**Android Based Online Course Registration (MSc)**

A project report submitted to the Department of Computer Science and Engineering, Jahangirnagar University in partial fulfillment of the requirements for the degree of

Master of Science (MSc) in Computer Science.

**Submitted By**

**Kazi Ahmadullah**

CSE 201701045

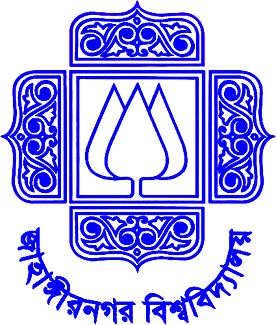
**Supervisor**

**Dr. Abu Sayed Md. Mostafizur Rahaman**

Professor

Department of Computer Science and Engineering

Jahangirnagar University

****

March 2018

Department of Computer Science and Engineering

Jahangirnagar University

**Declaration**

I do, hereby declare that the work presented in this project report is done by us under the Supervisor of **Dr. Abu Sayed Md. Mostafizur Rahaman**,Professor of Computer Science and Engineering Department, Jahangirnagar University, Savar, Dhaka. We also declare that neither this project nor any part thereof has been submitted elsewhere for the award of any degree or diploma.

**Submitted By-**

**Kazi Ahmadullah**

ID: CSE201701045

PMSCS Batch: 13

Department of Computer Science and Engineering

Jahangirnagar University

Savar, Dhaka, Bangladesh

**APPROVAL OF ACCEPTANCE**

This project report written by **Kazi Ahmadullah** (ID:CSE201701045) entitled “**Android Based Online Course Registration (MscReg)**” is submitted to the PMSCS Program, Department of Computer Science and Engineering, Jahangirnagar University in partial fulfilment of the requirements for the degree of master of Science in Computer Science. The project is done under the supervision of **Dr. Abu Sayed Md. Mostafizur Rahman**, Professor of Computer Science and Engineering Department, Jahangirnagar University.

We have examined this report and recommend its acceptance.

**Dr. Abu Sayed Md. Mostafizur Rahaman**

Professor and 1st Examiner

Department of Computer Science and Engineering

Jahangirnagar University

Savar, Dhaka, Bangladesh

**Sanjit Kumar Saha**

Assistant Professor and 2nd Examiner

Department of Computer Science and Engineering

Jahangirnagar University

Savar, Dhaka, Bangladesh

**Accepted By**

**Liton Jude Rozario**

Professor and Coordinator, PMSCS program 2017

Department of Computer Science and Engineering

Jahangirnagar University

Savar, Dhaka, Bangladesh

**Acknowledgement**

The authors show signification and indescribable gratefulness to their supervisor, Associate Professor, Department of Computer Science and Engineering Dr. Abu Sayed Md. Mostafizur Rahman for his outstanding help contribution is giving suggestion and encouragement and guiding in the right direction throughout the development process. He has been a great source of effective and feasible ideas, profound knowledge and all time feedback for the author.

The authors also like to thank all the teacher of CSE department of Jahangirnagar University who helped the authors providing guidelines to perform the work.

**Authors**

Kazi Ahmadullah

**Abstract**

This report is concern the development of assistant for taken course from online. This application is based on Android, an open source mobile platform. This application does not need much internet connection and also don’t need to take a lot of input. Student can use it to finding course, registration course, login profile, view profile. All these can be done using an android mobile phone in offline. Internet connection is only needed for course registration.

**Table of Contents**

|  |  |
| --- | --- |
| Declaration | i |
| Approval of acceptance | ii |
| Acknowledgement | iii |
| Abstract | iv |
| Table of contents | v |
| List of Figures | vii |

|  |  |  |  |
| --- | --- | --- | --- |
| **Chapter 1 : Introduction** | | | |
|  | 1.1 Introduction to Android | | 1 |
|  | 1.2 Android application | | 1 |
|  | 1.3 Project summary | | 1 |
|  | 1.3.1 Objective | | 1 |
|  | 1.3.2 Practical uses | | 1 |
|  | 1.4 Technology Used | | 2 |
| **Chapter 2 : State of the art** | | |  |
|  | 2.1 Existing applications | | 3 |
|  | 2.2 Limitation of existing system | | 4 |
|  | 2.3 Scope of proposed | | 4 |
|  | 2.4 Limitation of proposed system | | 4 |
| **Chapter 3 : System analysis** | | |  |
|  | 3.1 User Requirements | | 5 |
|  | 3.2 System requirements | | 5 |
|  | 3.3 SRC ( Software Requirement Specification | | 5 |
| **Chapter 4 : System design** | | |  |
|  | 4.1 System design | | 7 |
|  | 4.2 User Diagram | | 7 |
|  | 4.3 Sequence Diagram | | 8 |
|  | 4.4 Class Diagram | | 12 |
| **Chapter 5 : Implementation** | | |  |
|  | 5.1 Implementation | | 16 |
|  | 5.1.1 Android studio | | 16 |
|  | 5.1.2 SQLite | | 16 |
|  | 5.1.3 Shared Preferences | | 17 |
|  | 5.1.4 Android manifest | | 17 |
|  | 5.2 Error | | 17 |
|  | 5.3 Screenshot | | 18 |
| **Chapter 6 : Result & Testing** | | | 19 |
| **Chapter 7** | | |  |
|  | | Conclusion | 22 |
|  | | Future of work | 22 |
| **Reference** | | | 23 |
| **Appendix** | | | 24 |

