REQUIREMENTS ANALYSIS DOCUMENT

October 16, 2016

Software Design COMS3009

FindMeTutor Android Application

Proposed idea by:
Shaneel James-718840
Jadon Manilal-815050
Jared Naidoo - 719238
Krupa Prag - 782681
Nivek Ranjith - 802119

Contents

1 Executive Summary

The problem that we face is that students are in need of extra lessons and tutorials outside of standard lessons provided by the university. We propose a system that can connect students in search of tutors.
The results of the requirements analysis are documented below. This document completely describes the system in terms of the requirements. This document serves as a contextual basis between the client and the developer.

2 INTRODUCTION

2.0.1 Purpose of the system

The purpose of the FindMeTutor application is to provide a convenient means for tutors and students who are looking for tutors to be able to connect within a particular tertiary institute.

2.0.2 Scope of the system

Our team, working on the FindMeTutor application, envisions a successful product to be an Android Application which will be at a students disposal in order to improve their grades and achieve their academic dreams. With limited resources, a stringent budget and capped time, we aim to execute this task in an economical fashion.
This goal will be achieved by making use of agile methodology. We will be able to set short term targets to achieve deliverables within sprints, with a long term goal being to present the FindMeTutor Android Application.

2.0.3 Objective and success criteria of the project

The FindMeTutor Android application will be seen as successful if it facilitates a platform on which tutors and students can meet. We have great hope that the result of this would mean better results obtained by the students, and a manner in which tutors can generate some income and gain some job experience.

2.0.4 Definitions, acronyms, and abbreviations

1. App - abbreviation for application.
2. Application - is a piece of software
3. Android - is a mobile operating system developed by Google.
4. OS - abbreviation for operating system.
5. Operating system - is a collection of software that communicates with hardware and allows other programs to run on it.
6. Java - is a high-level programming language
7. UI - abbreviation for User interface
8. User interface/GUI - is the means in which a person controls a software application or hardware device.

9. ID - abbreviation for identity
11.User ID - the idenity that uniquely identifies someone on a computer system.
12. Sign in - when asked to enter username and password information. A sign in/login is a combination of information that authenticates a user's identity.
13. SDK - abbreviation for Software Development Kit
14 Software Development Kit - collection of software used for developing applications for a specific device or operating system. 15 Sessions - A tutorial session created by a student.

2.0.5 References

1. http://techterms.com/definition (2016-08-08)

3 CURRENT SYSTEM

3.0.1 Overview

Currently, there are many students in search of tutors to help them with particular courses with which they require some support, as well as fellow students or tutors who are available to tutor particular courses of study. However, the problem that is faced on hand is that either pool (students and tutors) are struggling to find each other.

4 PROPOSED SYSTEM

4.1 Overview

FindMeTutor app will be a platform through which students and tutors can meet in order to resolve the current situation.

FindMeTutor app will facilitate the following two registration categories:

1.Student looking for tutors they are able to register on the app with merely some personal details (demographic data, email and password).
2. Tutor - those who would like to tutor can register on the app by simply filling in some details with respect to the fields of study they are particularly comfortable to tutor.

4.2 Functional requirements - "Shall lists"

Describes the high-level functionality of the system

Overview of Requirements

Item	**Requirements to implement**
0	Proposal
1	Student registration and login
2	Tutor registration and login
3	Administrator registration
4	Administrator capabilities
5	Students able to request a tutor
6	Tutor able to accept/reject tutor requests
7	Student able to handle 'Upcoming events'
8	Tutor able to handle 'Upcoming events'
9	Student able to rate a tutor
10	Student able to check-in/check-out of tutor sessions
11	Tutors able to check-in/check-out of tutor sessions
12	Student funding and payment

13	Tutor funding and payment
14	Students able to add/ remove subjects enrolled in
15	Tutors able to add/ remove subjects able to tutor
16	Login of student and tutors
17	Students and tutors can log their tutorial sessions
18	Students and tutors can mark off completed sessions

