_POST['submit'])&& isset($_POST['category']) && $_POST['category']
!==''){
      $val1 = $_POST['category'];  $val3 = $_POST['price'];  $set ='';
      try {              // Use to add or update subject price
      foreach ($query2 as $row2) {
        if ($row2['subject_id']==$val1){
          $set = '1';
        }else if ($row2['subject_id']!==$val1){
          $set = '0';
        }else{
          header("location:setcourse.php");
        }
      }
      if ($set==1){
      $addsubject = $conn->prepare("UPDATE `subject_offer` SET
`price_perhrs`=:v3 WHERE `subject_id`=:v1 AND `tutor_id` =:v2");
      }else if ($set==0){
        $delQ2 = $conn->prepare('DELETE FROM `subject_offer` WHERE
`subject_id`=:ct AND `tutor_id` = :id ');
        $delQ2->bindParam(':ct', $_POST['category'], PDO::PARAM_INT);
        $delQ2->bindParam(':id', $_SESSION["identity"], PDO::PARAM_INT);
        $delQ2->execute();
        $addsubject = $conn->prepare("INSERT INTO `subject_offer` (`subject_id`,
`tutor_id`, `price_perhrs`) VALUES ( :v1 , :v2 , :v3)");
      }
      $addsubject->bindParam(':v1', $val1, PDO::PARAM_INT);
      $addsubject->bindParam(':v2', $_SESSION["identity"], PDO::PARAM_INT);
      $addsubject->bindParam(':v3', $val3, PDO::PARAM_INT);
      $addsubject->execute();
      header("location:setcourse.php");
    }
  catch(PDOException $e)
      {
      echo "Error: " . $e->getMessage();
      }
      $conn = null;
  }
```

	(isset($_POST['submit'])&& isset($_POST['category']) && $_POST['category'] !=='')
1	T
2	F

Black Box Testing

The drop down list contains the list of subjects stores in the database. Range slider is to slide the price range of the adding subject.

Input Condition	Valid Equivalence Class	Valid Boundaries	Invalid Equivalence Class	Invalid Boundaries
Subject				
Content	**Drop down list**	**Select a subject from the drop down list(1)**		**Did not select a subject from the drop down list.(3)**
Price				
Content	**Range Slider**	**Adjust slider at a certain range(2)**		**Did not adjust slider at a certain range(4)**

Valid Test Cases	Boundaries Covered
	(1)
	(2)

Invalid Test Cases	Boundaries Covered
	(3)
	(4)

Appendix C: Documents

C.1 PID

Project Initiation Document

Supervisor	Ang Sau Loong
Group Name	**Sam**
Project Title	**Find Lecturer**
Student Names & ID	**Ong Wei Cheng 0188666** **Bryan Lim 0192968** **Kenneth Chong 0191676** **Yeap Chi Hang 0188332** **Go Yik Yek 0188039**

Aim of project :

To allow students to find available lecturer around your area for tuition within your price ranges in addition students are able to select tuition venue from the list given.

Rationale for project :

1. Students are able to find available lecturer for private tuition or group tuition.
2. Students are able to find lecturers with a certain amount of price ranges for tuition.
3. Parents are able to know their children's study progress and their study materials.
4. Tuition venue can be selected from the list given.
5. Tutors does not need to find students by advertising using flyers.

The main challenge is :

1. Scheduling the class.
2. User security is not guarantee of safety.
3. Sorting time of the class.
4. Technically complicated like designing search algorithms.

Type of product to be produced :

Php, Visual studio code, Notepad, Subline Text, Xampp, Php myAdmin and Responsive Web Design or Android Studio

Resources required :

Computer and internet

Any external body involved ? If so, who ?

Students' Signatures :

Figure 51 : PID

C.2 Draft UI

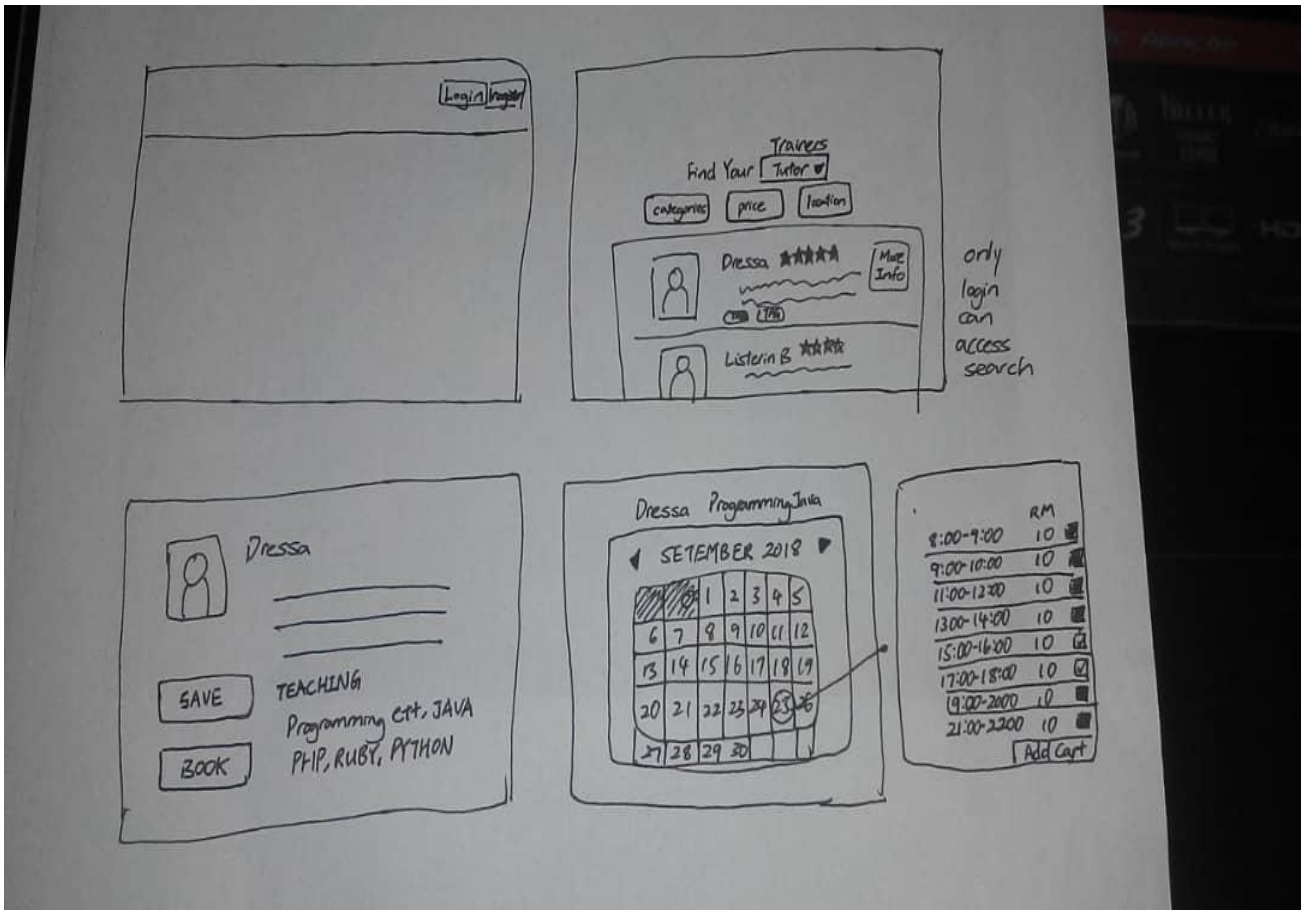


Figure 52: Draft User Interface

Reference

1. Marissa Higgins, 2016, Technology in the 90's versus today, Bustle, viewed 16 September 2018, <https://www.bustle.com/articles/187994-technology-in-the-90s-versus-today>

2. Margaret Rouse, 2017, What is online data backup, TechTarget, viewed 16 September 2018, <https://searchdatabackup.techtarget.com/definition/online-data-backup>

3. Gregory Hamel, 2018, Advantages and disadvantages of traditional file organization, Chron, viewed 16 September 2018, <https://smallbusiness.chron.com/advantages-disadvantages-traditional-file-organization-41400.html>

4. Christine, 2017, The evolution of communication technology, SmarterWare, viewed 17 September 2018, <http://smarterware.org/2017/01/evolution-communication-technology/>

5. Karehka Ramey, 2012, 10 Uses of technology in our daily life, useoftechnology, viewed 17 September 2018, <https://www.useoftechnology.com/technology-today-tomorrow/>

6. Sara Ashley O'Brien, Nelli Black, Curt Devine and Drew Griffin, 2018, CNN investigation: 103 Uber drivers accused of sexual assault or abuse, CNN Business, viewed 17 September 2018, <https://money.cnn.com/2018/04/30/technology/uber-driver-sexual-assault/index.html>
7. Team Clarizen, 2018, What are the objectives of project management, Clarizen, viewed 17 September 2018, <https://www.clarizen.com/objectives-of-project-management/>

8. Margaret Rouse, 2018, functional requirements, WhatIs.com, viewed 18 September 2018, <https://whatis.techtarget.com/definition/functional-requirements/>.

9. John Spacey, 2017, 14 Types of Technical Feasibility, Simplicable, viewed 25 September 2018, <https://simplicable.com/new/technical-feasibility/>.

10. Gaston, 2018, Introduction to Recommender Systems in 2018, viewed 17 October 2018, <https://tryolabs.com/blog/introduction-to-recommender-systems/>

11. Recommender-System, Content-based Filtering, recommender-systems.org, viewed 17 October 2018, <http://recommender-systems.org/content-based-filtering/>.

12. N Lakshmipathi Ananth, Bhanu Prakash Bhattula, A REVIEW ON RECOMMENDATION SYSTEM USING RATING DATASET, International Journal of Pure and Applied Mathematics, viewed 19 October 2018, <https://acadpubl.eu/jsi/2017-116-5-7/articles/5/23.pdf/>.

13. Ellie Kutsevol, 2018, Facilitating a System of Engagement with Ratings Feature, colibra, viewed viewed 19 October 2018, <https://www.collibra.com/blog/facilitating-a-system-of-engagement-with-ratings-feature/>

14. Margaret Rouse, 2018, Definition of Functional Requirements, TechTarget, viewed 19 October 2018, <https://whatis.techtarget.com/definition/functional-requirements>

15. Andrew Powell-Morse, 2016, Waterfall Model: What Is It and When Should You Use It?, Airbrake, viewed 19 October 2018, <https://airbrake.io/blog/sdlc/waterfall-model>

16. https://www.evirtualservices.com/economical-feasibility

17. https://www.energy.gov/eere/fuelcells/technological-feasibility-and-cost-analysis

18. Margaret Rouse, 2015, Use Case Diagram, TechTarget, viewed 20 October 2018, <https://whatis.techtarget.com/definition/use-case-diagram>

19. Maria Ericsson, 2004, Activity diagrams: What they are and how to use them, IBM, viewed 20 October 2018, <https://www.ibm.com/developerworks/rational/library/2802.html>

20. Margaret Rouse, 2007, Definition of Gantt Chart, TechTarget, viewed 20 October 2018, <https://searchsoftwarequality.techtarget.com/definition/Gantt-chart>

21. Margaret Rouse, 2018, Definition of entity relationship diagram (ERD), TechTarget, viewed 20 October 2018, <https://searchdatamanagement.techtarget.com/definition/entity-relationship-diagram-ERD>

22. Daniel Nations, 2018, What Exactly Is a Web Application?, viewed 20 October 2018, <https://www.lifewire.com/what-is-a-web-application-3486637>

23.

24 Hours Tutor

Website User Manual

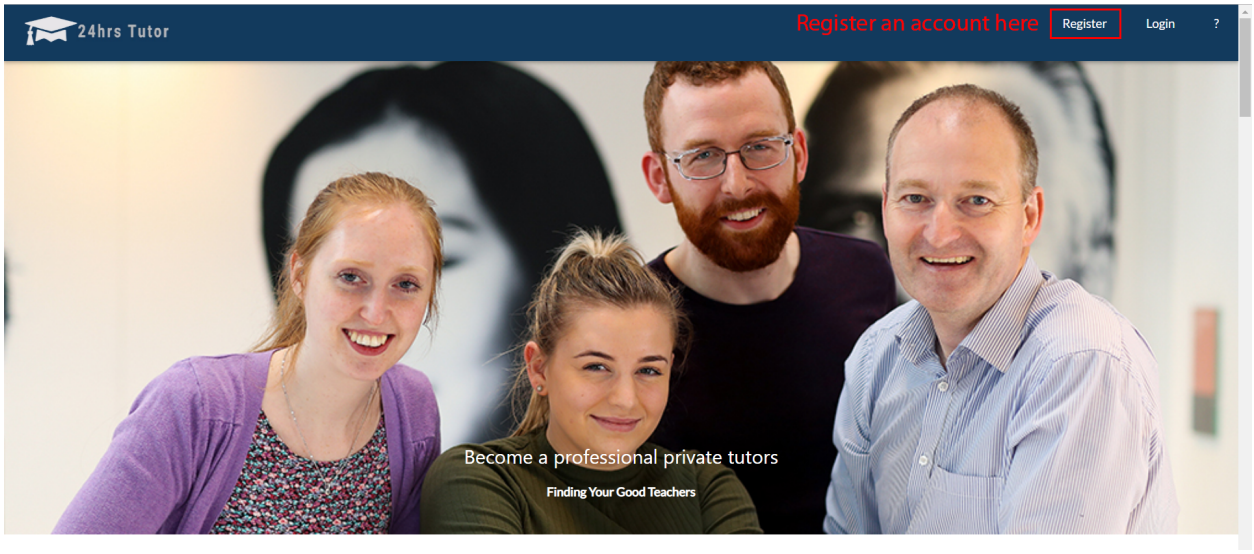
Introduction

The 24 Hours Tutor (24hrs Tutor) is a finding tutor for tuition system that is aimed to help both students and tutors. As for students, we aimed to help them by improving their academic grades and gaining new knowledge in other fields such as not related from school works. In addition as for tutors, we aimed to help them by securing a job.