|  |  |  |
| --- | --- | --- |
| **List of Figures** | | |
| **Figure No** | **Title** | **Page No** |
| **4.2** | Use Diagram for MscReg | 7 |
| **4.3** | Sequence Diagram for MscReg | 11 |
| **4.4** | Class Diagram for MscReg | 15 |
| **5.2** | Error Running app | 17 |
| **5.3.1** | Splash Screen | 18 |
| **5.3.2** | Login Page | 18 |
| **5.3.3** | Registration Page | 18 |
| **6.2.1** | MScReg Splash screen | 19 |
| **6.2.2** | MScReg Registration page | 19 |
| **6.2.3** | Required Data for MscReg | 20 |
| **6.2.4** | MscReg login page | 20 |
| **6.2.5** | MscReg Student Home Page | 21 |
| **6.2.6** | MscReg Add Course Page | 21 |
|  |  |  |

**CHAPTER 1**

**INTRODUCTION**

**1.1 Introduction to Android**

Android is a java-based operating system which runs on Linux 2.6 kernel. The system is very lightweight and full featured. Today android is the world’s most popular mobile OS.

**1.2 Android application**

An android application is a software application running on the android platform. Android application is designed for a smartphone or a tablet PC running on the android OS. Android application easily available in developer's website or Google play store or Amazon Appstore and so on.

**1.3 Project summary**

WFore is Android-based registration software that helps you to register courses online. It is ideal online registration programs. The weather data are including daily temperature, daily pressure and weekly rainfall. It's helpful for all kind of people who are keen to know the weather before go outside from home or any work.

**1.3.1 Objective**

Responsive view with different shape of android device. Attractive outlook from website. Response as soon as possible from online weather forecast platforms.

* System will allow weather forecast of people in different area.
* System has inbuilt validation system to validate the entered data.
* Fast, flexible, and less prone to errors and reduce expenses and save time.

**1.3.2 Practical uses**

In WFore different category people will be the users and the businessman may be the secondary user.

**1.4 Technology used**

**Project tools**

|  |  |
| --- | --- |
| Front end tools : | Android Studio |
| Back end tools : | SQLite |
| Platform : | Android |

**Software requirements**

|  |  |
| --- | --- |
| Application frontend | Android Studio |
| Database backend | SQLite |
| Office automation tools | Google Docs, |
| OS | Windows 7 |

**Hardware requirements (Recommended)**

|  |  |
| --- | --- |
| OS Version | Windows |
| Ram | 4 GB RAM minimum, 8 GB RAM recommended |
| Disk space | 500 MB disk space for Android Studio, at least 1.5 GB for Android SDK, emulator system images, and caches |
| Java version | Java Development Kit (JDK) 7 |
| Screen resolution | 1280x800 minimum screen resolution |

**Chapter 2**

**State of the art**

**2.1 Existing applications**

There are several applications for weather forecast.

**Ex 1 : DIU - Smart Student**

This app for the people of different area with all must have features as a weather forecast. .The simplest method of forecasting the weather, persistence, relies upon today’s conditions to forecast the conditions tomorrow i.e. the weather data are including daily temperature, daily pressure and weekly rainfall.

**Ref 1 (DIU - Smart Student)**

**Ex 2 : Online Courses**



Watch the weather online from your smartphone. Earn by using this app and also get weather notification for which city you choose or current city you are right now.

**Ref 2 (Online Courses)**

**2.2 Limitation of existing system**

The current system available on market is use only for online weather forecast. But existing system cannot choose online weather for different area it can choose online for only change city. This type of application helps for weather details information. Some application are not available all country.

**2.3 Scope of proposed system**

In this application user (different category of people and businessman) can do a lot of things to have knowledge about weather.

* User can change city and check the details of temperature, humanity, uv index and wind.
* After successful submission, system will give unique city weather for each user.

**2.4 Limitation of proposed system**

This project only works on devices, which have android operating system. If internet connection is disabled then user cannot change city. The record’s value could not be shared with others via mail or share it. Also could not be downloaded records as doc or pdf format.

**Chapter 3**

**System analysis**

**3.1 User Requirements :**

User:

1. Change city.
2. Forecast weather and weather details.

**3.2 System requirements:**

* Current city are automatically available.
* Choose option for change city.
* Temperature in Celsius and Fahrenheit.
* Temperature in hourly, weekly and daily pressure.
* Date and time also generated.
* Details weather also show like humidity, visibility and uv index.
* Wind and pressure are also show.

