1. **PRELIMINARY INVESTIGATION**

**1.1 SYNOPSIS**

**SynTech-X**

**Introduction**

Syntech-X is an inter-college festival,organized by R.D. National College’s Computer Science department.The fun-filled fest lasts two days, jampacked with pool of activities and events, ranging from technical events like:Blind Coding,Minute to Code it,Swiftkeys to Gaming events like: Counter Strike,NFS,Carrom,Table-Tennis as well as Cultural events like: Fashion Show,Dance Competitions,Lab Crap and many more.

**Scope**

R.D. National College’s Computer Science department wants an Android Application where students can download the app and can know about events in SynTech-X, history and Registration details. The users of the app would get daily notification of the Daily Cyber-Security tips and would be notified of the upcoming Events. People can view the various types of events and their details. Student who would like to participate can also signup and do pre-registration and register themselves, and have the access to directly call or email the respective events organizing members.

● Online Registeration

● Daily Notification

● Events Upadate Notification

● Commitees Details

● History of SynTech-X

● Managing Committee

● Gallery

**Modules**

● Event’s

● Commitees

● History

● Gallery

**Database Module**

● Events

● Gallery

● Commitees

**1.2 Organizational Overview:**

Syntech-X is an inter-college festival,organized by R.D. National College’s Computer Science department Bandra west.

● Syntech-X is an android application where student’s can view all the information about the Syntech-X.

● It is the next step from the R.D. National College’s Computer Science department to attract students in getting touch with the Annual Fest and take pride of it.

● Student’s can know about the Fest .

● Student’s can also register and participate.

● Student’s will be getting daily Cyber-Security tips .

● Student’s will be getting Notification from the app about new events in Syntech-X.

● Student’s can View all The events rules and regulations and the timings from the app.

**1.3 Working of the current system :**

● The present website is handled manually.

● Student’s can see the details about Syntech-X .

**1.4 Limitations of the current system:**

● Lack of security.

● Time consuming and high costing.

● Large amount of paper work is needed and difficult to handle.

**1.5 The Proposed System:**

● The proposed system gives all the information about the Syntech-X.

● App is user friendly to all ages with it’s look and feel.

● To allow user to stay connected to Syntech-X.

**1.6 Advantages of the Proposed system:**

● User friendly

● Gives all information about the Syntech-X attracting young Student’s.

**1.7 Tools and Technologies to be used:**

1) ANDROID STUDIO

2)NETBEANS

3)MYSQL

4)GlassFish Server 4.0

**Android studio:-**

Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development.It is available for download on Windows, macOS and Linux based operating systems.

The following features are provided in the current stable version:

1)Gradle-based build support

2)Android-specific refactoring and quick fixes

3)Lint tools to catch performance, usability, version compatibility and other problems

4)ProGuard integration and app-signing capabilities

5)Template-based wizards to create common Android designs and components

6)A rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations

7)Support for building Android Wear apps

8)Built-in support for Google Cloud Platform, enabling integration with Firebase Cloud Messaging (Earlier 'Google Cloud Messaging') and Google App Engine

9)Android Virtual Device (Emulator) to run and debug apps in the Android studio.

**NetBeans IDE 8.1**

NetBeans IDE is the official IDE for Java 8. With its editors, code analyzers, and converters, you can quickly and smoothly upgrade your applications to use new Java 8 language constructs, such as lambdas, functional operations, and method references.Batch analyzers and converters are provided to search through multiple applications at the same time, matching patterns for conversion to new Java 8 language constructs.With its constantly improving Java Editor, many rich features and an extensive range of tools, templates and samples, NetBeans IDE sets the standard for developing with cutting edge technologies out of the box.Design GUIs for Java SE, HTML5, Java EE, PHP, C/C++, and Java ME applications quickly and smoothly by using editors and drag-and-drop tools in the IDE. Keeping a clear overview of large applications, with thousands of folders and files, and millions of lines of code, is a daunting task. NetBeans IDE provides different views of your data, from multiple project windows to helpful tools for setting up your applications and managing them efficiently, letting you drill down into your data quickly and easily, while giving you versioning tools via Subversion, Mercurial, and Git integration out of the box.

**MySql**

MySQL is an open-source relational database management system (RDBMS). Its name is a combination of "My", the name of co-founder Michael Widenius's daughter, and "SQL", the abbreviation for Structured Query Language. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation. For proprietary use, several paid editions are available, and offer additional functionality.

MySQL is a central component of the LAMP open-source web application software stack (and other "AMP" stacks). LAMP is an acronym for "Linux, Apache, MySQL, Perl /PHP/ Python". Applications that use the MySQL database include: TYPO3, MODx, Joomla, WordPress, Simple Machines Forum, phpBB, MyBB, and Drupal. MySQL is also used in many high-profile, large-scale websites, including Google (though not for searches), Facebook, Twitter, Flickr, and YouTube.

**GlassFish Server**

GlassFish is a Java application server project created by Sun Microsystems that allows many developers to generate enterprise technologies that are convenient and scalable, as well as additional services that can be installed based on preference. It is a free, dual-licensed software under the GNU General Public License (GPL) and the Common Development and Distribution License (CDDL). GlassFish was acquired by Oracle in 2010.

GlassFish was developed based on a source code that was released by Sun and Oracle’s TopLink persistence system. The project was launched in 2005 and the first version that supported Java EE 5 was released in 2006.   
  
The reference implementation of Java EE is GlassFish, so it supports JMS, JavaServer Pages, Enterprise JavaBeans, RMI, JPA and servlets. Because of its nature, developers can create scalable and portable applications that easily integrate with legacy systems and technologies.

**1.8 Feasibility study:**

A feasibility study is an analysis of how successfully a project can be completed.

It is the initial design stage of any project, which brings together the elements of knowledge.

All activities of feasibility study are directed towards helping answer the question

"Should we proceed with the proposed project idea?"

"Does the proposed system contributes to the overall objectives for which the system was proposed for?"

**Technical feasibility:-**

Technical feasibility is concerned with specifying the equipment’s and the software to satisfy the user requirements.

The aim of technical feasibility is to support the cost of the company to undertake a technical study into:-

The system is very much feasible with its technical aspect as there is not much computing resource required to build the system.

Making changes in the system regarding updating product details can be easily done as the admin will have a complete understanding of the system's content and the tools which are used for developing the system.

**Operational feasibility:-**

In operational feasibility, we attempt to ensure that every user can access the system easily.

The ease to use the system will help to increase the operational importance of the system,as there will be not much computing expertise required to use the system and a person with minimum computing knowledge can use the system very effectively.

The proposed system will really benefit the organisation as the system could be maintained by the admin itself and there will not be requirement for any additional staff for maintaining the system.

The overall response of the system will also increase as there will be more number of users affiliated with the system in the near future.

**Economic feasibility:-**

Economic analysis is the most frequently used method for evaluating the effectiveness of a new system.

The proposed system can be developed at a minimum cost and resource.

The system can assure a good beneficial cost to the organisation.

The savings that would arise from the beneficial cost of the system can be used to improve the system's performance in future.

**CHP 2-SYSTEM ANALYSIS**

* Event Table
* Entity Relationship Diagram
* Class Diagram
* Object Diagram
* Use Case Diagram
* Activity Diagram
* State Chart Diagram
* Sequence Diagram

**2.1 Event Table**

Events are objects or messages used when a software components wants to notify a state change to other components.

An Event model is a software architecture (a set of classes and interfaces)that determines how components occur.

On the event source side:-

* create and describe events
* trigger (or fire)events
* distribute events to interested components

On the event listener side:-

* subscribe to event sources
* react to events when received
* remove the subscription to event sources when desired

Terminology often used refers to:-

* Event Source or Provider:-the sender of events
* Event:-the object sent
* Event Listener or Event Sink or Consumer:-the receiver of events

**For Admin:-**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **EVENT** | **TRIGGER** | **SOURCE** | **RESONSE** | **DESTINATION** |
| Admin logs in | Admin panel | Admin | Confirmed as admin | Admin |
| Admin adds new event | Event details | Admin | Event added | Admin |
| Admin removes Event | Event details | Admin | Event removed | Admin |
| Admin updates Event details | New details | Admin | Event details updated | Admin |
| Feedback from user | Customer queries | Admin | Queries successfully viewed | Admin |

**For User :-**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **EVENT** | **TRIGGER** | **SOURCE** | **RESONSE** | **DESTINATION** |
| Member visits the homepage | Homepage | Registered user | Display homepage | Registered user |
| Member logs in | Login page | Registered user | Successfully logged in | Registered user |
| Member searches for a Event | Event list | Registered user | Display Event | Registered user |
| Member selects a Event to view | Event description | Registered user | Event details displayed | Registered user |
| Member logs out | Logout | Registered user | Homepage | Registered user |

**2.2 Entity Relationship Diagram**

In software engineering,an entity relationship model(ER model)is a data model for describing the data or information aspects of a business domain or its process requirements,in an abstract way that leads itself to ultimately being implemented in a database such as a relational database. The main components of ER models are entities (things) and the relationships that can exist among them.

An entity- relationship model is a systematic way of describing and defining a business process.The process is modelled as components (entities) that are linked with each other by relationships that express the dependencies and requirements between them,such as: one building may be divided into zero or more apartments,but one apartment can only be located in one building. Entities may have various properties (attributes) that characterize them.Diagram created to represent these entities, attributes and relationships graphically are called entity -relationship diagrams.

An ER model is typically implemented as a database.In the case of a relational database,which stores data in tables,every row of each table represents one instance of an entity.Some data fields in these tables point to indexes in other tables;such pointers represent the relationship.

**Limitations:-**

* ER models assume information content that can readily be represented in a relational database.They describe only a relational structure for this information.
* They are inadequate for systems in which the information cannot readily be represented in relational form,such as with semi-structured data

**Diagram:-**



**2.3 Class Diagram**

The class diagram is a static diagram.It represents the static view of an application.Class diagram is not only used for visualising, describing and documenting different aspects of a system but also for constructing executable code of the software application

The class diagram describes the attributes and operations of a class and also the constraints imposed on the system.The class diagrams are widely used in the modelling of object oriented systems because they are the only UML diagrams which can be mapped directly with object oriented languages.

The class diagram shows a collection of classes,interfaces, associations, collaborations and constraints. It is also known as a structural diagram.

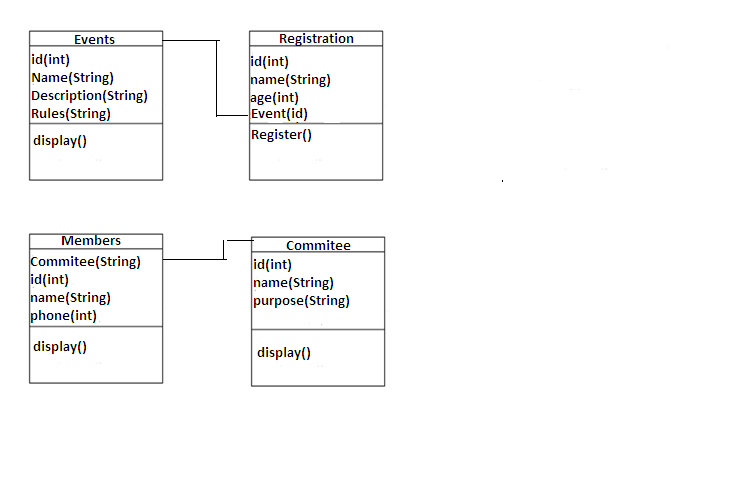
**Purpose**:-

The purpose of the class diagram is to model the static view of an application.The class diagrams are the only diagrams which can be directly mapped with object oriented languages and thus widely used at the time of construction.

The UML diagrams like activity diagram, sequence diagram can only give the sequence flow of the application but class diagram is a bit different.So it is the most popular UML diagram in the coder community.

So the purpose of the class diagram can be summarised as :-

* Analysis and design of the static view of an application.
* Describe responsibilities of a system.
* Base for component and deployment diagrams.
* Forward and reverse engineering.



