REQUIREMENTS ANALYSIS DOCUMENT

August 11, 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 INTRODUCTION

1.0.1 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.

1.0.2 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's 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.

1.0.3 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.

1.0.4 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. It comprises of system software, or the fundamental files your computer needs to boot up and function.
6. Boot up - Processes started when computer is turned on
7. Java - is a high-level programming language

8. UI - abbreviation for User interface
9. User interface - is the means in which a person controls a software application or hardware device. 10. 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.

1.0.5 References

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

2 CURRENT SYSTEM

2.0.1 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.

3 PROPOSED SYSTEM

3.1 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, faculty registration details, security answer 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.

3.2 Functional requirements - "Shall lists"

Describes the high-level functionality of the system

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	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	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 administrator to update a student account	UC-US
RQ3.2	The system shall allow a administrator to view a student account details	UC-VS
RQ3.3	The system shall allow a administrator to mark a student as deleted	UC-DS
RQ3.4	The system shall allow a administrator to update a tutor account	UC-UT
RQ3.5	The system shall allow a administrator to view a tutor account details	UC-VT
RQ3.6	The system shall allow a administrator to mark a tutor account as deleted	UC-DT

3.3 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 pitched as an Android application which is supported by multiple devices (smartphones and tablets) that has Android as an OS. This will allow for the application to be easily accessible to students and tutors as majority of the students have access to these 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 the personalised accounts of the application user will lose their preference choices of subjects selected on sign-up.

3.3.3 Performance

The response time between the UI and the server will be(??). The expected volume of data (??). 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 immediately when they encounter a topic they require assistance in.

3.3.4 Supportability

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 hope page will be customized to the particular topics in which they are enrolled in or signed up to tutor, with respect to whether they are students or tutors.

3.3.7 Packaging

3.3.8 Legal

3.4 System models

3.4.1 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. However, Joe is also looking for a 3rd year Mathematics tutor to help him with some of his mathematics courses. 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 student and 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; any tutors who have uploaded material on this particular topic (Calculus I) will come up. Mary may find these resources useful, but if Mary does not get the assistance she requires from the material available, and would like to seek a personal tutor, Mary will click the 'Request tutor' button and 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 is now able to communicate with Joe to make further arrangements of time of meeting, venue and also to flag him of what sections she needs assistance in. On the other hand, Joe requires an 'Emergency lesson' this is a last minute lesson before his test which is scheduled to take place in a day's time. Tutors eligible to tutor Joe's course will be sent an 'Urgent request'. A tutor that accepts this request will communicate with Joe through the messaging facility which is made available. If the only possible means of having a lesson before the test is to Skype each other, then the tutor and Joe will be redirected to a skype page which will facilitate this conference.

3.4.2 Use cases models

Use cases
Create Student UC-CS
Update Student UC-US
View Student UC-VS
Archive Student UC-DS

Create Tutor UC-CT
Update Tutor UC-UT
View Tutor UC-VT
Archive Tutor UC-DT

Use Case UC-CS: Create Student	
Related Requirements:	RQ1.1
Initiating actor:	Student
Actor goal:	To register on FindMeTutor
Participating Actors:	External Database System
Preconditions:	N/A
Postconditions:	Student is created
Flow of activities:	
1. Student indicates sign up 2. System displays student sign up form 3. Student enters demographic data, faculty registration details, security answer and password 4. System sends demographic data, faculty registration details, security answer and password to external database system 5. Student is created	

Use Case UC-US: Update Student	
Related Requirements:	RQ1.2, RQ3.1
Initiating actor:	Student or Administrator
Actor goal:	To update student demographic data, faculty registration details, security answer or password
Participating Actors:	External Database System
Preconditions:	Student exists
Postconditions:	Student is updated
Flow of activities:	
1. Student/Administrator requests to update Student 2. System displays form to update Student 3. Student/Administrator enters demographic data, faculty registration details or security answer 4. System sends demographic data, faculty registration details, security answer or password to external database system 5. Student is updated	

Use Case UC-CT: Create Tutor	
Related Requirements:	RQ2.1
Initiating actor:	Tutor
Actor goal:	To register on FindMeTutor
Participating Actors:	External Database System
Preconditions:	N/A
Postconditions:	Tutor is created
Flow of activities:	
1. Tutor indicates sign up 2. System displays tutor sign up form 3. Tutor enters demographic data, courses tutored, security answer and password 4. System sends demographic data, courses tutored details, security answer and to external database system 5. Tutor is created	

Use Case UC-UT: Update Tutor	
Related Requirements:	RQ2.2, RQ3.4
Initiating actor:	Tutor or Administrator
Actor goal:	To update Tutor demographic data, courses tutored, security answer or password
Participating Actors:	External Database System
Preconditions: Tutor exists	
Postconditions:	Tutor is updated
Flow of activities:	
1. Tutor/Administrator requests to update Tutor 2. System displays form to update Tutor 3. Tutor/Administrator enters demographic data, faculty registration details or security answer 4. System sends demographic data, faculty registration details, security answer or password to external database system 5. Tutor is updated	

