CS-223
SOFTWARE UNIT TESTING DOCUMENT

for

Project 8
Classroom Visualisation App-2

Prepared by:
Group-2
Harshit Agrawal - 160101030
Harshit Gupta - 160101031
Harshit Sharma - 160101032

April 22, 2018

Contents

1 Introduction

This document contains the complete unit testing of our app Classroom Visualization (project-8). Unit testing is undertaken after a module has been coded and successfully reviewed. The code testing team in isolation tests different units and modules of the system. Different members of the code testing team have submitted their reports. In short, this document is meant to equip the reader with the bugs and shortcomings in the working of the app .

1.1 Black Box Testing Team profile

The code testing team comprises of the following members, all of whom are Undergraduates currently pursuing Bachelor of Technology at Indian Institute of Technology Guwahati, India in the Department of Computer Science and Engineering. All of the members are currently in the sophomore year.

1. Mitansh jain
2. Sujoy Ghosh
3. Daman Tekchandani

2 Equivalence Class Partitioning

2.1 Add student module

Input : Student ID , Student Name , Internet connectivity
Equivalence Classes :

1. Positive integer Student ID , name , internet connectivity
2. Non integer or negative Student ID , name , internet connectivity
3. Existing Student ID , name , internet connectivity
4. Fields of Student ID empty , internet connectivity
5. Student ID , name , No internet connectivity

Expected Results :

1. Student successfully enrolled
2. Error message displayed
3. NO change in the database
4. No new entry addded in the database
5. Add feature disabled

Boundary Value :

- Student ID = 0

2.2 Select Seat module

Input : Student ID , Seat number , Internet connectivity
Equivalence Classes :

1. Valid Student ID , Non empty selection (a seat selected among the choices given) , Student not already seated , internet connectivity
2. Valid Student ID , Empty selection (no seat selected) , internet connectivity
3. Invalid Student ID , internet connectivity

4. Valid Student ID , Student already seated , internet connectivity
5. Empty Student ID field , internet connectivity
6. No internet connectivity

Expected Results :

1. Seat allocated to the corresponding Student ID
2. Error message displayed
3. "Enter a valid Student ID " Error message displayed
4. "Duplicate data " error message displayed
5. Warning given " No student selected "
6. Select seat feature disabled

Boundary Value :

- Since there are discrete values in this Equivalence class , we have no boundary values for them.

2.3 Classroom layout generator module

Input : Number of seats , Internet connectivity
Equivalence Classes :

1. Number of seats is a positive integer , internet connectivity
2. Number of seats is not a positive integer , internet connectivity
3. Field of number of seats is empty , internet connectivity
4. No internet connectivity

Expected Results :

1. Layout generated with the given number of seats
2. "Enter a valid number " error displayed
3. No seat layout generated
4. Layout generate feature disabled

Boundary Value :

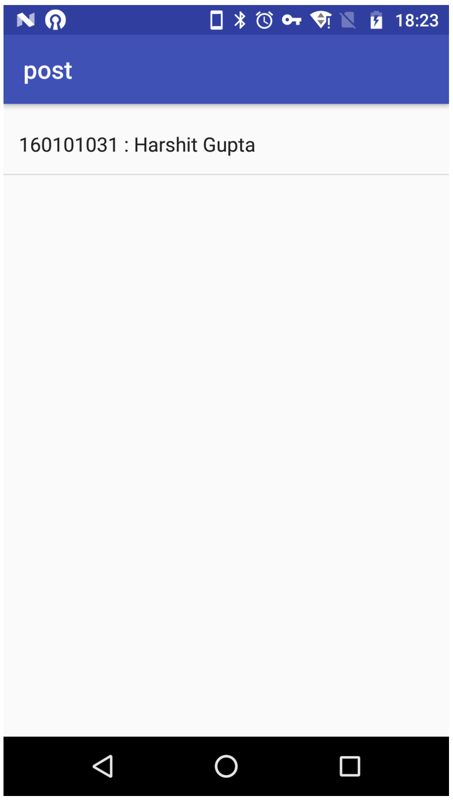
- Number of Setas = 0

3 Black Box Testing

Now, selecting one representative value from each equivalence class, the test suit obtained is as follows:

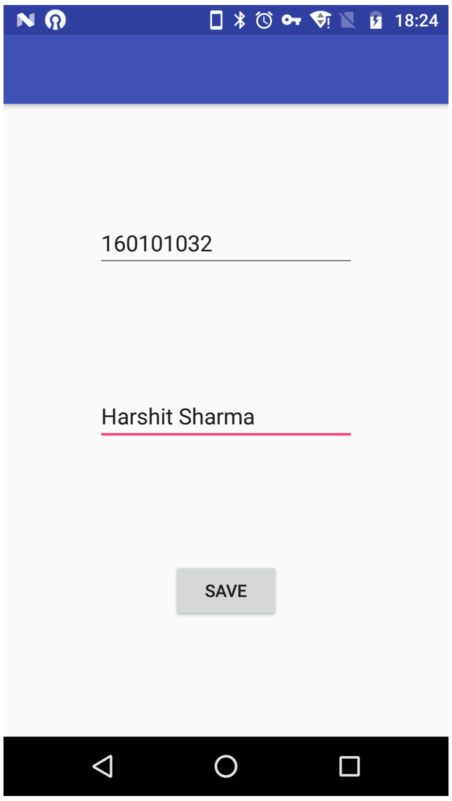
3.1 Add student module

1. a) Student ID : 160101032
   b) Student Name : Harshit Sharma
   c) Internet Connectivity = True



160101031 : Harshit Gupta

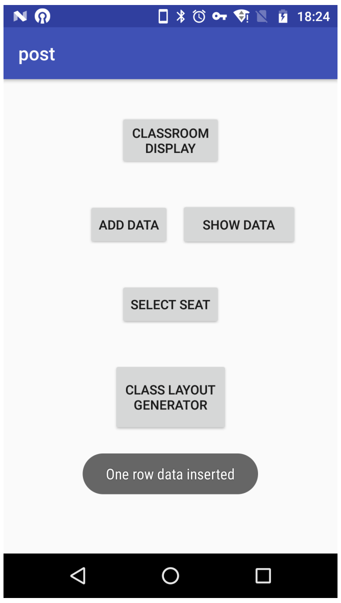
(a) Initial Database



160101032

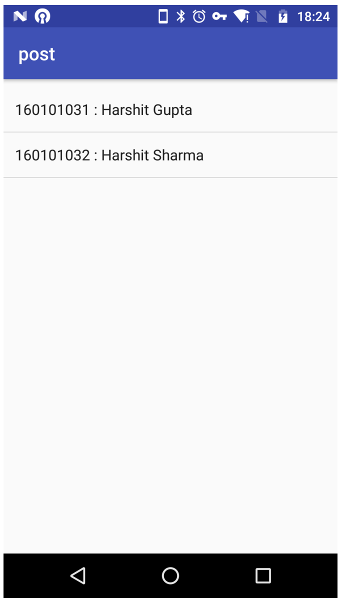
Harshit Sharma

(b) Add Data Screen



One row data inserted

(a) Success Message



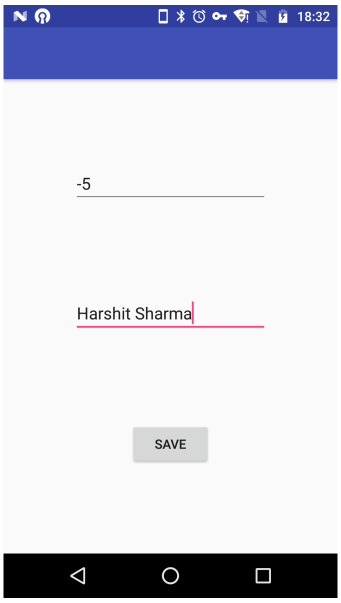
160101031 : Harshit Gupta

160101032 : Harshit Sharma

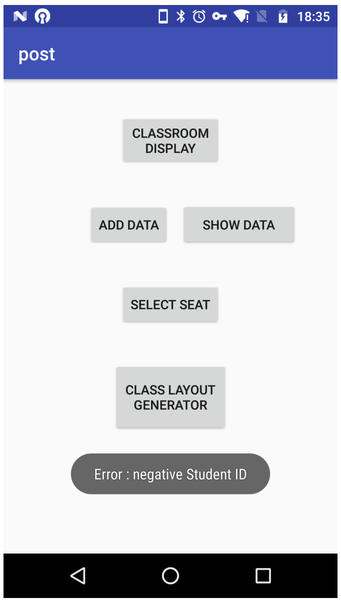
(b) Show Data Screen

- Number of seats is a positive integer , internet connectivity
- Expected Result : Classroom layout should be generated with the given number of seats
- App Output: The layout is successfully generated with the given no. of seats.

2. a) Student ID : -5
   b) Student Name : Harshit Sharma
   c) Internet Connectivity = True



(a) Negative Student ID

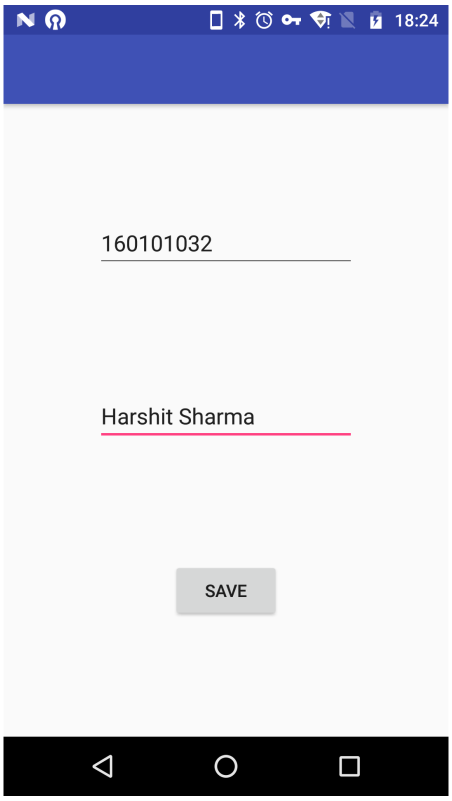


Error : negative Student ID

(b) Error Displayed

- Non integer or negative Student ID , name , internet connectivity
- Expected Result : Error message should be displayed regarding the invalid student id.
- App Output : Error is displayed “Negative Student Id”

