Appendix A

User Guide

A.1 Instructions

To run the system please follow the steps below:

1. Start backend/nodeserver/server.js
2. Start backend/MLServer/MLServer.jar
3. Start IPS_Admin app and create buildings+rooms.
4. Measure rooms using the IPS_Admin. After the measurements are done, press the learn button in BuildingActivity
5. The MuseumGuide app can now be used to locate the device.

Appendix B

Example Files

B.1 domain.pddl

```
(define (domain museum)
(:requirements :typing :durative-actions :fluents)
(:types
   exhibit person - object
)
(:predicates
   (at ?p - person ?e - exhibit)
   (path ?e1 ?e2 - exhibit)
   (want-to-see ?e - exhibit)
   (visited ?e - exhibit)
   (open)
)
(:functions
   (time-to-walk ?e1 ?e2 - exhibit)
   (time-to-see ?e - exhibit)
   (excitement ?e - exhibit)
   (seen)
)


(:durative-action walk
  :parameters
   (?p - person
    ?e1 ?e2 - exhibit)
  :duration (= ?duration (time-to-walk ?e1 ?e2))
  :condition
   (and (over all (path ?e1 ?e2)) (at start (at ?p ?e1)) (over all (open)))
  :effect
   (and (at start (not (at ?p ?e1))) (at end (at ?p ?e2))))


(:durative-action view
  :parameters
   (?p - person
    ?e - exhibit)
  :duration (= ?duration (time-to-see ?e))
  :condition
   (and (over all (at ?p ?e)) (at start (want-to-see ?e)) (over all (open)))
  :effect
   (and (at start (not (want-to-see ?e))) (at start (increase (seen) (excitement ?e))))
)



)
```

B.2 problem.pddl

```
(define (problem simplemuseum)
(:domain museum)
(:objects
   visitor - person
   e0 e1 e2 - exhibit
)
(:init
(want-to-see e0)
(want-to-see e1)
(want-to-see e2)
(path e0 e1) (path e1 e0)
(= (time-to-walk e0 e1) 0)
(= (time-to-walk e1 e0) 0)
(path e0 e2) (path e2 e0)
(= (time-to-walk e0 e2) 0)
(= (time-to-walk e2 e0) 0)
(path e1 e2) (path e2 e1)
(= (time-to-walk e1 e2) 0)
(= (time-to-walk e2 e1) 0)
(= (time-to-see e0) 16)
(= (time-to-see e1) 12)
(= (time-to-see e2) 0)
(at visitor e0)
(= (seen) 0)
(= (excitement e0) 1)
(= (excitement e1) 1)
(= (excitement e2) 1)
(open)
(at 10000 (not (open)))
)
(:goal (and
;(visited e0)
;(visited e1)
;(visited e2)
)

)

(:metric maximize (seen))

)
```

B.3 learning.arff

```
@relation room


@attribute 54:22:f8:17:9a:db NUMERIC

@attribute 00:91:f9:4b:44:e0 NUMERIC

@attribute e2:91:f9:4b:44:e0 NUMERIC

@attribute c0:3e:0f:85:ab:9d NUMERIC

@attribute 00:91:f9:4b:44:e1 NUMERIC

@attribute 22:91:f9:4b:44:e1 NUMERIC

@attribute 2c:39:96:b4:c3:a6 NUMERIC

@attribute 44:e9:dd:6d:36:83 NUMERIC

@attribute 5c:7d:5e:0e:4b:cf NUMERIC

@attribute c2:91:f9:4b:44:e0 NUMERIC

@attribute class
    {5713ab468794d11d17c53d59,5713aa0e8794d11d17c53d57,5713aa6b8794d11d17c53d58,5713abae8794d11d17c53d5a}

@data

-47.0,-66.0,-60.0,-54.0,-76.0,-76.0,-86.0,-80.0,-67.0,-64.0,5713ab468794d11d17c53d59

-47.0,-66.0,-60.0,-48.0,-78.0,-78.0,-80.0,-84.0,-67.0,-64.0,5713ab468794d11d17c53d59

-45.0,-64.0,-63.0,-56.0,-88.0,-87.0,-88.0,-81.0,-79.0,-64.0,5713aa0e8794d11d17c53d57

-42.0,-71.0,-66.0,-52.0,-86.0,-86.0,-84.0,-83.0,-63.0,-69.0,5713aa0e8794d11d17c53d57

-47.0,-77.0,-74.0,-61.0,-90.0,0.0,-77.0,-79.0,-69.0,-73.0,5713aa6b8794d11d17c53d58

-54.0,-76.0,-74.0,-60.0,-90.0,0.0,-77.0,-74.0,-72.0,-73.0,5713aa6b8794d11d17c53d58

-28.0,-72.0,-67.0,-60.0,-90.0,-90.0,-73.0,0.0,-47.0,-73.0,5713abae8794d11d17c53d5a

-35.0,-70.0,-70.0,-59.0,-91.0,-90.0,-71.0,-87.0,-54.0,-73.0,5713abae8794d11d17c53d5a
```