**2.4 Object Diagram**

Object diagrams are derived from class diagrams so object diagrams are dependent upon class diagrams.

Object diagrams represent an instance of class diagram.The basic concepts are similar for class diagrams and object diagrams.Object diagrams also represent the static view of a system but this static view is a snapshot of the system at a particular moment.

Object diagrams are used to render a set of objects and their relationships as an instance.

**Purpose**:-

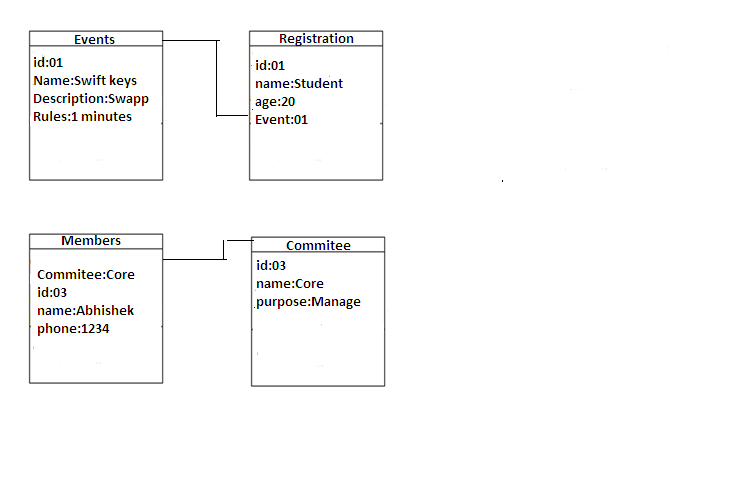
The purpose of a diagram should be understood clearly to implement it practically.The purpose of object diagrams are similar to class diagrams.

The difference is that a class diagram represents an abstract model consisting of classes and their relationships.But an object diagram represents an instance at a particular moment which is concrete in nature.

It means the object diagram is more close to the actual system behaviour.The purpose is to capture the static view of a system at a particular moment.

So the purpose of the object diagram can be summarised as:-

* Forward and reverse engineering
* Object relationships of a system
* Static view of an interaction
* Understand object behaviour and their relationship from practical perspective.



**2.5 Use Case Diagram**

To model a system the most important aspect is to capture the dynamic behaviour.To clarify a bit in details, dynamic behaviour means the behaviour of the system when it is running/operating.

So only static behaviour is not sufficient to model a system rather dynamic behaviour is more important than static behaviour.Now as we have to discuss that the use case diagram is dynamic in nature there should be some internal or external factors for making the interaction.

These internal and external agents are known as actors.So use case diagrams consists of actors,use cases and their relationships.The diagram is used to model the system/subsystem of an application.A single use case diagram captures a particular functionality of a system.

So to model the entire system numbers of use case diagrams are used.

**Purpose:-**

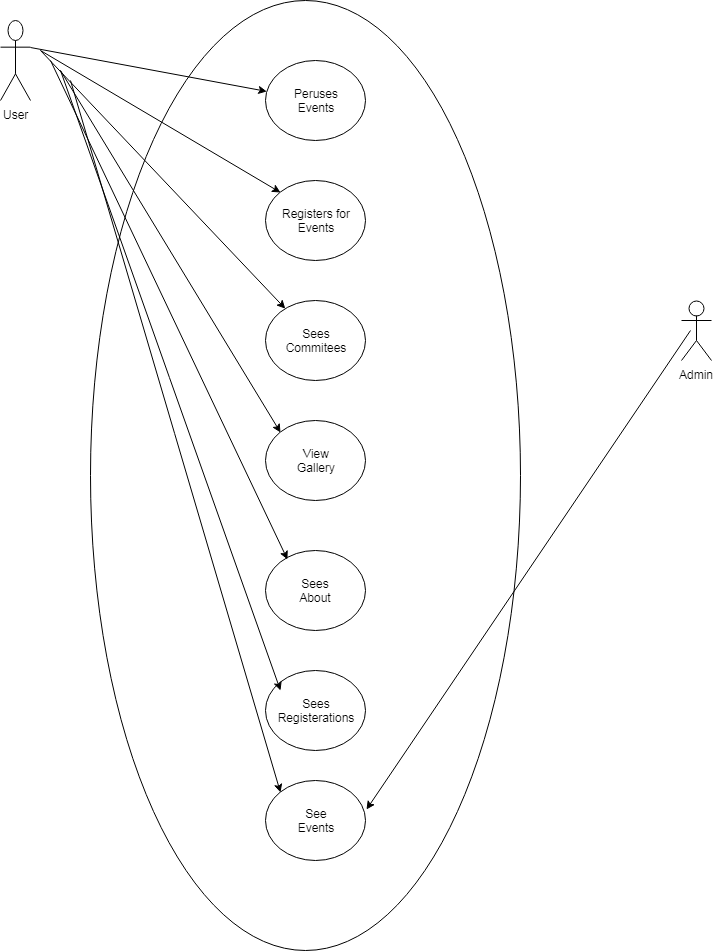
The purpose of use case diagram is to capture the Dayna aspect of a system.But this definition is too generic to describe the purpose.

Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements.So when a system is analysed to gather it's functionalities use cases are prepared and actors are identified.

Now when the initial task is complete use case diagrams are modelled to present the outside view.

So in brief the purpose of use case diagram can be as follows:-

* Used to gather requirements of a system.
* Used to get an outside view of a system.
* Identify external and internal factors influencing the system.
* Show the interacting among the requirements are actors.



**2.6 Activity Diagram**

Activity diagram is another important diagram in UML to describe dynamic aspects of the system.

Activity diagram is basically a flow chart to represent the flow from one activity to another activity.The activity can be described as an operation of the system.

So the control flow is drawn from one operation to another.This flow can be sequential,branched or concurrent.Activity diagram deals with all type of flow control by using different elements like fork,join etc.

**Purpose:-**

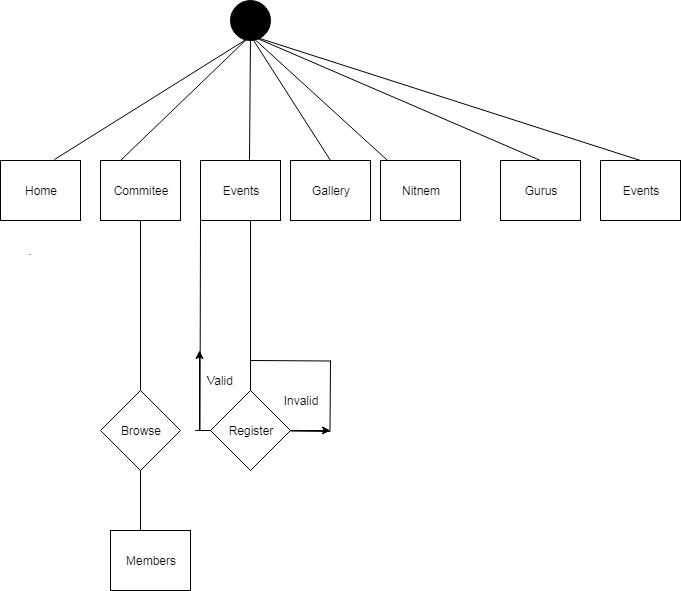
It captures the dynamic behaviour of the system.Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing dynamic nature of a system but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in activity diagram is the message part.

It does not show any message flow from one activity to another. Activity diagram is some time considered as the flow chart. Although the diagram looks like a flow chart but it is not. It shows different flow like parallel,branched,concurrent and single.

So the purpose can be described as:

* Draw the activity flow of system.
* Describe the sequence from one activity to another.
* Describe the parallel,branched and concurrent flow of the system.



**2.7 State Chart Diagram**

The name of the diagram itself clarifies the purpose of the diagram and other details. It describes different states of a component in a system. The states are specific to a component/object of a system.

A State Chart Diagram describes a state machine. Now to clarify it state machine can be defined as a machine which defines different states of an object and these states are controlled by external or internal events.

As State Chart Diagram defines states it is used to model lifetime of an object.

**Purpose:-**

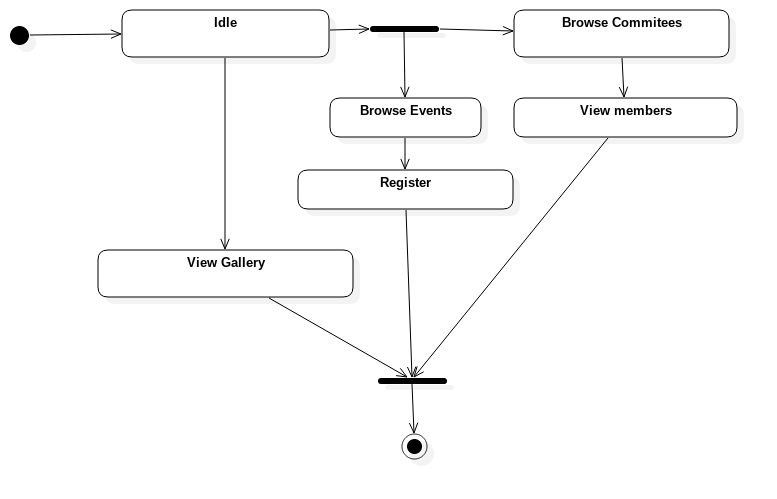
State Chart Diagram is one of the five UML diagrams used to model dynamic nature of a system.They define different states of an object during its lifetime. And these states are changed by events. So State Chart Diagrams are useful to model reactive systems. Reactive systems can be defined as a system that responds to external or internal events.

State Chart Diagram describes the flow of control from one state to another state. States are defined as a condition in which an object exists and it changes when some event is triggered.So the most important purpose of State Chart Diagram is to model life time of an object from creation to termination.

State Chart Diagram are also used for forward and reverse engineering of a system.But the main purpose is to model reactive system.

Following are the main purpose of using State Chart Diagrams:-

* To model dynamic aspect of a system.
* To model life time of a reactive system.
* To describe different states of an object during its life time.
* Define a state machine to model of an object.

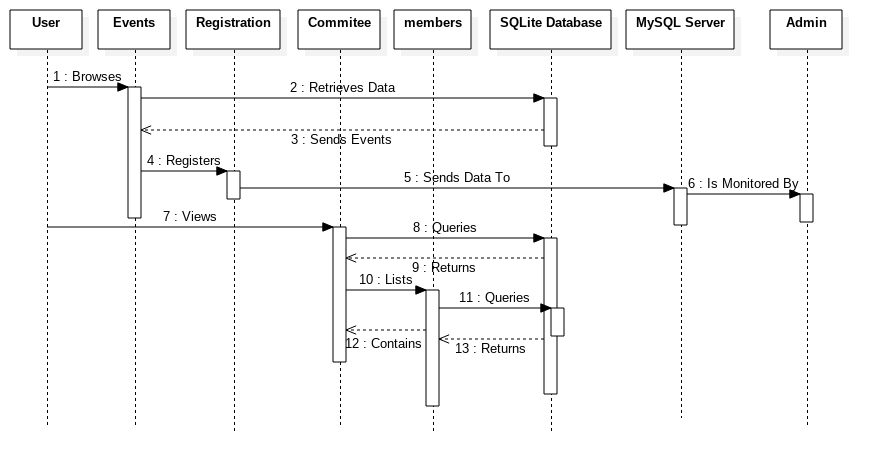


**2.8 Sequence Diagram**

A Sequence Diagram is an interaction diagram that shows how processes operate with one another and what is their order. It is a construct of a Message Sequence Chart. A Sequence Diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence Diagrams are sometimes called event diagrams or event scenarios.

A Sequence Diagram shows parallel vertical lines(lifelines),different processes or objects that live simultaneously and as horizontal arrows,the messages exchanged between them,in order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

If the lifeline is that of an object,it demonstrates a role.Leaving the instance name blank can represent anonymous and unnamed instances.Messages,written with horizontal arrows with the message name written above them,display interaction.Solid arrow heads represent synchronous calls,open arrow heads represent asynchronous messages, and dashed lines represent reply messages.If a caller sends a synchronous message,it must wait until the message is done,such as invoking a subroutine. If a caller sends an asynchronous message,it can continue processing and doesn’t have to wait for a response.Asynchronous calls are present in multithreaded applications and in message-oriented middleware.Activation boxes,or method-call boxes,are opaque rectangles drawn on top of lifelines to represent that processes are being performed response to the message(Execution Specification in UML).