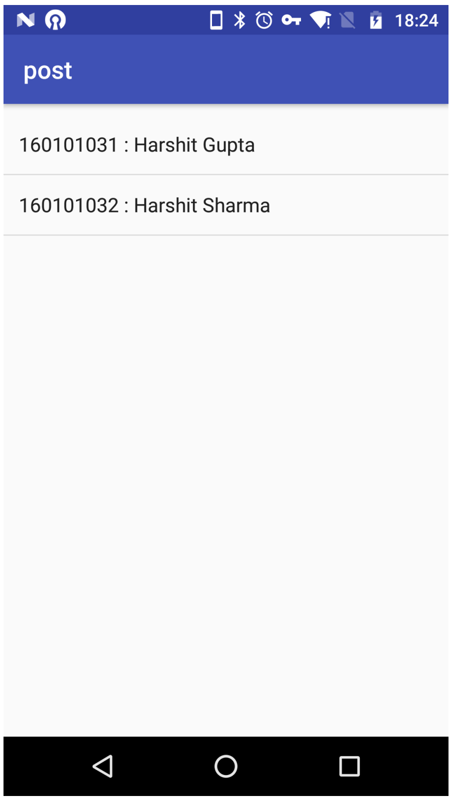
3. a) Student ID : 160101032
   b) Student Name : Harshit Sharma
   c) Internet Connectivity = True



160101032

Harshit Sharma

(a) Adding Existing Record



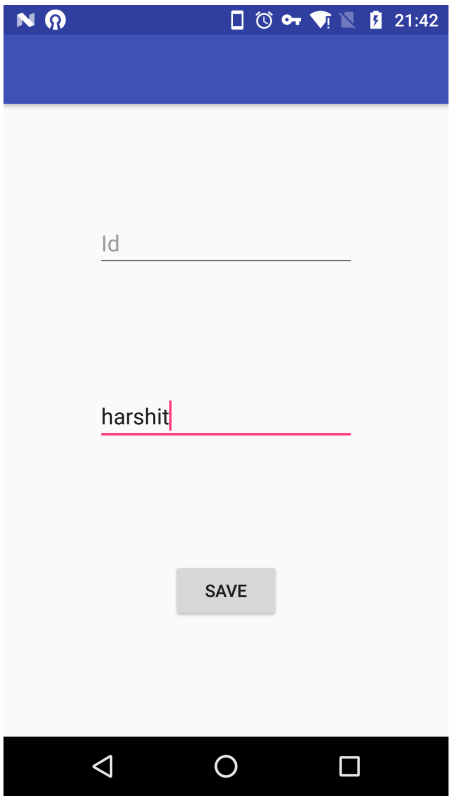
160101031 : Harshit Gupta

160101032 : Harshit Sharma

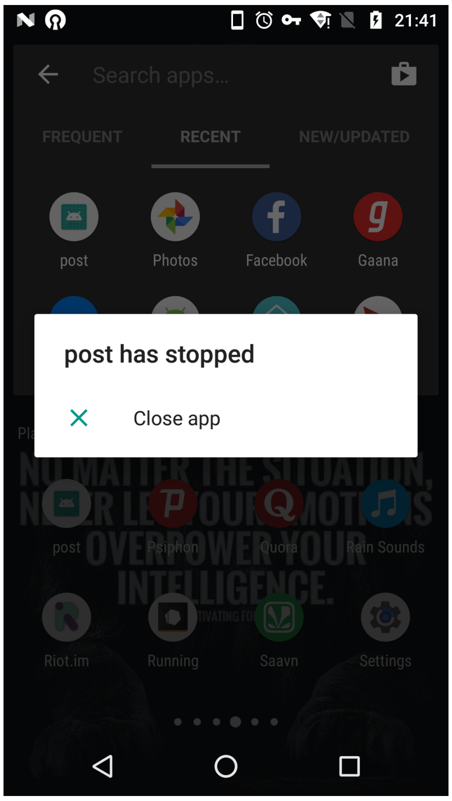
(b) No change in the database

- Existing Student ID , name , internet connectivity
- Expected Expected Result : No change should be done to the database.
- App Output : Though a message is displayed that row is added but there is no change in the database.

4. a) Student ID :
   b) Student Name : harshit
   c) Internet Connectivity = True



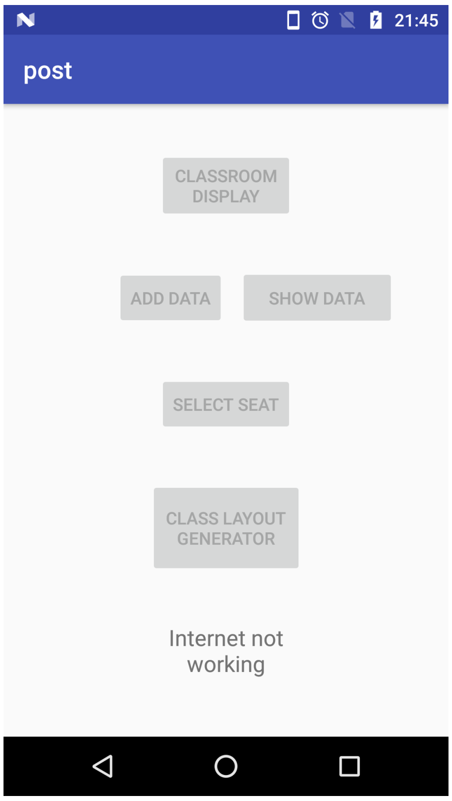
(a) Empty Student ID



(b) Crash Screen

- Fields of Student ID empty , internet connectivity
- Expected Result : No new entry should be added in the database
- Output : The app stops working when the field of student ID is left empty.

5. a) Student ID : 160101032
   b) Student Name : Harshit Sharma
   c) Internet Connectivity = False



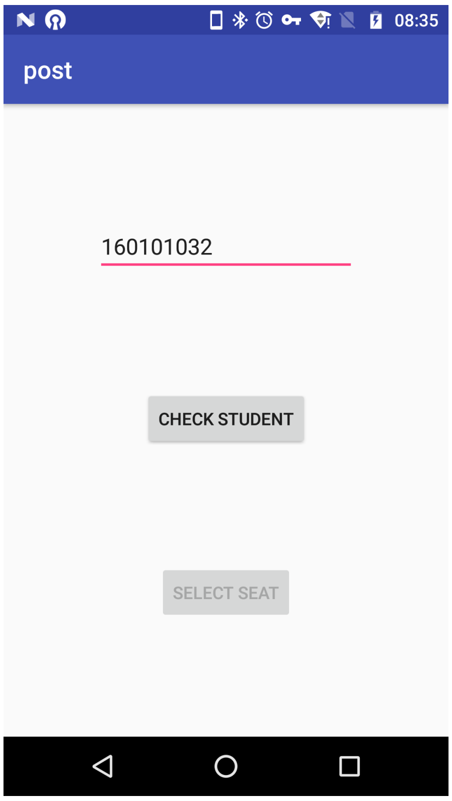
Internet not working

Figure 3.6: Features disabled if internet is not working

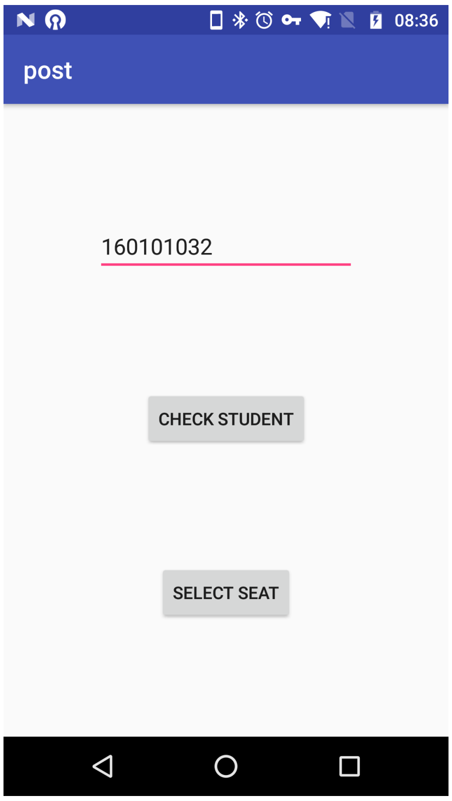
- Student ID , name , No internet connectivity
- Expected Result : The add student feature should be disabled
- App Output : Add Student button is disabled when there is no internet connectivity.

3.2 Select Seat module

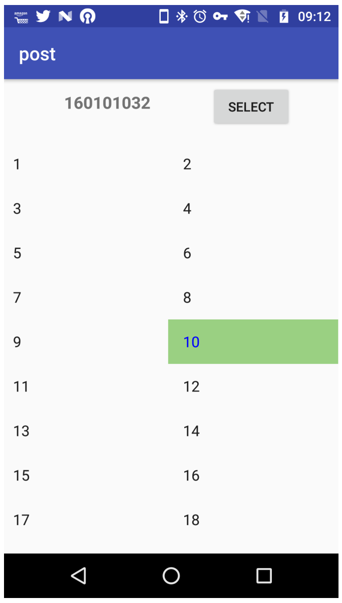
1. a) Student ID : 160101032
   b) Seat Number : 10
   c) Internet Connectivity = True



(a) Valid Student ID



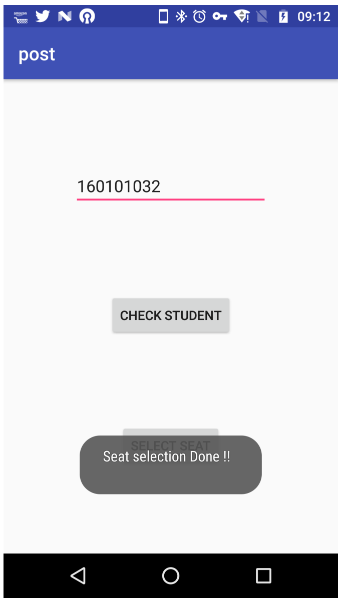
(b) Select Seat button enabled



160101032

1	2
3	4
5	6
7	8
9	10
11	12
13	14
15	16
17	18

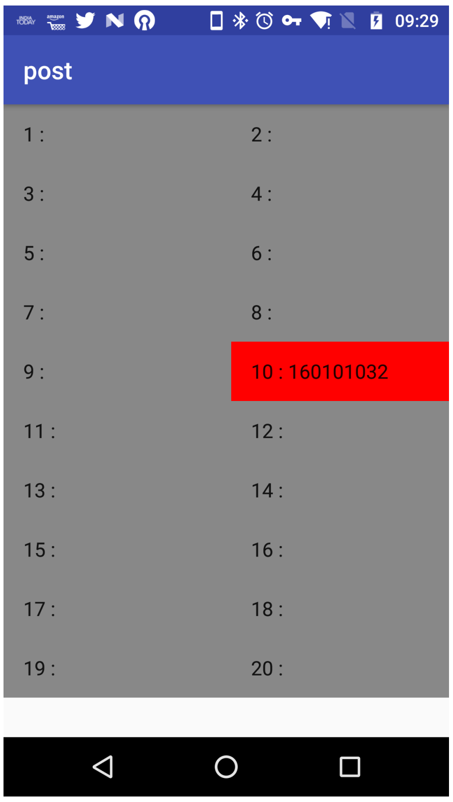
(a) Seat selection screen



160101032

Seat selection Done !!