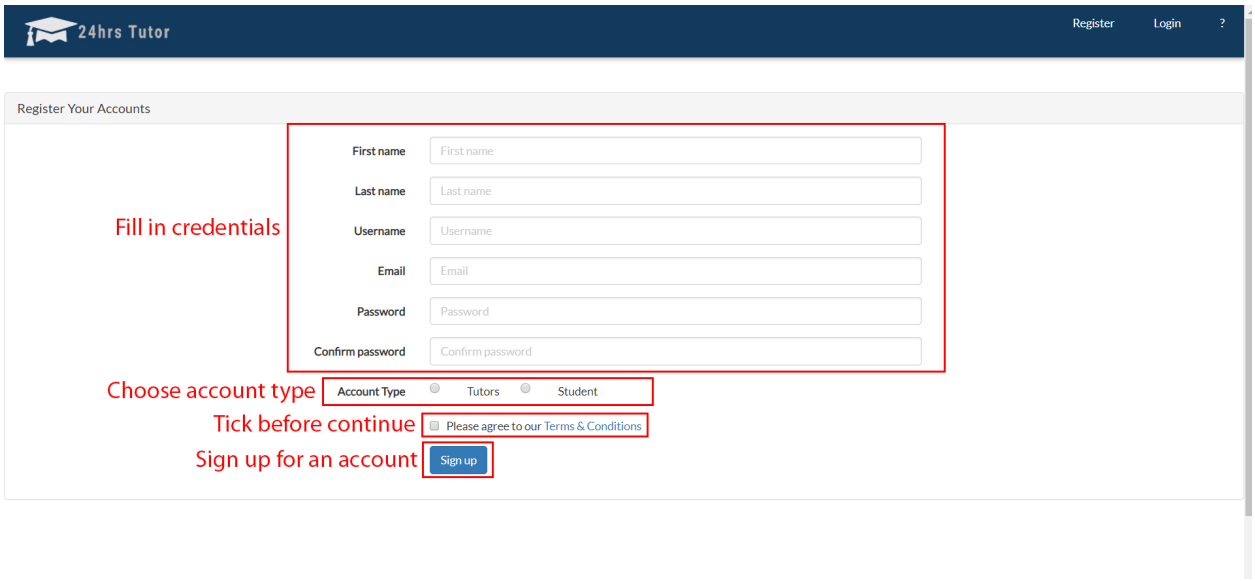
This document will guide users through necessary steps required to achieve the user's aim.

Table of Contents

i. Register (Student/Tutor)



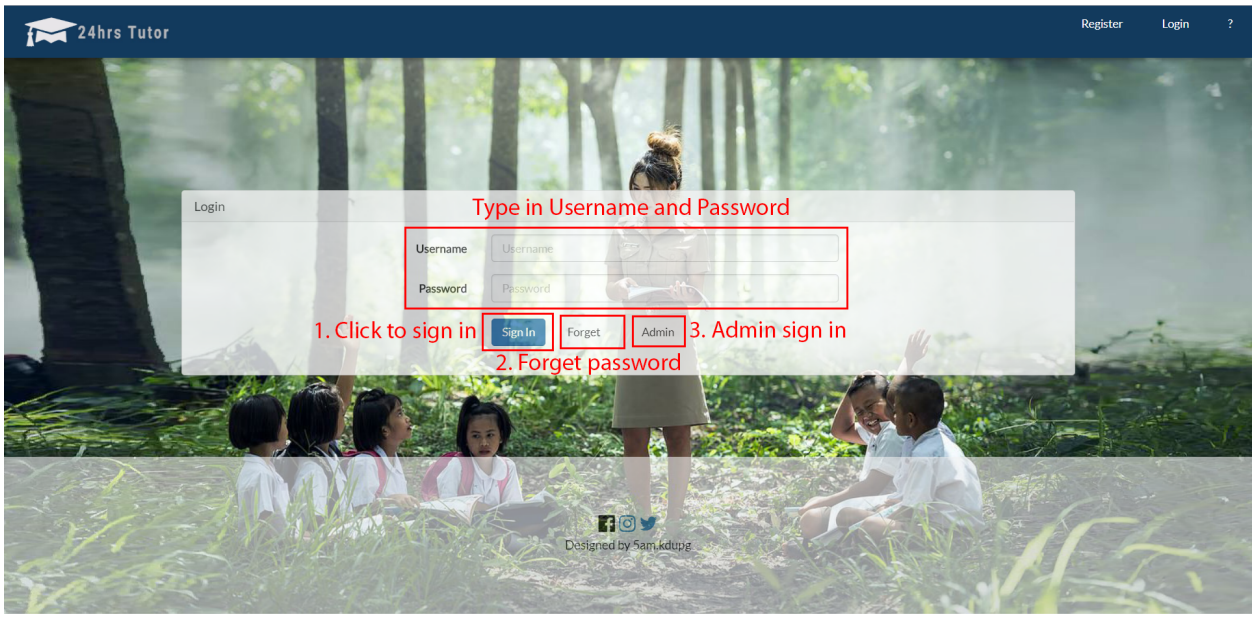
Users can find the register button at the top right corner of the home page.



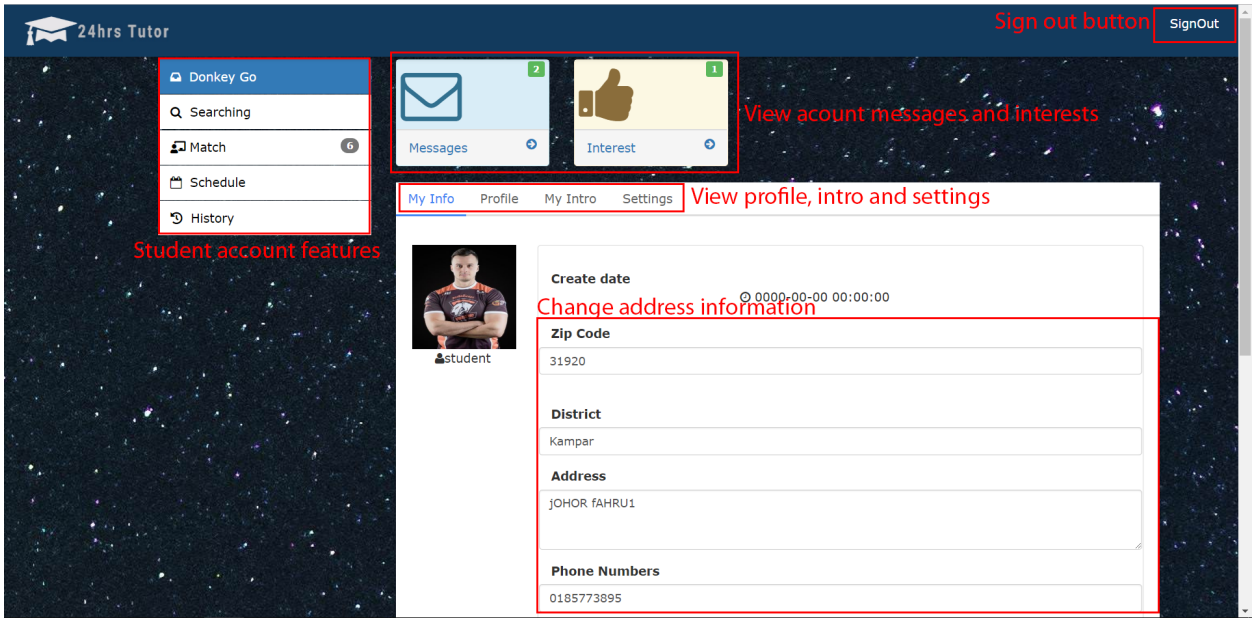
After clicking on the register button, users will be directed to the register page. Users are required to fill in their credentials in the empty text boxes. After filling in everything, users will have to choose their account type whether they are signing up as a student or a tutor. Before proceeding, users are required to click on the checkbox to agree with the terms and conditions of the system. After that, users will be able to proceed to sign up for an account. If users are interested in registering as a tutor, users will be prompted an email by the system to follow and carry out required steps to be a tutor.

ii. Login

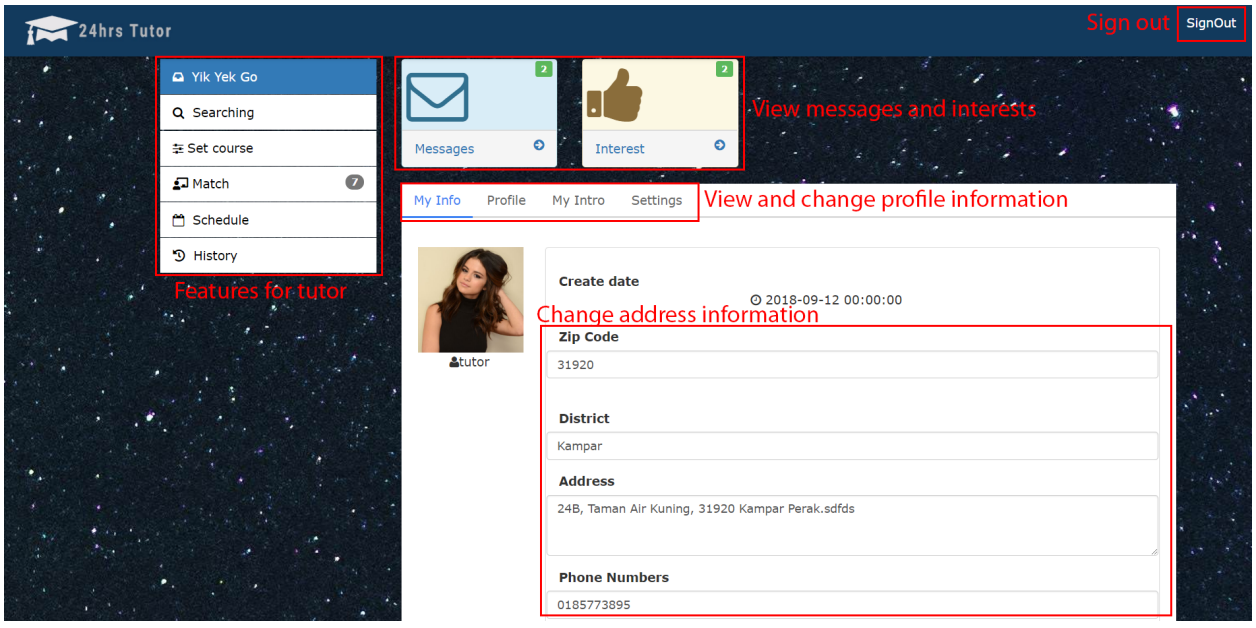
Users can log in through through the main menu by clicking on the Login button at the top right corner of the page.



After that, users will be redirected to the login page where users will have to fill up their registered username and password. After entering the credentials, users will have to click at the 1. Sign in button to continue. If users forget their password, they can click on 2. Forget password button to reset their password. 3. Admin button is for admins to sign in to the system.

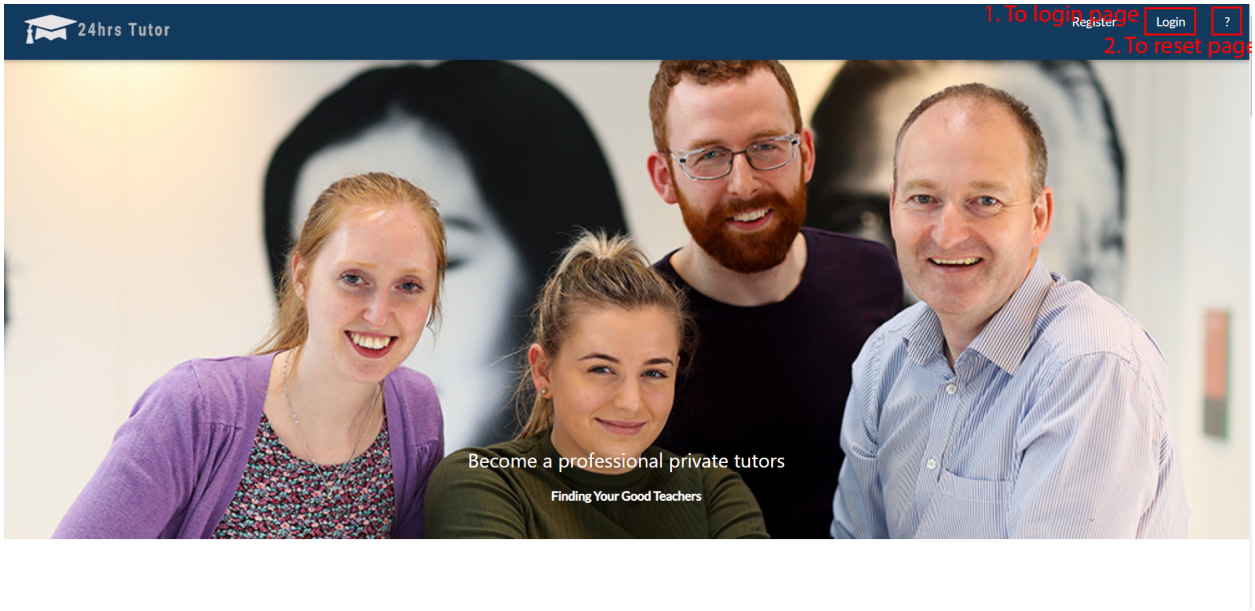


After users pressed the sign in button, users will be signed in to their account and will be redirected to the profile page. Users can view their messages and interests and use the student account features in the profile page. Users can also view their information and or change their information in the profile page.

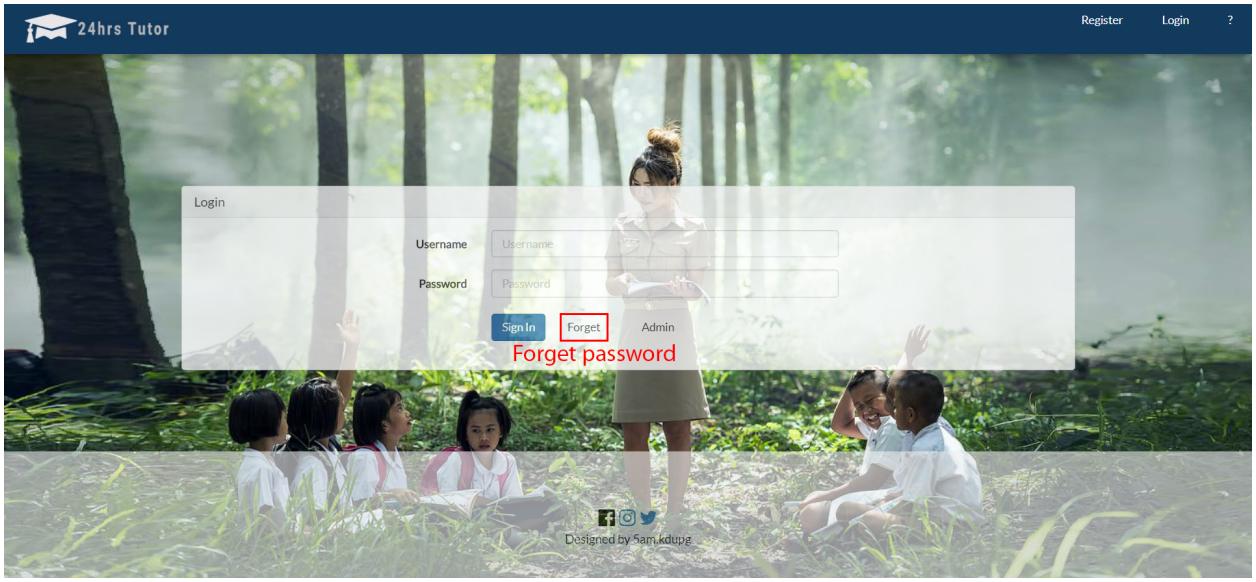


If the user's username and password is verified as a tutor, they will automatically be directed to the tutor profile site where they have different features but the same functions of viewing information, messages, interests and changing personal information.