**CHP 3 :- System Diagram**

* Component Diagram
* Package Diagram
* System Flow Chart
* Structured Chart
* Deployment Diagram

**3.1 Component Diagram**

Component Diagram are different in terms of nature and behaviour.Component diagrams are used to model physical aspects of a system.

Component diagrams are used to visualize the organization and relationships among components in a system.These diagrams are also used to make executable systems.

**Purpose:-**

Component diagram is a special kind of diagram in UML.The purpose is also different from all other diagrams discussed so far. It does not describe the functionality of the system but it describes the component used to make those functionalities

So from that point component diagrams are used to visualize the physical components in a system.These components are libraries,packages,files,etc.

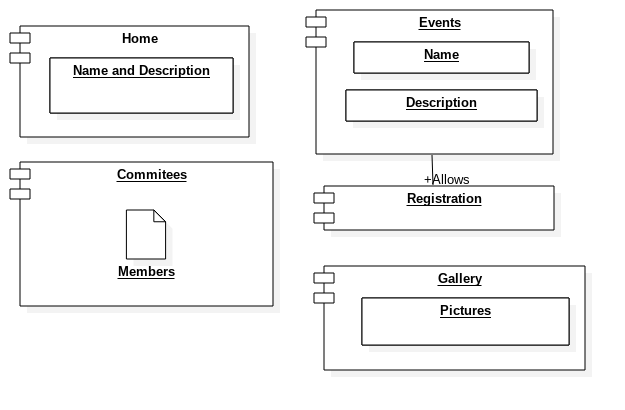
Component diagrams can also be described as a static implementation view of system.Static implementation represents the organization of the components at a particular moment.

A single Component Diagram cannot represent the entire system but a collection of diagrams are used to represent the whole.

So the purpose of the component diagram can be summarized as:

* Visualize the components of a system
* Construct executable by using forward and reverse engineering
* Describe the organization and relationships of the components.

**Diagram:-**



**3.2 Package Diagram**

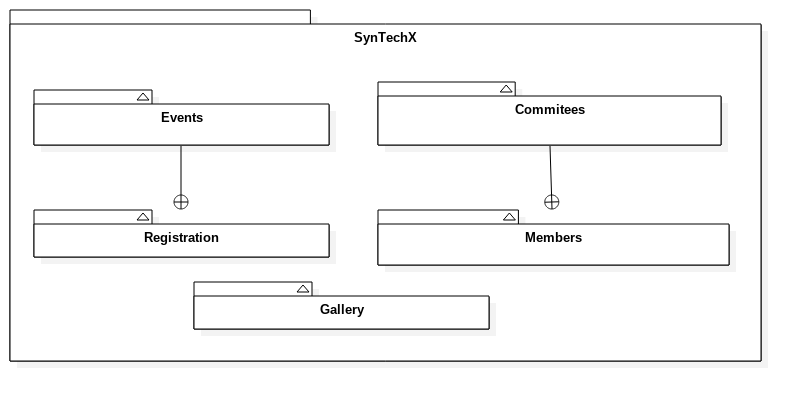
When modellinga large scale system,you would probably be working with a high volume of model elements. They describe a model from different views and different phases,hence are in different types.

UML package helps to organize and arrange model elements and diagrams into logical groups,through which you can manage a chunk of project data together.

You can also use packages to represent different views of the systems architecture .In addition,developers can use package to model the physical package or namespace structure of the application to build.

Package Diagram visualizes packages and depicts the dependency, Import,access,generalization,realization and merge relationships between them.Package diagram enables you to gain a high level understandingof the collaboration among model elements through analysing the relationships among their parent package. also helps explain the systems architecture from a broad view.

**Diagram:-**



**3.3 System Flow Chart Diagram**

System Flow Charts are a way of displaying how data flows in a system and how decisions are made to control events.

To illustrate this, symbols are use .They are connected together to show what happens to data and where it goes. The basic ones include:symbols used in flow charts.

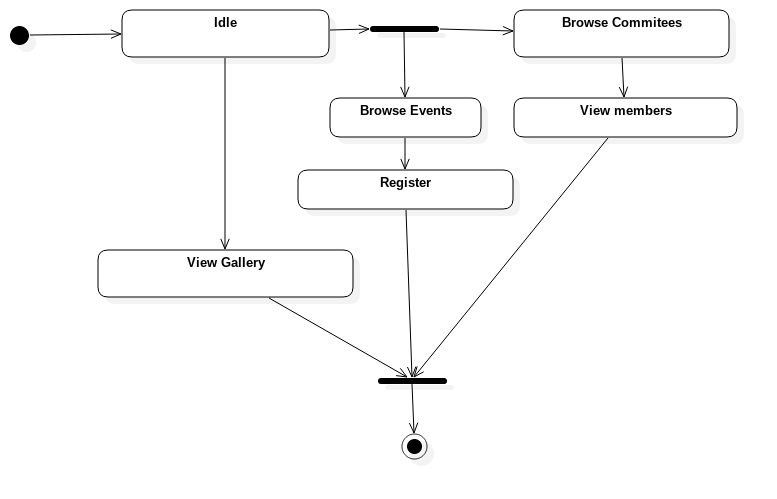
The flow of data generally goes from top to bottom and left to right and depicts the sequence of processing steps along these data lines.

The following are examples of some of the symbols used in system flowcharts: A system flowchart shows the key input and output associated with the program.The shape of symbols indicate the type of input or output devices.

The type of diagram dictates the flow chart symbols that are used. The terminator symbols marks the starting or ending point of the system.

A flow chart is a formalized graphic representation of program logic sequence,work or manufacturing process,organization.

A graphical representation of the sequence of operations in an information system or program .Information system flow charts show how data flows from source



**3.4 Structure Chart Diagram**

A structure chart (SC) in software engineering and organization theory,is a chart which shows the breakdown of a system to its lowest manageable levels.

They are used in structured programming to arrange modules into a tree.Each module is represented by a box, which contains the module’s name.A structured chart(SC) in software engineering and organizational theory ,is chart which shows the breakdown of a system to its lowest manageable levels.

They are used in structured programming to arrange program modules into tree.Each module is represented by a box,which contains the module’s name.

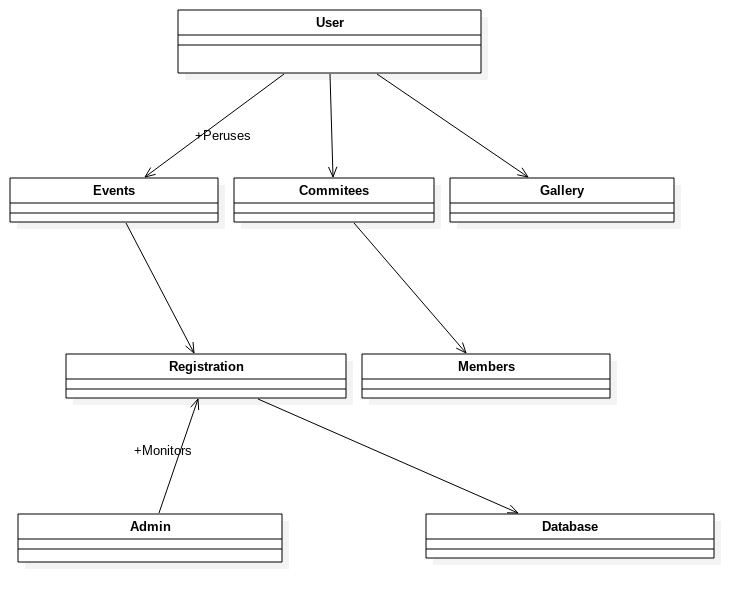
Structure diagram is a chart derived from data flow chart diagram.The system structure chart represents hierarchical structure of modules.

A structure chart depicts

The size and complexity of the system, and number of readily identifiable functions and modules within each functions and whether each identifiable function is a manageable entity or should be broken down into smaller components.

A structure chart is also used to diagram associated elements that comprise a run stream or thread. It is often developed as a hierarchal

Diagram,but other representations are allowable.



**3.5 Deployment Diagram**

Deployment diagrams are used to visualize the topology of the physical components of a system where the software components are deployed.

So deployment diagrams are used to describe the static deployment view of a system.Deployment diagrams consist of nodes and their relationships.

**Purpose:-**

The name Deployment itself describes the purpose of the diagram.Deployment diagrams are used for describing the hardware components where software components are deployed.Component diagrams and deployment diagrams are closely related.

Component diagrams are used to describe the components and deployment diagrams shows how they are deployed in hardware.

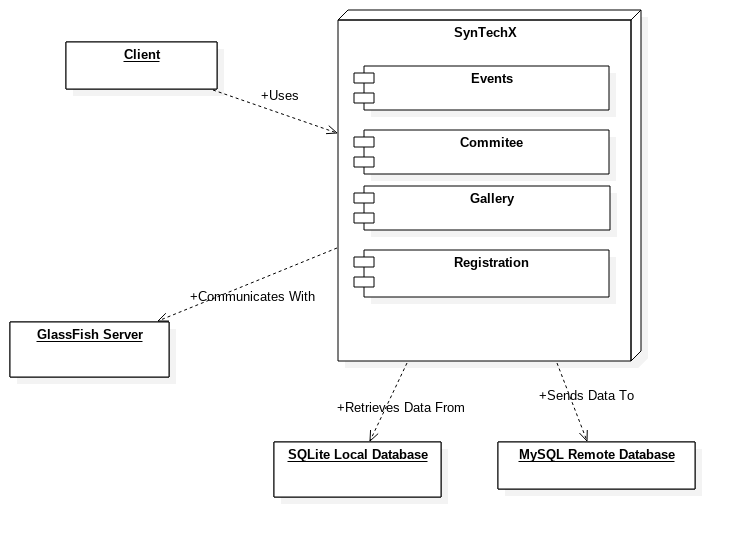
UML is mainly designed to focus on software artefacts of a system.But these two diagrams are special diagrams used to focus on software components and hardware components.

So most of the UML diagrams are used to handle logical components but deployment are made to focus on hardware topology of a system.Deployment diagrams are used by the system engineers.