(b) Success message

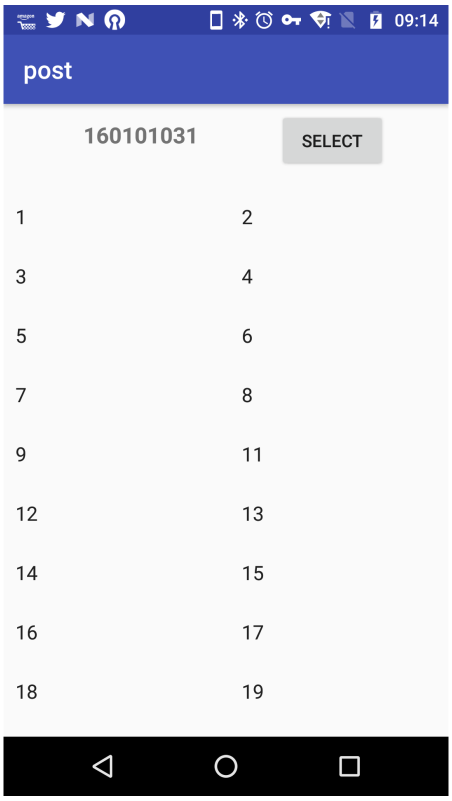


1 :	2 :
3 :	4 :
5 :	6 :
7 :	8 :
9 :	10 : 160101032
11 :	12 :
13 :	14 :
15 :	16 :
17 :	18 :
19 :	20 :

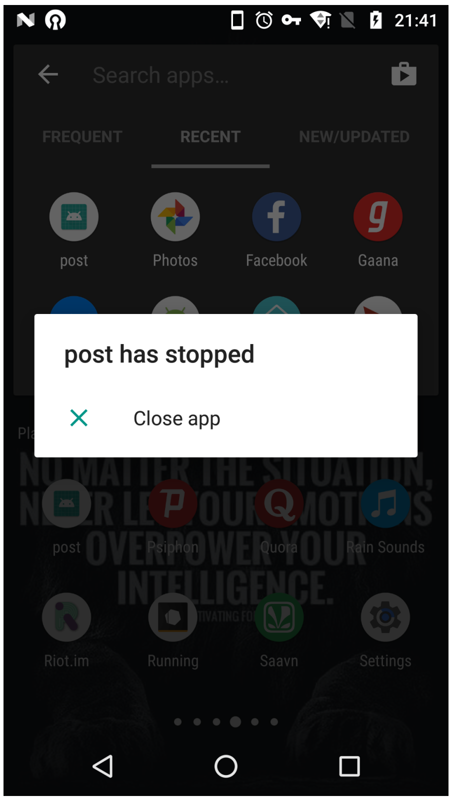
Figure 3.9: Show Class Screen

- Valid Student ID , Non empty selection (a seat selected among the choices given) , Student not already seated , internet connectivity
- Expected Result : Seat should be allocated to the corresponding Student ID.
- App Output : The seat selected is allocated to the corresponding Student ID. A message is displayed “Seat Selection done”.

2. a) Student ID : 160101031
   b) Seat Number :
   c) Internet Connectivity = True



(a) No seat selected

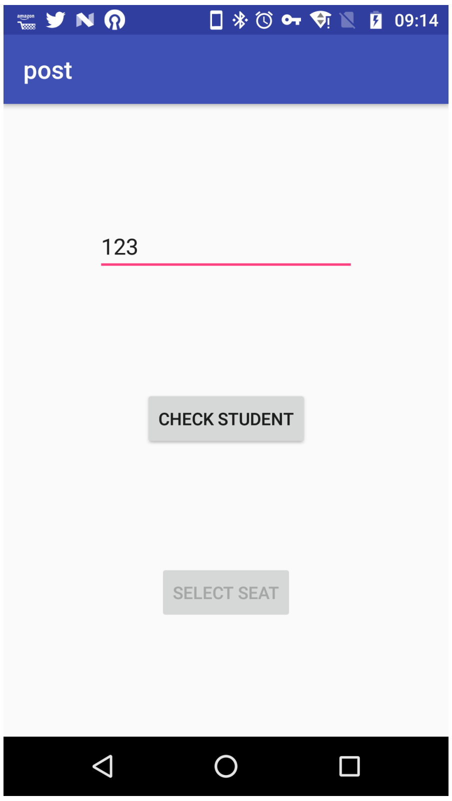

post has stopped

Close app

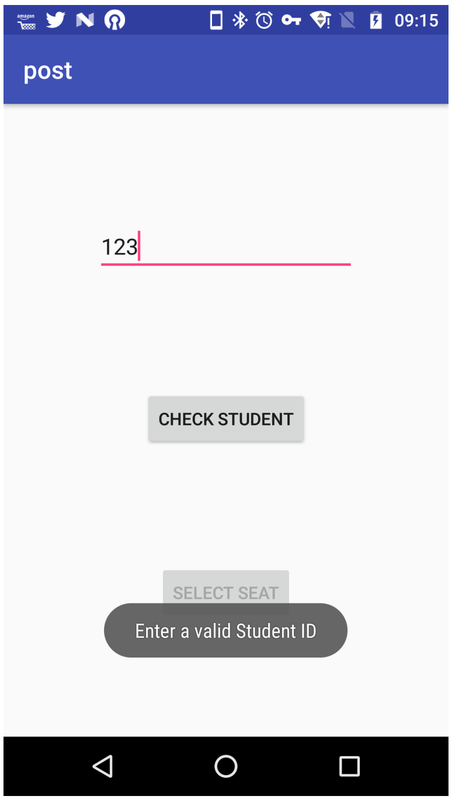
(b) Crash Screen

- Valid Student ID , Empty selection (no seat selected) , internet connectivity
- Expected Result : Error message should be displayed regarding no seat being selected.
- App Output : The app stops working if no seat is selected and select button is pressed.

3. a) Student ID : 123
   b) Seat Number : 12
   c) Internet Connectivity = True



(a) invalid Student ID

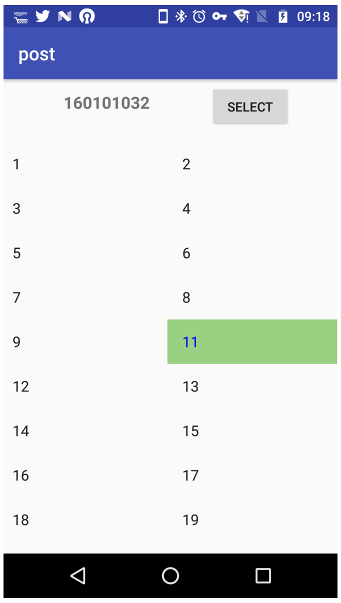


Enter a valid Student ID

(b) Error message displayed

- Invalid Student ID , internet connectivity
- Expected Expected Result : Error message displayed regarding the student ID being invalid.
- App Output : Error is displayed "Please enter a valid Student Id".

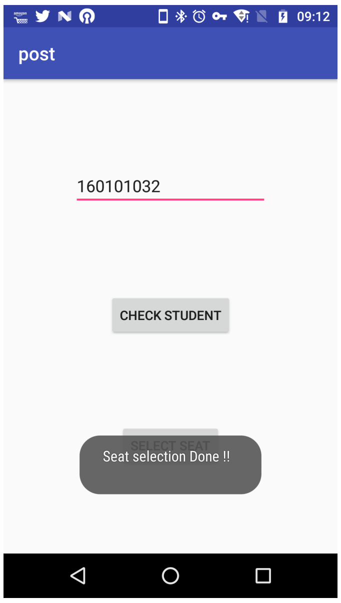
4. a) Student ID : 160101032
   b) Seat Number : 10
   c) Internet Connectivity = True


post

160101032 SELECT

1	2
3	4
5	6
7	8
9	11
12	13
14	15
16	17
18	19

(a) Select seat screen


post

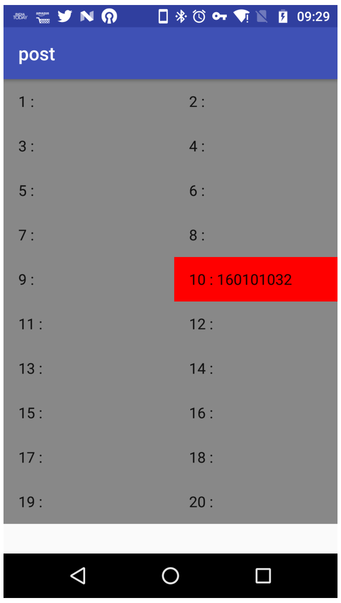
160101032

CHECK STUDENT

SELECT SEAT

Seat selection Done !!

(b) Show Data Screen

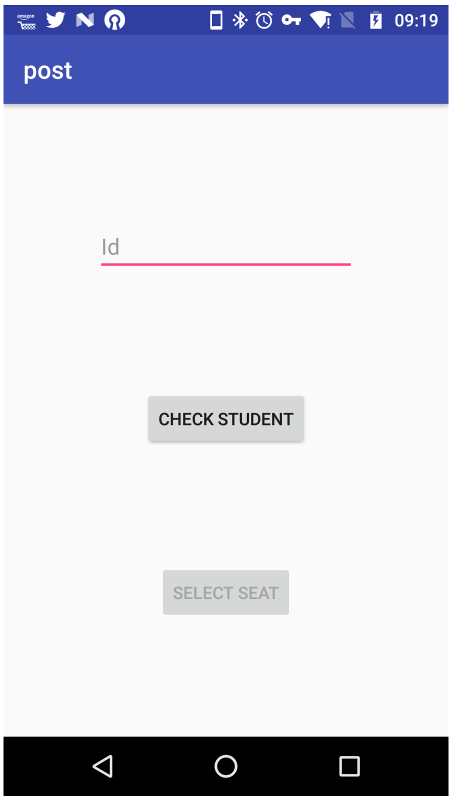

post

1 :	2 :
3 :	4 :
5 :	6 :
7 :	8 :
9 :	10 : 160101032
11 :	12 :
13 :	14 :
15 :	16 :
17 :	18 :
19 :	20 :

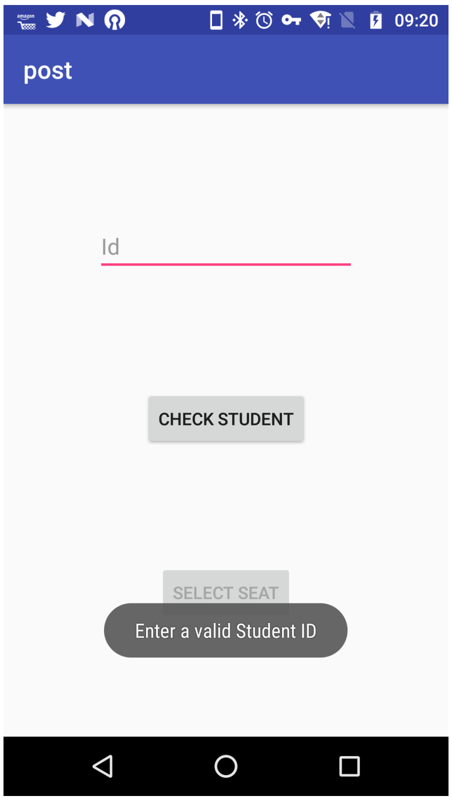
Figure 3.13: No change in the seat database

- Valid Student ID , Student already seated , internet connectivity
- Expected Result : Error message should be displayed regarding duplicate data.
- App Output : Though a message is shown regarding the seat being selected but no change occurs in the seat database (displaying the classroom after selecting the seat).