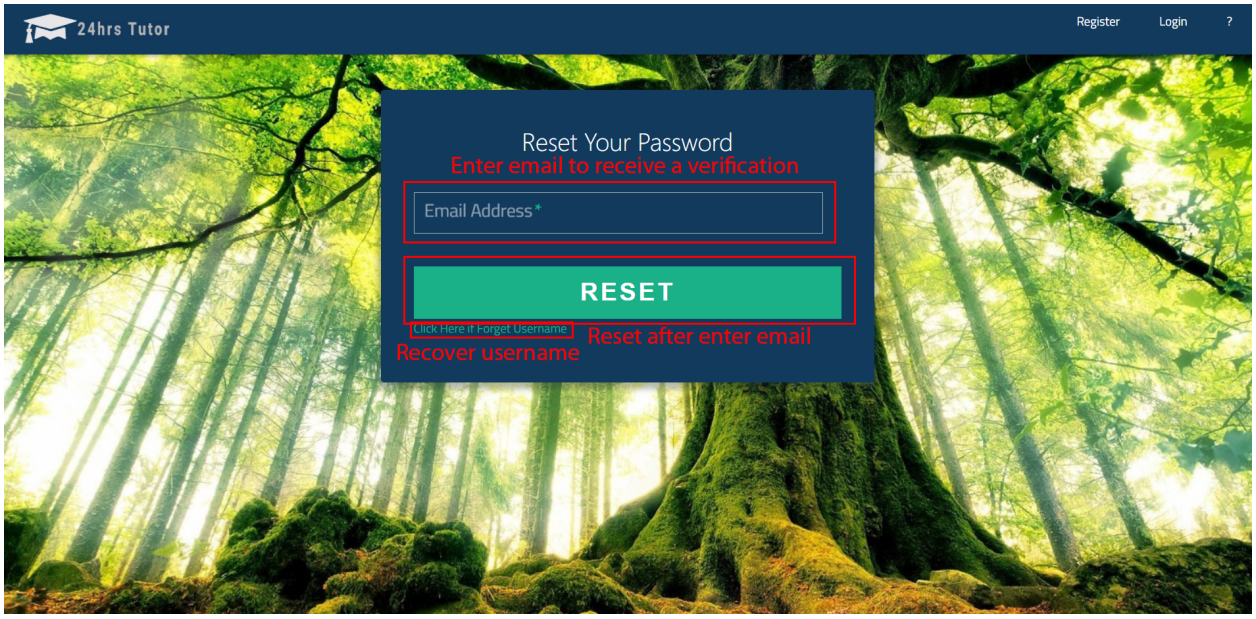
iii. Reset password and username



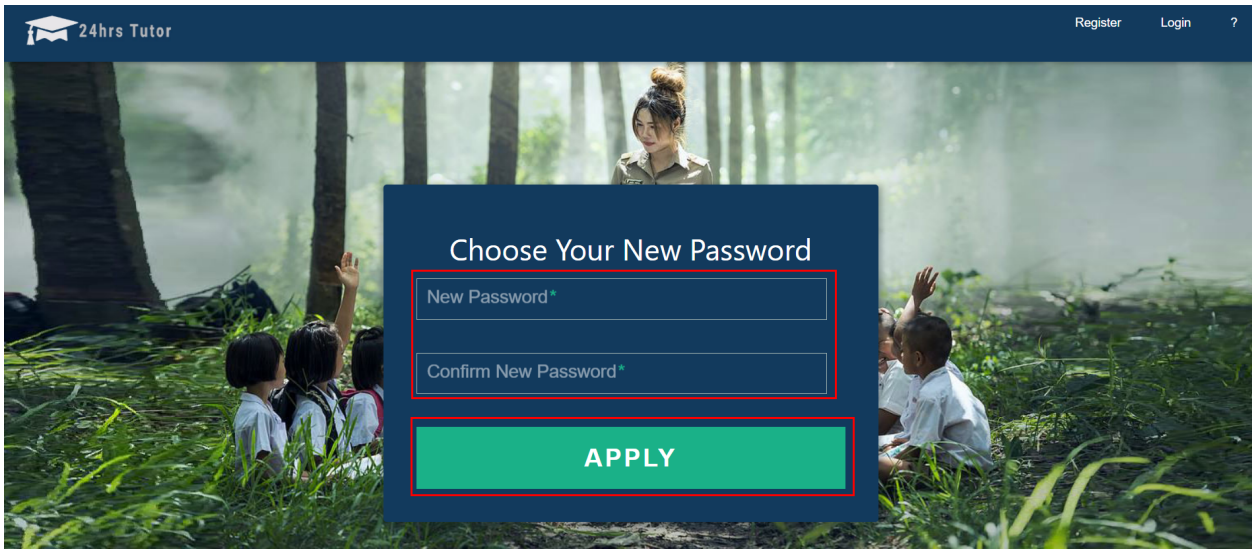
Users will have to first try to login to their account by clicking 1. login button or the ? button beside login and will be redirected to reset page.



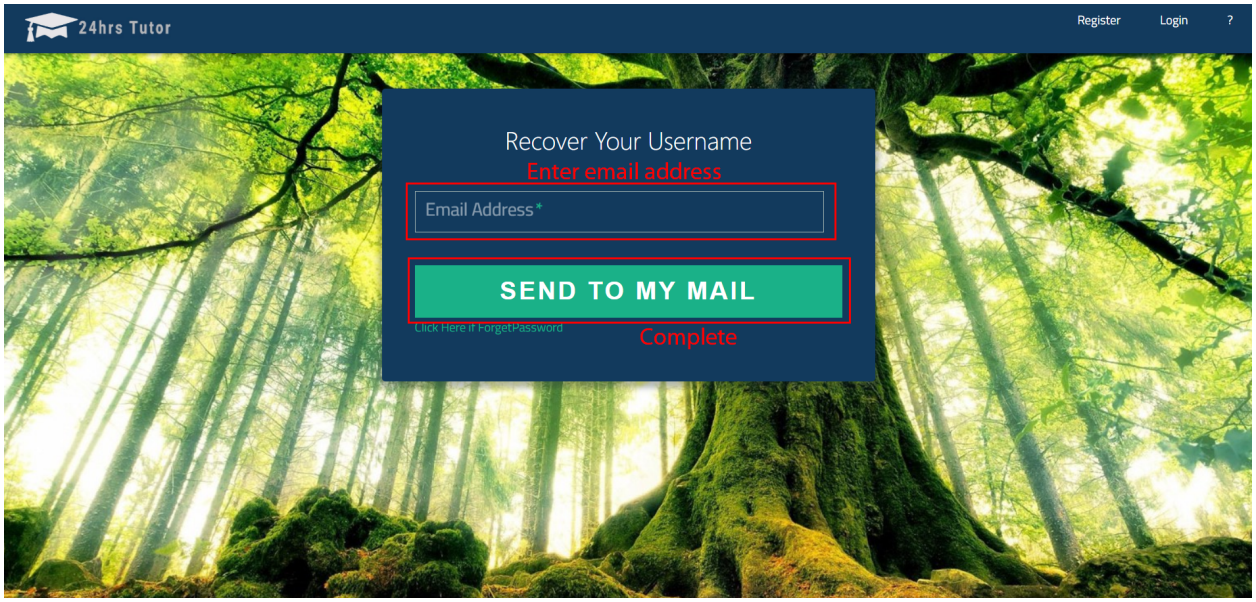
Users will be able to reset their password through the forget button in login page.



Users will be redirected to a page where they will be required to enter their email address to receive a verification email to reset their password. After entering the email address, users will just have to click on the reset button to continue. Besides that, users can also recover their forgotten username.



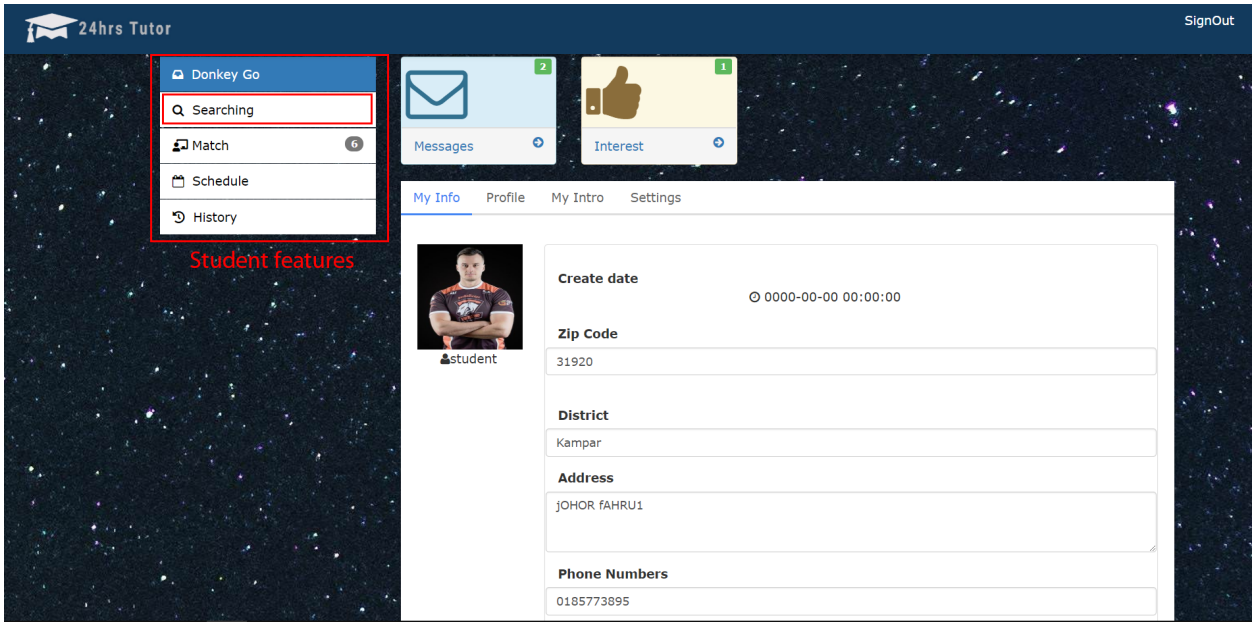
After clicking on the link provided in the email, users will now be able to enter a new desired password for their account. Users will have to enter twice to confirm their new password is correct and proceed to apply.



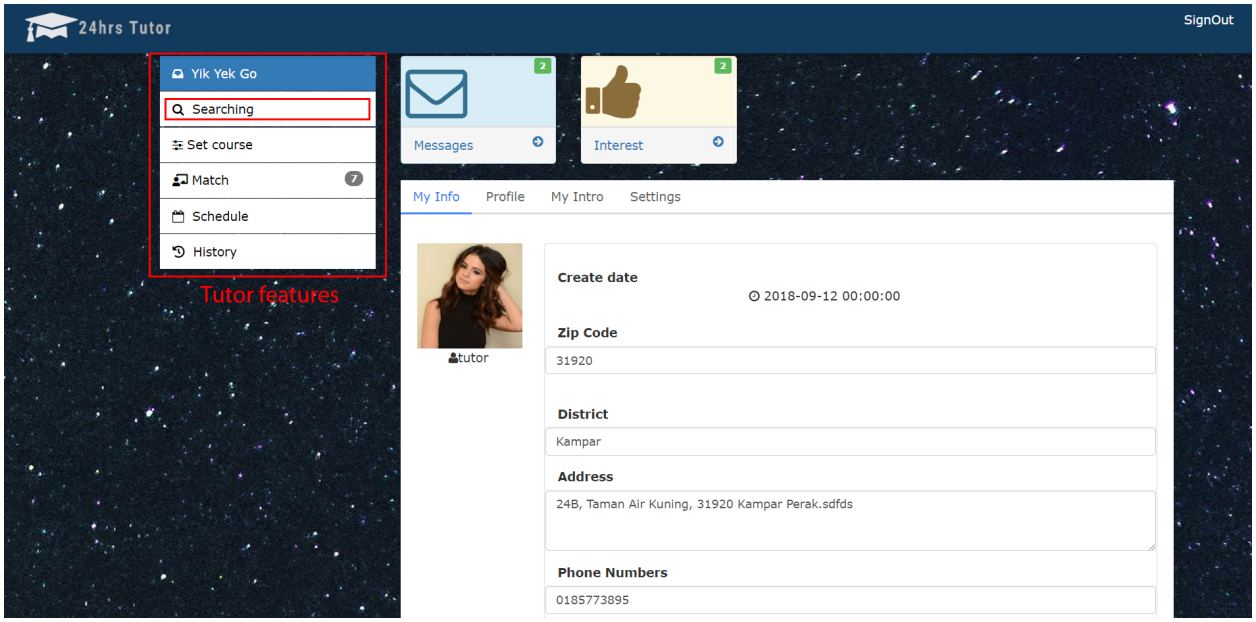
By clicking on the forget username option, users will be redirected to a page where users will also have to enter their email to receive a verification message. After entering your email, complete the step by clicking the send to my mail button.

iv. Searching

Users will first have to login to their account by following the login steps.