Requirement	**Functional Requirement**	**Use Case**
RQ1.1	The system shall allow a student to register	UC-CS
RQ1.2	The system shall allow a student to update their account eg update password	UC-US
RQ1.3	The system shall allow a student to view their account details	UC-VS
RQ1.4	The system shall allow a student to mark their account as deleted	UC-DS
RQ2.1	The system shall allow a tutor to register	UC-CT
RQ2.2	The system shall allow a tutor to update their account eg password update	UC-UT
RQ2.3	The system shall allow a tutor to view their account details	UC-VT
RQ2.4	The system shall allow a tutor to mark their account as deleted	UC-DT
RQ3.1	The system shall allow a student to request a tutor	UC-RT
RQ3.2	The system shall allow a student to choose a tutor from a list	UC-RT
RQ4.1	The system shall allow a tutor to accept a request	UC-RT
RQ4.2	The system shall allow a tutor to reject a request	UC-RT
RQ5.1	The system shall allow a student to rate a tutor	UC-R
RQ6.1	The system shall allow a student to check-in	UC-CI
RQ6.2	The system shall allow a student to check-out	UC-CO
RQ7.1	The system shall allow a tutor to check-in	UC-CI
RQ7.2	The system shall allow a tutor to check-out	UC-CO
RQ8.1	The system shall allow a student to add funds	UC-CF
RQ8.2	The system shall allow a student to view funds	UC-VF
RQ8.3	The system shall allow a student to update funds	UC-UF
RQ9.1	The system shall allow a tutor to add funds	UC-CF

RQ9.2	The system shall allow a tutor to view funds	UC-VF
RQ9.3	The system shall allow a tutor to update funds	UC-UF
RQ10.1	The system shall allow a student to add subjects	UC-CSb
RQ11.1	The system shall allow a tutor to add subjects	UC-CSb
RQ12.1	The system shall allow a student to remove subjects	UC-DSb
RQ12.2	The system shall allow a tutor to remove subjects	UC-DSb
RQ13.1	The system shall allow a student to login	UC-L
RQ13.2	The system shall allow a tutor to login	UC-L
RQ14.1	The system shall allow a student to view their upcoming sessions	UC-VSes
RQ14.2	The system shall allow a tutor to view their upcoming sessions	UC-VSes
RQ15.1	The system shall allow a tutor to mark a session as done	UC-DSes
RQ15.2	The system shall allow a student to mark a session as done	UC-DSes
RQ16.1	The system shall allow a tutor to mark a session as cancelled	UC-DSes
RQ16.2	The system shall allow a student to mark a session as cancelled	UC-DSes
RQ17.1	The system shall allow a student to view a subject	UC-RSb
RQ17.2	The system shall allow a tutor to view a subject	UC-RSb
RQ18.1	The system shall allow a students account to automatically be deducted from their account to make a payment to the tutor, once they both have checked out	UC-FP
RQ18.2	The system shall allow a tutor receive funds from a student once they both check-out of a tutorial session	UC-FP

4.3 Functional requirements specification

4.3.1 Stakeholders

- Students
- Tutors
- University (Facculty of Science : School of Computer Science and Applied Mathematics)

4.3.2 Actors and Goals

- Students - a registered user looking for a tutor
- Tutors - a registered user, whom is approachable to tutor particular subjects
- Database - records details of students and tutors. Stores required information with regards to tutorials for student and tutor.

4.4 Non-functional requirements

Describes the user-level requirements that are not directly related to the functionality.

3.3.1 Usability
The application will be user friendly as it will be an Android application which is supported by multiple devices (android smartphones and android tablets). This will allow for the application to be easily accessible to students and tutors as majority of students have access to android devices.
3.3.2 Reliability
The probability that the system will be able to process work correctly and completely without being aborted.
In the case of system failure, the damage that could be caused could be such where a user will not be able to use the app during system failure.
3.3.3 Performance
The response time between the UI and the server will be optimised. The expected volume of user activity will peak at the end of each academic term within the tertiary institute when examinations/tests will be approaching, while on a regular basis the application will be utilised when students who feel the need to get assistance when they encounter a topic they require assistance in.
3.3.4 Supportability
The App will be facilitated over a spectrum of Android platform versions. The SDK supports 14-24.
3.3.5 Implementation
Our team has implemented the agile methodology in order to obtain our final goal of building the FindMeTutor application. For each sprint we will set targets of what we would like to achieve, with the objective of using these

milestones to be building blocks towards our final goal.

3.3.6 Interface

The UI will be made in Android studio. The set up will be simple and neat. The app will be used by students who will be using the app in order to search for a tutor which is suitable to tutor, hence, with this intention, to prevent furthering the overwhelmed feeling, the app will not be clutered and simple to use. The 'user-friendly' experience provided by the UI, will allow the user to interact with the app in a natural and intuitive way.
Each user's home page will be customized to display there upcoming tutorial sessions.