5. a) Student ID :
   b) Seat Number :
   c) Internet Connectivity = True



(a) Empty Student ID

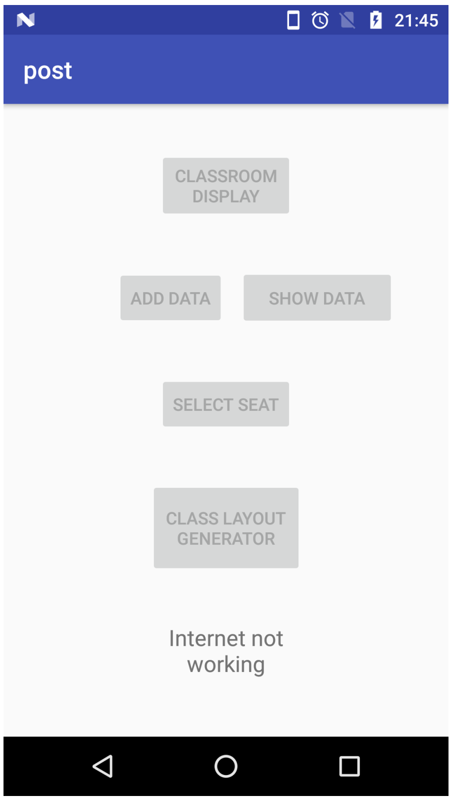


Enter a valid Student ID

(b) Error message

- Empty Student ID field , internet connectivity
- Expected Result : Warning should be given regarding no student being selected
- App Output : Error is displayed "Please enter a valid Student Id".

6. a) Student ID : 160101032
   b) Seat Number : 10
   c) Internet Connectivity = False



Internet not working

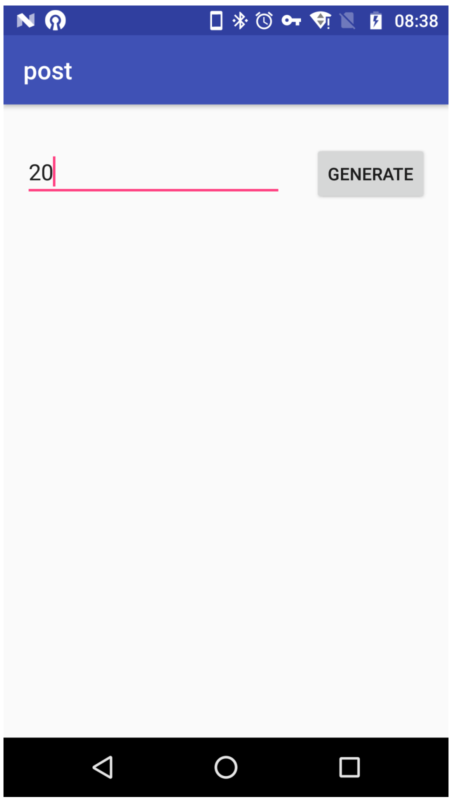
Figure 3.15: Features disabled if internet is not working

- No internet connectivity
- Expected Result : Select seat feature should be disabled
- App Output : The select seat button is disabled.

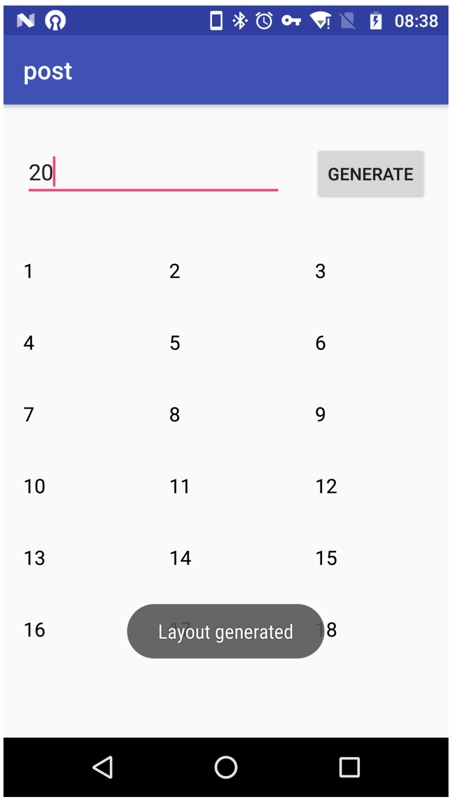
3.3 Classroom layout generator module

1. a) Seat Number : 20

   b) Internet Connectivity = True



(a) Valid Seat Number

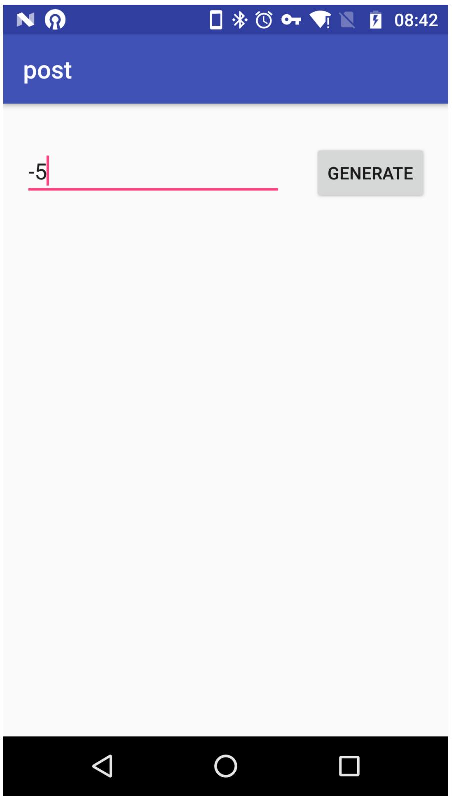


(b) Layout Screen

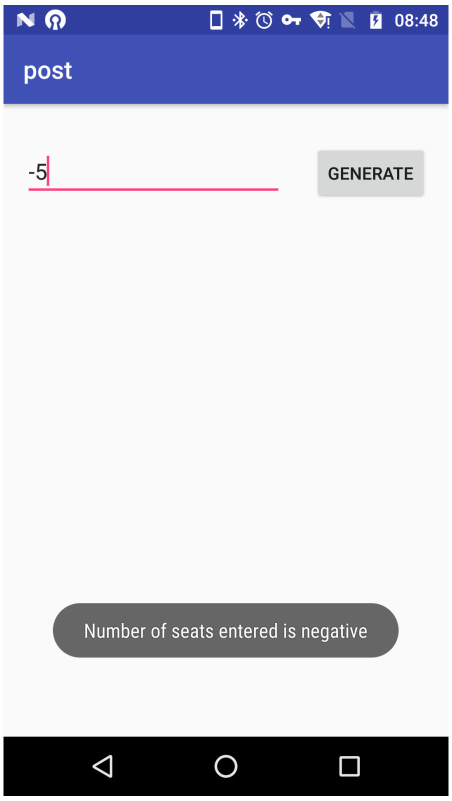
- Number of seats is a positive integer , internet connectivity
- Expected Result : Classroom layout should be generated with the given number of seats
- App Output: The layout is successfully generated with the given no. of seats.

2. a) Seat Number : -5

   b) Internet Connectivity = True



(a) Negative seat number

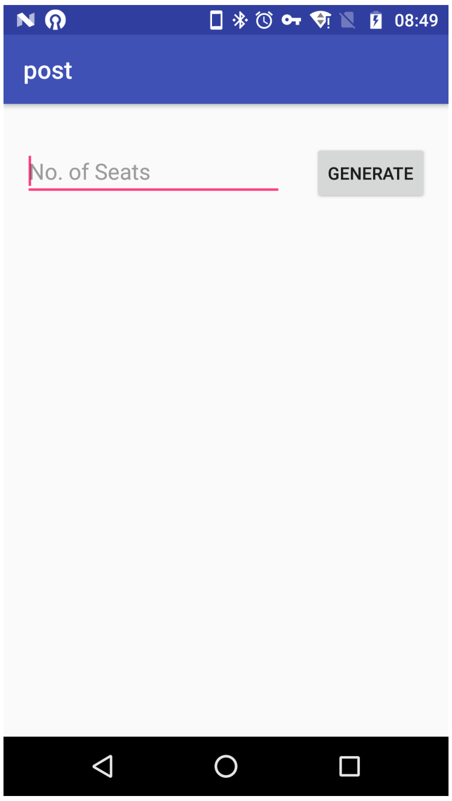


Number of seats entered is negative

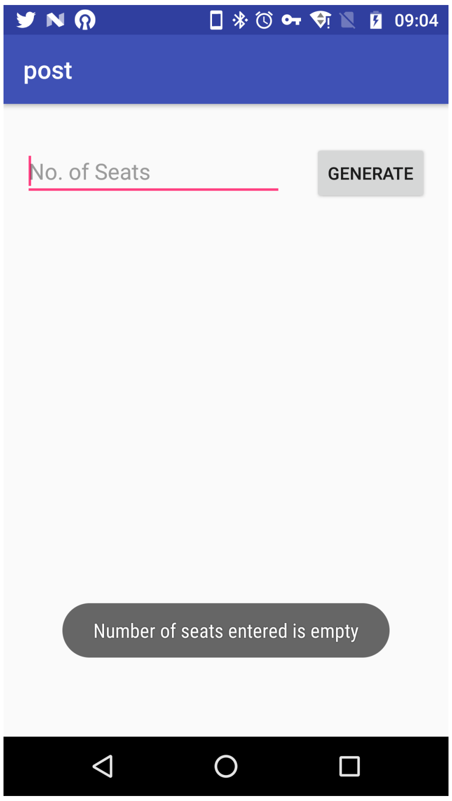
(b) Error message displayed

- Number of seats is not a positive integer , internet connectivity
- Expected Result :Error should be displayed regarding the number being invalid.
- App Output: Error message is displayed "The number entered is negative".