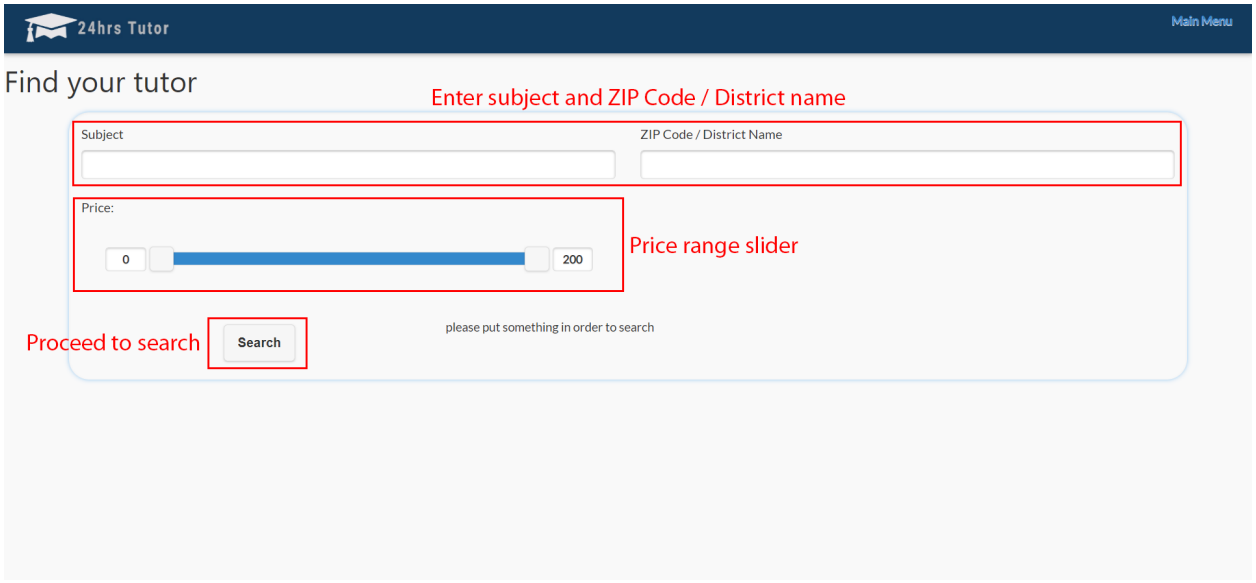


Student profile page

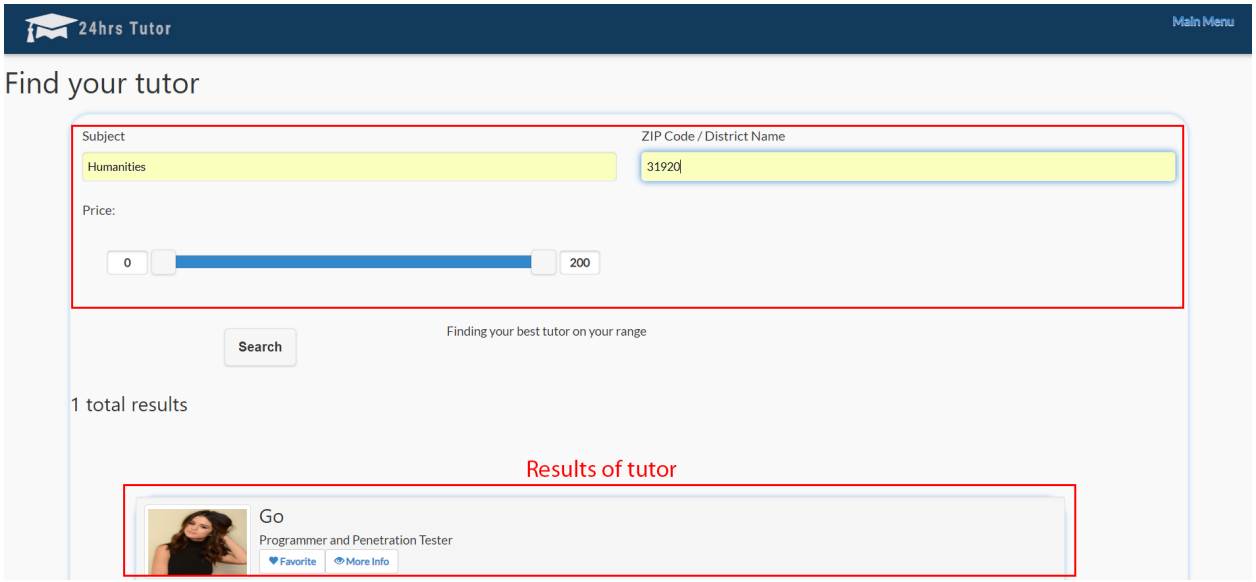


Tutor profile page

After users have been redirected to their profile page, users will be able to find the searching feature in the features box.



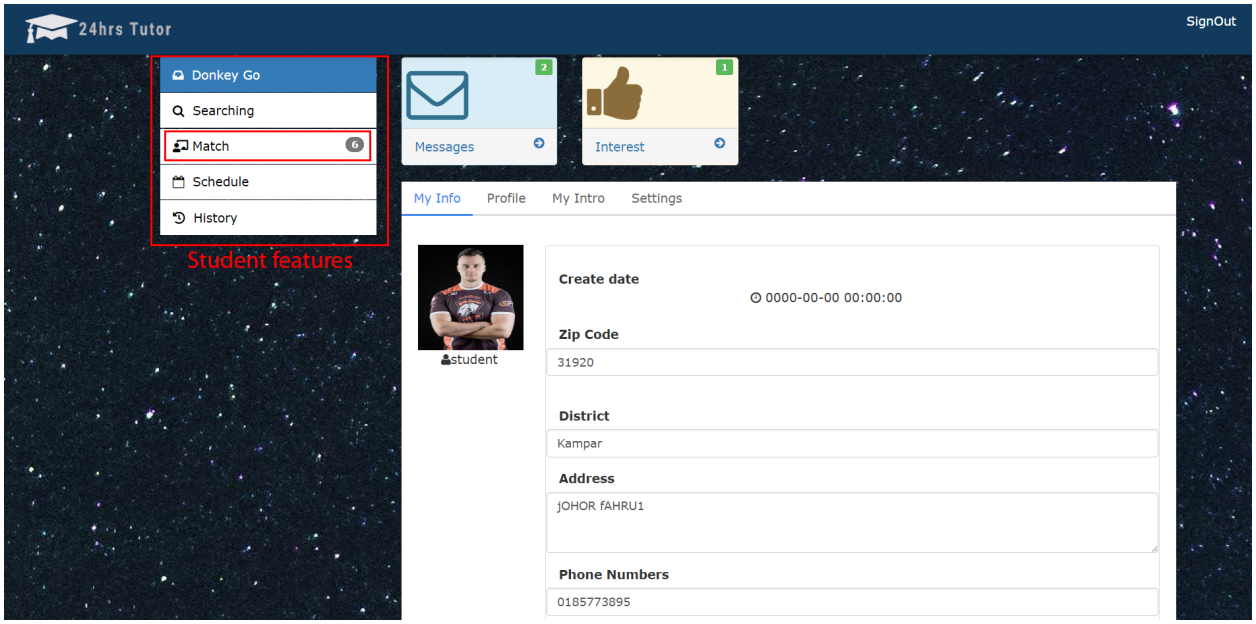
Users will then be redirected to the searching page where users will have to choose the desired subject and enter a valid ZIP Code / District name and move the price range slider to change the desired price. After finishing, click on search to proceed.



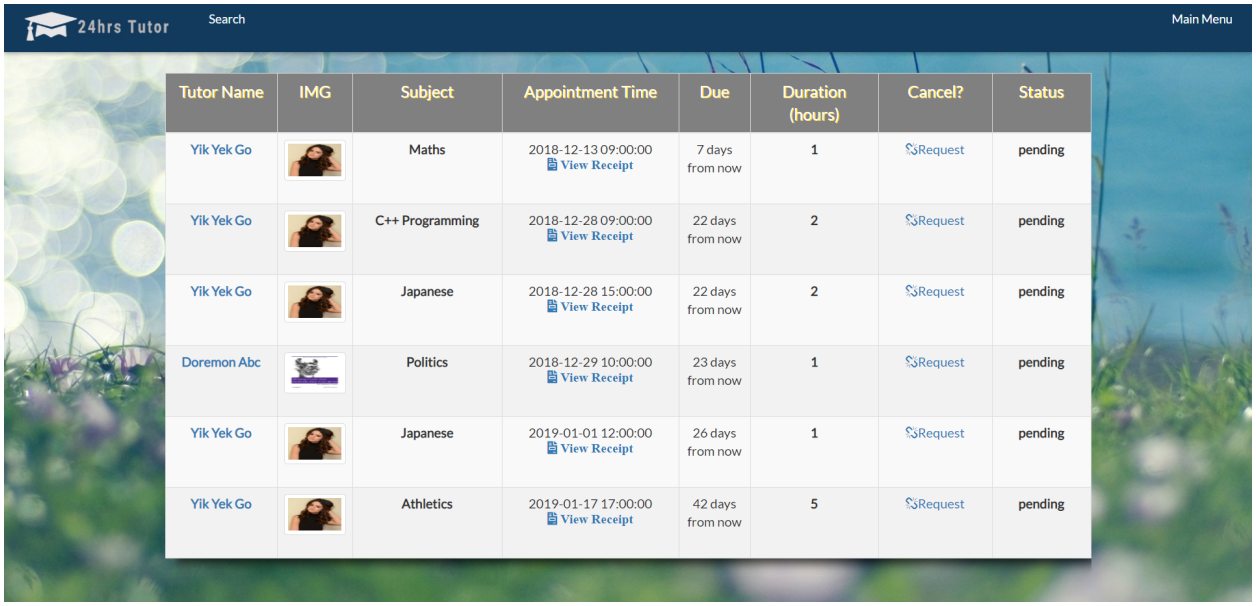
The results of tutor and their information will be shown below.

v. Matching

Users will first have to login to their account by following the login steps.



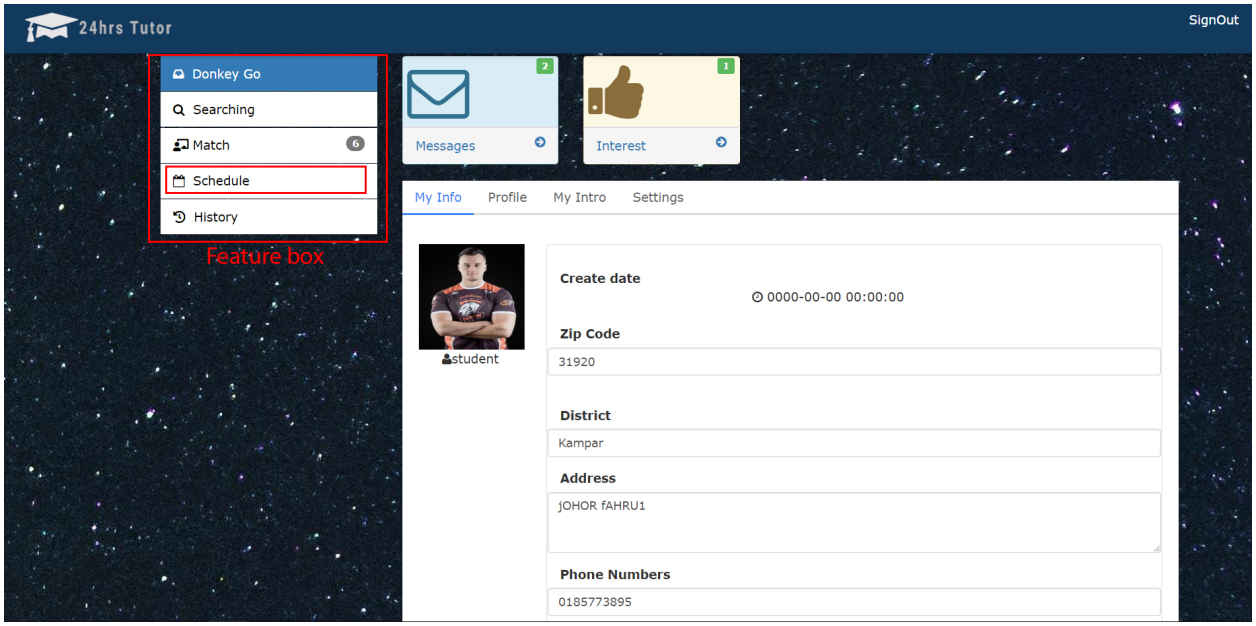
Users will be able to find the matching features in the section shown above.



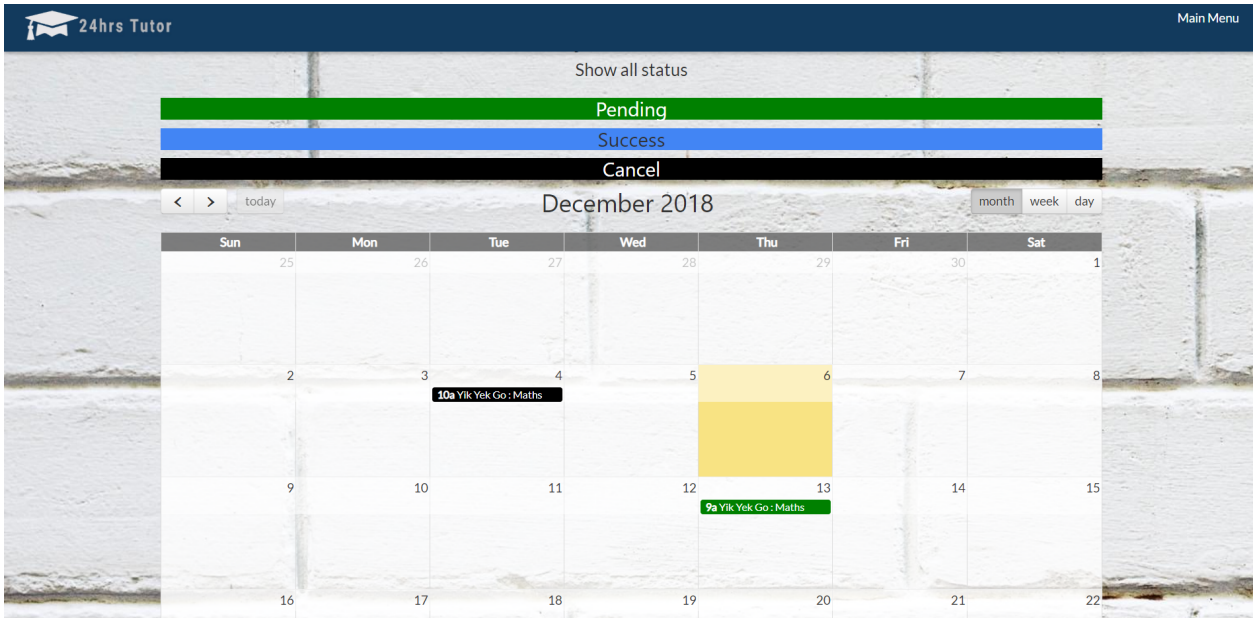
Users will be redirected to matching page and show the results of matched tutors.

vi. Schedule (Student)

Users will first have to login into their accounts.