3.3.7 Packaging

Android studio for development
Adobe illustrator and photoshop for App graphics - FindMeTutor logo

4.5 Acceptance Testing

Acceptance Test general

- ACT0.01 Ensure working connection between the applcation and it's database (pass: database stores information and app displays required data from database)
- ACT1.01 Challenge registered student to re-register with taken username (fail)
- ACT1.02 Challenge new student to register with new identifier (pass)
- ACT1.03 Challenge registered student to update their account (pass)
- ACT1.04 Challenge registered student to view their account (pass)
- ACT2.01 Challenge registered tutor to re-register with taken username (fail)
- ACT2.02 Challenge new tutor to register (pass)
- ACT2.03 Challenge registered tutor to update their account (pass)
- ACT2.04 Challenge registered tutor to view their account (pass)

- ACT3.01 Challenge registered administrator to view a registered student's account (pass)
- ACT3.02 Challenge registered administrator to update a registered student's account (pass)
- ACT3.03 Challenge registered administrator to mark a registered student's account as deleted (pass)
- ACT3.04 Challenge registered administrator to view a registered tutor's account (pass)
- ACT3.05 Challenge registered administrator to update a registered tutor's account (pass)
- ACT3.06 Challenge registered administrator to mark a registered tutor's account as deleted (pass)
- ACT4.01 Challenge registered administrator to re-register with taken username(fail)
- ACT4.02 Challenge new administrator to register (pass)
- ACT4.03 Challenge registered administrator to update their account (pass)
- ACT4.04 Challenge registered administrator to view their account (pass)
- ACT5.01 A student requesting a tutor can request a tutor for a date/time which has already passed (fail)
- ACT5.02 A student requesting a tutor can request a tutor for a subject that they have not added to their enrolled subjects (fail)
- ACT5.03 A student requesting a tutor can request tutorials for various topics at a given time. ie. Student can request tutor for subject B while request tutor status for subject A is still pending (pass)
- ACT5.04 A student requesting a tutor can delete a request for a tutorial, if no longer require it. (fail)
- ACT5.05 A student receiving a tutor acceptance notification, can agree to a tutorial session from the list of tutors who have agreed (pass)

- ACT5.06 A student receiving a tutor acceptance notification, can disagree to tutorial . ie . They can decide to no longer want to continue with the tutorial, or have selected another tutor who has accepted their request.(pass)
- ACT 6.01 A tutor can accept/reject a tutor request(pass)
- ACT 7.01 A student can add an event to their upcoming events (pass)
- ACT 7.02 A student can modify an event to their upcoming events (pass)
- ACT 7.03 A student can delete an event to their upcoming events (pass)
- ACT 7.04 A student can view an event to their upcoming events (pass)
- ACT 8.01 A tutor can add an event to their upcoming events (pass)
- ACT 8.02 A tutor can modify an event to their upcoming events (pass)
- ACT 8.03 A tutor can delete an event to their upcoming events (pass)
- ACT 8.04 A tutor can view an event to their upcoming events (pass)
- ACT 9.01 A student can rate a tutor that they have previously had a tutorial session with. (pass)
- ACT 9.02 A student can rate a tutor that they have not previously had a tutorial session with. (fail)
- ACT 10.01 A student can check-in once at tutorial venue(pass)
- ACT 11.01 A tutor can check-in once at tutorial venue(pass)
- ACT 12.01 A student can pay in funds to their account balance (pass)
- ACT 12.02 A student can view balance of their account funds (pass)
- ACT 13.01 A tutor can receive funds toward their account baance - from student tutorial session payments (pass)
- ACT 13.02 A tutor can view the balance of their account funds (pass)
- ACT 14.01 A student can add subjects in which they are enrolled(pass)

- ACT 14.02 A tutor can add subjects in which they are fit to tutor(pass)
- ACT 15.01 A student can remove subjects in which they are no longer enrolled(pass)
- ACT 15.02 A tutor can remove subjects in which they no longer would want to tutor(pass)
- ACT 16.01 A student who is registered can login (pass)
- ACT 16.02 A student who is not registered can login (fail)
- ACT 16.03 A tutor who is registered can login (pass)
- ACT 16.04 A tutor who is not registered can login (fail)
- ACT 17.01 A student can view their upcoming tutorial sessions(pass)
- ACT17.02 A tutor can view their upcoming tutorial sessions(pass)
- ACT 18.01 A student can mark off completed tutorial sessions(pass)
- ACT18.02 A tutor can mark off completed tutorial sessions(pass)

