

# **REQUIREMENTS ANALYSIS DOCUMENT**

September 2, 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

<b>1</b>	<b>3</b>
<b>2 INTRODUCTION</b>	<b>4</b>
2.0.1 Purpose of the system . . . . .	4
2.0.2 Scope of the system . . . . .	4
2.0.3 Objective and success criteria of the project . . . . .	4
2.0.4 Definitions, acronyms, and abbreviations . . . . .	4
2.0.5 References . . . . .	5
<b>3 CURRENT SYSTEM</b>	<b>5</b>
3.0.1 Overview . . . . .	5
<b>4 PROPOSED SYSTEM</b>	<b>5</b>
4.1 Overview . . . . .	5
4.2 Functional requirements - "Shall lists" . . . . .	6
4.3 Non-functional requirements . . . . .	8
4.4 System models . . . . .	9
4.4.1 Scenario . . . . .	9
4.4.2 Use cases models . . . . .	10
4.4.3 Use case diagrams . . . . .	16
4.4.4 Analysis object model . . . . .	20
4.4.5 Dynamic model . . . . .	20
4.4.6 User interface navigational paths and screen mock-ups . . . . .	20
4.4.7 Operational requirements . . . . .	20

# 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. 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 identity 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.

## **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, 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.

## 4.2 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 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 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 eg update password	UC-UT
RQ3.5	The system shall allow a administrator to view a tutor account details	UC-VT

RQ3.6	The system shall allow an administrator to mark a tutor account as deleted	UC-DT
RQ4.1	The system shall allow an administrator to register	UC-CA
RQ4.2	The system shall allow an administrator to update their account	UC-UA
RQ4.3	The system shall allow an administrator to view their account details	UC-VA
RQ4.4	The system shall allow an administrator to mark their account as deleted	UC-DA
RQ5.1	The system shall allow a student to request a tutor	UC-RT
RQ5.2	The system shall allow a student to choose a tutor from a list	UC-CT
RQ6.1	The system shall allow a tutor to accept a request	UC-AR
RQ6.2	The system shall allow a tutor to reject a request	UC-DR
RQ7.1	The system shall allow a student to add events to their personal 'Upcoming events'	UC-CE
RQ7.2	The system shall allow a student to view their upcoming events	UC-VE
RQ7.3	The system shall allow a student to update their upcoming events	UC-UE
RQ7.4	The system shall allow a student to delete their upcoming events	UC-DE
RQ8.1	The system shall allow a tutor to add events to their personal 'Upcoming events'	UC-CE
RQ8.2	The system shall allow a tutor to view their upcoming events	UC-VE
RQ8.3	The system shall allow a tutor to update their upcoming events	UC-UE
RQ8.4	The system shall allow a tutor to delete their upcoming events	UC-DE
RQ9.1	The system shall allow a student to rate a tutor	UC-R
RQ9.1	The system shall allow a student to check-in	UC-CI
RQ9.2	The system shall allow a student to check-out	UC-CO
RQ10.1	The system shall allow a tutor to check-in	UC-CI
RQ10.2	The system shall allow a tutor to check-out	UC-CO
RQ11	The system shall allow a student to add funds	

## **4.3 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 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 immediately 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 cluttered 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 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**

Android studio for development

Adobe illustrator and photoshop for App graphics - FindMeTutor logo

## **4.4 System models**

### **4.4.1 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.

#### **4.4.2 Use cases models**

##### **Use cases**

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  
Create Administrator UC-CA  
Update Administrator UC-UA  
Read Administrator UC-VA  
Archive Administrator UC-DA  
Request Tutor UC-RT  
Choose Tutor UC-CT  
Create Event UC-CE  
Update Event UC-UE  
Read Event UC-VE  
Archive Event UC-DE  
Rate Tutor UC-R  
Accept Request UC-AR  
Archive Request UC-DR  
Check-in UC-CI  
Check-out UC-CO  
Add Subject UC-CSb  
Archive Subject UC-DSb

Note: We have classed the database as a component of our system//

<b>Use Case UC-CS: Create Student</b>	
Related Requirements:	RQ1.1
Initiating actor:	Student
Actor goal:	To register on FindMeTutor
Participating Actors: N/A	
Preconditions:	N/A
Postconditions:	Student is created
Flow of activities:	
<ol style="list-style-type: none"> <li>1. Student indicates sign up</li> <li>2. System displays student sign up form</li> <li>3. Student enters demographic data,student number, email address, contact number and password</li> <li>4. System stores demographic data,student number, email address, contact number and password</li> <li>5. System sends confirmation email to student</li> <li>6. Student indicates confirmation</li> <li>7. Student is created</li> </ol>	
<b>Use Case UC-US: Update Student</b>	
Related Requirements:	RQ1.2, RQ3.1
Initiating actor:	Student or Administrator
Actor goal:	Update student demographic data,student number,student email address, student contact number or student password
Participating Actors:	N/A
Preconditions:	Student exists and is not marked as deleted
Postconditions:	Student is updated
Flow of activities:	
<ol style="list-style-type: none"> <li>1. Student/Administrator requests to update Student</li> <li>2. System reads Student</li> <li>3. System displays form to update Student</li> <li>4. Student/Administrator enters student demographic data,student number,student email address, student contact number or student password</li> <li>5. System stores student demographic data,student number,student email address, student contact number or student password</li> <li>6. Student is updated</li> </ol>	