3. a) Seat Number :
   b) Internet Connectivity = True



(a) Empty Seat number field



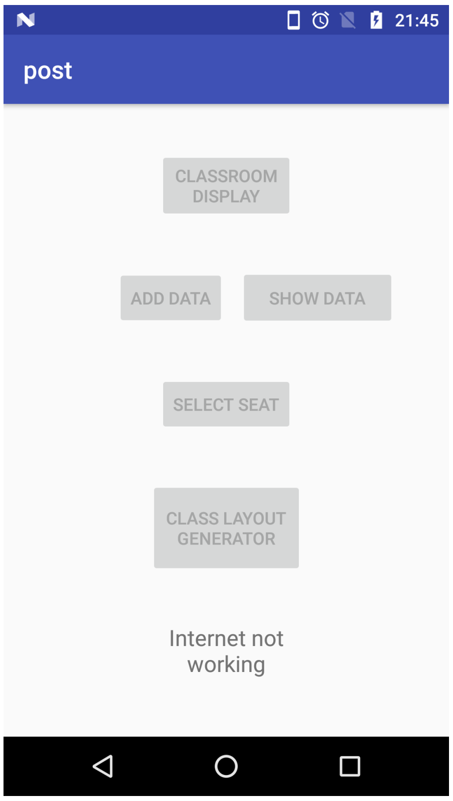
Number of seats entered is empty

(b) Error message displayed

- Field of number of seats is empty , internet connectivity
- Expected Result : No seat layout should be generated
- App Output: Error message is displayed " No. of seats field is empty".

4. a) Seat Number : 20

   b) Internet Connectivity = False



Internet not working

Figure 3.19: Features disabled if internet is not working

- No internet connectivity
- Expected Result : Layout generate feature should be disabled
- App Output: Generate classroom layout button is disabled.

4 Boundary Value Analysis

4.1 Add student module

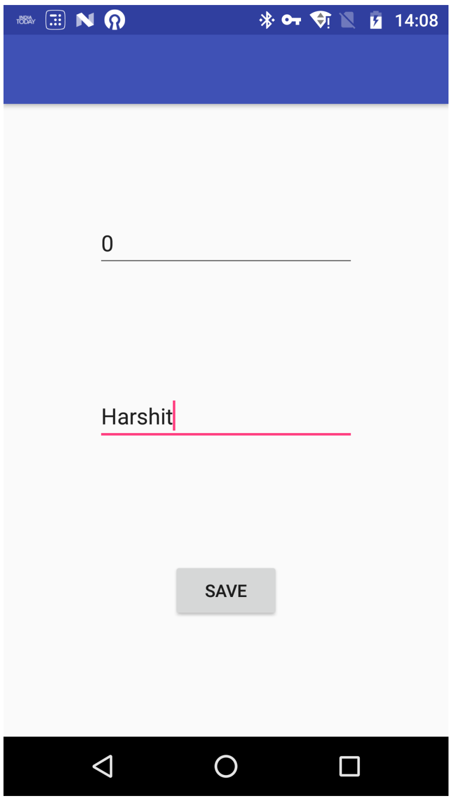
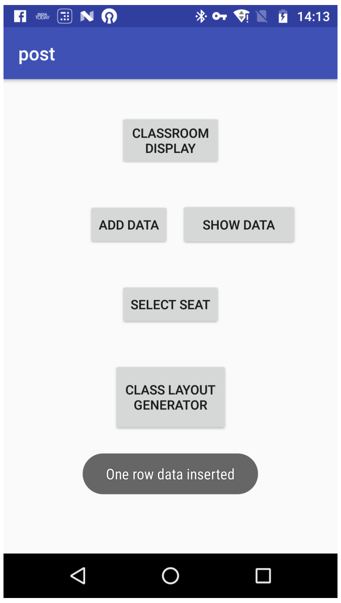
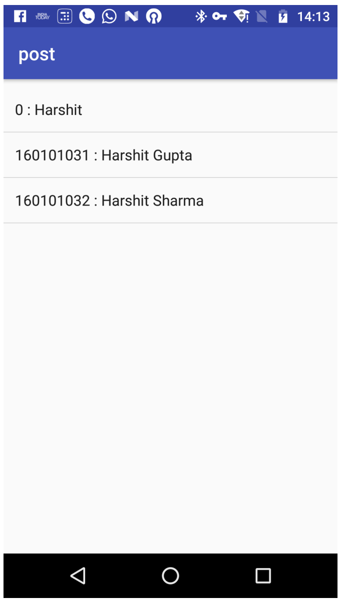


Figure 4.1: Add student screen



One row data inserted

Figure 4.2: Student with ID 0 inserted



0 : Harshit
160101031 : Harshit Gupta
160101032 : Harshit Sharma

Figure 4.3: Updated Database

- Student ID = 0 , name , internet connectivity
- Expected Result : Student should be successfully enrolled in the course
- App Output : A new row corresponding to student id = 0 is added in the database and pops up the message as shown above.

4.2 Classroom layout generator module

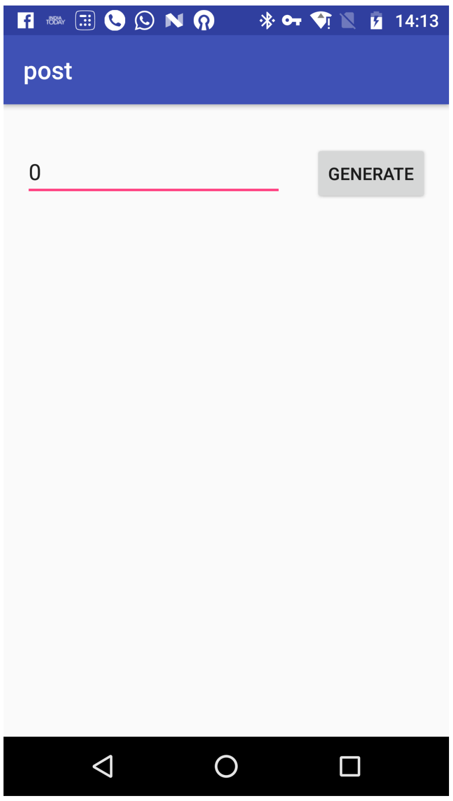


Figure 4.4: Number of seats = 0

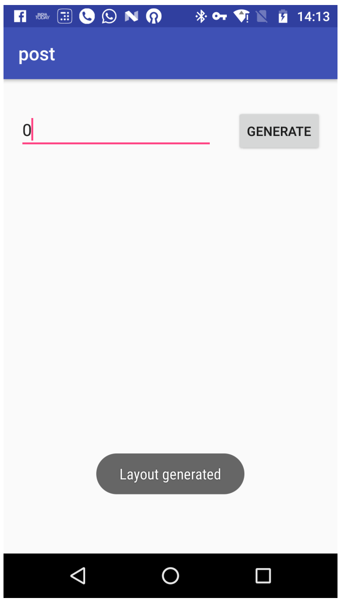


Figure 4.5: Layout generated

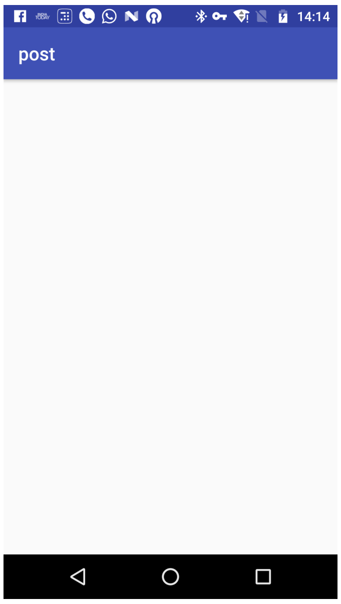


Figure 4.6: No seats are added as number of seats were zero

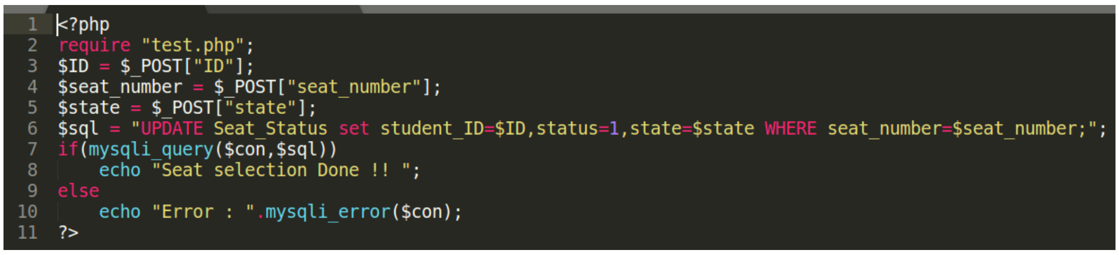
- Number of seats = 0 , internet connectivity
- Expected Result : No seats should be generated.
- App Output: Layout with the no. of seats is generated. The Display Classroom screen displays an empty screen as there are no seats in the classroom.

5 White Box Testing

Notation followed for describing CFG		
1 .	A→B	This means a direct edge between A and B eg - 5→6 means the path from node 5 to node 6
2.	A⇒B	This means a continuous path from A to B eg - 5⇒8 means the path from node 5 to node 8 i.e 5 to 6 , then 6 to 7 , then 7 to 8

5.1 Seat Status

This PHP script accepts Student ID and the seat selected by the student and update the database Also the script gives error if database connection is not established



```php
<?php
require "test.php";
$ID = $_POST["ID"];
$seat_number = $_POST["seat_number"];
$state = $_POST["state"];
$sql = "UPDATE Seat_Status set student_ID=$ID,status=1,state=$state WHERE seat_number=$seat_number;";
if(mysqli_query($con,$sql))
    echo "Seat selection Done !! ";
else
    echo "Error : ".mysqli_error($con);
?>
```