4.6 System models

4.6.1 Scenario

For instance, there is a student - Joe Soap - who is currently doing his 3rd year of study in computer science. Joe would like to generate some income from tutoring first and second year mathematics modules. We also know that the student, Mary Smith, is a first year astronomy student who is looking for a mathematics tutor. The FindMeTutor app will be ideal to resolve the problems faced in this particular scenario. Joe will register on the application as a tutor, on registering, he will select what he is capable and willing to tutor - first and second year mathematics. On the other hand, we will have Mary register as a student. Mary will then be able to search for the course she needs assistance in, for example Calculus I. Mary will click the 'Request tutor' button and specify Calculus I as a subject as well as a date and time, this will send a request to all those who have registered to tutor Calculus I. Joe Soap will be part of the list of tutors

approached. Joe accepts the request. Mary is notified of this and of any other Calculus I tutors who accept the request, Marry is able to select Joe Soap to confirm a tutorial session. Marry and Joe independently need to 'check-in' and 'checkout' before and after the tutorial respectively.

4.6.2 Use cases models

Use Cases:

Use cases name	Use case
Create Student	UC-CS
Update Student	UC-US
Read Student	UC-VS
Archive Student	UC-DS
Create Tutor	UC-CT
Update Tutor	UC-UT
Read Tutor	UC-VT
Archive Tutor	UC-DT
Request Tutor	UC-RT
Read Session	UC-VSes
Archive Session	UC-DSes
Rate Tutor	UC-R
Check-in	UC-CI
Check-out	UC-CO
Add Subject	UC-CSb
Read Subject	UC-RSb
Archive Subject	UC-DSb
Login	UC-L
Update Funds	UC-AF
Read Funds	UC-VF
Confirm Funds	UC-CF
Fund Payment	UC-FP

Use Case Descriptions:

Use Case UC-CS: Create Student	
Related Requirements:	RQ1.1
Initiating actor:	Student
Actor goal:	To register on FindMeTutor
Participating Actors:	N/A
Preconditions:	Student must be enrolled in the university
Postconditions:	Student is created
Flow of activities:	
1. Student indicates sign up 2. System displays student sign up form 3. Student enters demographic data, student number, email address, contact number and password 4. System stores demographic data,student number, email address, contact number and password 5. System sends confirmation email to student 6. Student indicates confirmation 7. Student is created	

Use Case UC-US: Update Student	
Related Requirements:	RQ1.2
Initiating actor:	Student
Actor goal:	Update student demographic data,student number,student email address, student contact number ,student password or current funds
Participating Actors:	N/A
Preconditions:	Student exists and is not marked as deleted
Postconditions:	Student is updated
Flow of activities:	
1. Student requests to update Student 2. System reads Student 3. System displays form to update Student 4.Student enters student demographic data,student number,student email address, student contact number, student password or student changes profile picture 5. System stores student demographic data,student number,student email address, student contact number, student password or new profile picture 6. Student is updated	

Use Case UC-CT: Create Tutor	
Related Requirements:	RQ2.1
Initiating actor:	Tutor
Actor goal:	To register on FindMeTutor
Participating Actors:	N/A
Preconditions:	Tutor must be registered in the university
Postconditions:	Tutor is created
Flow of activities:	
1. Tutor indicates sign up 2. System displays tutor sign up form 3. Tutor enters demographic data, tutor email address, tutor contact number and tutor password 4. System stores demographic data, tutor email address, tutor contact number and tutor password 5. System sends confirmation email to Tutor 6. Tutor indicates confirmation 7. Tutor is created	

Use Case UC-UT: Update Tutor	
Related Requirements:	RQ2.2
Initiating actor:	Tutor
Actor goal:	To update Tutor demographic data, tutor email address, tutor contact number, tutor password or profile picture
Participating Actors:	N/A
Preconditions:	Tutor exists
Postconditions:	Tutor is updated
Flow of activities:	
1. Tutor requests to update Tutor 2. System reads Tutor 3. System displays form to update Tutor 4. Tutor enters Tutor demographic data, tutor email address, tutor contact number, tutor password or changes profile picture 5. System stores tutor demographic data, tutor email address, tutor contact number or tutor password 6. Tutor is updated	