**3.3 SRC (Software Requirement Specification)**

Case 1 – Select city

In Dhaka or other city people are automatically find his/her current location.

Case 2 – Choose option for change city

The user can change city that the want .They find the weather details from this.

Case 3 – Forecast weather

The weather data are including daily temperature, daily pressure and weekly rainfall.

Case 4 – Weather details

The weather data are including feels like humidity, visibility, uv-index etc.

After successful choosing city then automatic weather detail generate.

**Chapter 4**

**System design**

**4.1 System design:**

System design is the most important part of a project. System design define the architecture, components, modules, interfaces and data for a system to satisfy specified requirement. System design could be divided into three part :

The first one is Conceptual design tells what the system should do.

The second one is Logical design tells what the system should look to the user.

And the last one is Physical design tells how the system should be built.

**4.2 Use case Diagram**

Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. Hence, when a system is analyzed to gather its functionalities, use cases are prepared and actors are identified.

**Student**

**Fig 4.2: Use case Diagram for MScReg**

The use case diagram in this case consists of actors such as student and order veriﬁcation system. The following describes the use cases:

* New students register with the system.
* Existing students log on to the MScReg and can browse the course catalog.
* A student can select courses and add them to semester course list.
* A student can delete course to taken semester course list.
* A student get department notice from server.
* An administrator approves the order and enrolls the student in the courses.
* The system notiﬁes the student of enrollment via e-mail.

**4.3 Sequence Diagram**

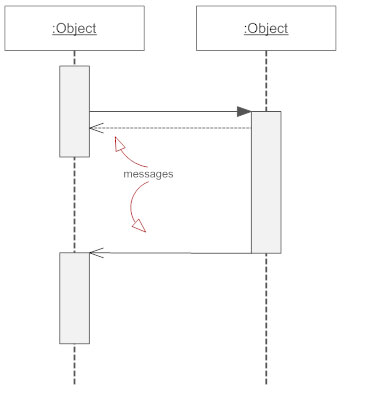
**Class Roles or Participants**  
Class roles describe the way an object will behave in context. Use the UML object symbol to illustrate class roles, but don't list object attributes.

Object symbol - Sequence diagram

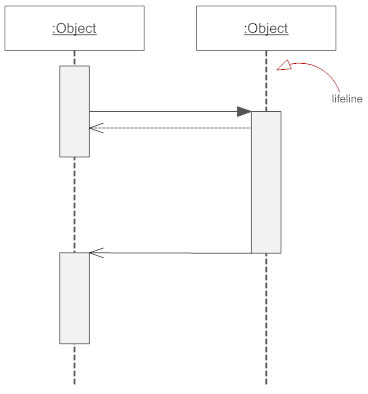
**Activation or Execution Occurrence**  
Activation boxes represent the time an object needs to complete a task. When an object is busy executing a process or waiting for a reply message, use a thin gray rectangle placed vertically on its lifeline.



**Messages**  
Messages are arrows that represent communication between objects. Use half-arrowed lines to represent asynchronous messages. Asynchronous messages are sent from an object that will not wait for a response from the receiver before continuing its tasks. For message types, see below.



**Lifelines**  
Lifelines are vertical dashed lines that indicate the object's presence over time.



**Destroying Objects**  
Objects can be terminated early using an arrow labeled "<< destroy >>" that points to an X. This object is removed from memory. When that object's lifeline ends, you can place an X at the end of its lifeline to denote a destruction occurrence.

**Loops**  
A repetition or loop within a sequence diagram is depicted as a rectangle. Place the condition for exiting the loop at the bottom left corner in square brackets [ ].

Types of Messages in Sequence Diagrams

**Synchronous Message**  
A synchronous message requires a response before the interaction can continue. It's usually drawn using a line with a solid arrowhead pointing from one object to another.

Synchronous message - Sequence diagram

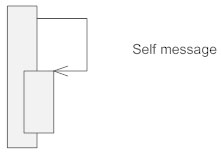
**Asynchronous Message**  
Asynchronous messages don't need a reply for interaction to continue. Like synchronous messages, they are drawn with an arrow connecting two lifelines; however, the arrowhead is usually open and there's no return message depicted.

Simple messageAsyncrhonous message

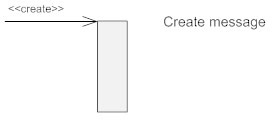
**Reply or Return Message**  
A reply message is drawn with a dotted line and an open arrowhead pointing back to the original lifeline.

Reply messages

**Self Message**  
A message an object sends to itself, usually shown as a U shaped arrow pointing back to itself.



**Create Message**  
This is a message that creates a new object. Similar to a return message, it's depicted with a dashed line and an open arrowhead that points to the rectangle representing the object created.