Figure 5.1: Php script to add select a seat

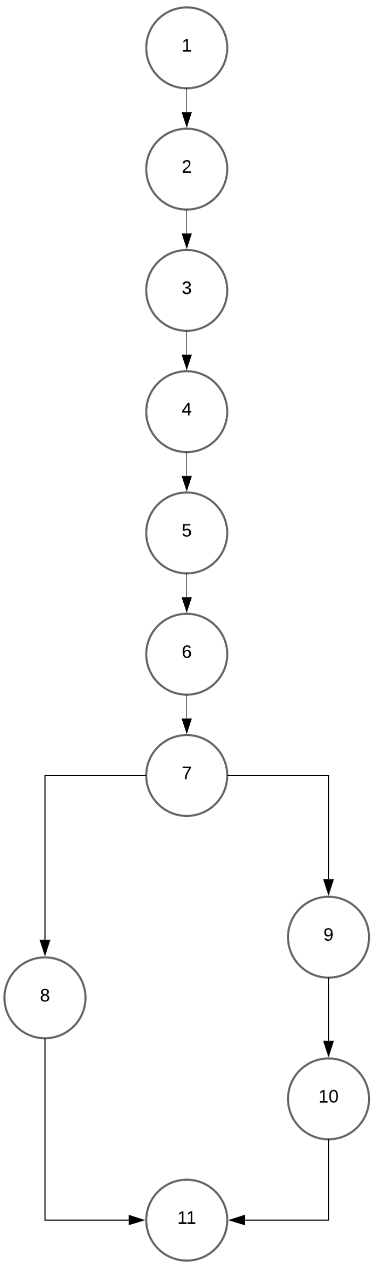


Figure 5.2: CFG of the code snippet

V = 11
E = 11
No . of bounded region = 1
E-V+2 = No. of bounded region +1 = 2

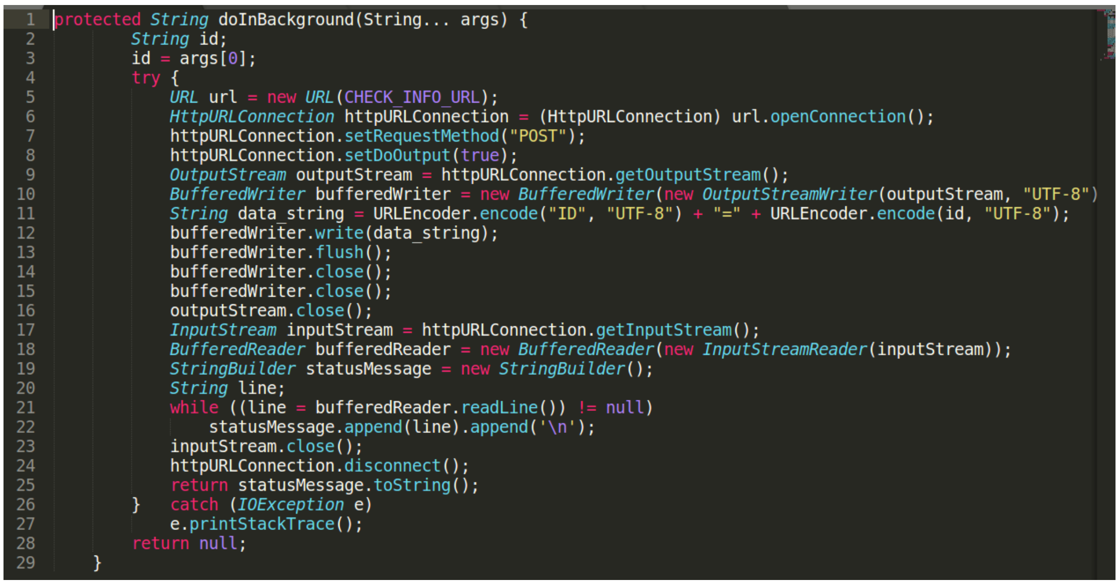
Test cases :

a) (1⇒7→9→10→11): Database connection established
b) (1⇒7→8→11) : Database connection not established

5.2 Background

The following code snippet starts a background process of opening a HTTP url connection and connect to the database via the post method request and then the request data is sent encoded in a data string The data is then read using the input stream and buffer reader and the response string is returned by the function . The function handles the input output exceptions generated by the output stream and input stream buffer

This do-in-background function is common to the all activities of the app and is a crucial part of communication to the database



```java
protected String doInBackground(String... args) {
        String id;
        id = args[0];
        try {
            URL url = new URL(CHECK_INFO_URL);
            HttpURLConnection httpURLConnection = (HttpURLConnection) url.openConnection();
            httpURLConnection.setRequestMethod("POST");
            httpURLConnection.setDoOutput(true);
            OutputStream outputStream = httpURLConnection.getOutputStream();
            BufferedWriter bufferedWriter = new BufferedWriter(new OutputStreamWriter(outputStream, "UTF-8")
            String data_string = URLEncoder.encode("ID", "UTF-8") + "=" + URLEncoder.encode(id, "UTF-8");
            bufferedWriter.write(data_string);
            bufferedWriter.flush();
            bufferedWriter.close();
            bufferedWriter.close();
            outputStream.close();
            InputStream inputStream = httpURLConnection.getInputStream();
            BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(inputStream));
            StringBuilder statusMessage = new StringBuilder();
            String line;
            while ((line = bufferedReader.readLine()) != null)
                statusMessage.append(line).append('\n');
            inputStream.close();
            httpURLConnection.disconnect();
            return statusMessage.toString();
        }   catch (IOException e)
            e.printStackTrace();
        return null;
    }
```

Figure 5.3: Database connection

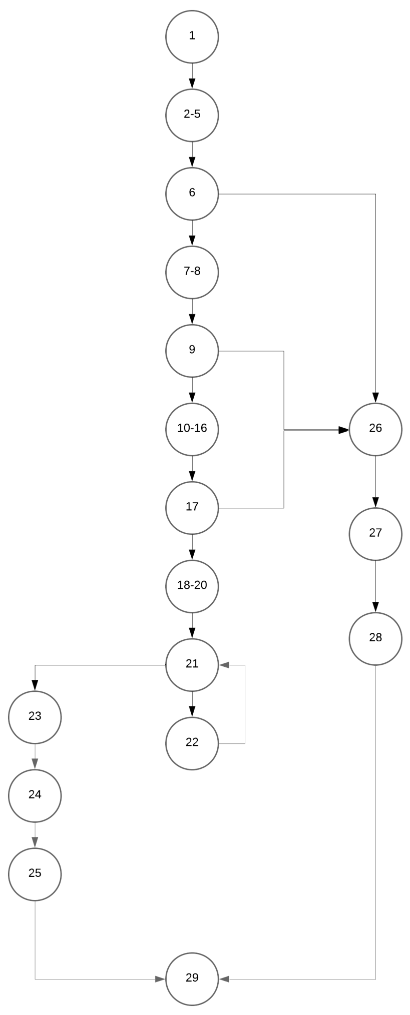


Figure 5.4: CFG of the code snippet

V = 17
E = 20
No . of bounded region = 4
E-V+2 = No. of bounded region +1 = 5
Test cases :

a) (1⇒6→26→27→28→29) : If the url connection is not established then the function throws an exception
b) (1⇒6⇒9→26→27→28→29) : If the output stream is null then the function throws an exception
c) (1⇒6⇒9⇒17→26→27→28→29) : If the input stream is null then the func-

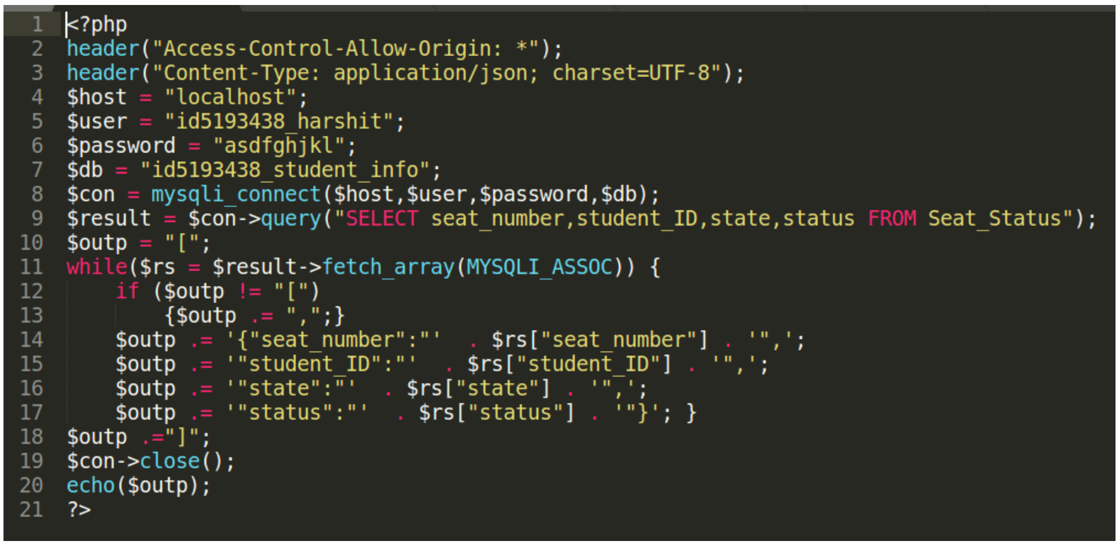
tion throws an exception

d) (1⇒21→23→24→25→29) : if all connections are fine and response from the database is empty .

e) (1⇒21→22→21→23→24→25→29) : if all connections are fine and response from the database is non empty .

5.3 Display

The following script establishes a connection to the seat info database and fetches the seat information in a Json formatted string .



```php
<?php
header("Access-Control-Allow-Origin: *");
header("Content-Type: application/json; charset=UTF-8");
$host = "localhost";
$user = "id5193438_harshit";
$password = "asdfghjkl";
$db = "id5193438_student_info";
$con = mysqli_connect($host,$user,$password,$db);
$result = $con->query("SELECT seat_number,student_ID,state,status FROM Seat_Status");
$outp = "[";
while($rs = $result->fetch_array(MYSQLI_ASSOC)) {
    if ($outp != "[")
        {$outp .= ",";}
    $outp .= '{"seat_number":"'  . $rs["seat_number"] . '",';
    $outp .= '"student_ID":"'  . $rs["student_ID"] . '",';
    $outp .= '"state":"'  . $rs["state"] . '",';
    $outp .= '"status":"'  . $rs["status"] . '"}'; }
$outp .="]";
$con->close();
echo($outp);
?>
```