Use Case UC-DS: Archive Student	
Related Requirements:	RQ1.4
Initiating actor:	Student
Actor goal:	To delete Student
Participating Actors:	N/A
Preconditions:	Student exists
Postconditions:	Student is Archived
Flow of activities:	
1. Student requests to delete Student 2. System reads Student 3. System displays confirmation message 4. Student enters confirmation 5. System marks student as archived 6. Student is archived	

Use Case UC-DT: Archive Tutor	
Related Requirements:	RQ2.4
Initiating actor:	Tutor
Actor goal:	To delete Tutor
Participating Actors:	N/A
Preconditions:	Tutor exists
Postconditions:	Tutor is Archived
Flow of activities:	
1. Tutor requests to delete Tutor 2. System displays confirmation message 3. System reads Tutor 4. Tutor enters confirmation 5. System marks Tutor as archived 6. Tutor is archived	

Use Case UC-VT: Read Tutor	
Related Requirements:	RQ2.3
Initiating actor:	Tutor, Student
Actor goal:	To view Tutor
Participating Actors:	N/A
Preconditions:	Tutor exists
Postconditions:	Tutor is viewed
Flow of activities:	
1. Tutor/Student requests to view Tutor 2. System reads tutor 3. System displays Tutor 4. Tutor is viewed	

Use Case UC-VS: Read Student	
Related Requirements:	RQ1.3
Initiating actor:	Student, Tutor
Actor goal:	To view Student
Participating Actors:	
Preconditions:	Student exists
Postconditions:	Student is viewed
Flow of activities:	
1. Student/Tutor requests to view Student 2. System reads Student 3. System displays Student 4. Student is viewed	

Use Case UC-L: Login	
Related Requirements:	RQ13.1, RQ13.2
Initiating actor:	Student or Tutor
Actor goal:	To login
Participating Actors:	N/A
Preconditions:	initiating actor exists
Postconditions:	initiating actor is logged in
Flow of activities:	
1. System prompts for student email and password 3. initiating actor enters student email and password 4. System reads Student or Tutor to check validity 5. If a valid user of the system, Student/Tutor is logged on	

Use Case UC-RT: Request Tutor	
Related Requirements:	RQ3.1, RQ3.2, RQ4.1, RQ4.2
Initiating actor:	Student
Actor goal:	To request a Tutor
Participating Actors:	Tutor
Preconditions:	Student exists, Student must be registered for one or more subjects, Tutor exists, Student has available funds
Postconditions:	Student requests Tutor
Flow of activities:	
1. Student indicates that he/she wishes to request a tutor 2. System reads Subject and Student If Student has no available funds or is not registered to a subject 3. System displays message indicating that Student can not request a Tutor Else if Student has available funds 4. System prompts for date, time, description and subject of the tutorial 5. Student enters date, time and description and subject of the tutorial 6. System reads and prompts Tutors who tutor the subject the Student has indicated he wishes to request a tutor for 7. Tutors accept or reject System prompt 8. System displays Tutors who have accepted 9. Student selects a Tutor from the displayed Tutors 10. Tutor has been requested	

Use Case UC-CSb: Add Subject	
Related Requirements:	RQ10.1, RQ11.1
Initiating actor:	Student, Tutor
Actor goal:	To register for a subject
Participating Actors:	N/A
Preconditions:	Initiating actor exists
Postconditions:	Initiating actor is registered for a subject
Flow of activities:	
1. Initiating actor indicates that he/she wants to register for a subject 2. System reads subjects 3. System displays Student Subject Table 4. Student selects subject 5. System stores subject in Student Subject Table 6. Student is registered for the subject	

Use Case UC-DSb: Archive Subject	
Related Requirements:	RQ12.1, RQ12.2
Initiating actor:	Student, Tutor
Actor goal:	To un-register for a subject
Participating Actors:	N/A
Preconditions:	Initiating actor exists
Postconditions:	Initiating actor is un-registered from a subject
Flow of activities:	
1. Initiating actor indicates that he/she wants to remove a subject 2. System displays initiating actor Subject Table 3. Initiating actor selects subject 4. System removes subject from initiating actor Subject Table 5. Initiating actor is un-registered from subject	