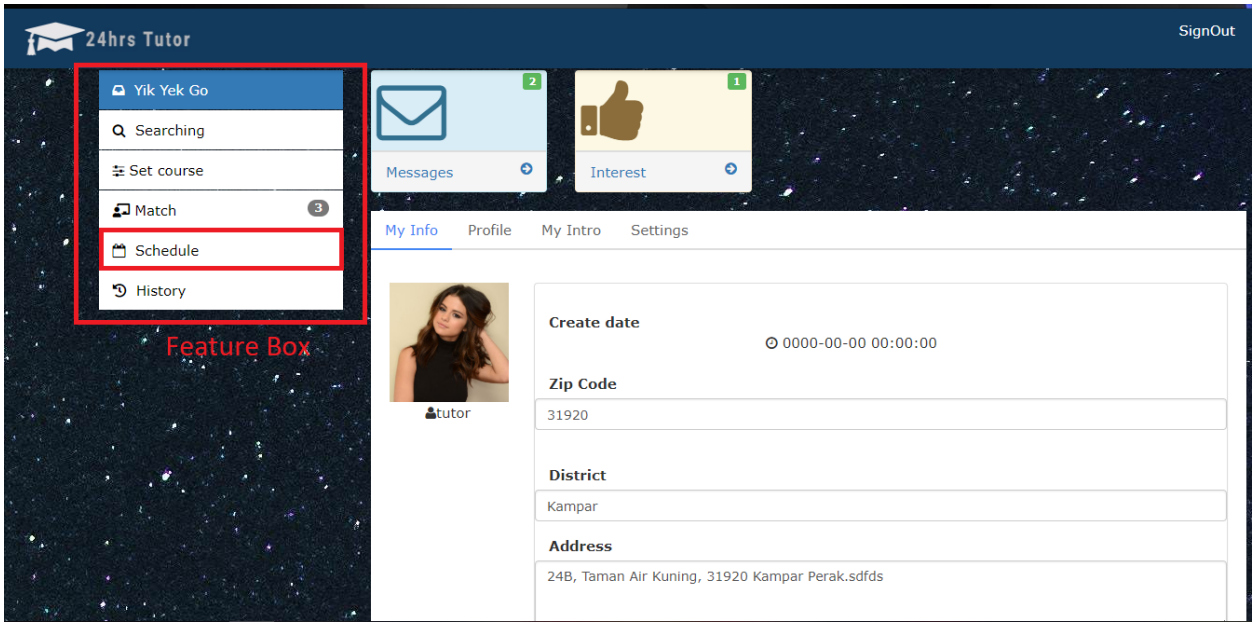
Users will find the schedule feature in the features box.



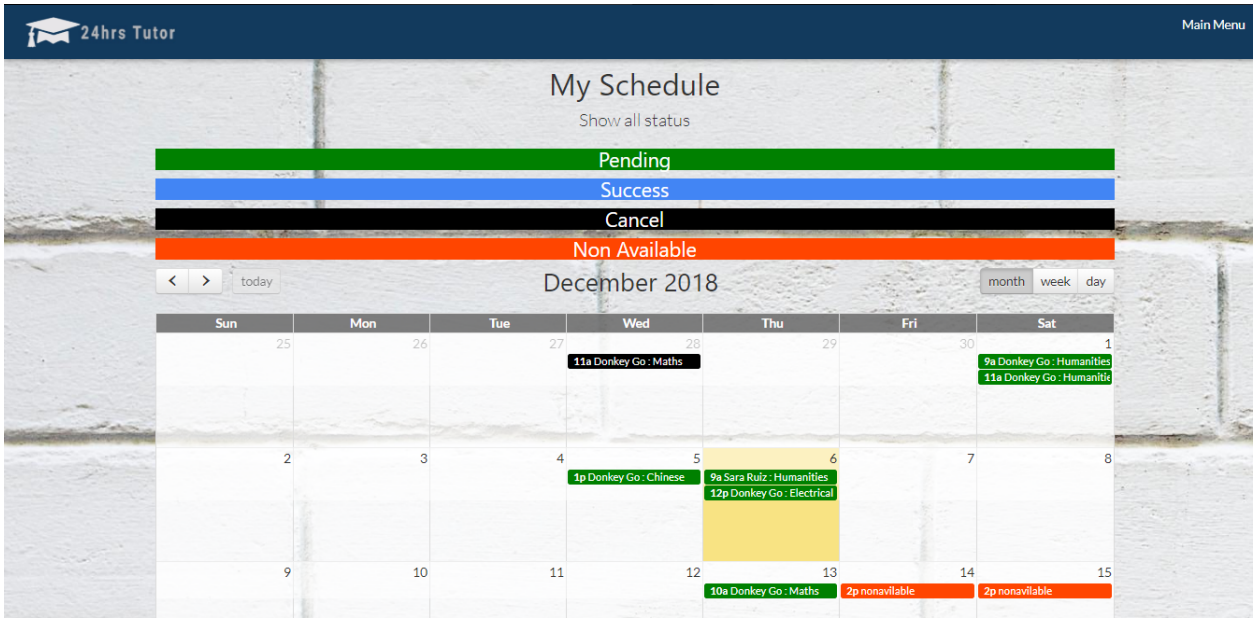
Users will be able to view their schedule.

Vii. Schdule (Tutors)

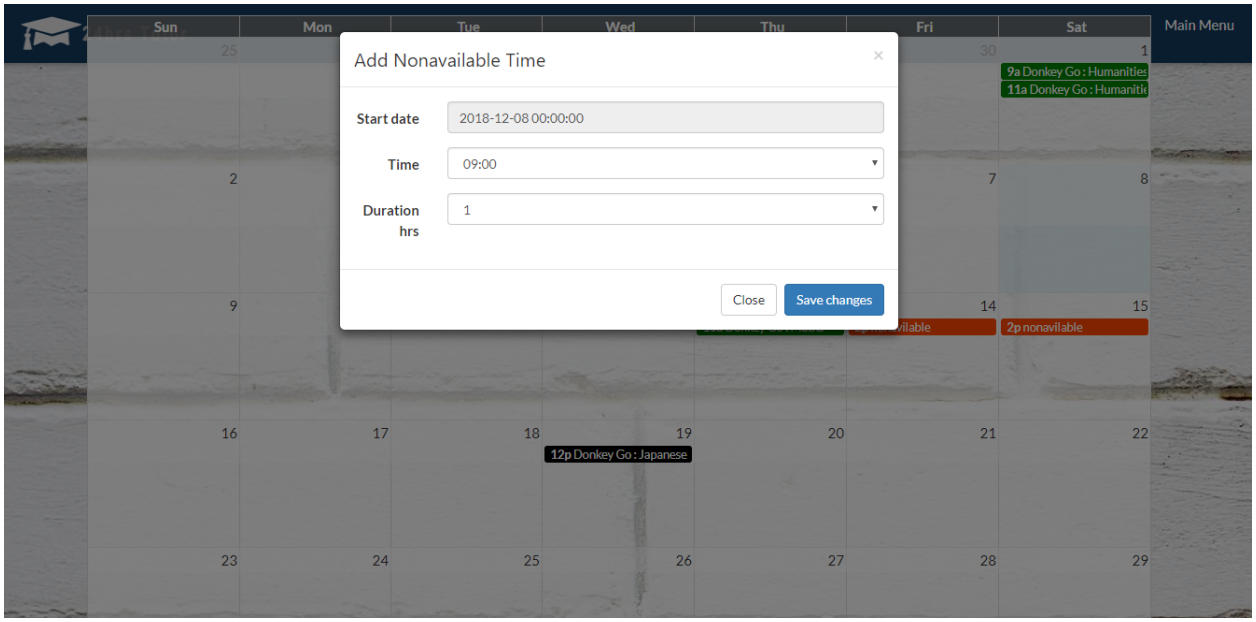
Users will first have to login into their accounts.



Users will find the schedule feature in the features box.

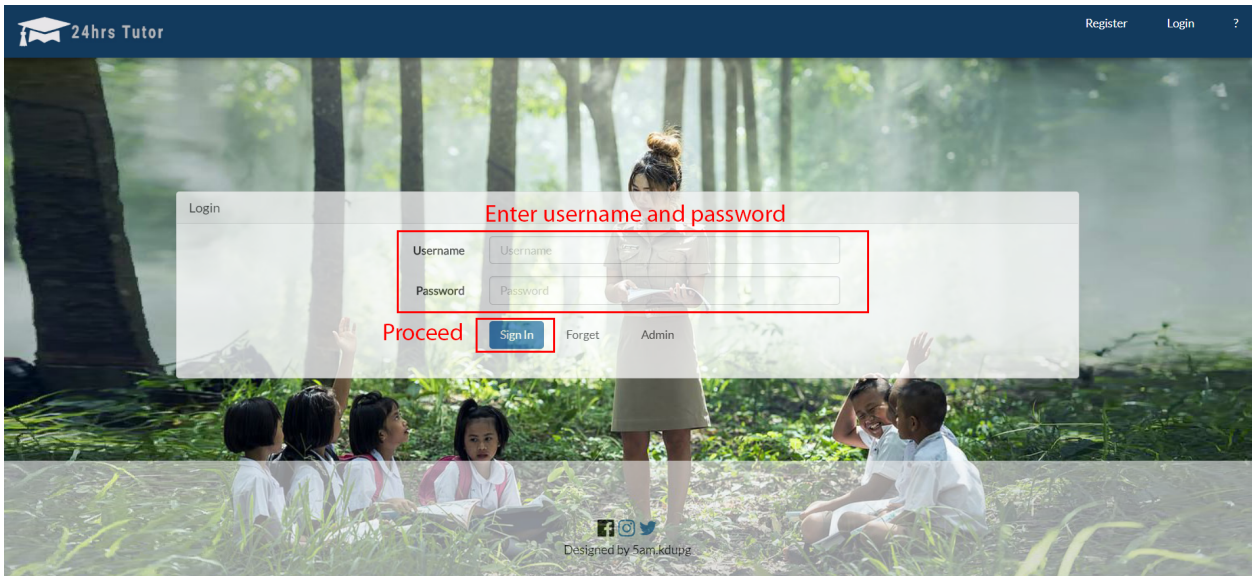


Users are able to view and edit their timetable.

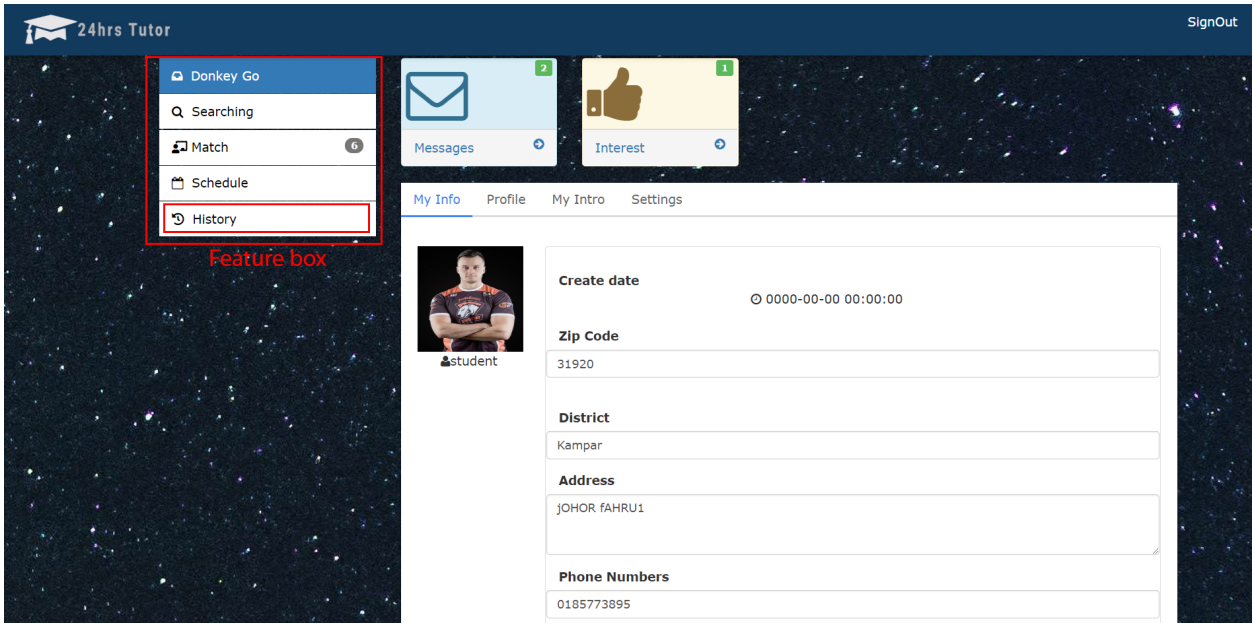


Once the user has clicked onto one of the empty dates then it will pop up a settings which allows the user to set their non available time in the system. Afterwards the user has set their non available time in the system press the save changes button, then the empty date on the calendar becomes red in colour.