Use Case UC-DS: Archive Student	
Related Requirements:	RQ1.4, RQ3.3
Initiating actor:	Student or Administrator
Actor goal:	To delete Student
Participating Actors:	External Database System
Preconditions: Student exists	
Postconditions:	Student is Archived
Flow of activities:	
1. Student/Administrator requests to delete Student 2. System displays confirmation message 3. Student/Administrator enters confirmation 4. System sends archive Student signal to external database system 5. Student is archived	

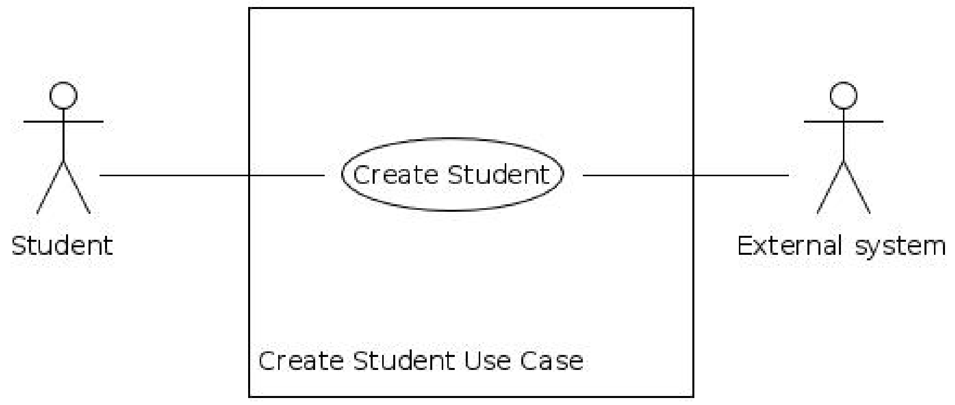
Use Case UC-DT: Archive Tutor	
Related Requirements:	RQ2.4, RQ3.6
Initiating actor:	Tutor or Administrator
Actor goal:	To delete Tutor
Participating Actors:	External Database System
Preconditions: Tutor exists	
Postconditions:	Tutor is Archived
Flow of activities:	
1. Tutor/Administrator requests to delete Tutor 2. System displays confirmation message 3. Tutor/Administrator enters confirmation 4. System sends archive Tutor signal to external database system 5. Tutor is archived	

Use Case UC-VT: View Tutor	
Related Requirements:	RQ2.3, RQ3.5
Initiating actor:	Tutor or Administrator
Actor goal:	To view Tutor
Participating Actors:	External Database System
Preconditions: Tutor exists	
Postconditions:	Tutor is viewed
Flow of activities:	
1. Tutor/Administrator requests to view Tutor 2. System displays Tutor 3. Tutor is viewed	

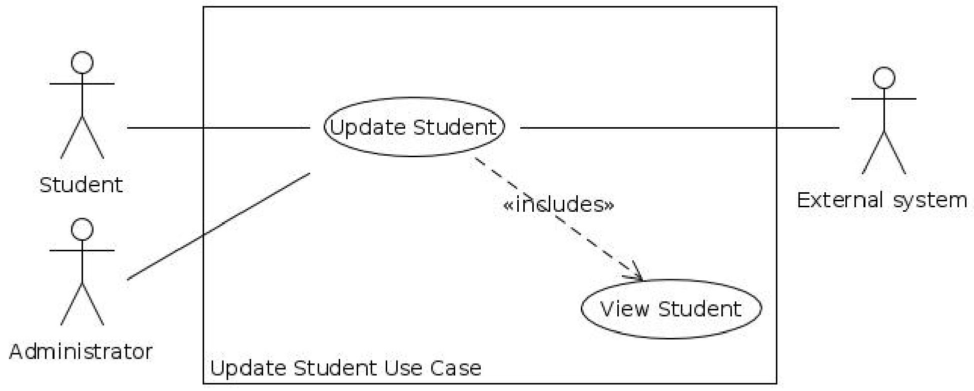
Use Case UC-VS: View Student	
Related Requirements:	RQ1.3, RQ3.2
Initiating actor:	Student or Administrator
Actor goal:	To view Student
Participating Actors:	External Database System
Preconditions: Student exists	
Postconditions:	Student is viewed
Flow of activities:	
1. Student/Administrator requests to view Student 2. System displays Student 3. Student is viewed	