Use Case UC-RSb: Read Subject	
Related Requirements:	RQ17.1, RQ17.2
Initiating actor:	Student or Tutor
Actor goal:	To view subject
Participating Actors:	N/A
Preconditions:	Initiating actor exists, Subject exists
Postconditions:	Subject is viewed
Flow of activities:	
1. Initiating actor indicates that he/she wants to view a subject 2. System reads subject 3. System displays subject 4. Subject is viewed	

Use Case UC-VE: Read Session	
Related Requirements:	RQ14.1, RQ14.2
Initiating actor:	Student or Tutor
Actor goal:	To view a session
Participating Actors:	Tutor or Student
Preconditions:	initiating actor exists
Postconditions:	A session is viewed by the initiating actor
Flow of activities:	
1. Initiating actor indicates that he/she wants to view a session 2. System reads session 3. System displays a session 4. A session is viewed	

Use Case UC-DE: Archive Session	
Related Requirements:	RQ15.1, RQ15.2, RQ16.1, RQ16.2
Initiating actor:	Student, Tutor
Actor goal:	To cancel a session
Participating Actors:	Tutor or Student
Preconditions:	Initiating actor exists, A session exists and is not completed
Postconditions:	A session is cancelled
Flow of activities:	
1. A student indicated that he/she wishes to cancel a session 2. System prompts for confirmation 3. Student/tutor confirms 4. An email is sent to tutor/student indicating that a session was cancelled 5. A session is cancelled	

Use Case UC-CI: Check In	
Related Requirements:	RQ6.1, RQ7.1
Initiating actor:	Student, Tutor
Actor goal:	To check in, indicating the beginning of a tutorial session
Participating Actors:	Student or Tutor(not the same as the initiating actor)
Preconditions:	Initiating actor exists
Postconditions:	The initiating actor is checked in
Flow of activities:	
1. Initiating actor indicates that he/she has arrived at the venue where the tutorial session will take place. 2. The system reads the initiating actors device location 3. The system stores the initiating actors device location 4. Participating actor indicates that he/she has arrived at the venue where the tutorial session will take place 5. The system reads the participating actors device location 6. The system stores the participating actors device location 7. The system marks the initiating actor and the participating actor as checked in	
Extensions:	
6.1 If the initiating actors device location is not within a set radius of participating actors the system invokes step 1 of flow of activities	

Use Case UC-CO: Check Out	
Related Requirements:	RQ6.2, RQ7.2
Initiating actor:	Student, Tutor
Actor goal:	To check out from a tutorial session
Participating Actors:	Student or Tutor(not the same as the initiating actor)
Preconditions:	Initiating actor exists, participating actor exists, initiating actor is checked in, participating actor is checked in
Postconditions:	Initiating actor and secondary actor is checked out
Flow of activities:	
1. Initiating actor indicates that the tutorial has ended 2. Participating actor indicates that the tutorial has ended 3. Invokes remove funds from Student 4. Invoked add funds to Tutor 5. Invokes Rate Tutor 6. Initiating actor and participating actor is checked out	

Use Case UC-R: Rate Tutor	
Related Requirements:	RQ5.1
Initiating actor:	Student
Actor goal:	To rate a Tutor
Participating Actors:	Tutor
Preconditions:	Initiating actor exists, participating session actor exists, a session between tutor and student is complete
Postconditions:	A Tutor is rated, a tutors rating is changed
Flow of activities:	
1. Student indicates that he/she wishes to rate a Tutor 2. System prompts for rating 3. Student selects rating 4. System stores rating 5. A Student has rated a Student	