viii. History



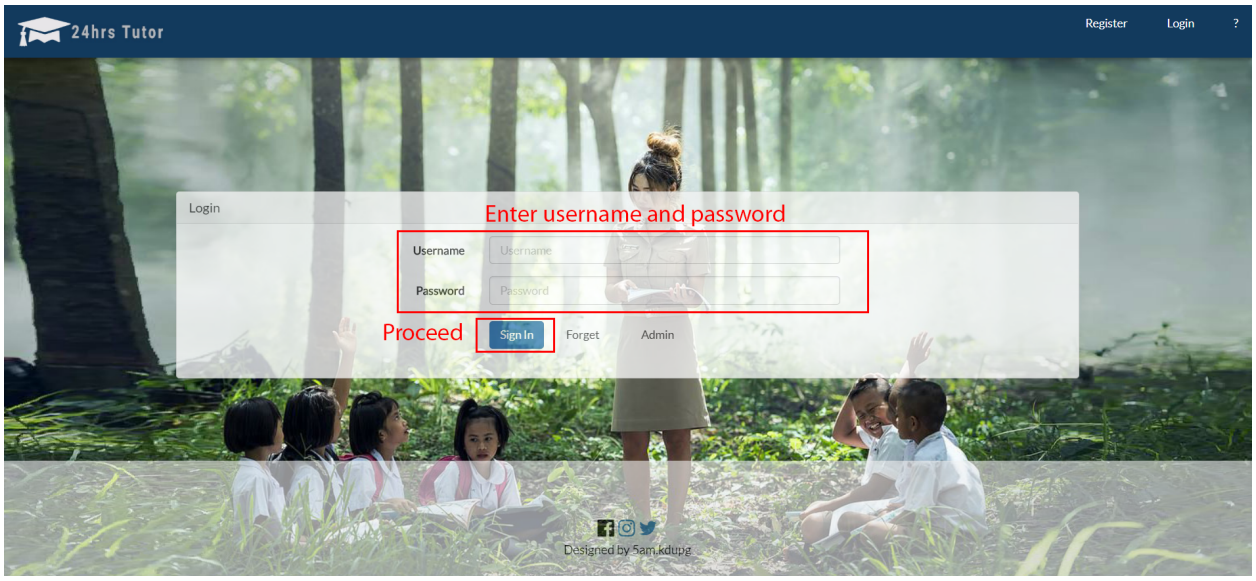
Users will first have to login into their accounts



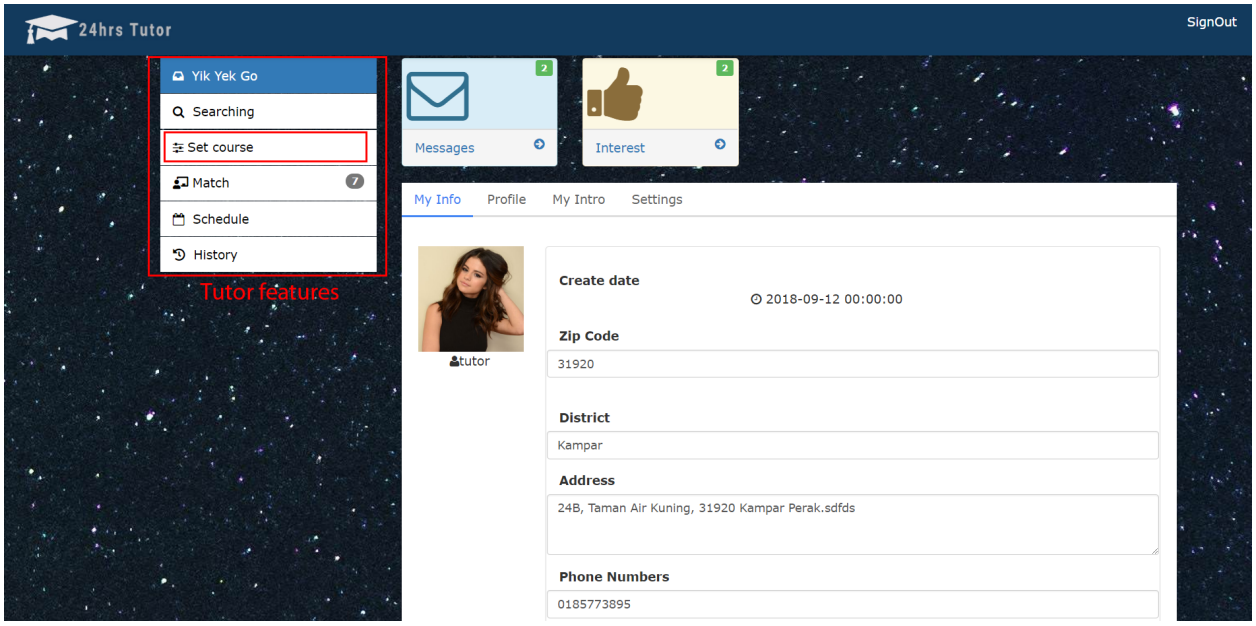
Users will then find the history feature in the features box.

In the history page, users will be able to view the details of lessons. The date and the status of the lesson, whether it was cancelled or it was successful. Users can also view more info about the lesson by clicking the info button.

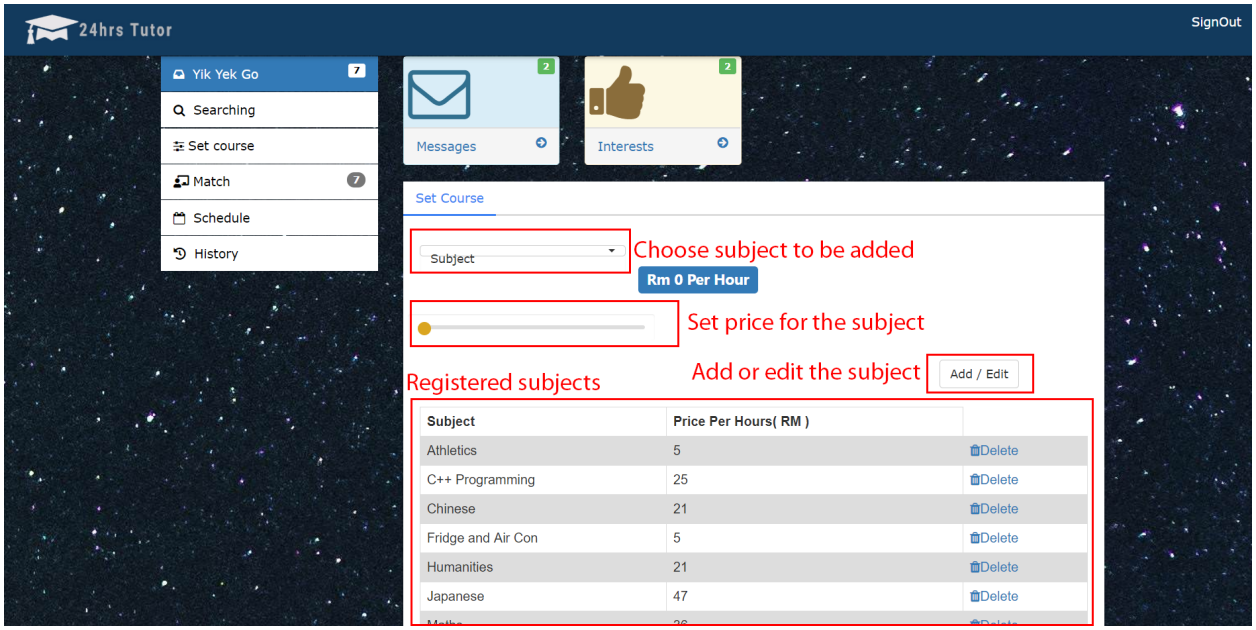
ix. Set course (Tutor)



First login to a tutor account.



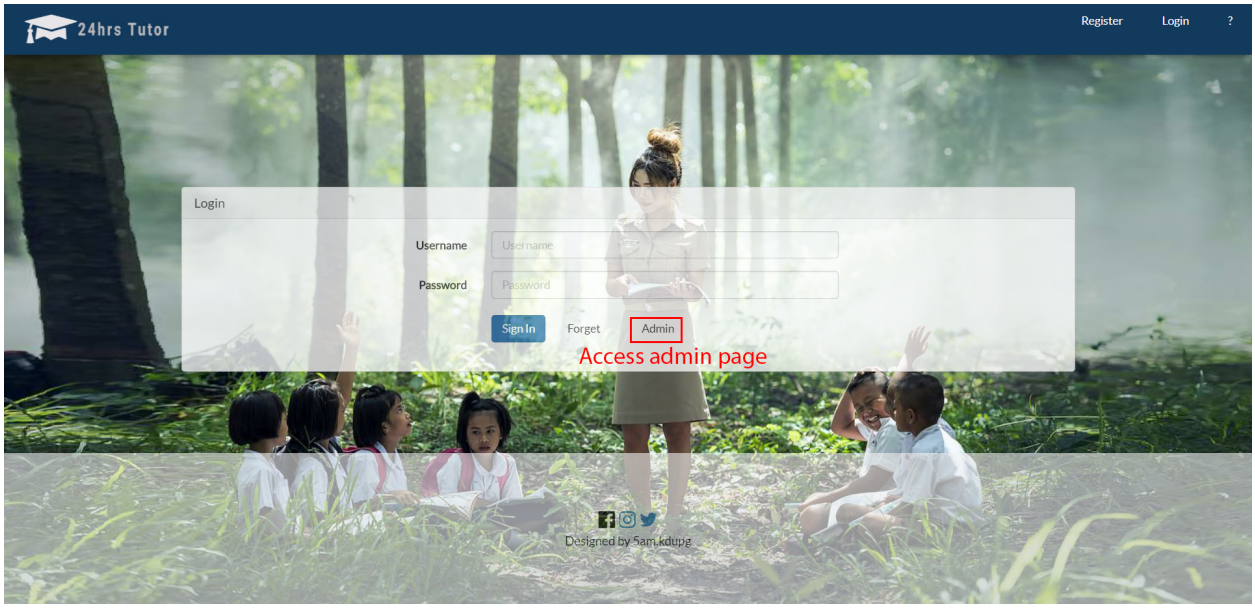
Users will be able to set courses if they are logged in to a tutor account.



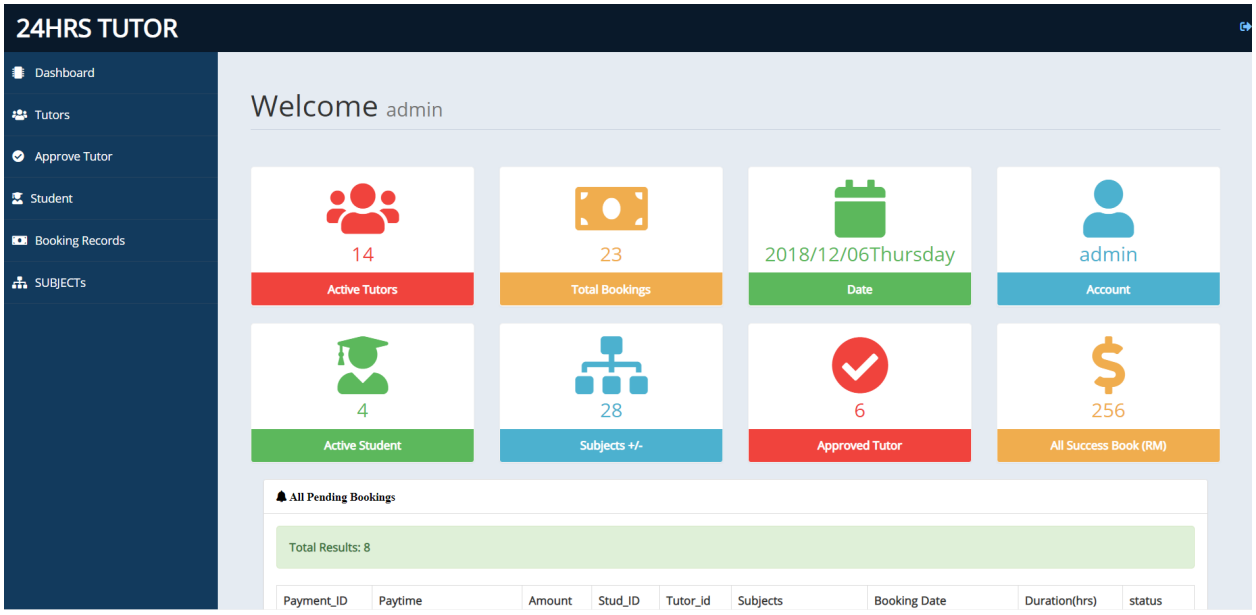
The set course feature allows tutors to choose the subjects they wanted to add and set a price for the subject. If the subject chosen is already a registered subject, then it will only change the price of the subject. Users will also be able to view the subjects the registered below.

x. Admin Login

Users can gain access to the admin page through through the main menu by clicking on the Login button at the top right corner of the page.

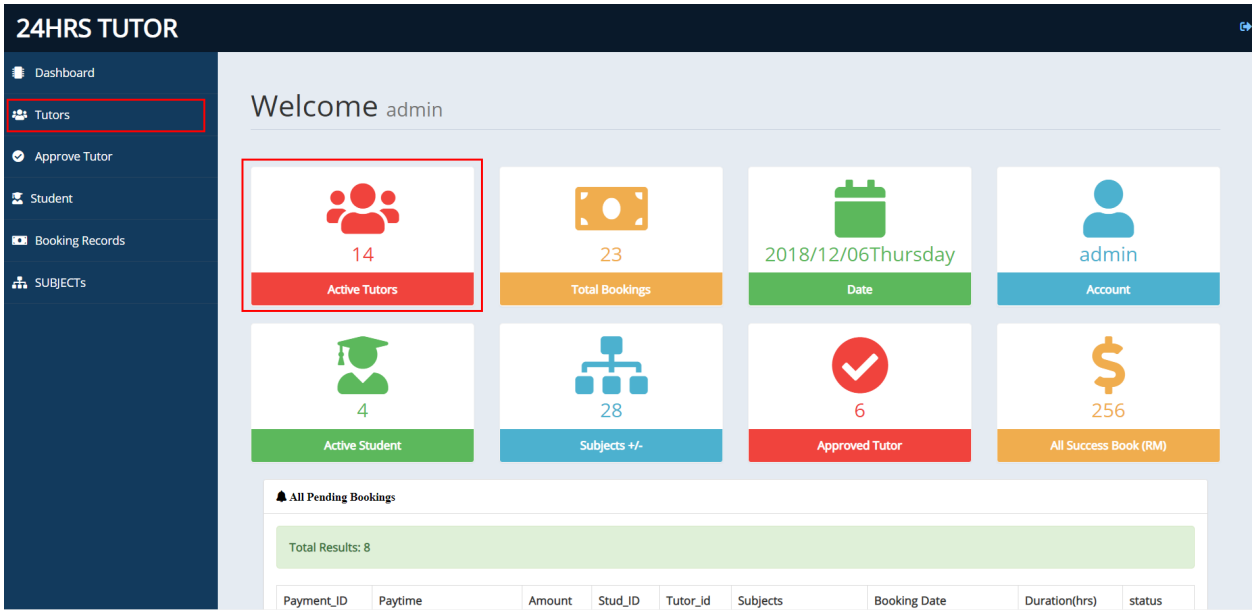


The admin page is accessible after clicking on the admin button.

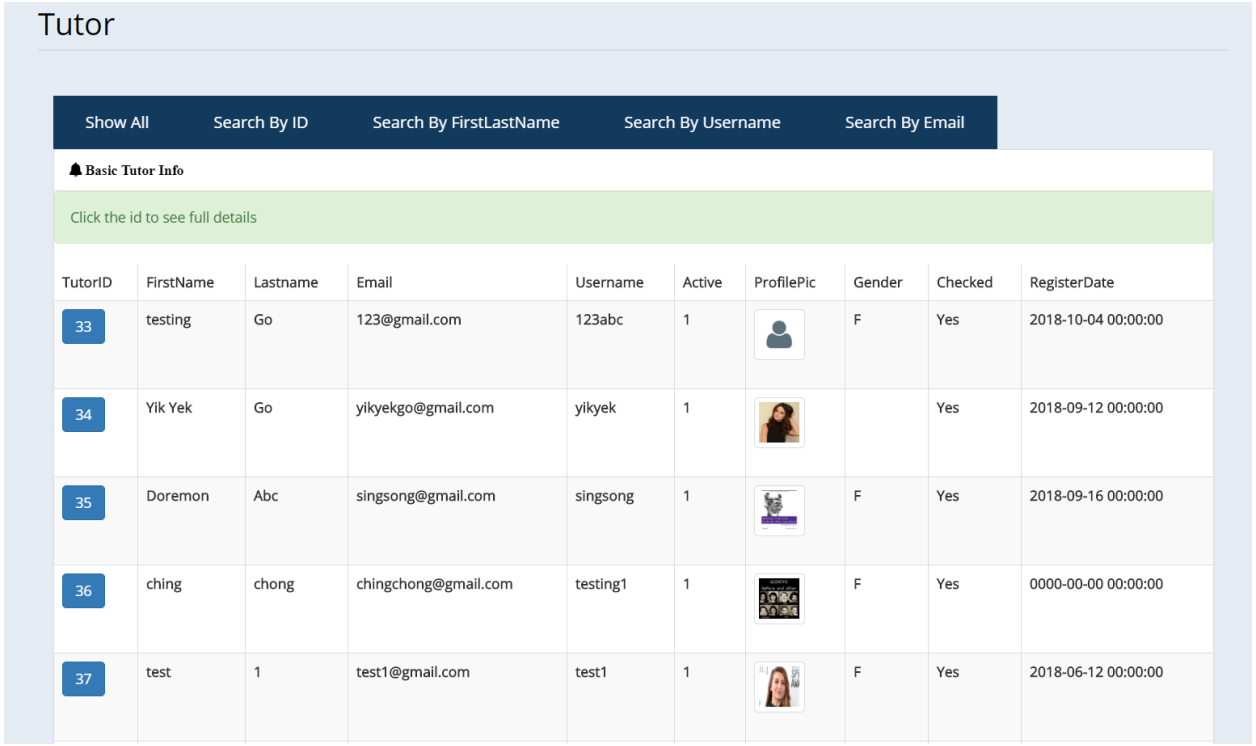


Users will then be redirected to the admin page of the system and view the dashboard once accessed.