It can be described as follows:-

* Visualize hardware topology of a system.
* Describe the hardware components used to deploy software components.
* Describe runtime processing nodes

**Diagram:-**



**CHAPTER 4: SYSTEM CODING**

4.1 Site Map

4.2 Data Dictionary

4.3 Source Code

**4.1 SITE MAP**

● Online Registeration

● Daily Notification

● Events Upadate Notification

● Commitees Details

● History of SynTech-X

● Managing Committee

● Gallery

**Modules**

● Event’s

● Commitees

● History

● Gallery

**Database Module**

● Events

● Gallery

● Commitees

**4.2 DATA DICTIONARY**

**TABLES:**

**Register**

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Key** |
| uname | varchar(20) |  |
| phone | bigint(10) |  |
| email | varchar(50) | primary key |
| address | varchar(150) |  |
| pwd | varchar(10) |  |
| ques | varchar(40) |  |
| ans | varchar(30) |  |

**Events**

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Key** |
| id | varchar(20) |  |
| name | Varchar(20) |  |
| desc | varchar(30) |  |
| date | longblob |  |
| loc | Varchar(30) |  |
|  |  |  |
|  |  |  |

**Admin**

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Key** |
| Name | varchar(20) |  |
| pass | varchar(20) |  |

**4.3 Source code**

**Homepage.xml**

*#commitee\_item\_layout.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"*

*android:orientation="horizontal"*

*android:layout\_width="match\_parent"*

*android:layout\_height="match\_parent"*

*android:paddingLeft="48dp"*

*android:paddingRight="48dp"*

*android:paddingTop="16dp"*

*android:paddingBottom="16dp">*

*<ImageView*

*android:layout\_width="0dp"*

*android:layout\_weight="1"*

*android:layout\_height="50dp"*

*android:src="@drawable/ic\_gallery"*

*android:id="@+id/commitee\_item\_logo"*

*android:layout\_gravity="center\_vertical"*

*android:layout\_margin="5dp"*

*android:scaleType="fitCenter"*

*/>*

*<TextView*

*android:layout\_width="0dp"*

*android:layout\_weight="7"*

*android:layout\_height="match\_parent"*

*android:textSize="20sp"*

*android:text="Commitee"*

*android:textColor="@color/content"*

*android:paddingStart="16dp"*

*android:id="@+id/commitee\_item\_text"*

*android:clickable="true"*

*android:focusable="true"*

*android:gravity="center\_vertical"*

*android:background="?attr/selectableItemBackground"*

*/>*

*</LinearLayout>*

*#activity\_event\_page.xml*

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

*<android.support.design.widget.CoordinatorLayout 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"*

*tools:context="com.antimony.scriptoo.syntechx.EventActivity"*

*android:fitsSystemWindows="true"*

*>*

*<android.support.design.widget.AppBarLayout*

*android:id="@+id/event\_app\_bar"*

*android:layout\_width="match\_parent"*

*android:layout\_height="250dp"*

*android:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar"*

*>*

*<android.support.design.widget.CollapsingToolbarLayout*

*android:layout\_width="match\_parent"*

*android:layout\_height="match\_parent"*

*app:layout\_scrollFlags="scroll|exitUntilCollapsed"*

*>*

*<ImageView*

*android:layout\_width="match\_parent"*

*android:layout\_height="match\_parent"*

*android:scaleType="centerCrop"*

*app:layout\_collapseMode="parallax"*

*android:id="@+id/event\_page\_header"*

*/>*

*<android.support.v7.widget.Toolbar*

*android:layout\_width="match\_parent"*

*android:layout\_height="?attr/actionBarSize"*

*android:id="@+id/event\_page\_toolbar"*

*app:layout\_collapseMode="pin"*

*/>*

*</android.support.design.widget.CollapsingToolbarLayout>*

*</android.support.design.widget.AppBarLayout>*

*<include layout="@layout/content\_event\_page" />*

*<android.support.design.widget.FloatingActionButton*

*android:id="@+id/fab"*

*android:layout\_width="wrap\_content"*

*android:layout\_height="wrap\_content"*

*android:layout\_margin="@dimen/fab\_margin"*

*app:layout\_anchor="@id/event\_app\_bar"*

*app:layout\_anchorGravity="bottom|end"*

*android:onClick="register"*

*app:srcCompat="@drawable/ic\_register" />*

*</android.support.design.widget.CoordinatorLayout>*

*#header\_commitees\_listview.xml*

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

*<RelativeLayout*

*xmlns:android="http://schemas.android.com/apk/res/android" android:layout\_width="match\_parent"*

*android:layout\_height="match\_parent">*

*<ImageView*

*android:layout\_width="match\_parent"*

*android:layout\_height="200dp"*

*android:src="@drawable/header\_commitees"*

*android:scaleType="centerCrop"*

*android:id="@+id/blahblah"*

*/>*

*<TextView*

*android:layout\_width="wrap\_content"*

*android:layout\_height="wrap\_content"*

*android:text="COMMITEES"*

*android:textColor="@color/white"*

*android:layout\_alignBottom="@+id/blahblah"*

*android:layout\_centerInParent="true"*

*android:layout\_marginBottom="30dp"*

*android:textSize="20sp"*

*/>*

*</RelativeLayout>*

*#untitled.py*

*import os*

*content = ""*

*files = [f for f in os.listdir('.') if os.path.isfile(f)]*

*print("\n")*

*for f in files:*

*file = open(f, "r")*

*print(f)*

*content+="#"+f+"\n\n"*

*content+=file.read()*

*content+="\n\n"*

*file.close()*

*final = open("SourceCode.txt", "w")*

*final.write(content)*

*final.close()*

*#activity\_commitee.xml*

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

*<ScrollView 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"*

*tools:context="com.antimony.scriptoo.syntechx.CommiteeActivity"*

*android:background="@drawable/commitee\_activity\_background"*

*>*

*<LinearLayout*

*android:layout\_width="match\_parent"*

*android:layout\_height="wrap\_content"*

*android:orientation="vertical"*

*>*

*<android.support.v7.widget.Toolbar*

*android:layout\_width="match\_parent"*

*android:layout\_height="?attr/actionBarSize"*

*android:background="@android:color/transparent"*

*android:id="@+id/commitee\_activity\_toolbar"*

*/>*

*<LinearLayout*

*android:layout\_width="match\_parent"*

*android:layout\_height="wrap\_content"*

*android:id="@+id/commitee\_page\_relative\_layout"*

*android:orientation="vertical"*

*android:padding="16dp"*

*>*

*<TextView*

*android:layout\_width="match\_parent"*

*android:layout\_height="wrap\_content"*

*android:id="@+id/commitee\_page\_title"*

*android:text="Title"*

*android:textColor="@color/title"*

*android:textSize="50sp"*

*android:textStyle="bold"*

*android:paddingBottom="16dp"*

*android:paddingLeft="6dp"*

*/>*

*<android.support.v7.widget.CardView*

*android:layout\_width="match\_parent"*

*android:layout\_height="wrap\_content"*

*android:layout\_margin="6dp"*

*app:cardCornerRadius="5dp"*

*android:elevation="10dp"*

*android:id="@+id/commitee\_page\_role\_card"*

*>*

#EventsFragment.java

package com.antimony.scriptoo.syntechx;

import android.os.Bundle;

import android.support.annotation.Nullable;

import android.support.v4.app.Fragment;

import android.support.v7.widget.Toolbar;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.ListView;

import java.util.ArrayList;

/\*\*

\* Created by antimony on 12/10/17.

\*/

public class EventsFragment extends Fragment {

Toolbar toolbar;

ListView listView;

ArrayList<EventClass> array;

// NestedScrollView nestedScrollView;

public static EventsFragment getInstance(){

EventsFragment fragment = new EventsFragment();

return fragment;

}

@Override

public void onActivityCreated(@Nullable Bundle savedInstanceState) {

super.onActivityCreated(savedInstanceState);

listView = (ListView)getActivity().findViewById(R.id.events\_fragment\_listview);

DatabaseHelper db = new DatabaseHelper(getContext());

array = db.getEvents();

db.close();

EventsAdapter adapter = new EventsAdapter(getContext(), array);

listView.setAdapter(adapter);

((MainNavActivity)getActivity()).setAttributes(

getResources().getColor(R.color.grey),

getResources().getColor(R.color.grey),

"Events"

);

}

@Nullable

@Override

public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {

return inflater.inflate(R.layout.fragment\_events, container, false);

}

}

#HomeFragment.java

package com.antimony.scriptoo.syntechx;

import android.content.Intent;

import android.net.Uri;

import android.os.Bundle;

import android.support.annotation.Nullable;

import android.support.v4.app.Fragment;

import android.support.v7.widget.Toolbar;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.Button;

/\*\*

\* Created by antimony on 12/5/17.

\*/

public class HomeFragment extends Fragment {

public static HomeFragment getInstance(){

HomeFragment fragment = new HomeFragment();

return fragment;

}

@Override

public void onActivityCreated(@Nullable Bundle savedInstanceState) {

super.onActivityCreated(savedInstanceState);

((MainNavActivity)getActivity()).setAttributes(

getResources().getColor(R.color.colorPrimary),

getResources().getColor(R.color.colorPrimaryDark),

"SynTech-X"

);

}

@Nullable

@Override

public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {

return inflater.inflate(R.layout.fragment\_home, container, false);

}

}

#CommiteeClass.java

package com.antimony.scriptoo.syntechx;

import android.database.Cursor;

import android.provider.ContactsContract;

import java.io.Serializable;

import java.util.ArrayList;

/\*\*

\* Created by antimony on 12/11/17.

\*/

public class CommiteeClass implements Serializable{

String name, role, image, id;

public CommiteeClass(Cursor c){

id = Integer.toString(c.getInt(c.getColumnIndex(DatabaseHelper.COMMITEE\_ID)));

// id = "5";

name = c.getString(c.getColumnIndex(DatabaseHelper.COMMITEE\_NAME));

role = c.getString(c.getColumnIndex(DatabaseHelper.COMMITEE\_ROLE));

image = c.getString(c.getColumnIndex(DatabaseHelper.COMMITEE\_LOGO));

}

}

#DialogMemberDetails.java

package com.antimony.scriptoo.syntechx;

import android.content.Intent;

import android.net.Uri;

import android.os.Bundle;

import android.support.annotation.Nullable;

import android.support.v7.app.AppCompatActivity;

import android.support.v7.widget.DialogTitle;

import android.util.Log;

import android.view.View;

import android.view.Window;

import android.widget.TextView;

import android.widget.Toast;

import com.nostra13.universalimageloader.core.ImageLoader;

import de.hdodenhof.circleimageview.CircleImageView;

/\*\*

\* Created by antimony on 12/17/17.

\*/

public class DialogMemberDetails extends AppCompatActivity {

Intent recievedIntent;

TextView name;

CircleImageView dp;

DialogTitle title;

String number, email;

@Override

protected void onCreate(@Nullable Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.member\_details\_card\_dialog);

recievedIntent = getIntent();

name = findViewById(R.id.member\_dialog\_name);

dp = findViewById(R.id.member\_dialog\_image);

setTitle(recievedIntent.getStringExtra("EVENTNAME"));

name.setText(recievedIntent.getStringExtra("MEMBERNAME"));

ImageLoader imageLoader = ImageLoader.getInstance();

imageLoader.displayImage(recievedIntent.getStringExtra("IMAGEURL"), dp);

number = "tel:" + recievedIntent.getStringExtra("MEMBERNUMBER");

email = "mailto:" + recievedIntent.getStringExtra("MEMBEREMAIL");

}

public void callLaga(View v){

if(recievedIntent.getStringExtra("MEMBERNUMBER")!=null) {

Intent i = new Intent(Intent.ACTION\_DIAL);

i.setData(Uri.parse(number));

startActivity(i);

}else {

Toast.makeText(this, "Number unavailable", Toast.LENGTH\_SHORT).show();

}

}

public void emailKar(View v){

if(recievedIntent.getStringExtra("MEMBEREMAIL")!=null) {

Intent i = new Intent(Intent.ACTION\_SENDTO);

i.setData(Uri.parse(email));

startActivity(i);

}else{

Toast.makeText(this, "Email unavailable", Toast.LENGTH\_SHORT).show();

}

}

}

#EventClass.java

package com.antimony.scriptoo.syntechx;

import android.database.Cursor;

import com.antimony.scriptoo.syntechx.DatabaseHelper;

import java.io.Serializable;

import java.util.HashMap;

import java.util.Map;

/\*\*

\* Created by antimony on 12/10/17.