Figure 5.5: PHP script to fetch the information of the seat database

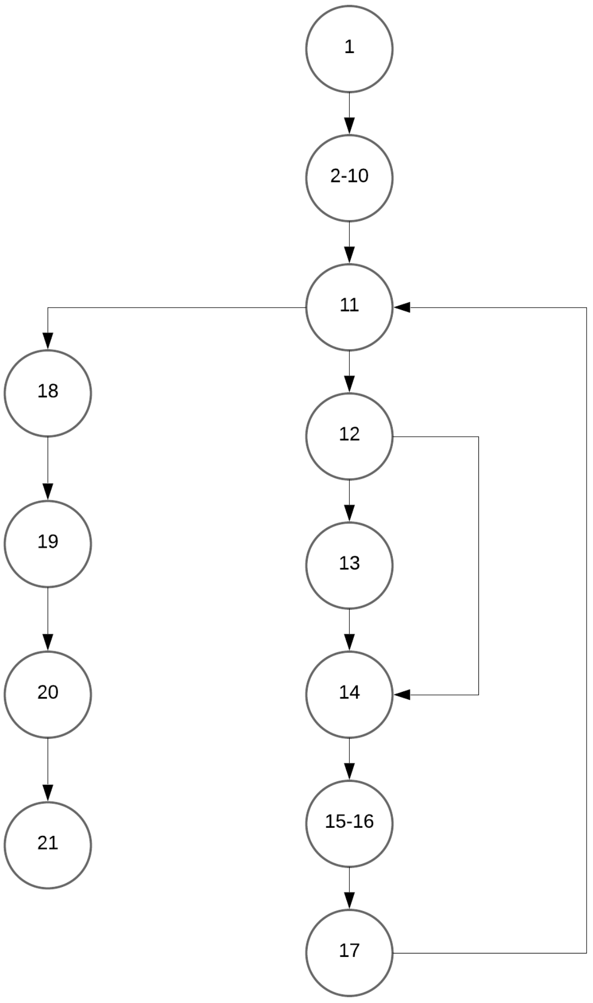


Figure 5.6: CFG of the code snippet

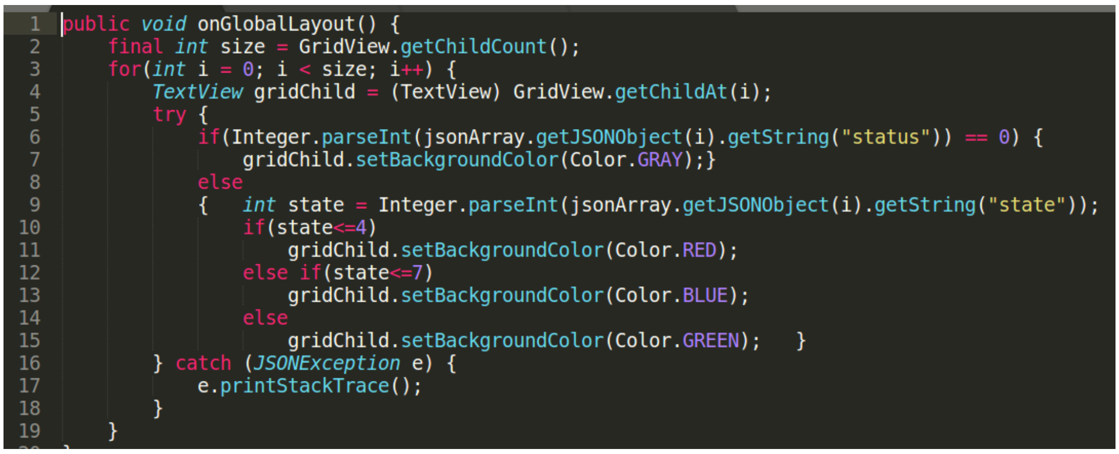
V = 12
E = 13
No . of bounded region = 2
E-V+2 = No. of bounded region +1 = 3
Test cases :

a) $(1 \Rightarrow 11 \rightarrow 18 \Rightarrow 21)$: if the response of the Sql query (seat database) is empty .
b) $(1 \Rightarrow 11 \rightarrow 12 \rightarrow 13 \rightarrow 14 \rightarrow 15 \Rightarrow 17 \rightarrow 11 \rightarrow 18 \Rightarrow 21)$: if the response of the Sql query (seat database) is non - empty and if the response is the last entry of the Json array .

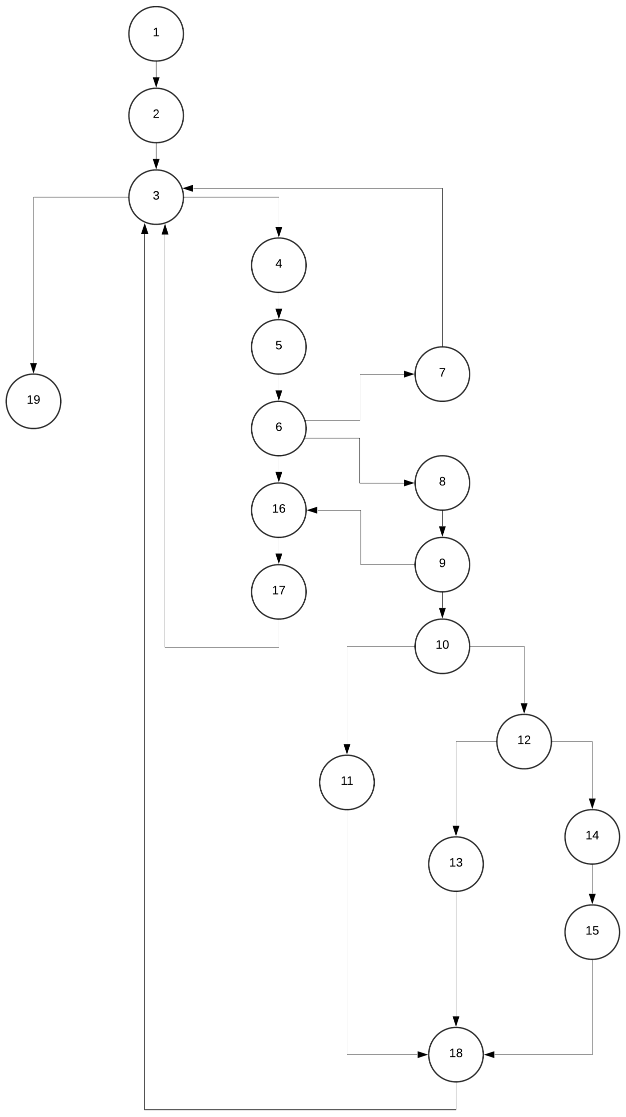
c) (1⇒11→12→14→15⇒17→11→18⇒21) : if the response of the Sql query (seat database) is non - empty and if the if the response is not the last entry of the Json array .

5.4 Classroom State View

On the basis of the seat database state value the function assigns the color to the seat in the Classroom View .



```java
public void onGlobalLayout() {
    final int size = GridView.getChildCount();
    for(int i = 0; i < size; i++) {
        TextView gridChild = (TextView) GridView.getChildAt(i);
        try {
            if(Integer.parseInt(jsonArray.getJSONObject(i).getString("status")) == 0) {
                gridChild.setBackgroundColor(Color.GRAY);}
            else
            {   int state = Integer.parseInt(jsonArray.getJSONObject(i).getString("state"));
                if(state<=4)
                    gridChild.setBackgroundColor(Color.RED);
                else if(state<=7)
                    gridChild.setBackgroundColor(Color.BLUE);
                else
                    gridChild.setBackgroundColor(Color.GREEN);   }
        } catch (JSONException e) {
            e.printStackTrace();
        }
    }
```



V = 19
E = 24
No . of bounded region = 6
E-V+2 = No. of bounded region +1 = 7
Test cases :

a) If the seat is unoccupied then the color of the seat is grey
b) If the seat is occupied then it is further subdivided into three sub cases :
    i. If the state is between 1 to 4 the color is red

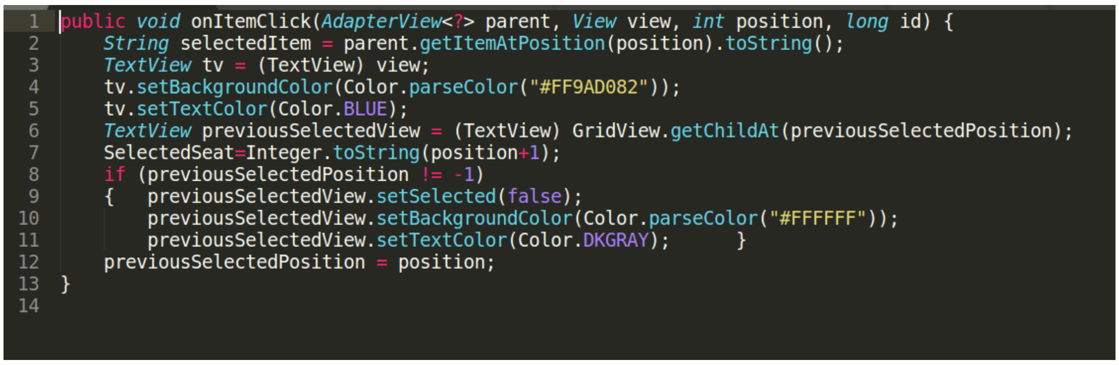
ii. If the state is between 5 to 7 the colour is blue

iii. If the state is between 8 to 10 the colour is green

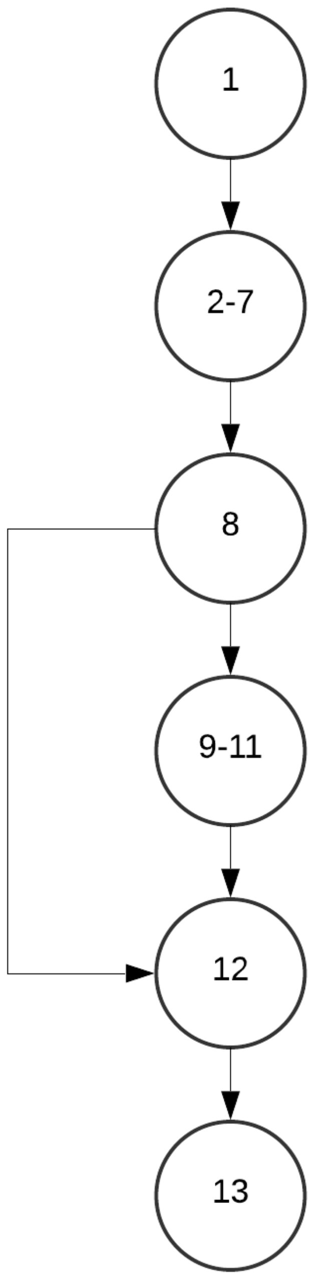
c) other test cases include exception due to due to the variables holding null values .