**Delete Message**  
This is a message that destroys an object. It can be shown by an arrow with an x at the end.

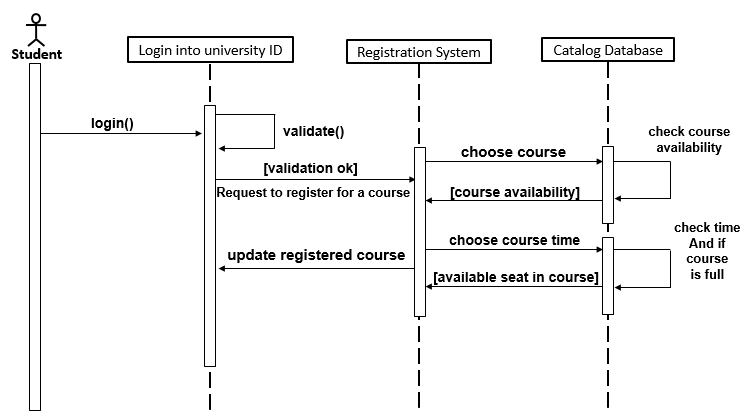
Delete message

**Found Message**  
A message sent from an unknown recipient, shown by an arrow from an endpoint to a lifeline.

Found message

**Lost Message**  
A message sent to an unknown recipient. It's shown by an arrow going from a lifeline to an endpoint, a filled circle or an x.

Lost message



**Fig 4.3: Sequence Diagram for MScReg**

Objects are discovered by examining the use cases and scenarios and grouped into classes. Each class should have a definition which states the purpose of the class. Packages are created to hold logical groups of classes. Classes and packages are drawn in the Logical View of the tool. The following packages and classes have been created for the registration system:

**People**

* StudentInfo--Information about the student actor needed by the registration system (for example, name, address, phone, idNumber, major, gradDate).

**UniversityArtifacts**

* CourseOffering--Specific information about selections for a semester (for example, daysOffered, timeOfDay, location).
* StudentSchedule--Output report containing the list of registered course offerings generated when a student registers for a course.
* CourseRoster--Output report containing the list of registered students for a specific course offering generated for a professor.

**Interfaces**

* RegistrationForm--Form which provides the capability for a student to select registration options.
* Add/DropForm--Form which provides the capability for a student to modify a course schedule.
* CourseSelectionForm--Form which provides the capability for a professor to add/drop courses to teach.
* StudentMaintenanceForm--Form which provides the capability for the registrar to add/delete/modify student information.
* CourseMaintenanceForm--Form which provides the capability for the registrar to add/delete/modify course and course offering information.

**4.4 Class Diagram:**

Classes

Classes represent an abstraction of entities with common characteristics. Associations represent the relationships between classes.

Illustrate classes with rectangles divided into compartments. Place the name of the class in the first partition (centered, bolded, and capitalized), list the attributes in the second partition (left-aligned, not bolded, and lowercase), and write operations into the third.



Active Classes

Active classes initiate and control the flow of activity, while passive classes store data and serve other classes. Illustrate active classes with a thicker border.



Visibility

Use visibility markers to signify who can access the information contained within a class. Private visibility, denoted with a - sign, hides information from anything outside the class partition. Public visibility, denoted with a + sign, allows all other classes to view the marked information. Protected visibility, denoted with a # sign, allows child classes to access information they inherited from a parent class.



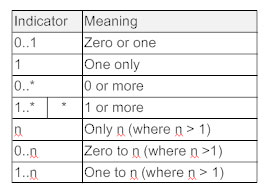
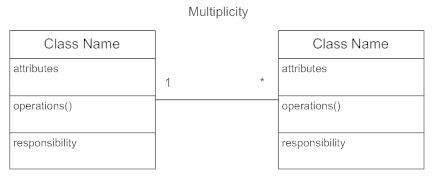
Associations

Associations represent static relationships between classes. Place association names above, on, or below the association line. Use a filled arrow to indicate the direction of the relationship. Place roles near the end of an association. Roles represent the way the two classes see each other.



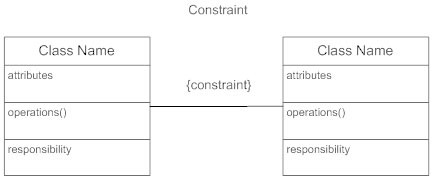
Multiplicity (Cardinality)

Place multiplicity notations near the ends of an association. These symbols indicate the number of instances of one class linked to one instance of the other class. For example, one company will have one or more employees, but each employee works for just one company.



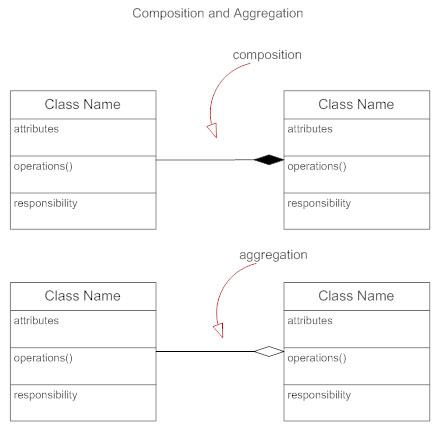
Constraint

Place constraints inside curly braces {}.