\*/

public class EventClass implements Serializable {

public String name, rules, description, timing, date, location, background;

public EventClass(Cursor c){

name = c.getString(c.getColumnIndex(DatabaseHelper.EVENT\_NAME));

background = c.getString(c.getColumnIndex(DatabaseHelper.EVENT\_BACKGROUND));

rules = c.getString(c.getColumnIndex(DatabaseHelper.EVENT\_RULES));

description = c.getString(c.getColumnIndex(DatabaseHelper.EVENT\_DESCRIPTION));

timing = c.getString(c.getColumnIndex(DatabaseHelper.EVENT\_TIMINGS));

date = c.getString(c.getColumnIndex(DatabaseHelper.EVENT\_DATE));

location = c.getString(c.getColumnIndex(DatabaseHelper.EVENT\_LOCATION));

}

}

#untitled.py

import os

content = ""

files = [f for f in os.listdir('.') if os.path.isfile(f)]

print("\n")

for f in files:

file = open(f, "r")

print(f)

content+="#"+f+"\n\n"

content+=file.read()

content+="\n\n"

file.close()

final = open("SourceCode.txt", "w")

final.write(content)

final.close()

#EventsAdapter.java

package com.antimony.scriptoo.syntechx;

import android.content.Context;

import android.content.Intent;

import android.support.annotation.NonNull;

import android.support.annotation.Nullable;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.ArrayAdapter;

import android.widget.ImageView;

import android.widget.RelativeLayout;

import android.widget.TextView;

import com.nostra13.universalimageloader.core.ImageLoader;

import java.util.ArrayList;

public class EventsAdapter extends ArrayAdapter<EventClass> {

private static class ViewHolder{

TextView name;

ImageView image;

RelativeLayout root;

}

public EventsAdapter(Context context, ArrayList<EventClass> events){

super(context, 0, events);

}

@NonNull

@Override

public View getView(final int position, @Nullable View convertView, @NonNull ViewGroup parent) {

final EventClass event = getItem(position);

//GET EVENT FROM ARRAYADAPTER

ViewHolder viewHolder;

if(convertView == null){

viewHolder = new ViewHolder();

convertView = LayoutInflater.from(getContext()).inflate(R.layout.event\_card, parent, false);

viewHolder.name = convertView.findViewById(R.id.event\_card\_title);

viewHolder.image = convertView.findViewById(R.id.events\_card\_background);

viewHolder.root = convertView.findViewById(R.id.event\_card\_root);

convertView.setTag(viewHolder);

}else{

viewHolder = (ViewHolder) convertView.getTag();

}

//INFLATE THE VIEW

viewHolder.name.setText(event.name);

ImageLoader imageLoader = ImageLoader.getInstance();

imageLoader.displayImage(event.background, viewHolder.image);

viewHolder.root.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent i = new Intent(getContext(), EventActivity.class);

i.putExtra("EVENTOBJECT", event);

getContext().startActivity(i);

}

});

return convertView;

}

}

#NotificationService.java

package com.antimony.scriptoo.syntechx;

import android.app.NotificationManager;

import android.content.Context;

import android.support.v4.app.NotificationCompat;

import android.support.v4.app.NotificationManagerCompat;

import android.util.Log;

import com.google.firebase.messaging.FirebaseMessagingService;

import com.google.firebase.messaging.RemoteMessage;

/\*\*

\* Created by antimony on 12/20/17.

\*/

public class NotificationService extends FirebaseMessagingService {

private static final String TAG = "FCM Service";

@Override

public void onMessageReceived(RemoteMessage remoteMessage) {

NotificationCompat.Builder mBuilder = new NotificationCompat.Builder(this);

mBuilder.setSmallIcon(R.drawable.finance);

mBuilder.setContentTitle(remoteMessage.getNotification().getTitle());

mBuilder.setContentText(remoteMessage.getNotification().getBody());

NotificationManager nmanager = (NotificationManager) getSystemService(Context.NOTIFICATION\_SERVICE);

nmanager.notify(0, mBuilder.build());

Log.d(TAG, "From: " + remoteMessage.getFrom());

Log.d(TAG, "Notification Message Body: " + remoteMessage.getNotification().getBody());

}

}

#GalleryFragment.java

package com.antimony.scriptoo.syntechx;

import android.os.Bundle;

import android.support.annotation.Nullable;

import android.support.v4.app.Fragment;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.view.animation.Animation;

import android.view.animation.AnimationUtils;

import android.widget.ImageView;

import android.widget.ViewFlipper;

import com.nostra13.universalimageloader.core.ImageLoader;

/\*\*

\* Created by antimony on 12/17/17.

\*/

public class GalleryFragment extends Fragment {

public ViewFlipper simpleViewFlipper;

public ViewFlipper simpleViewFlipper2;

public ViewFlipper simpleViewFlipper3;

public Animation in, out;

public void members(){

// get The references of ViewFlipper

simpleViewFlipper = (ViewFlipper) getActivity().findViewById(R.id.view\_flipper\_member); // get the reference of ViewFlipper

String[] Members = getResources().getStringArray(R.array.Members);

// loop for creating ImageView's

for (int i = 0; i < Members.length; i++) {

// create the object of ImageView

ImageLoader imageLoader = ImageLoader.getInstance();

ImageView imageView = new ImageView(getContext());

imageLoader.displayImage(Members[i], imageView);

simpleViewFlipper.addView(imageView);

}

// Declare in and out animations and load them using AnimationUtils class

simpleViewFlipper.setInAnimation(in);

simpleViewFlipper.setOutAnimation(out);

simpleViewFlipper.setFlipInterval(3000);

simpleViewFlipper.setAutoStart(true);

simpleViewFlipper.startFlipping();

}

public void flipper16(){

// get The references of ViewFlipper

simpleViewFlipper = (ViewFlipper) getActivity().findViewById(R.id.view\_flipper\_16); // get the reference of ViewFlipper

String[] y2016 = getResources().getStringArray(R.array.y2016);

// loop for creating ImageView's

for (int i = 0; i < y2016.length; i++) {

// create the object of ImageView

ImageLoader imageLoader = ImageLoader.getInstance();

ImageView imageView = new ImageView(getContext());

imageLoader.displayImage(y2016[i], imageView);

simpleViewFlipper.addView(imageView);

}

// Declare in and out animations and load them using AnimationUtils class

simpleViewFlipper.setInAnimation(in);

simpleViewFlipper.setOutAnimation(out);

simpleViewFlipper.setFlipInterval(3000);

simpleViewFlipper.setAutoStart(true);

simpleViewFlipper.startFlipping();

}

public void flipper15(){

// get The references of ViewFlipper

simpleViewFlipper2 = (ViewFlipper) getActivity().findViewById(R.id.view\_flipper\_15); // get the reference of ViewFlipper

String[] y2015 = getResources().getStringArray(R.array.y2015);

// loop for creating ImageView's

for (int i = 0; i < y2015.length; i++) {

// create the object of ImageView

ImageLoader imageLoader = ImageLoader.getInstance();

ImageView imageView = new ImageView(getContext());

imageLoader.displayImage(y2015[i], imageView);

simpleViewFlipper2.addView(imageView);

}

// Declare in and out animations and load them using AnimationUtils class

simpleViewFlipper2.setInAnimation(in);

simpleViewFlipper2.setOutAnimation(out);

simpleViewFlipper2.setFlipInterval(3500);

simpleViewFlipper2.setAutoStart(true);

simpleViewFlipper2.startFlipping();

}

public void flipper13(){

// get The references of ViewFlipper

simpleViewFlipper2 = (ViewFlipper) getActivity().findViewById(R.id.view\_flipper\_13); // get the reference of ViewFlipper

String[] y2013 = getResources().getStringArray(R.array.y2013);

// loop for creating ImageView's

for (int i = 0; i < y2013.length; i++) {

// create the object of ImageView

ImageLoader imageLoader = ImageLoader.getInstance();

ImageView imageView = new ImageView(getContext());

imageLoader.displayImage(y2013[i], imageView);

simpleViewFlipper2.addView(imageView);

}

// Declare in and out animations and load them using AnimationUtils class

simpleViewFlipper2.setInAnimation(in);

simpleViewFlipper2.setOutAnimation(out);

simpleViewFlipper2.setFlipInterval(4000);

simpleViewFlipper2.setAutoStart(true);

simpleViewFlipper2.startFlipping();

}

public void flipper14(){

// get The references of ViewFlipper

simpleViewFlipper2 = (ViewFlipper) getActivity().findViewById(R.id.view\_flipper\_14); // get the reference of ViewFlipper

String[] y2014 = getResources().getStringArray(R.array.y2014);

// loop for creating ImageView's

for (int i = 0; i < y2014.length; i++) {

// create the object of ImageView

ImageLoader imageLoader = ImageLoader.getInstance();

ImageView imageView = new ImageView(getContext());

imageLoader.displayImage(y2014[i], imageView);

simpleViewFlipper2.addView(imageView);

}

// Declare in and out animations and load them using AnimationUtils class

simpleViewFlipper2.setInAnimation(in);

simpleViewFlipper2.setOutAnimation(out);

simpleViewFlipper2.setFlipInterval(4500);

simpleViewFlipper2.setAutoStart(true);

simpleViewFlipper2.startFlipping();

}

public static GalleryFragment getInstance(){

GalleryFragment fragment = new GalleryFragment();

return fragment;

}

@Override

public void onActivityCreated(@Nullable Bundle savedInstanceState) {

super.onActivityCreated(savedInstanceState);

((MainNavActivity)getActivity()).setAttributes(

getResources().getColor(R.color.gallery\_teal),

getResources().getColor(R.color.gallery\_tealdark),

"Gallery"

);

in = AnimationUtils.loadAnimation(getContext(), android.R.anim.slide\_in\_left);

out = AnimationUtils.loadAnimation(getContext(), android.R.anim.slide\_out\_right);

flipper16();

flipper15();

flipper13();

flipper14();

members();

}

@Nullable

@Override

public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {

return inflater.inflate(R.layout.fragment\_gallery, container, false);

}

}

#MainActivity.java

package com.antimony.scriptoo.syntechx;

import android.content.Intent;

import android.database.SQLException;

import android.os.Build;

import android.support.annotation.NonNull;

import android.support.design.widget.BottomNavigationView;

import android.support.v4.app.Fragment;

import android.support.v4.app.FragmentTransaction;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.view.MenuItem;

import android.view.Window;

import android.view.WindowManager;

import com.nostra13.universalimageloader.core.DisplayImageOptions;

import com.nostra13.universalimageloader.core.ImageLoader;

import com.nostra13.universalimageloader.core.ImageLoaderConfiguration;

import java.io.IOException;

public class MainActivity extends AppCompatActivity {

BottomNavigationView bottom;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

bottom = findViewById(R.id.main\_bottom\_navigation);

listenToNavigation();

//DB LOADER INIT

DisplayImageOptions displayImageOptions = new DisplayImageOptions.Builder()

.cacheInMemory(true)

.cacheOnDisk(true)

.build();

ImageLoaderConfiguration config = new ImageLoaderConfiguration.Builder(this)

.defaultDisplayImageOptions(displayImageOptions)

.build();

ImageLoader.getInstance().init(config);

//DB INIT

DatabaseHelper dh = new DatabaseHelper(this);

//CREATE

try{

dh.createDatabase();

}catch (IOException ioe){

throw new Error("Unable to create");

}

setFragment(EventsFragment.getInstance());

}

public void listenToNavigation(){

bottom.setOnNavigationItemSelectedListener(new BottomNavigationView.OnNavigationItemSelectedListener() {

@Override

public boolean onNavigationItemSelected(@NonNull MenuItem item) {

Fragment selected = null;

switch (item.getItemId()){

//ABOUT PAGE

case R.id.navigation\_about:

selected = AboutFragment.getInstance();

break;

//COMMITEES PAGE

case R.id.navigation\_commitees:

selected = CommiteesFragment.getInstance();

break;

//GALLERY PAGE

case R.id.navigation\_gallery:

selected = GalleryFragment.getInstance();

break;

case R.id.navigation\_events:

selected = EventsFragment.getInstance();//

break;

}

setFragment(selected);

return true;

}

});

}

public void setColors(int primary, int primaryDark){

bottom.setBackgroundColor(primary);

if(Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.LOLLIPOP){

Window window = getWindow();

window.addFlags(WindowManager.LayoutParams.FLAG\_DRAWS\_SYSTEM\_BAR\_BACKGROUNDS);

window.setStatusBarColor(primaryDark);

}

}

public void setFragment(Fragment selected){

FragmentTransaction transaction = getSupportFragmentManager().beginTransaction();

transaction.replace(R.id.main\_frame\_layout, selected);

transaction.commit();

}

}

#CommiteesAdapter.java

package com.antimony.scriptoo.syntechx;

import android.content.Context;

import android.content.Intent;

import android.support.annotation.NonNull;

import android.support.annotation.Nullable;

import android.util.Log;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.ArrayAdapter;

import android.widget.ImageView;

import android.widget.TextView;

import java.util.ArrayList;

/\*\*

\* Created by antimony on 12/11/17.