5.5 Select Seat

The function takes the choice of seat of the student as an input via touch on the screen . The seat selected is shown by a change of color of the selected Seat .



```java
public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
    String selectedItem = parent.getItemAtPosition(position).toString();
    TextView tv = (TextView) view;
    tv.setBackgroundColor(Color.parseColor("#FF9AD082"));
    tv.setTextColor(Color.BLUE);
    TextView previousSelectedView = (TextView) GridView.getChildAt(previousSelectedPosition);
    SelectedSeat=Integer.toString(position+1);
    if (previousSelectedPosition != -1)
    {   previousSelectedView.setSelected(false);
        previousSelectedView.setBackgroundColor(Color.parseColor("#FFFFFF"));
        previousSelectedView.setTextColor(Color.DKGRAY);      }
    previousSelectedPosition = position;
}
```



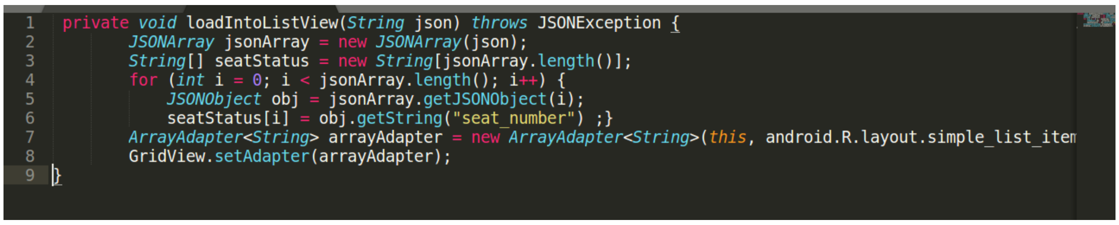
V = 6
E = 6
No . of bounded region = 1

E-V+2 = No. of bounded region +1 = 2
Test cases :

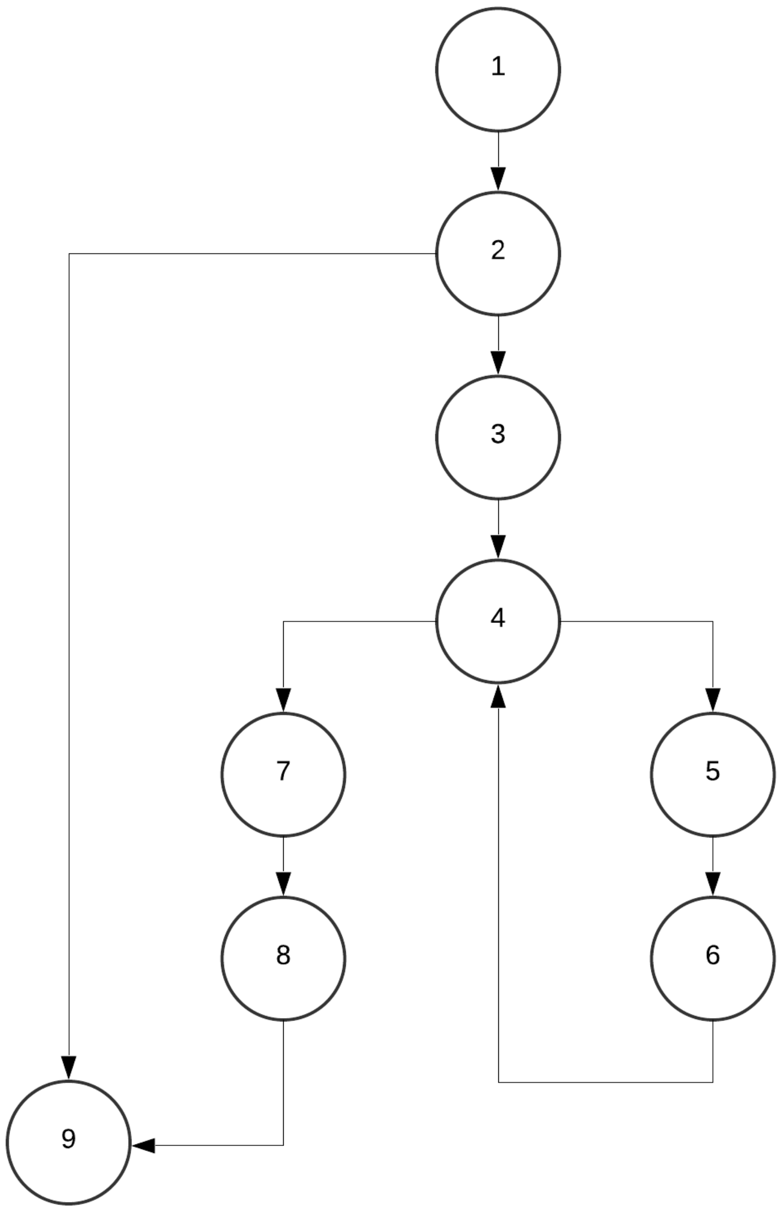
a) (1⇒7→8→9⇒13) : if the seat is selected
b) (1⇒7→8→12⇒13) : if the seat is not selected

5.6 Load List View

The following function takes a Json string as a parameter and loads the Grid view by the values supplied in the Json string



```
private void loadIntoListView(String json) throws JSONException {
        JSONArray jsonArray = new JSONArray(json);
        String[] seatStatus = new String[jsonArray.length()];
        for (int i = 0; i < jsonArray.length(); i++) {
            JSONObject obj = jsonArray.getJSONObject(i);
            seatStatus[i] = obj.getString("seat_number") ;}
        ArrayAdapter<String> arrayAdapter = new ArrayAdapter<String>(this, android.R.layout.simple_list_item
        GridView.setAdapter(arrayAdapter);
}
```



V = 9

E = 10

No . of bounded region = 2

E-V+2 = No. of bounded region +1 = 3

Test cases :

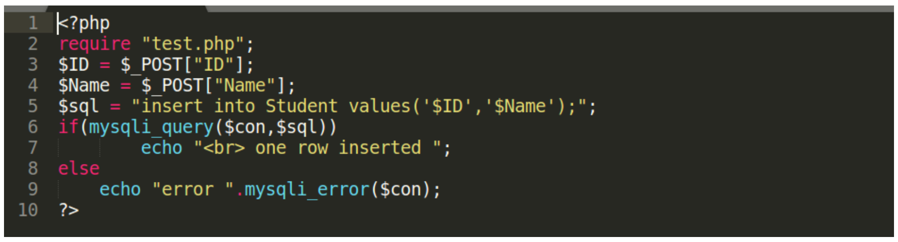
a) (1→2→9) : When the string json is null , it throws and exception

b) (1→2→3→4→7⇒9) : when the string json is not null and seat database is empty .

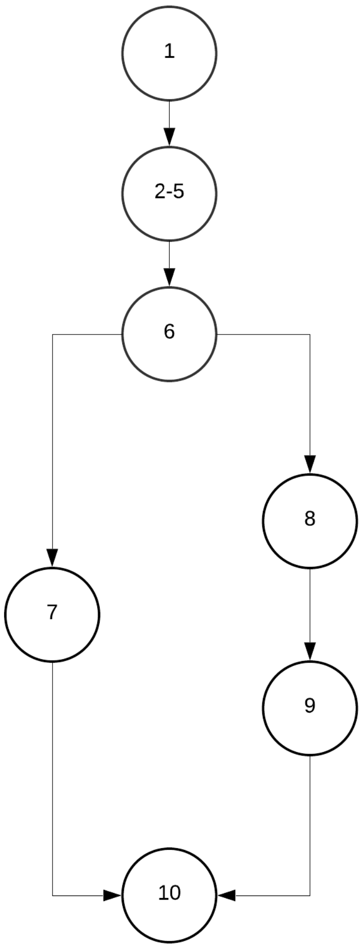
c) (1→2→3→4→5→6→4→7→9) : when the string json is not null and seat database non empty .

5.7 Add Student

The following PHP scripts takes Student ID and Name as an input in the POST request sent and inserts the data in the student information database .



```php
<?php
require "test.php";
$ID = $_POST["ID"];
$Name = $_POST["Name"];
$sql = "insert into Student values('$ID','$Name');";
if(mysqli_query($con,$sql))
        echo "<br> one row inserted ";
else
    echo "error ".mysqli_error($con);
?>
```



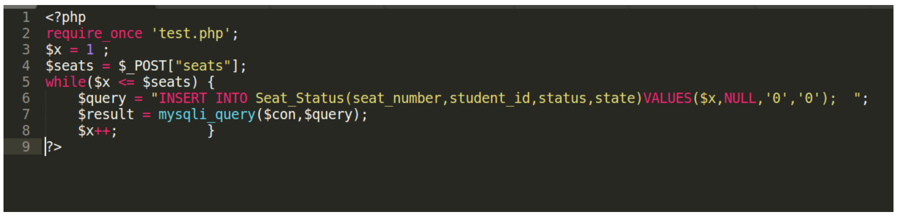
V = 7
E = 7
No . of bounded region = 1
E-V+2 = No. of bounded region +1 = 2

Test cases:

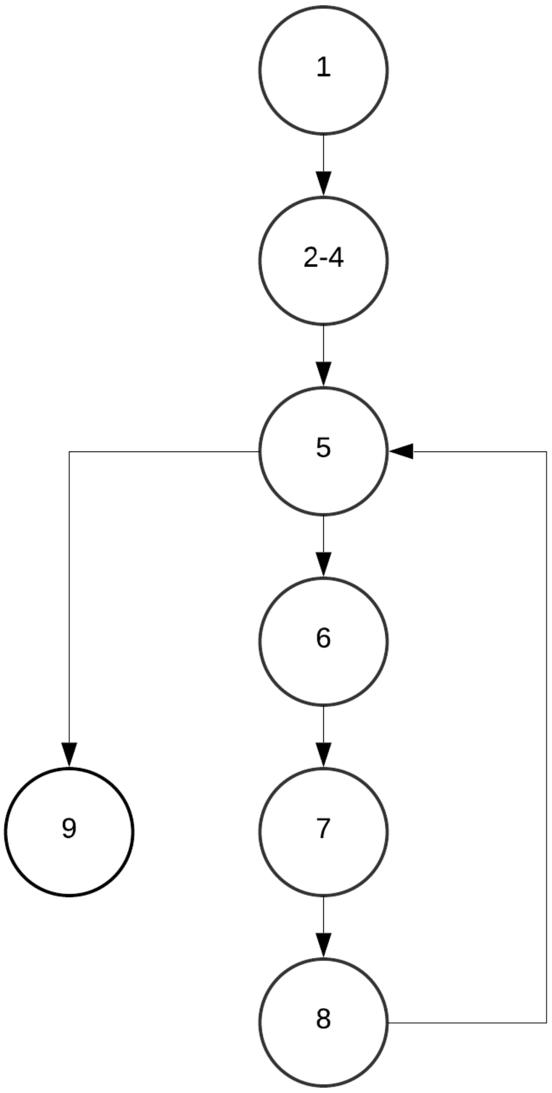
a) (1⇒6→7→10): If valid input is given and connection is established successfully .
b) (1⇒6→8→9→10): If invalid input is given or connection is not established successfully .

5.8 Classroom Layout Generator

The following PHP scripts takes number of seats as an input in the POST request sent and poplulates/initialises the seat database with the given number of seats .



```php
<?php
require_once 'test.php';
$x = 1 ;
$seats = $_POST["seats"];
while($x <= $seats) {
    $query = "INSERT INTO Seat_Status(seat_number,student_id,status,state)VALUES($x,NULL,'0','0');  ";
    $result = mysqli_query($con,$query);
    $x++;           }
?>
```



V = 7
E = 7
No . of bounded region = 1
E-V+2 = No. of bounded region +1 = 2

Test cases :

a) (1⇒5→9) : If the number of seats is less then one

b) (1⇒5→6⇒8→5→9) : if the number of seats supplied is greater than equal to one .

6 Conclusion

The black box and white box testing were quite successful in the black box testing . It was found that there are some loopholes which may lead to the crashing of the app that is the error handling should have been handled more carefully. The app crashed a couple of times while testing In white box testing, we traversed along the linear independent paths and tested the modules. Expected output matched with observed one in all cases.