<b>Use Case UC-CT: Create Tutor</b>	
Related Requirements:	RQ2.1
Initiating actor:	Tutor
Actor goal:	To register on FindMeTutor
Participating Actors:	N/A
Preconditions:	N/A
Postconditions:	Tutor is created
Flow of activities:	
<ol style="list-style-type: none"> <li>1. Tutor indicates sign up</li> <li>2. System displays tutor sign up form</li> <li>3. Tutor enters demographic data, tutor email address, tutor contact number and tutor password</li> <li>4. System stores demographic data, tutor email address, tutor contact number and tutor password</li> <li>5. System sends confirmation email to Tutor</li> <li>6. Tutor indicates confirmation</li> <li>7. Tutor is created</li> </ol>	
<b>Use Case UC-UT: Update Tutor</b>	
Related Requirements:	RQ2.2, RQ3.4
Initiating actor:	Tutor or Administrator
Actor goal:	To update Tutor demographic data, tutor email address, tutor contact number and tutor password
Participating Actors:	N/A
Preconditions:	Tutor exists
Postconditions:	Tutor is updated
Flow of activities:	
<ol style="list-style-type: none"> <li>1. Tutor/Administrator requests to update Tutor</li> <li>2. System reads Tutor</li> <li>3. System displays form to update Tutor</li> <li>4. Tutor/Administrator enters Tutor demographic data, tutor email address, tutor contact number or tutor password</li> <li>5. System stores tutor demographic data, tutor email address, tutor contact number or tutor password</li> <li>6. Tutor is updated</li> </ol>	

<b>Use Case UC-DS: Archive Student</b>	
Related Requirements:	RQ1.4, RQ3.3
Initiating actor:	Student or Administrator
Actor goal:	To delete Student
Participating Actors:	N/A
Preconditions:	Student exists
Postconditions:	Student is Archived
Flow of activities:	
<ol style="list-style-type: none"> <li>1. Student/Administrator requests to delete Student</li> <li>2. System reads Student</li> <li>3. System displays confirmation message</li> <li>4. Student/Administrator enters confirmation</li> <li>5. System marks student as archived</li> <li>6. Student is archived</li> </ol>	
<b>Use Case UC-DT: Archive Tutor</b>	
Related Requirements:	RQ2.4, RQ3.6
Initiating actor:	Tutor or Administrator
Actor goal:	To delete Tutor
Participating Actors:	N/A
Preconditions:	Tutor exists
Postconditions:	Tutor is Archived
Flow of activities:	
<ol style="list-style-type: none"> <li>1. Tutor/Administrator requests to delete Tutor</li> <li>2. System displays confirmation message</li> <li>3. System reads Tutor</li> <li>4. Tutor/Administrator enters confirmation</li> <li>5. System marks Tutor as archived</li> <li>6. Tutor is archived</li> </ol>	

<b>Use Case UC-VT: Read Tutor</b>	
Related Requirements:	RQ2.3, RQ3.5
Initiating actor:	Tutor or Administrator
Actor goal:	To view Tutor
Participating Actors:	N/A
Preconditions:	Tutor exists
Postconditions:	Tutor is viewed
Flow of activities:	
1. Tutor/Administrator requests to view Tutor 2. System reads tutor 3. System displays Tutor 4. Tutor is viewed	
<b>Use Case UC-VS: Read Student</b>	
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 reads Student 3. System displays Student 4. Student is viewed	

Use Case UC-RT: Request Tutor	
Related Requirements:	RQ5.1
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:	
<ol style="list-style-type: none"> <li>1. Student indicates that he/she wishes to request a tutor</li> <li>2. System reads Student</li> <li>If Student has no available funds or is not registered to a subject</li> <li>3. System displays message indicating that Student can not request a Tutor</li> <li>Else if Student has available funds</li> <li>5. System prompts for date, time, description and subject of the tutorial</li> <li>6. Student enters date, time and description and subject of the tutorial</li> <li>7. System prompts Tutors who tutor the subject the Student has indicated he wishes to request a tutor for</li> <li>8. Tutors accept or reject System prompt</li> <li>9. System displays Tutors who have accepted</li> <li>10. Student selects a Tutor from the displayed Tutors</li> <li>11. Tutor has been requested</li> </ol>	

#### 4.4.3 Use case diagrams



Use cases:

Use cases:

Use cases

#### **4.4.4 Analysis object model**

(Attached P17)

#### **4.4.5 Dynamic model**

(Attached P18)

#### **4.4.6 User interface navigational paths and screen mock-ups**

(Attached P19)

#### **4.4.7 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.