3.4.3 Use case diagrams

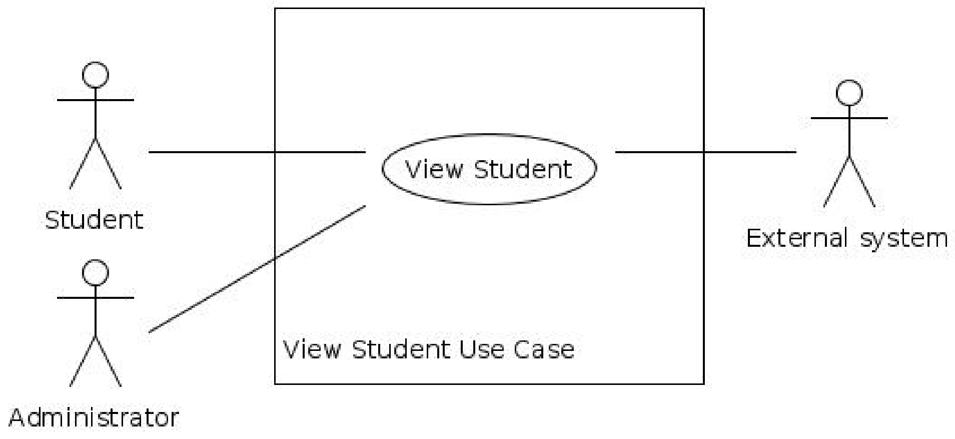
Use case diagram: Create Student



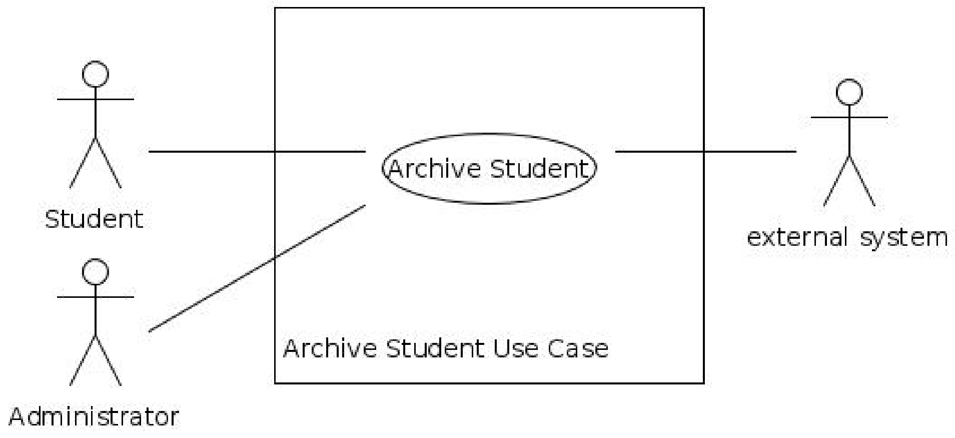
Use case diagram: Update Student



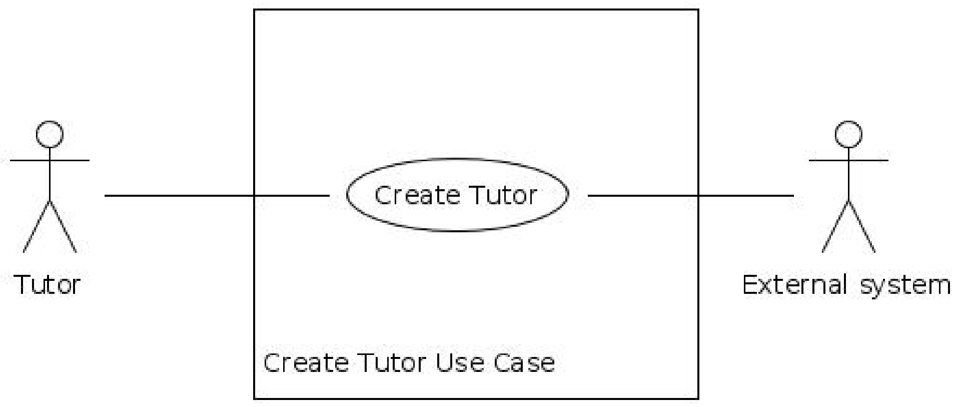
Use case diagram: View Student



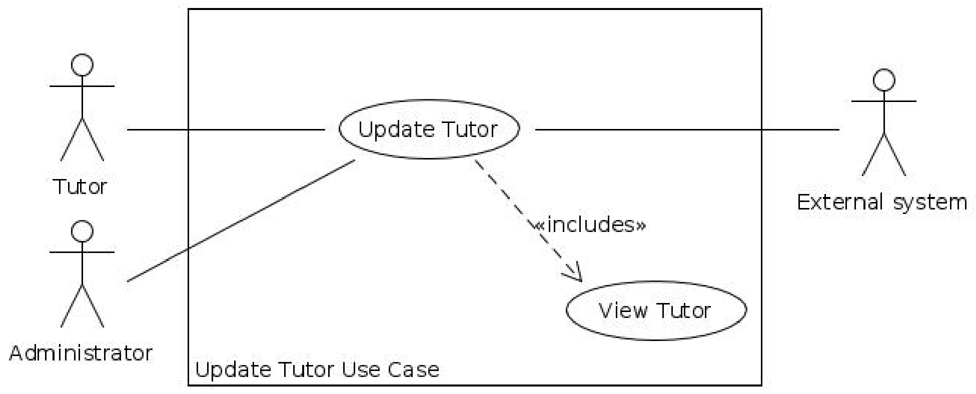
Use case diagram: Archive Student



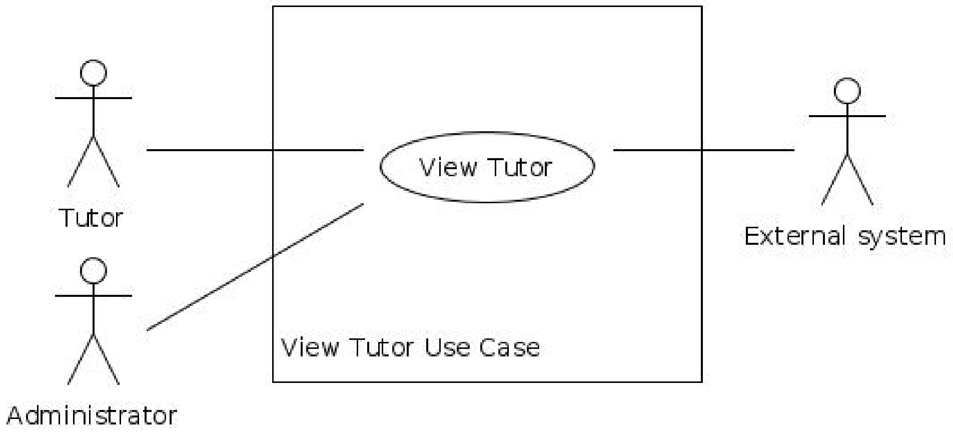
Use case diagram: Create Tutor



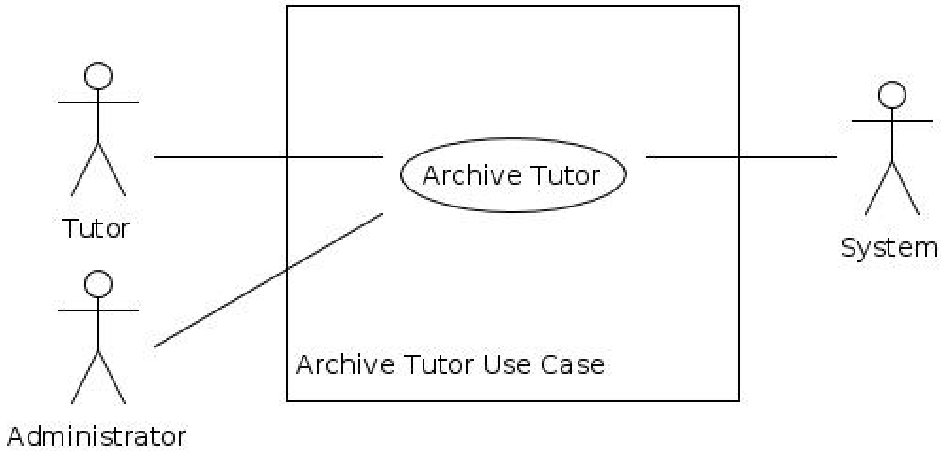
Use case diagram: Update Tutor



Use case diagram: View Tutor



Use case diagram: Archive Tutor



3.4.4 Analysis object model

3.4.5 Dynamic model

A dynamic model represents the behaviour of an object over time. It is used where the object's behaviour is best described as a set of states that occur in a defined sequence. The components of the dynamic model are:
States. The purchase order (PO) is modelled as passing through a set of states. The operations that can be performed on the PO are dependent on the state it is in. For example, information cannot be entered after it has reached the approved state.
State transitions. When purchase order data entry has been completed it transitions from the entering PO state to the pending approval state. State transitions are modelled as being instantaneous.
Events. Events trigger state transitions. For example, the order mailed event triggers a state transition from the approved to the placed state.
Actions. Actions occur on state transitions. For example, on a goods received event the action of notify purchaser is performed. Actions are modelled as instantaneous occurrences (contrast with activities).
Activities. An activity is performed while an object is in a specific state. For example, while in the entering PO state data is entered into the PO. Activities are modelled as occurring over a period of time (contrast with actions).
(Will add diagram)

3.4.6 User interface navigational paths and screen mock-ups

3.4.7 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.