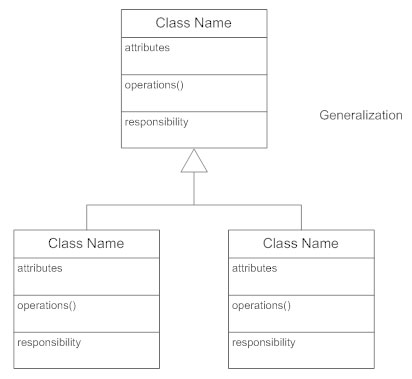
Composition and Aggregation

Composition is a special type of aggregation that denotes a strong ownership between Class A, the whole, and Class B, its part. Illustrate composition with a filled diamond. Use a hollow diamond to represent a simple aggregation relationship, in which the "whole" class plays a more important role than the "part" class, but the two classes are not dependent on each other. The diamond ends in both composition and aggregation relationships point toward the "whole" class (i.e., the aggregation).

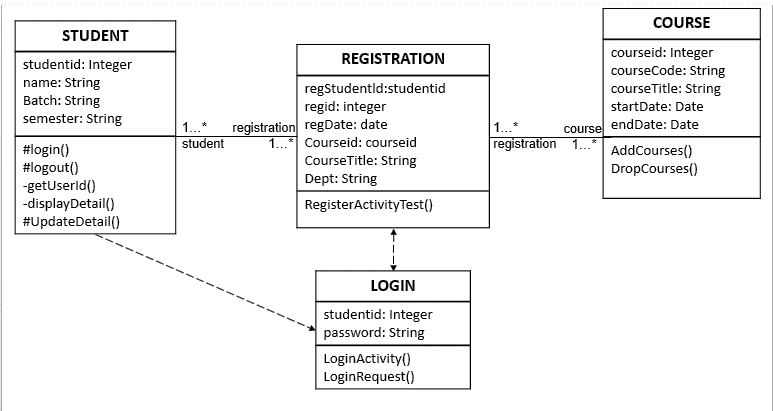


Generalization

Generalization is another name for inheritance or an "is a" relationship. It refers to a relationship between two classes where one class is a specialized version of another. For example, Honda is a type of car. So the class Honda would have a generalization relationship with the class car.



In real life coding examples, the difference between inheritance and aggregation can be confusing. If you have an aggregation relationship, the aggregate (the whole) can access only the PUBLIC functions of the part class. On the other hand, inheritance allows the inheriting class to access both the PUBLIC and PROTECTED functions of the superclass.



**Fig 4.4: Class Diagram for MScReg**

Class diagrams are created to graphically depict the packages and classes in the model. The Main class diagram typically contains only packages. Each package contains its own class diagrams. The Main class diagram for a package contains

the public classes of the package (classes that communicate with classes in other packages). Other class diagrams are created as needed. Class diagrams are contained in the Logical View of the tool.

Use cases and scenarios are examined to determine the relationships needed by the system. Relationships between classes are created and displayed on selected class diagrams.

**Chapter 5**

**Implementation**

**5.1 Implementation**

System implementation is the process of defining how the system should be built and ensuring that the system is operational and accomplish strategic objectives and goals. **Ref (System implementation )**

Implementation is most important part for building a system. In my project i use project tools, software requirement and hardware requirement.

**5.1.1 Android studio :**

Android Studio provides the fastest tools for building apps on every type of Android device. I use 3.0 version IDE for my project and It includes a bug fixes focused around gradle, the core IDE, and lint. Android studio provide world-class code editing, debugging, performance tooling, a flexible build system, and an instant build/deploy system all allow you to focus on building unique and high quality apps.



**Ref (Android studio )**

**5.1.2 SQLite :**

SQLite is an open source database. SQLite does not need to be installed as there is no setup procedure to use it. SQLite supports standard relational database feature like SQL syntax, transactions and prepared statements. The database requires limited memory at runtime ( approx 250 KByte ) which makes it a lightweight database to embed into other runtime.



**Ref ( SQLite )**

**5.1.3 Shared Preferences :**

Shared preferences is a way of storing data for android. It allows to save and retrieve data in the form of key,value pair.

In order to use shared preferences, need to call a method getSharedPreferences() that returns a SharedPreference instance pointing to the file that contains the values of preferences.

SharedPreferences sharedpreferences = getSharedPreferences(MyPREFERENCES, Context.MODE\_PRIVATE);

The first parameter is the key and the second parameter is the MODE.