\*/

public class CommiteesAdapter extends ArrayAdapter<CommiteeClass>{

private static class ViewHolder{

TextView name;

ImageView image;

}

public CommiteesAdapter(Context context, ArrayList<CommiteeClass> coms){

super(context, 0, coms);

}

@NonNull

@Override

public View getView(final int position, @Nullable View convertView, @NonNull ViewGroup parent) {

final CommiteeClass com = getItem(position);

ViewHolder viewHolder;

if (convertView == null){

viewHolder = new ViewHolder();

convertView = LayoutInflater.from(getContext()).inflate(R.layout.commitee\_item\_layout, parent, false);

viewHolder.name = convertView.findViewById(R.id.commitee\_item\_text);

viewHolder.image = convertView.findViewById(R.id.commitee\_item\_logo);

convertView.setTag(viewHolder);

}else{

viewHolder = (ViewHolder) convertView.getTag();

}

viewHolder.name.setText(com.name);

viewHolder.image.setImageResource(getContext().getResources().getIdentifier(

com.image.split("\\.png")[0],

"drawable",

getContext().getPackageName())

);

// Log.d("LOL", com.image.split("\\.png")[0]);

viewHolder.name.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent i = new Intent(getContext(), CommiteeActivity.class);

// i.putExtra("NAME", com.name);

// i.putExtra("ROLE", com.role);

i.putExtra("ID", position);

i.putExtra("COMMITEEOBJECT", com);

getContext().startActivity(i);

}

});

return convertView;

}

}

#EventActivity.java

package com.antimony.scriptoo.syntechx;

import android.content.Intent;

import android.os.Bundle;

import android.support.v7.app.AppCompatActivity;

import android.support.v7.widget.SwitchCompat;

import android.support.v7.widget.Toolbar;

import android.view.View;

import android.widget.ImageView;

import android.widget.TextView;

import com.nostra13.universalimageloader.core.ImageLoader;

public class EventActivity extends AppCompatActivity {

TextView description, rules, location, time;

ImageView header;

Toolbar toolbar;

EventClass event;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_event\_page);

toolbar = (Toolbar) findViewById(R.id.event\_page\_toolbar);

description = (TextView)findViewById(R.id.description\_textview);

rules= (TextView)findViewById(R.id.rules\_textview);

time= findViewById(R.id.event\_timing\_view);

location= (TextView)findViewById(R.id.location\_textview);

header = (ImageView)findViewById(R.id.event\_page\_header);

Intent i = getIntent();

setSupportActionBar(toolbar);

////

event = (EventClass)i.getSerializableExtra("EVENTOBJECT");

//SET ALL THE THINGS

getSupportActionBar().setTitle(event.name);

description.setText(event.description);

rules.setText(event.rules);

time.setText(event.date + ", " + event.timing);

location.setText(event.location);

ImageLoader loader = ImageLoader.getInstance();

loader.displayImage(event.background, header);

//TRANSPARENT STATUS BAR

/\* if(Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.KITKAT){

getWindow().setFlags(WindowManager.LayoutParams.FLAG\_LAYOUT\_NO\_LIMITS, WindowManager.LayoutParams.FLAG\_LAYOUT\_NO\_LIMITS);

}\*/

}

public void register(View v){

Intent i = new Intent(this, RegistrationActivity.class);

i.putExtra("EVENT\_NAME", event.name);

startActivity(i);

}

}

#RegisterInBackground.java

package com.antimony.scriptoo.syntechx;

import android.content.Context;

import android.content.Intent;

import android.os.AsyncTask;

import android.support.v7.app.AppCompatActivity;

import android.widget.Toast;

import java.io.BufferedInputStream;

import java.io.BufferedReader;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.net.HttpURLConnection;

import java.net.URL;/\*\*

\* Created by antimony on 3/20/18.

\*/

public class RegisterInBackground extends AsyncTask<String, Void, String>{

private Context context;

String resultString;

String ip\_address = "192.168.43.228";

// String ip\_address = "103.21.236.22";

public RegisterInBackground(Context context){

this.context = context;

}

@Override

protected String doInBackground(String... args) {

try{

String name = args[0];

String college = args[1];

String stream = args[2];

String age = args[3];

String contact\_num = args[4];

String email\_id = args[5];

String event\_name = args[6];

String urlstring = "http://" + ip\_address + ":8080/SynTechX/registration.jsp?"

+ "name=" + name

+ "&college=" + college

+ "&age=" + age

+ "&stream=" + stream

+ "&contact=" + contact\_num

+ "&email=" + email\_id

+ "&event=" +event\_name;

URL url = new URL(urlstring);

HttpURLConnection urlConnection = (HttpURLConnection) url.openConnection();

InputStream is = new BufferedInputStream(urlConnection.getInputStream());

BufferedReader reader = new BufferedReader(new InputStreamReader(is));

String x = "";

StringBuffer content = new StringBuffer();

while ( (x = reader.readLine()) != null){

content.append(x);

}

is.close();

reader.close();

return content.toString();

}catch (Exception e){

resultString = e.toString();

return resultString;

}

}

@Override

protected void onPostExecute(String result) {

String temp=result.trim();

Toast.makeText(context,temp,Toast.LENGTH\_SHORT).show();

if(temp.equals("HO GAYA WOW!")){

Toast.makeText(context,"Succesfully Registered!",Toast.LENGTH\_SHORT).show();

((AppCompatActivity)context).finish();

}else{

// Toast.makeText(context,"FAILED",Toast.LENGTH\_SHORT).show();

}

}

}

#RegistrationActivity.java

package com.antimony.scriptoo.syntechx;

import android.app.ProgressDialog;

import android.content.Intent;

import android.os.Bundle;

import android.os.PersistableBundle;

import android.support.annotation.Nullable;

import android.support.v7.app.AppCompatActivity;

import android.util.Log;

import android.view.View;

import android.webkit.WebResourceRequest;

import android.webkit.WebView;

import android.webkit.WebViewClient;

import android.widget.EditText;

import android.widget.TextView;

import android.widget.Toast;

/\*\*

\* Created by antimony on 12/20/17.

\*/

public class RegistrationActivity extends AppCompatActivity {

Intent recievedIntent;

TextView title;

EditText name\_view, college\_view, stream\_view, age\_view, contact\_num\_view, email\_id\_view;

String name, college, stream, age, contact\_num, email\_id, event\_name;

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_registration);

recievedIntent = getIntent();

event\_name = recievedIntent.getStringExtra("EVENT\_NAME");

title = findViewById(R.id.reg\_title);

name\_view = findViewById(R.id.reg\_name);

college\_view = findViewById(R.id.reg\_college);

stream\_view= findViewById(R.id.reg\_stream);

age\_view = findViewById(R.id.reg\_age);

contact\_num\_view = findViewById(R.id.reg\_contact);

email\_id\_view = findViewById(R.id.reg\_mail);

String title\_text = title.getText().toString() + event\_name;

title.setText(title\_text);

}

public void registerForEvent(View v){

try {

name = name\_view.getText().toString();

college = college\_view.getText().toString();

stream = stream\_view.getText().toString();

age = age\_view.getText().toString();

contact\_num = contact\_num\_view.getText().toString();

email\_id = email\_id\_view.getText().toString();

if(isValid()){

new RegisterInBackground(this).execute(name, college, stream, age, contact\_num, email\_id, event\_name);

Toast.makeText(this,

"Thank you, " + name + ". " +

"Registering for " + event\_name + ", please wait...",

Toast.LENGTH\_SHORT).show();

name\_view.setEnabled(false);

college\_view.setEnabled(false);

stream\_view.setEnabled(false);

age\_view.setEnabled(false);

contact\_num\_view.setEnabled(false);

email\_id\_view.setEnabled(false);

}

}catch (Exception e){

Toast.makeText(this, "Please enter all fields!", Toast.LENGTH\_SHORT).show();

}

}

public boolean isValid(){

boolean valid = true;

if(Integer.parseInt(age) > 99 || Integer.parseInt(age) < 10 ){

valid = false;

Toast.makeText(this, "Please enter a valid age!", Toast.LENGTH\_SHORT).show();

}

if(!(email\_id.contains("@") && email\_id.contains("."))){

valid = false;

Toast.makeText(this, "Please enter a valid email ID", Toast.LENGTH\_SHORT).show();

}

return valid;

}

}

#AboutFragment.java

package com.antimony.scriptoo.syntechx;

import android.content.Intent;

import android.net.Uri;

import android.os.Bundle;

import android.support.annotation.Nullable;

import android.support.v4.app.Fragment;

import android.support.v7.widget.Toolbar;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.Button;

public class AboutFragment extends Fragment {

Toolbar toolbar;

Button website\_button, developer\_button;

final String website\_url = "http://syntech-x.rdnational.ac.in/";

public static AboutFragment getInstance(){

AboutFragment fragment = new AboutFragment();

return fragment;

}

@Override

public void onActivityCreated(@Nullable Bundle savedInstanceState) {

super.onActivityCreated(savedInstanceState);

toolbar = (Toolbar)getActivity().findViewById(R.id.toolbar);

toolbar.setTitle(getString(R.string.app\_name));

website\_button = getActivity().findViewById(R.id.website\_button);

website\_button.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent i = new Intent(Intent.ACTION\_VIEW, Uri.parse(website\_url));

startActivity(i);

}

});

developer\_button = getActivity().findViewById(R.id.developer\_button);

developer\_button.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent i = new Intent(Intent.ACTION\_VIEW, Uri.parse("http://goo.gl/5WvbrQ"));

startActivity(i);

}

});

((MainNavActivity)getActivity()).setAttributes(

getResources().getColor(R.color.colorPrimary),

getResources().getColor(R.color.colorPrimaryDark),

"About"

);

}

@Nullable

@Override

public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {

return inflater.inflate(R.layout.fragment\_about, container, false);

}

}

#CommiteesFragment.java

package com.antimony.scriptoo.syntechx;

import android.os.Bundle;

import android.support.annotation.Nullable;

import android.support.design.widget.BottomNavigationView;

import android.support.v4.app.Fragment;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.ListView;

import java.util.ArrayList;

public class CommiteesFragment extends Fragment {

ListView listView;

ArrayList<CommiteeClass> array;

BottomNavigationView bottom;

@Override

public void onActivityCreated(@Nullable Bundle savedInstanceState) {

super.onActivityCreated(savedInstanceState);

listView = getActivity().findViewById(R.id.commitees\_list\_view);

bottom = getActivity().findViewById(R.id.main\_bottom\_navigation);

DatabaseHelper db = new DatabaseHelper(getContext());

array = db. getCommitees();

db.close();

CommiteesAdapter adapter = new CommiteesAdapter(getContext(), array);

View header = (View)getLayoutInflater().inflate(R.layout.header\_commitees\_listview, null);

listView.addHeaderView(header);

listView.setAdapter(adapter);

((MainNavActivity)getActivity()).setAttributes(getResources().getColor(R.color.commitees\_blue), getResources().getColor(R.color.commitees\_bluedark), "Committees");

}

public static CommiteesFragment getInstance(){

CommiteesFragment fragment = new CommiteesFragment();

return fragment;

}

@Override

public void onCreate(@Nullable Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

}

@Nullable

@Override

public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {

return inflater.inflate(R.layout.fragment\_commitees, container, false);

}

}

#MainNavActivity.java

package com.antimony.scriptoo.syntechx;

import android.os.Build;

import android.os.Bundle;

import android.support.design.widget.FloatingActionButton;

import android.support.design.widget.Snackbar;

import android.support.v4.app.Fragment;

import android.support.v4.app.FragmentTransaction;

import android.view.View;

import android.support.design.widget.NavigationView;

import android.support.v4.view.GravityCompat;

import android.support.v4.widget.DrawerLayout;

import android.support.v7.app.ActionBarDrawerToggle;

import android.support.v7.app.AppCompatActivity;

import android.support.v7.widget.Toolbar;

import android.view.Menu;

import android.view.MenuItem;

import android.view.Window;