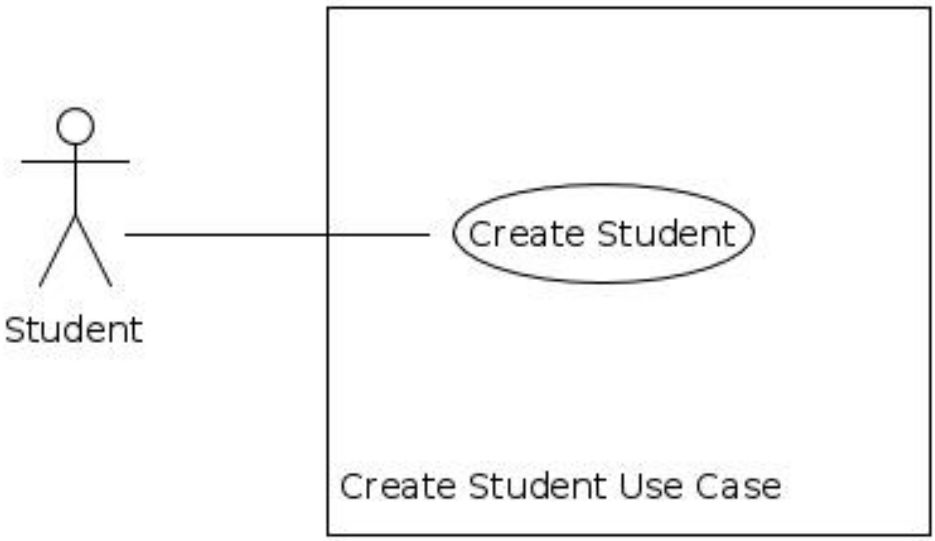
Use Case UC-CF: Confirm Funds	
Related Requirements:	RQ8.1, RQ9.1
Initiating actor:	Student, Tutor
Actor goal:	To cancel a session
Participating Actors:	
Preconditions:	Initiating actor exists
Postconditions:	
Flow of activities:	
1. 2. 3. 4. 5.	

Use Case UC-AF: Update Funds	
Related Requirements:	RQ8.3, RQ9.3
Initiating actor:	Student, Tutor
Actor goal:	To cancel a session
Participating Actors:	
Preconditions:	Initiating actor exists
Postconditions:	
Flow of activities:	
1. 2. 3. 4. 5.	

Use Case UC-VF: Read Funds	
Related Requirements:	RQ8.2, RQ9.2
Initiating actor:	Student, Tutor
Actor goal:	To cancel a session
Participating Actors:	
Preconditions:	Initiating actor exists
Postconditions:	
Flow of activities:	
1. 2. 3. 4. 5.	

4.6.3 Use case diagrams

Use case Diagram Request Tutor:



Use case Create Student:

Use case:

4.6.4 User stories

Identifier	User story	Size (points)
ST-1	As a student, I can register on the FindMeTutor application	5
ST-2	As a student, I can login to my account on the FindMeTutor application	5
ST-3	As a tutor, I can register on the FindMeTutor application	5
ST-4	As a tutor, I can login to my account on the FindMeTutor application	5
ST-6	As an administrator, I can register on the FindMeTutor application	5
ST-7	As an administrator, I can login to my account on the FindMeTutor application	5
ST-8	As a student, I can request a tutor	5
ST-9	As a student, I can add an event to my 'Upcoming events'	5
ST-10	As a student, I can ammend an event on my 'Upcoming events'	5
ST-11	As a student, I can delete an event on my 'Upcoming events'	5
ST-12	As a student, I can view an event on my 'Upcoming events'	5
ST-13	As a tutor, I can accept a tutor request	5
ST-14	As a tutor, I can reject a tutor request	5
ST-14	As a tutor, I can add an event to my 'Upcoming events'	5
ST-15	As a tutor, I can ammend an event on my 'Upcoming events'	5
ST-16	As a tutor, I can delete an event on my 'Upcoming events'	5
ST-17	As a tutor, I can view an event on my 'Upcoming events'	5
ST-18	As a student, I can select a tutor	5
ST-19	As a student, I can make contact with a tutor to make tutorial arrangements	5
ST-20	As a tutor, I can make tutorial arrangments with a student who has contacted me	5

ST-21	As a student, I can check-in to a tutorial session	5
ST-22	As a student, I can check-out to a tutorial session	5
ST-23	As a tutor, I can check-in to a tutorial session	5
ST-24	As a tutor, I can check-out to a tutorial session	5

4.6.5 Analysis object model

Appendix 1

4.6.6 Dynamic model

Appendix 2

4.6.7 User interface navigational paths and screen mock-ups

Appendix 3

4.6.8 Operational requirements

Operational requirements describe the non-business characteristics of an application.
3.5.1 Amazon Web Server - Web server to host the database
3.5.2 Android studio to design UI
3.5.3 GitHub to facilitate the build of the project among team members
3.5.4 MySql which is the database management system used to house and control the database.
3.5.5 phpMyAdmin which is used to interact with the database in a graphical user interface.

5 APPENDIX