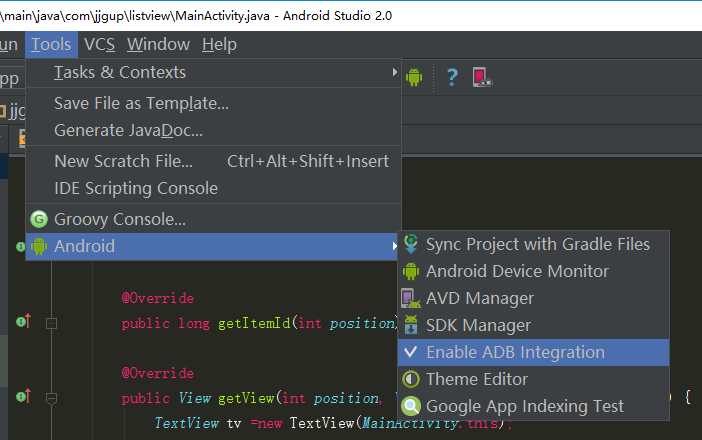
**5.1.4 Android manifest :**

Manifest is in root directory of android application. The manifest file provides essential information about your app to the Android system, which the system must have before it can run any of the app's code.It determines the processes that host the application components. It declares the permissions that the application must have in order to access protected parts of the API and interact with other applications. It also declares the permissions that others are required to have in order to interact with the application's components.

**5.2 Error :**

Problem 1 : In this MScReg need online for all activity.

Problem 2 : Error running app: Instant Run requires 'Tools | Android | Enable ADB integration' to be enabled



**Fig 5.2: Error Running app**

menu>> tools>>Android>>Enable ADB integration check it.

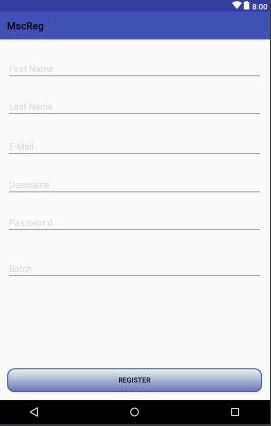
**5.3 Screenshot :**

****

**Fig 5.3.1: Splash Screen**

****

**Fig 5.3.2: Login Page**

****

**Fig 5.3.3: Registration Page**

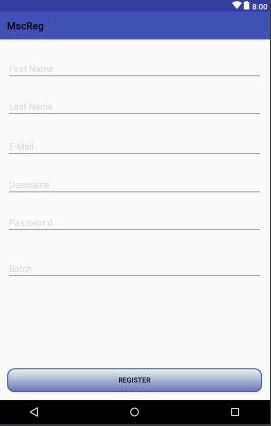
**Chapter 6**

**Result**



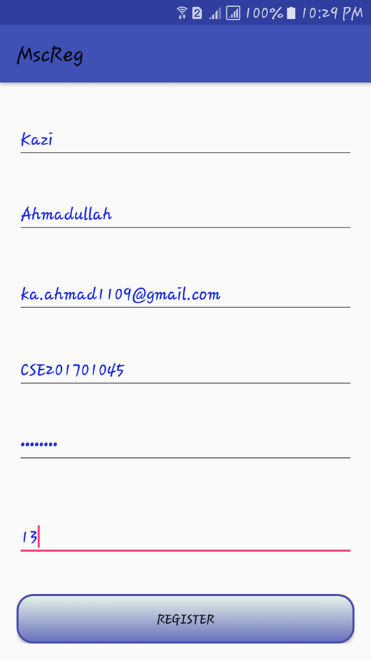
**Fig 6.2.1:** MScRegSplash screen

This is MScReg splash or welcome screen. It’s taking some time for staring MScReg app login page.



**Fig 6.2.2:** MScReg Registration page

This is MScReg Registration page for new student. Here new student creating account with university validation.



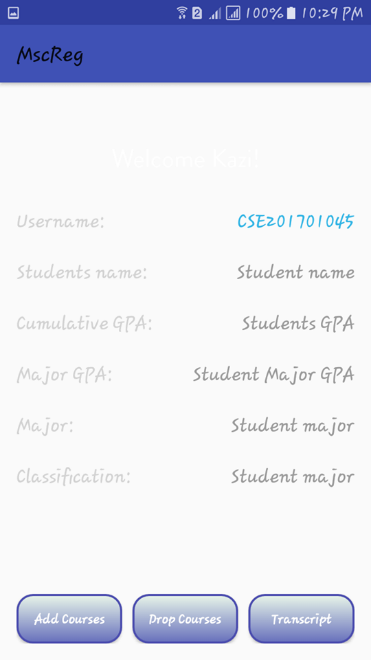
**Fig 6.2.3: Required Data for MScReg**

Registration from fill up carefully. Here contain required information for student registration.