**CHAP 5: SYSTEM IMPLEMENTATION**

5.1 Hardware and Software Requirements

5.2 Screen Layouts

**5.1 Hardware & Software Requirements**

**Hardware Requirements:**

The minimum Hardware and system Software requirements for development and using this is:

Processor: 1 GHz and above

RAM: Minimum requirement is 2 GB

**Software requirements:**

**Client side requirements**

Android version: 4.4 and above

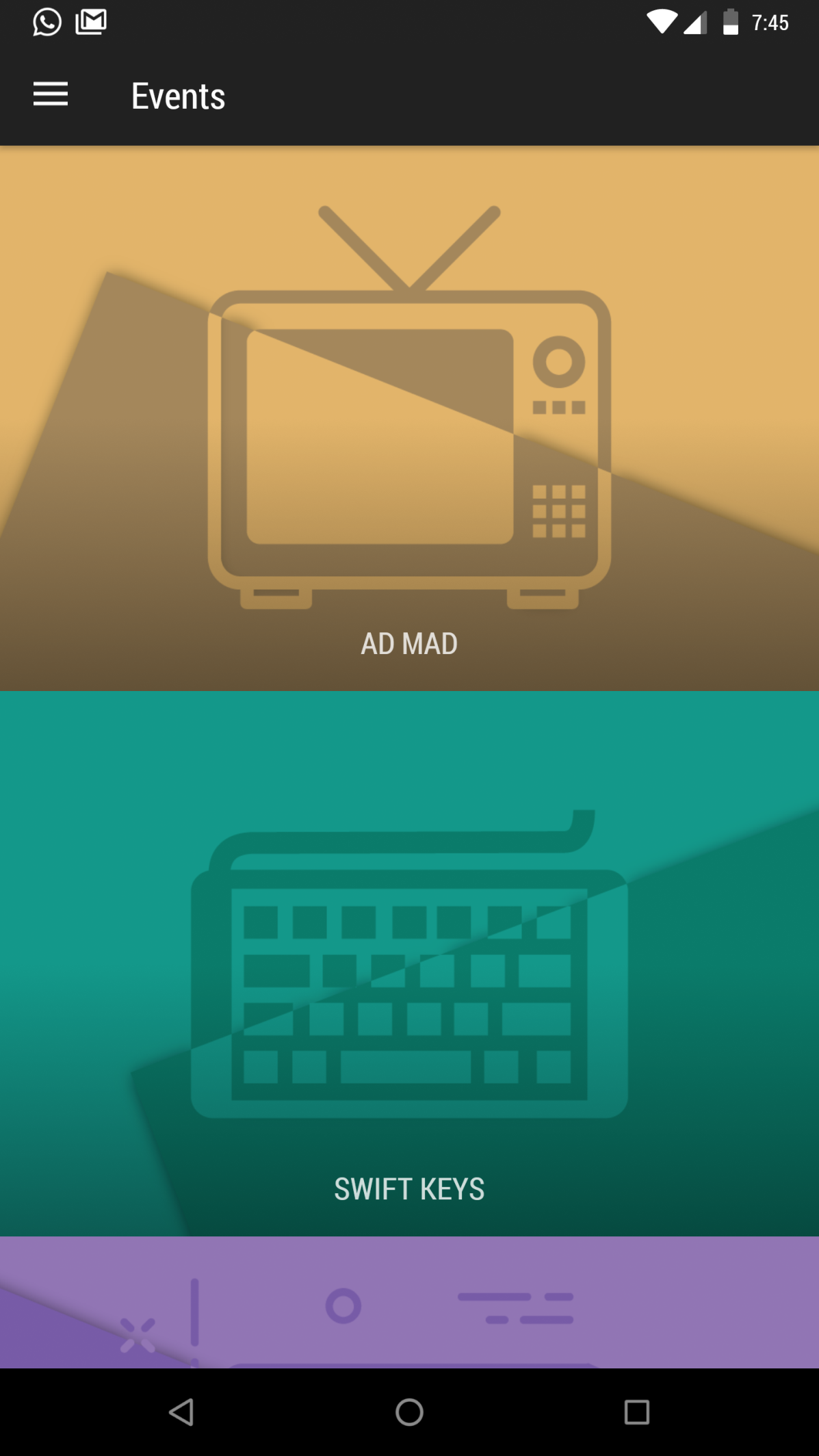
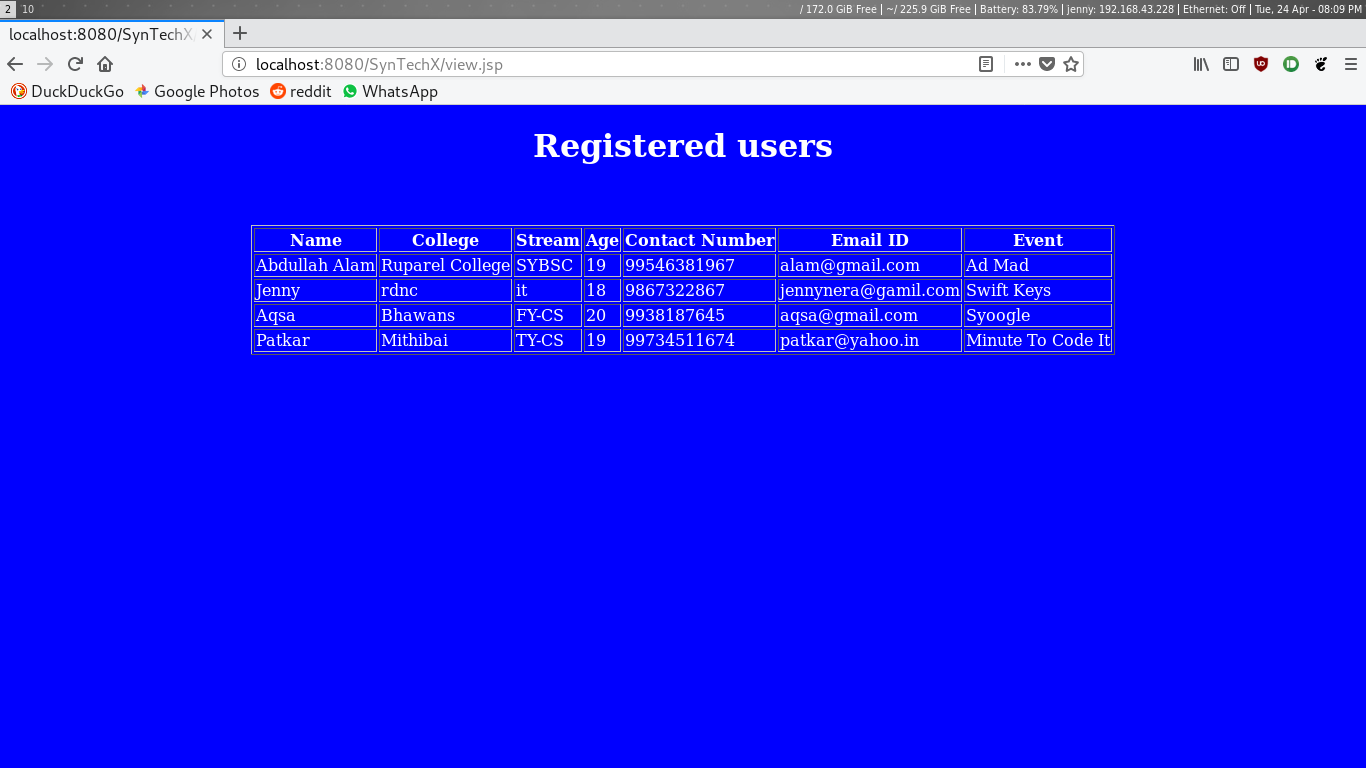
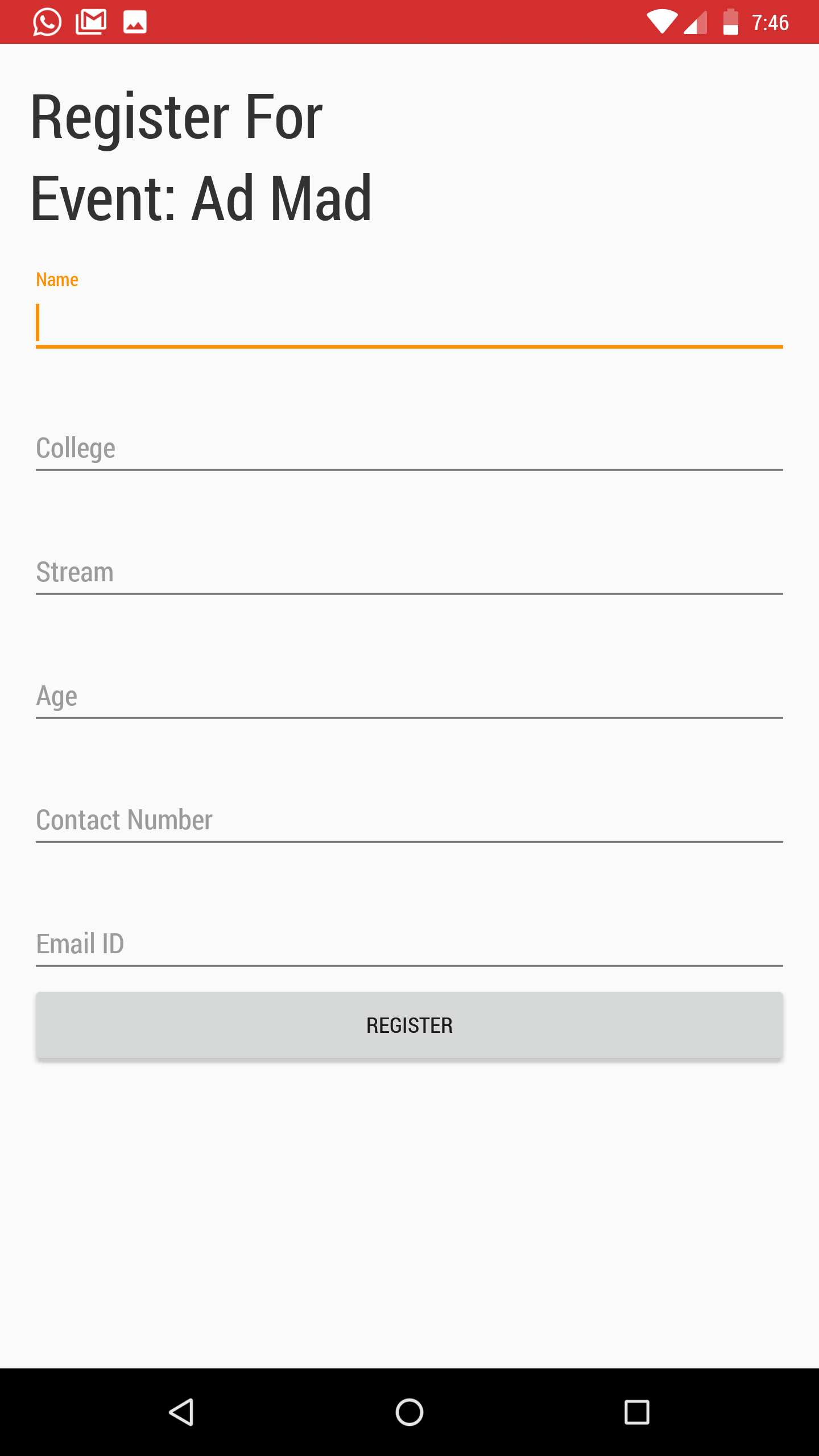
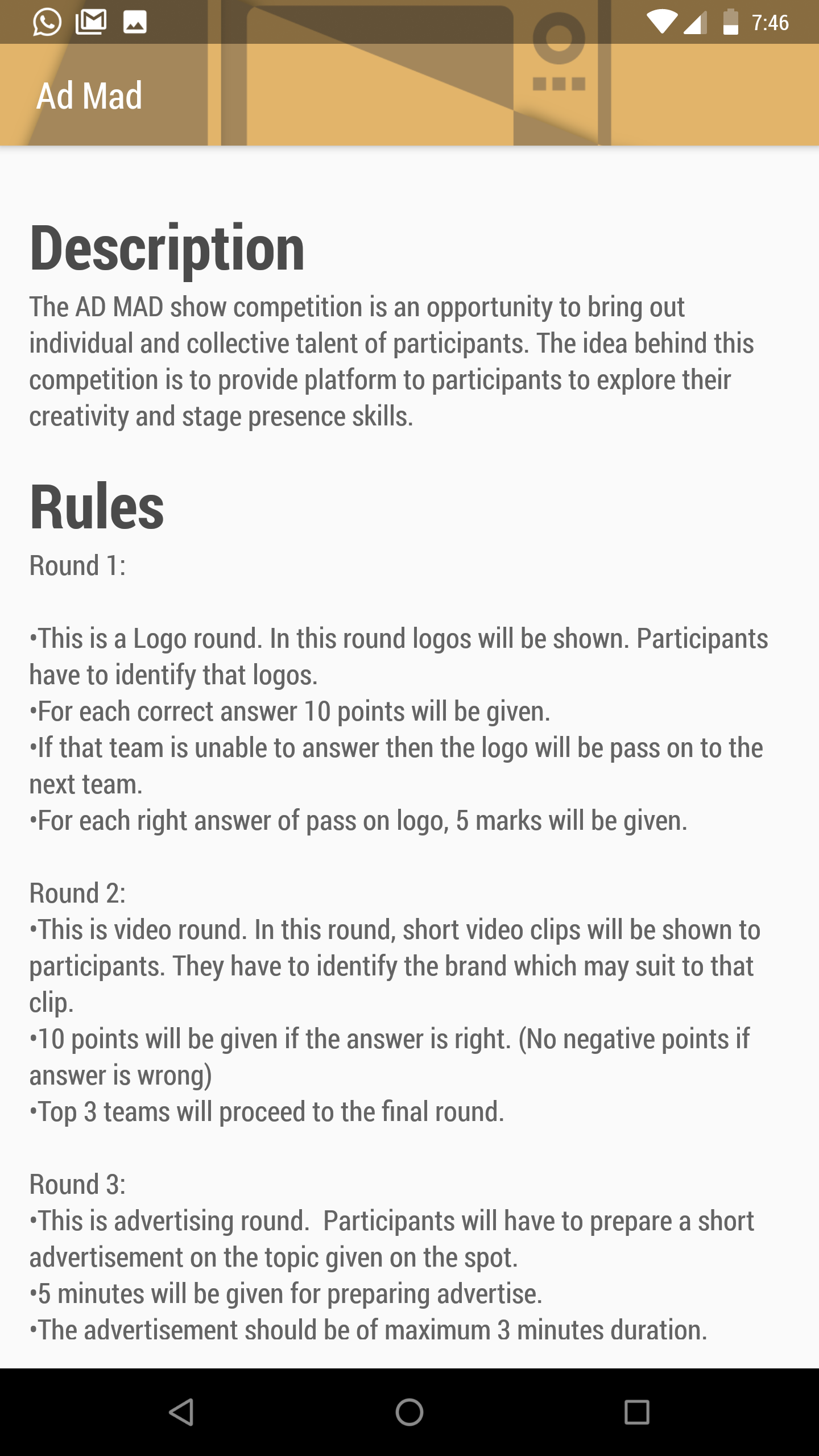
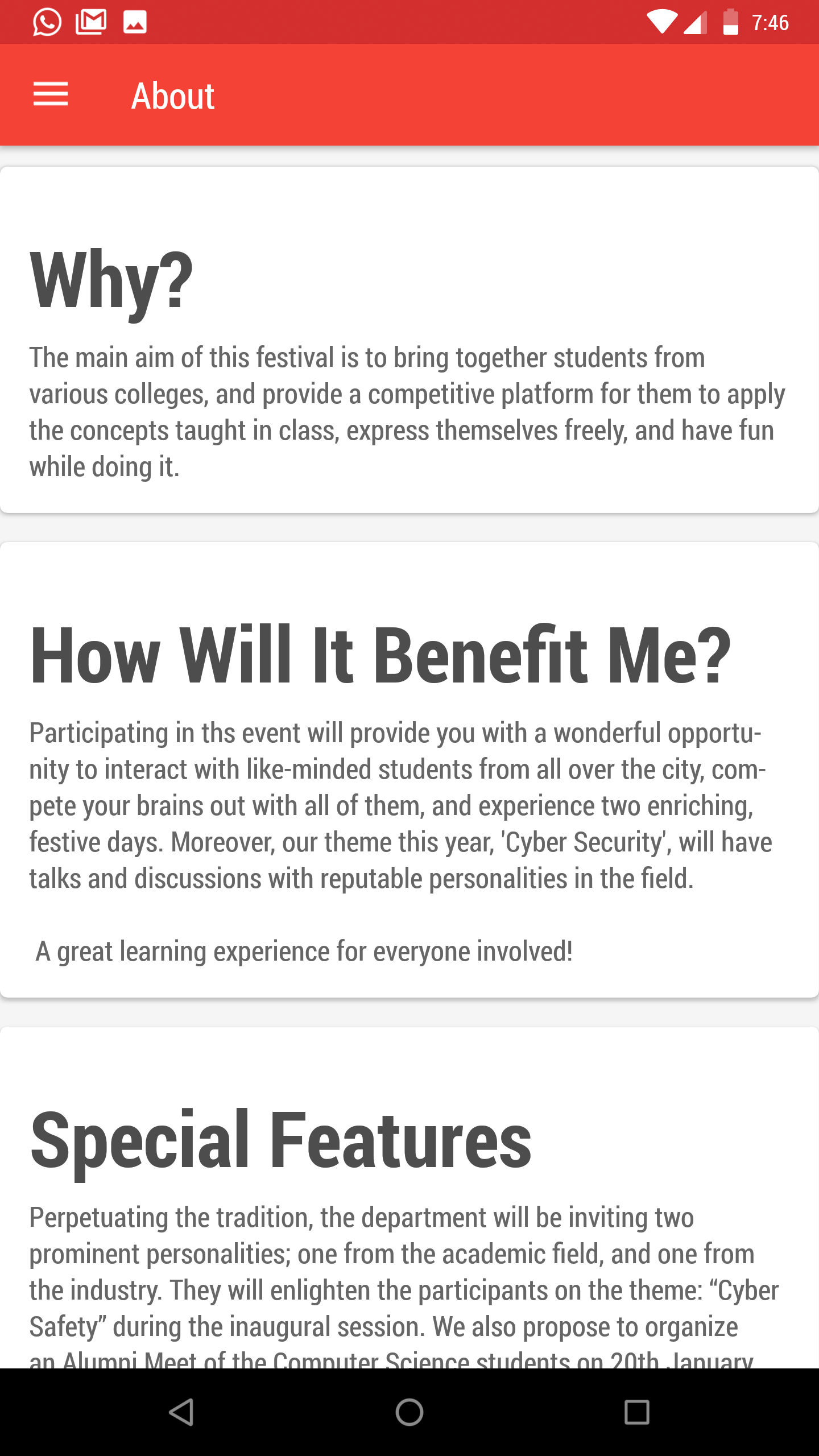
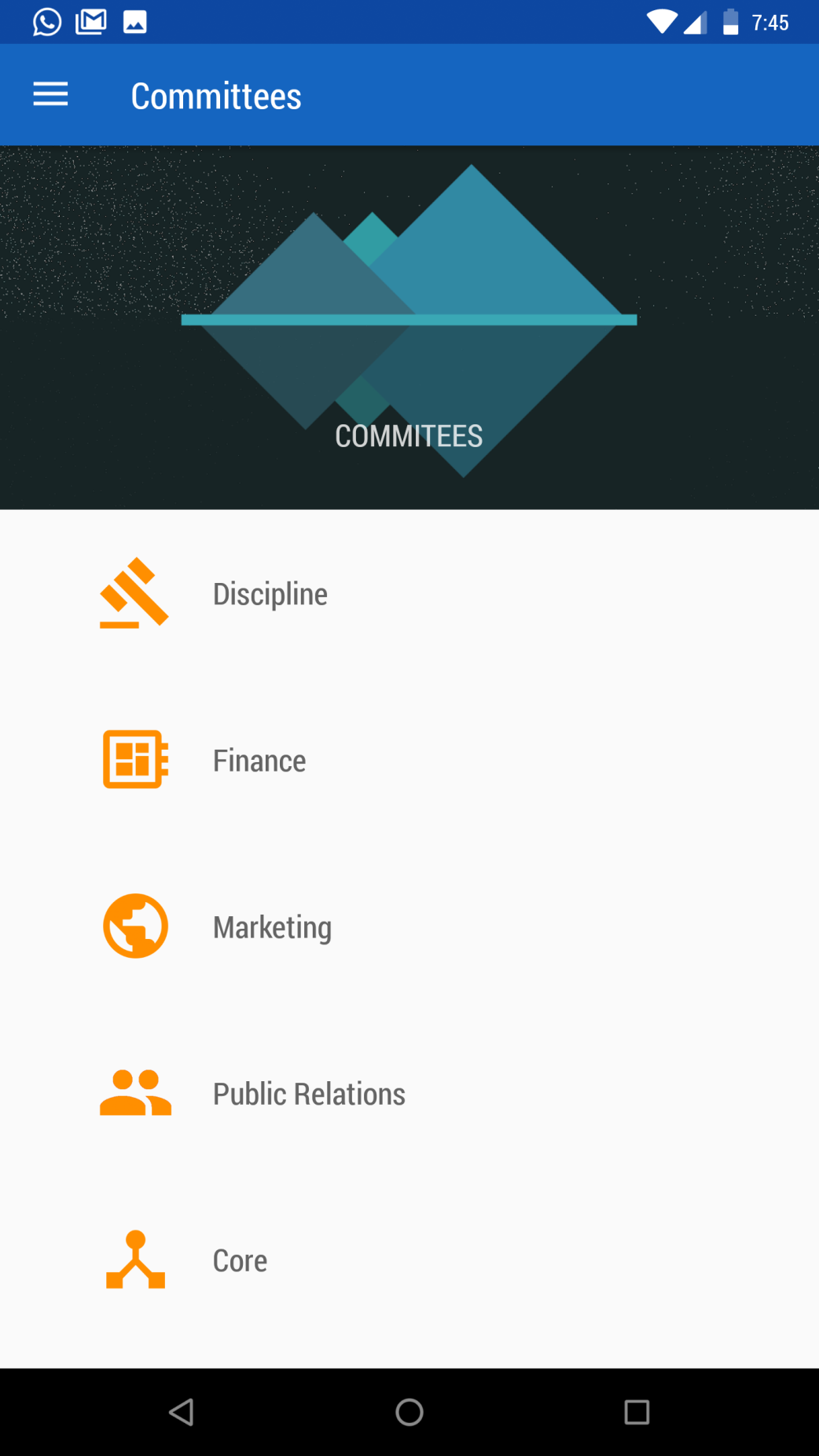
**Sever side requirements**

Windows 10

MySQL

GlassFish Server 4.0

**5.2 Screen Layouts**



**CHP 6. FUTURE ENHANCEMENT**

If there is any rectification or enhancement proposed by the user, the application will be modified and the desired module will be implemented into current application. Future enhancements include developing the following as stated below:

* To include more features as per requirements
* To include more event provisions.
* To include more real-time functionality for the events.
* To update the application as per the customer’s demands.

**CHP 7. REFRENCES AND BIBLIOGRAPHY**

**Website:**

* [**https://www.youtube.com**](https://www.youtube.com/)
* [**https://www.google.com**](https://www.google.com/)
* [**https://stackoverflow.com**](https://stackoverflow.com/)
* [**https://www.w3schools.com**](https://www.w3schools.com/)
* [**https://www.tutorialspoint.com**](https://www.tutorialspoint.com/)