xi. View active tutors

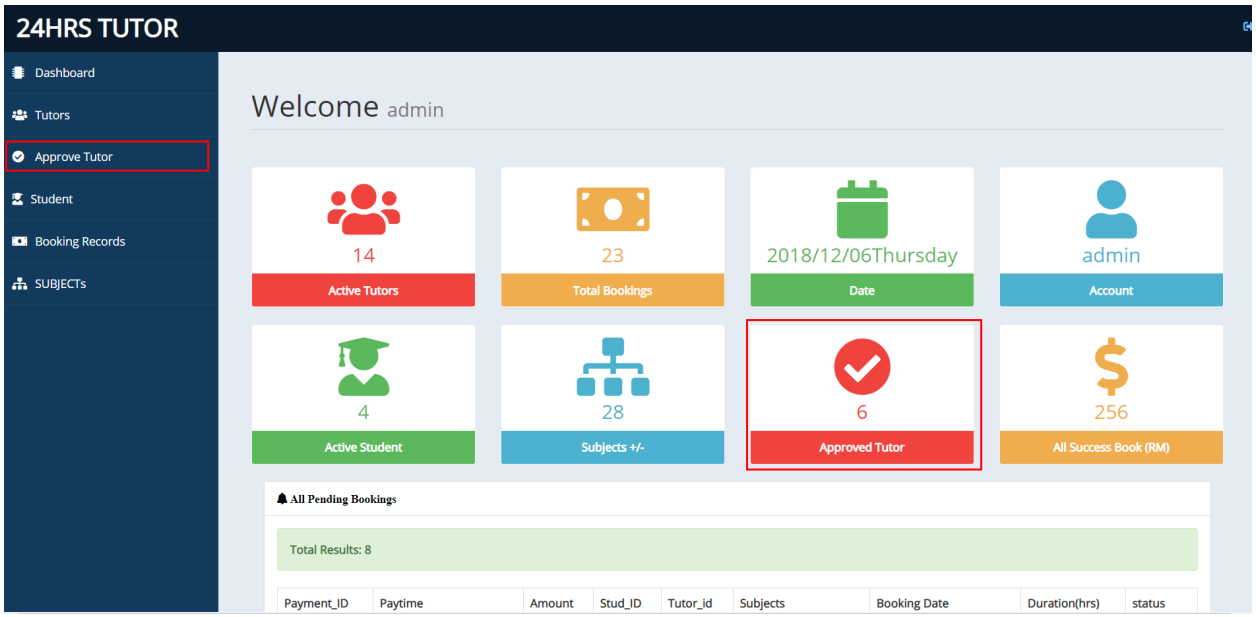


Admin will be able to view active tutors in the system

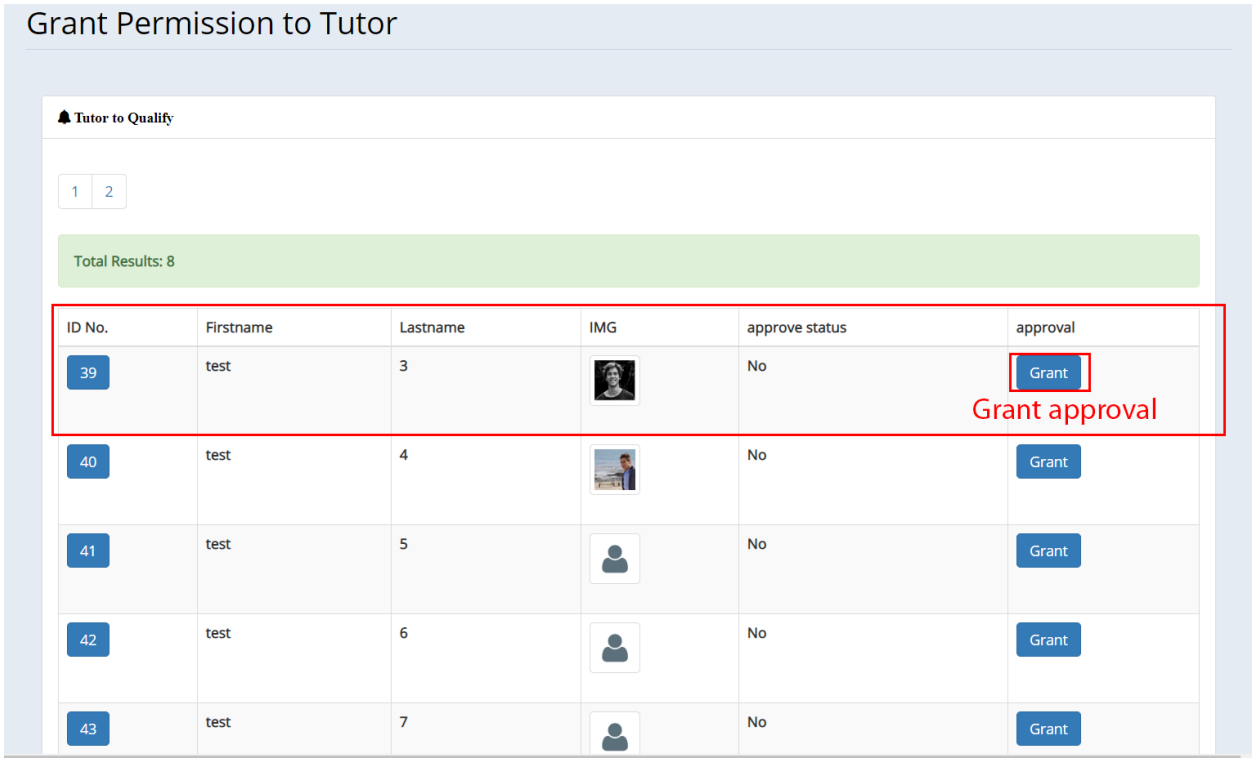


The results will be viewable after clicking.

xii. Approving tutors

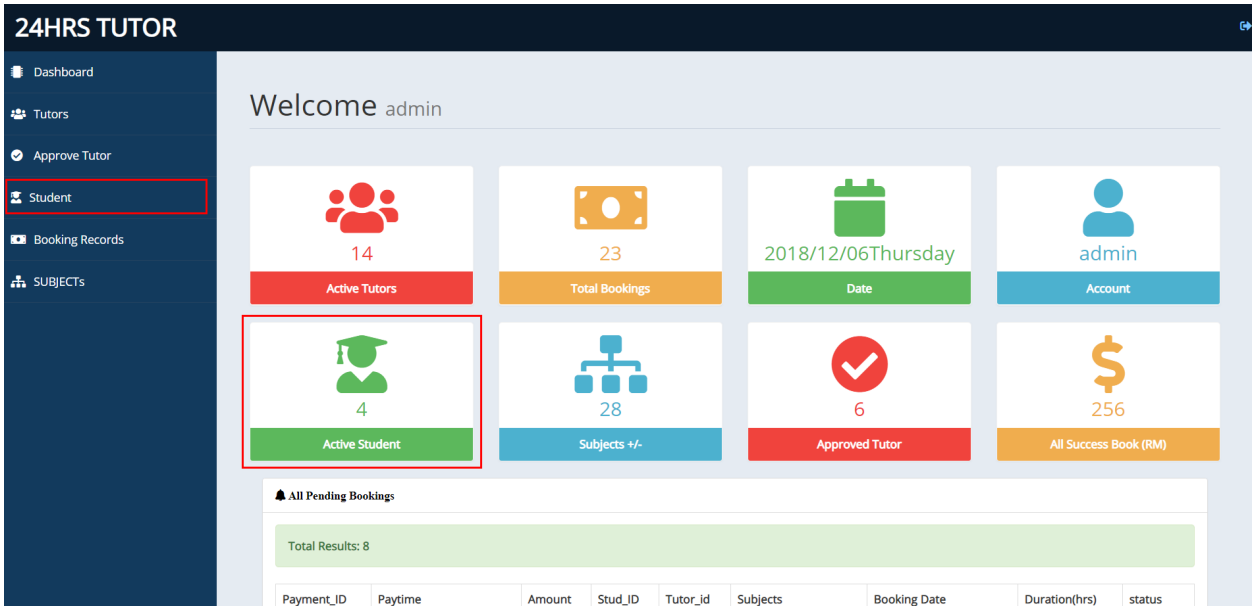


An admin has the power to grant permission for a tutor to teach. It can be done in the admin page.

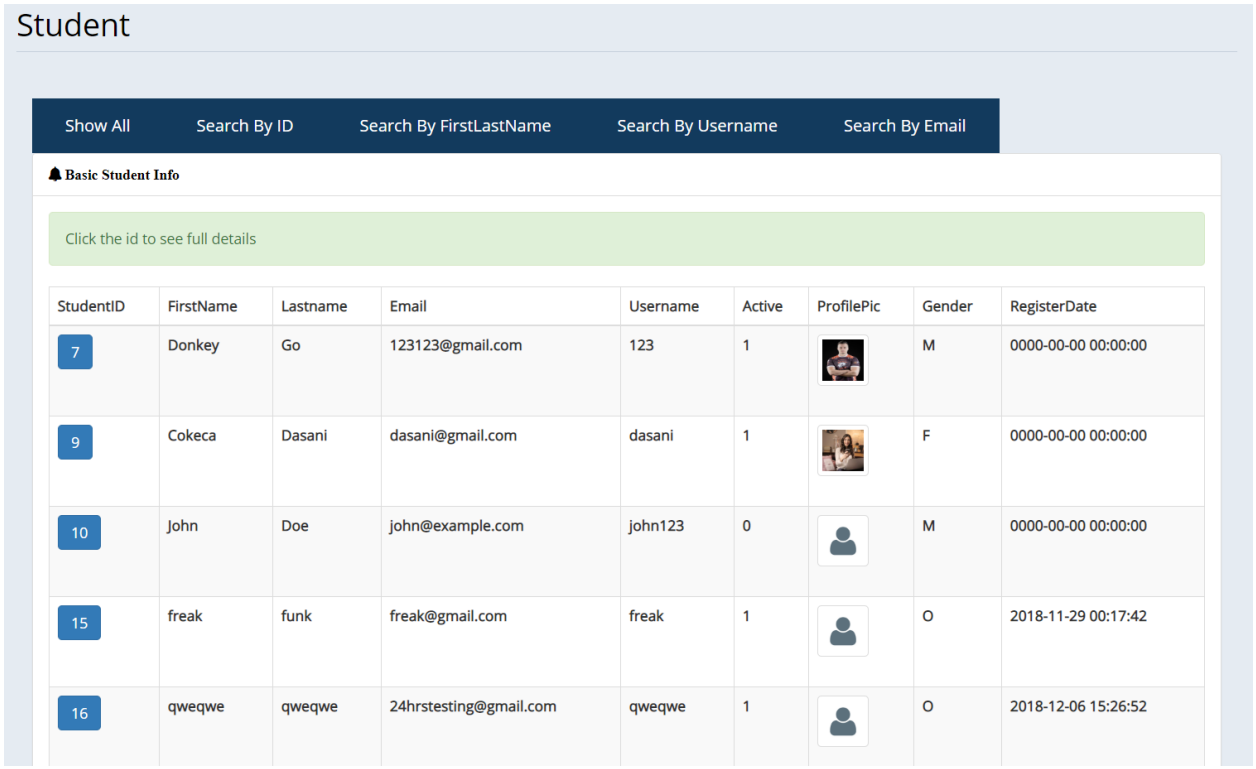


Admin will be able to view the details of the tutor as well as granting the permission for the tutor to teach.

xiii. View active students

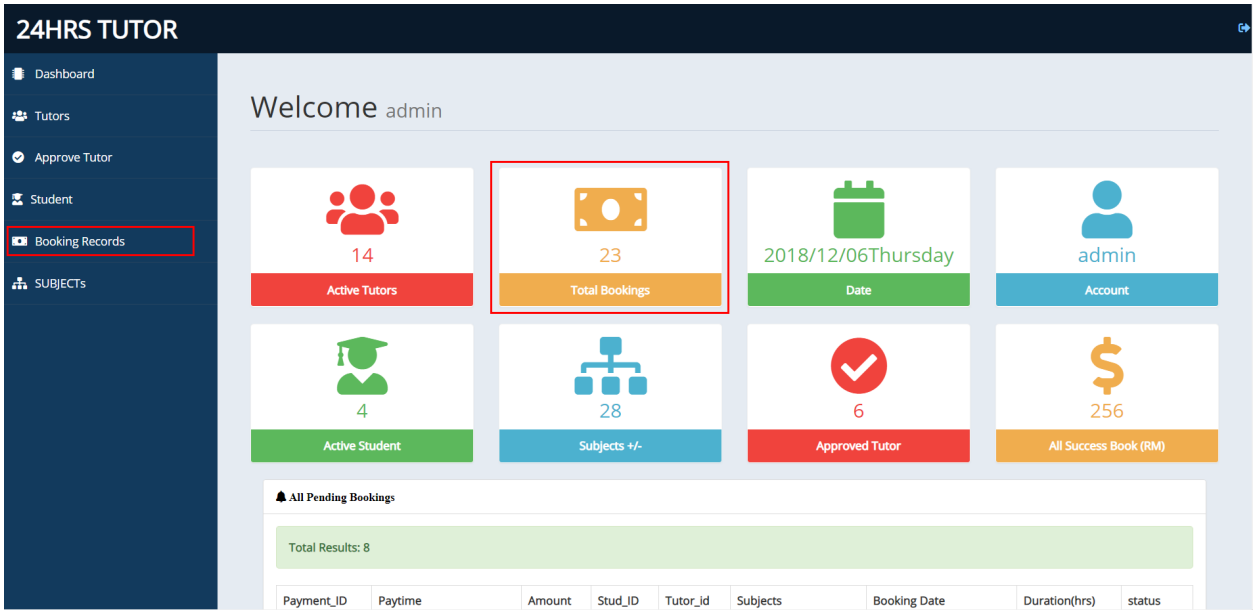


Admin will be able to view active students in the system.