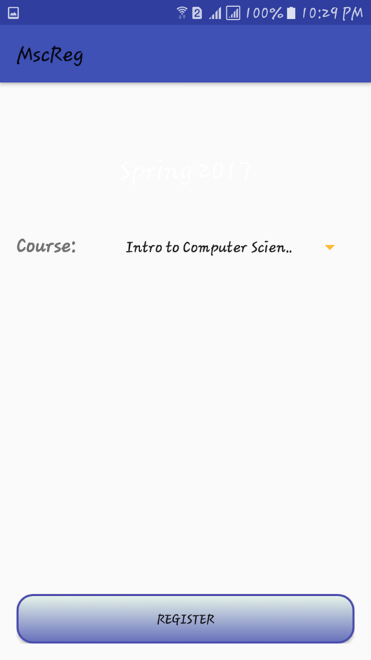
**Fig 6.2.4: MscReg login page**

After successfully registration. Then Login with “username” and “password”.



**Fig 6.2.5: MscReg Student Home Page**

Here there different button for course registration. “Add Course” used for taken course. “Drop Course” used for drop unwanted course.



**Fig 6.2.6: MscReg Add Course Page**

Here shows required course for semester. After select subject from spinner then click “Register” button.

**Chapter 7**

**Conclusion**

This project was first mobile based application which i made. By making this project i learnt basic of mobile development life cycle. It was an experience of systematically going through the project phases, planning the project and implementing the same. I worked under a wonderful guider who guide me. So overall it was greater learning experience for me.

* Includes computer and internet technology
* Eliminates traditional working procedure.
* Make the working procedure more accurate, faster and more efficient
* Gives better experience to students.

**Future of work**

* A user friendly interface in an enhanced version.
* Divide admin panel into teachers and administrator who will manage it.
* Make more protected and secure from spam, unwanted data hazard, secure user profile

**Reference**

1. Ref ( System implementation ) <http://www.uky.edu/~dsianita/695A&D/lecture5.html>
2. Ref ( Error 2 ) <http://stackoverflow.com/questions/9350269/error-java-lang-indexoutofboundsexception-invalid-index-0-size-is-0-in-my-prog>
3. Ref ( Android studio ) <https://developer.android.com/studio/index.html>
4. Ref ( SQLite ) <http://programmerguru.com/android-tutorial/what-is-sqlite/>
5. Ref ( Testing ) <https://en.wikipedia.org/wiki/Software_testing#Testing_methods>
6. WIKIPEDIA, The Free Encyclopedia, Multitier Architecture, cited on 29 April 2009, http://en.wikipedia.org/wiki/multitier\_architecture.
7. SOMMERVILLE I., Software Engineering, 7th Edition, 2004, ISBN: 0-321-21026-3, Pearson Education Limited, pp. 68.
8. KENDALL & KENDALL, Systems Analysis s and Design, International Edition, 5 Edition, 2002, ISBN: 0-13-042365-3, Pearson Education, Inc., pp. 245.
9. SPARX SYSTEMS, UML Tutorial, cited on 25th May 2009, http://www.sparxsystems.com/umltutorial.html.

**Appendix**

Gardel :

android {

compileSdkVersion 25

buildToolsVersion "25.0.0"

defaultConfig {

applicationId "com.registration.course.university"

minSdkVersion 19

targetSdkVersion 25

versionCode 1

versionName "1.0"

}

buildTypes {

release {

minifyEnabled false

proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'

}

}

}

dependencies {

compile fileTree(dir: 'libs', include: ['\*.jar'])

testCompile 'junit:junit:4.12'

compile 'com.android.support:appcompat-v7:25.0.0'

compile 'com.android.volley:volley:1.0.0'

compile 'com.android.support:design:25.0.0'

}

Activity\_login.xml

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

tools:context="com.registration.course.university.LoginActivity">

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="10dp"

android:layout\_marginRight="20dp"

android:layout\_marginLeft="20dp"

android:orientation="horizontal">

<ImageView

android:layout\_width="165dp"

android:layout\_height="165dp"

android:id="@+id/imageView"

android:background="@drawable/ju\_logo"/>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:orientation="vertical">

<TextView

android:layout\_width="200dp"

android:layout\_height="wrap\_content"

android:layout\_marginTop="20dp"

android:text="Jahangirnagar University"

android:textSize="15dp"

android:gravity="center"

android:textColor="#1f2ce6" />

<TextView

android:layout\_width="200dp"

android:layout\_height="wrap\_content"

android:layout\_marginTop="20dp"

android:text="Jahangirnagar University"

android:textSize="15dp"

android:gravity="center"

android:textColor="#1f2ce6" />

</LinearLayout>

</LinearLayout>

<TextView

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="PMSCS Program"

android:layout\_marginTop="10dp"

android:layout\_marginLeft="20dp"

android:layout\_marginRight="20dp"

android:id="@+id/etUniv"

android:textColor="#0c0c0d"

android:textSize="24dp"

android:textStyle="bold"/>

<TextView

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="Please Log in"

android:layout\_marginTop="10dp"

android:layout\_marginLeft="20dp"

android:layout\_marginRight="20dp"

android:textColor="#0c0c0d"

android:textSize="24dp"/>

<EditText

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/etUsername"

android:hint="Username"

android:textColor="#100f0f"

android:layout\_marginTop="20dp"

android:layout\_marginRight="20dp"

android:layout\_marginLeft="20dp"

android:textColorHint="#D3D3D3"

android:textSize="20sp" />

<EditText

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginRight="20dp"

android:layout\_marginLeft="20dp"

android:layout\_marginTop="20dp"

android:inputType="textPassword"

android:ems="10"

android:id="@+id/etPassword"

android:hint="Password"

android:textColor="#100f0f"

android:textColorHint="#D3D3D3"

android:textSize="20sp" />

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_marginTop="20dp"

android:layout\_marginLeft="20dp"

android:layout\_marginRight="20dp">

<Button

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_marginRight="10dp"

android:text="Login"

android:textStyle="bold"

android:id="@+id/bLogin"

android:background="@drawable/bt\_style"

android:textColor="#ffffff"

android:layout\_weight="1" />

<Button

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_marginLeft="10dp"

android:text="Create Account"

android:textStyle="bold"

android:id="@+id/tvRegisterHere"

android:background="@drawable/bt\_style"

android:textColor="#ffffff"

android:layout\_weight="1" />

</LinearLayout>

</LinearLayout>

Activity\_addcourses.xml

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:paddingBottom="@dimen/activity\_vertical\_margin"

android:paddingLeft="@dimen/activity\_horizontal\_margin"

android:paddingRight="@dimen/activity\_horizontal\_margin"

android:paddingTop="@dimen/activity\_vertical\_margin"

android:orientation="vertical"

tools:context="com.registration.course.university.AddCourses">

<CheckBox

android:id="@+id/checkBox2"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="20dp"

android:text="Structural programming language" />

<CheckBox

android:id="@+id/checkBox3"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="10dp"

android:text="Mathmatics" />

<CheckBox

android:id="@+id/checkBox1"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="10dp"

android:text="Object oriented language" />

<Button

android:id="@+id/button1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginTop="37dp"

android:layout\_gravity="center"

android:background="@drawable/bt\_style"

android:text="Save" />

</LinearLayout>

Activity\_dropcourses.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

android:orientation="vertical"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context="com.registration.course.university.DropCourses">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:textAppearance="?android:attr/textAppearanceLarge"

android:text="Drop Courses"

android:id="@+id/textView3"

android:layout\_gravity="center\_horizontal"

android:layout\_alignParentTop="true"

android:layout\_centerHorizontal="true"

android:layout\_marginTop="51dp" />

</RelativeLayout>

**LoginActivity.java**

package com.registration.course.university;

import android.content.Intent;

import android.graphics.Typeface;

import android.support.v7.app.AlertDialog;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.TextView;

import com.android.volley.RequestQueue;

import com.android.volley.Response;

import com.android.volley.toolbox.Volley;

import org.json.JSONException;

import org.json.JSONObject;

import org.w3c.dom.Text;

public class LoginActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_login);

// Font path

String fontPath = "fonts/Brandon\_reg.otf";

TextView univ = (TextView) findViewById(R.id.etUniv);

Typeface tf = Typeface.createFromAsset(getAssets(), fontPath);

univ.setTypeface(tf);

final EditText etUsername = (EditText) findViewById(R.id.etUsername);

final EditText etPassword = (EditText) findViewById(R.id.etPassword);

final Button bLogin = (Button) findViewById(R.id.bLogin);

final Button registerLink = (Button) findViewById(R.id.tvRegisterHere);

registerLink.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View view) {

Intent registerIntent = new Intent(LoginActivity.this, RegisterActivity.class);

LoginActivity.this.startActivity(registerIntent);

}

});

bLogin.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View view) {

final String username = etUsername.getText().toString();

final String password = etPassword.getText().toString();

Response.Listener<String> responseListener = new Response.Listener<String>() {

@Override

public void onResponse(String response) {

try {

JSONObject jsonResponse = new JSONObject(response);

boolean success = jsonResponse.getBoolean("success");

if(success){

String firstname = jsonResponse.getString("firstname");

Intent intent = new Intent(LoginActivity.this, UserAreaActivity.class);

intent.putExtra("firstname", firstname);

intent.putExtra("username", username);

LoginActivity.this.startActivity(intent);

}

else{

AlertDialog.Builder builder = new AlertDialog.Builder(LoginActivity.this);

builder.setMessage("Login Failed").setNegativeButton("Retry", null).create().show();

}

} catch (JSONException e) {

e.printStackTrace();

}

}

};

LoginRequest loginRequest = new LoginRequest(username, password, responseListener);

RequestQueue queue = Volley.newRequestQueue(LoginActivity.this);

queue.add(loginRequest);

}

});

}

}