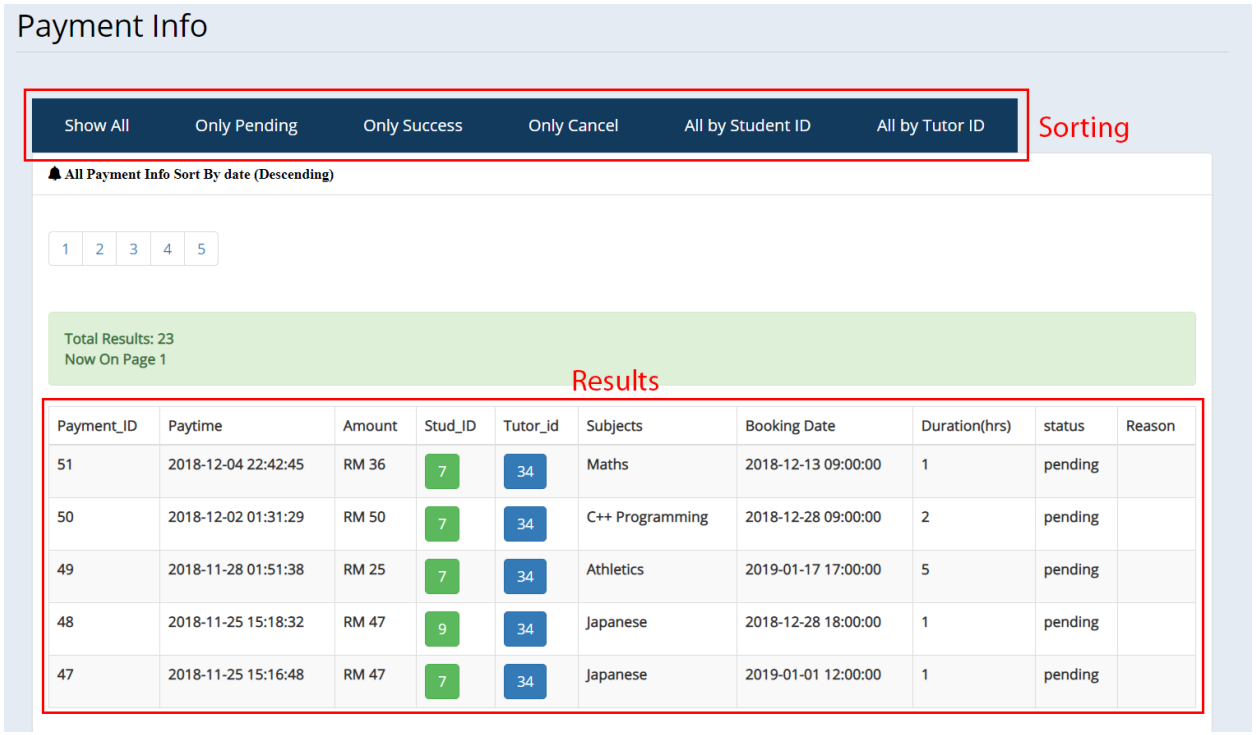


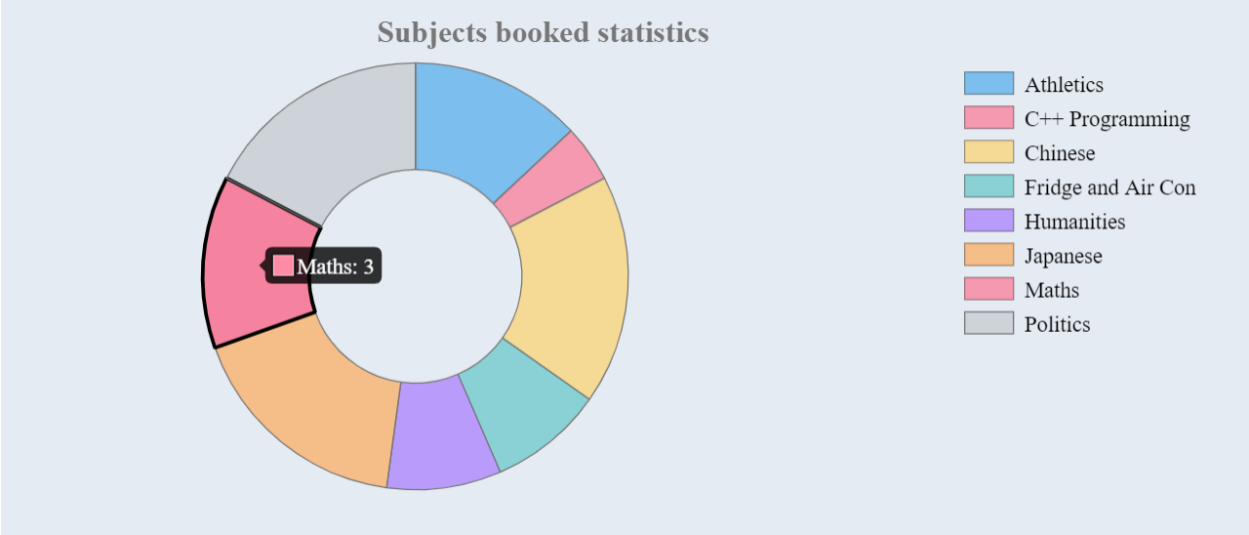
The results will be viewable after clicking.

xiv. View booking records



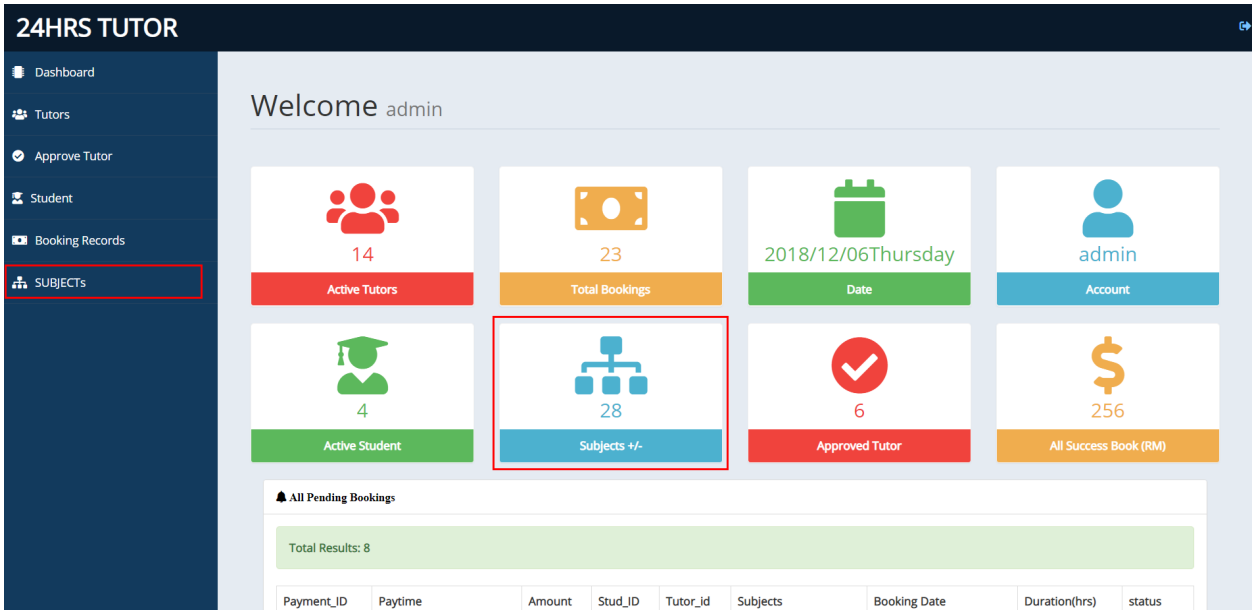
Admins will be able to view all the booking records by students.



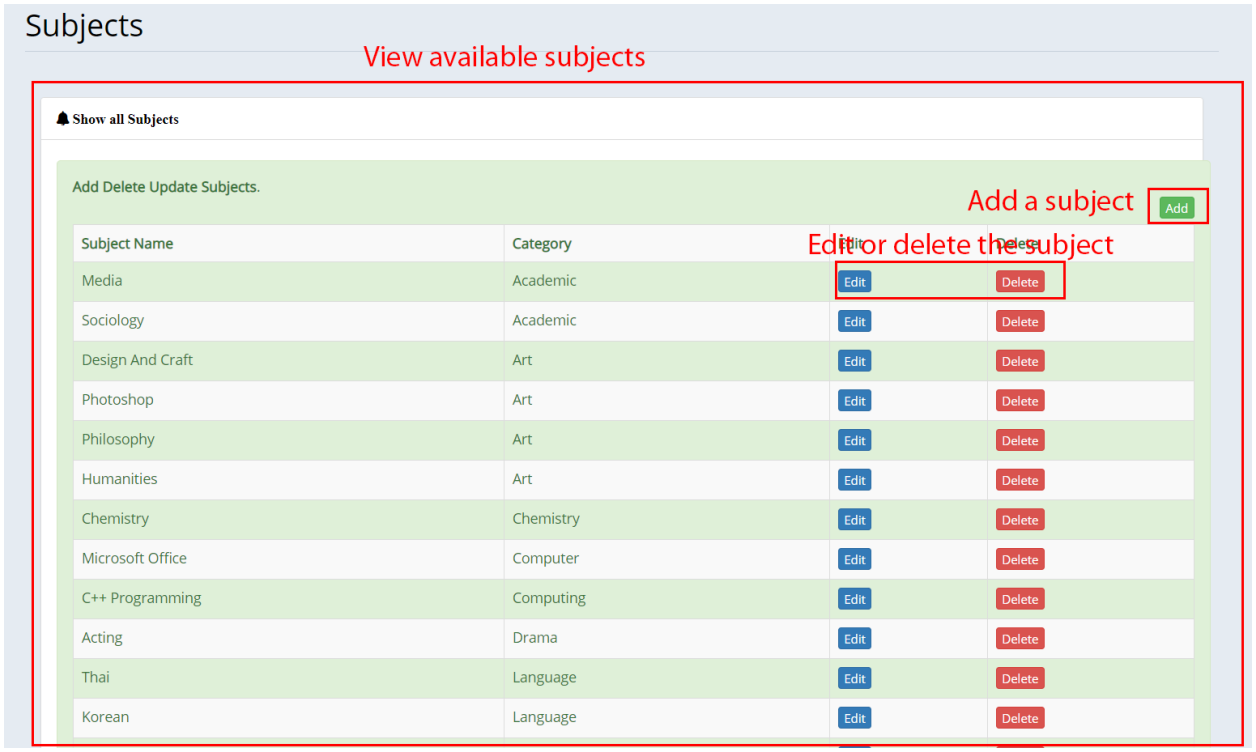


Admins will be able to view the results of specific student bookings by student id. Admins will also be able to view statistics of subjects booked.

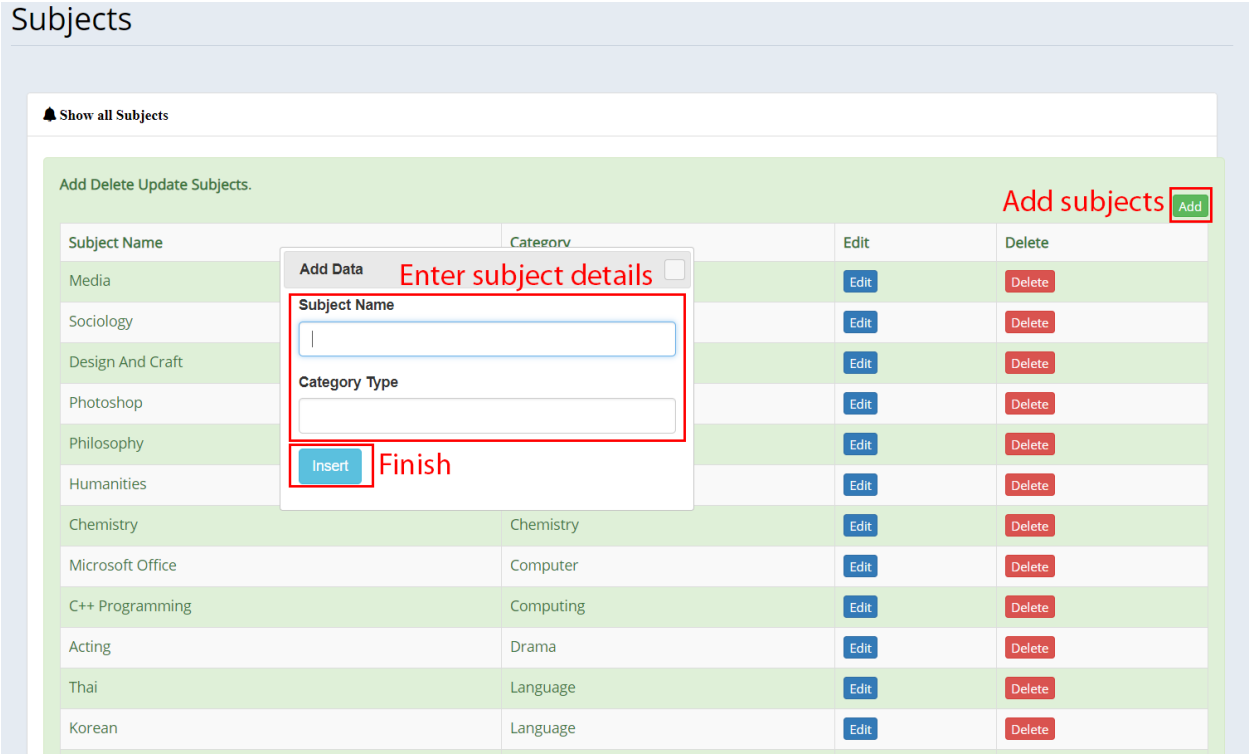
xv. Edit subjects



Admins will be able to add or edit subjects for tutors to choose.



Admins will be able to view available subjects. Edit, delete a certain subject can also be done. Adding a new subjects for tutors can also be done by the admins.



Admins have to enter the subject details in order to insert a subject.