B.4 unclassified.arff

```
@relation room


@attribute 54:22:f8:17:9a:db NUMERIC

@attribute 00:91:f9:4b:44:e0 NUMERIC

@attribute e2:91:f9:4b:44:e0 NUMERIC

@attribute c0:3e:0f:85:ab:9d NUMERIC

@attribute 00:91:f9:4b:44:e1 NUMERIC

@attribute 22:91:f9:4b:44:e1 NUMERIC

@attribute 2c:39:96:b4:c3:a6 NUMERIC

@attribute 44:e9:dd:6d:36:83 NUMERIC

@attribute 5c:7d:5e:0e:4b:cf NUMERIC

@attribute c2:91:f9:4b:44:e0 NUMERIC

@attribute class

    {5713ab468794d11d17c53d59,5713aa0e8794d11d17c53d57,5713aa6b8794d11d17c53d58,5713abae8794d11d17c53d5a}

@data

-41.0,-71.0,-67.0,-52.0,-86.0,-86.0,-86.0,-82.0,-63.0,-69.0,?
```

B.5 RPs.data

```
54:22:f8:17:9a:db

00:91:f9:4b:44:e0

e2:91:f9:4b:44:e0

c0:3e:0f:85:ab:9d

00:91:f9:4b:44:e1

22:91:f9:4b:44:e1

2c:39:96:b4:c3:a6

44:e9:dd:6d:36:83

5c:7d:5e:0e:4b:cf

c2:91:f9:4b:44:e0
```

B.6 rooms.data

```
5713ab468794d11d17c53d59

5713aa0e8794d11d17c53d57

5713aa6b8794d11d17c53d58

5713abae8794d11d17c53d5a
```

Appendix C

Source Code

C.1 backend/MLServer/five-seconds

```
#!/bin/bash

ulimit -t 1



./planner --optimise domain.pddl 5713a9028794d11d17c53d56_temp.pddl
```

C.2 backend/MLServer/domain.pddl

```
(:requirements :typing :durative-actions :fluents)
(:types
    exhibit person - object
)
(:predicates
    (at ?p - person ?e - exhibit)
    (path ?e1 ?e2 - exhibit)
    (want-to-see ?e - exhibit)
    (visited ?e - exhibit)
    (open)
)
(:functions
    (time-to-walk ?e1 ?e2 - exhibit)
    (time-to-see ?e - exhibit)
    (excitement ?e - exhibit)
    (seen)
)

(:durative-action walk
  :parameters
   (?p - person
    ?e1 ?e2 - exhibit)
  :duration (= ?duration (time-to-walk ?e1 ?e2))
  :condition
   (and (over all (path ?e1 ?e2)) (at start (at ?p ?e1)) (over all (open)))
  :effect
   (and (at start (not (at ?p ?e1))) (at end (at ?p ?e2))))

(:durative-action view
  :parameters
   (?p - person
    ?e - exhibit)
  :duration (= ?duration (time-to-see ?e))
  :condition
```

```
   (and (over all (at ?p ?e)) (at start (want-to-see ?e)) (over all (open)))

  :effect

   (and (at start (not (want-to-see ?e))) (at start (increase (seen) (excitement ?e))))

)




)
```

C.3 backend/MLServer/src/tools/Rectangle.java

```
import java.io.Serializable;


public class Rectangle implements Serializable{
    private Point coordinates;
    private double width, length;


    public Rectangle(Point coordinates, double width, double length){
        this.coordinates = coordinates;
        this.width = width;
        this.length = length;
    }


    public Point getCoordinates() {
        return coordinates;
    }


    public double getWidth() {
        return width;
    }


    public double getLength() {
        return length;
    }


    public void setCoordinates(Point coordinates) {
        this.coordinates = coordinates;
    }


    public void setWidth(double width) {
        this.width = width;
    }


    public void setLength(double length) {
        this.length = length;
    }


    public Point getCenter(){
        return new Point((width+coordinates.getX())/2,(length +coordinates.getY())/2);
    }


}
```

C.4 backend/MLServer/src/tools/Point.java

```java
import java.io.Serializable;


public class Point implements Serializable {
    private double x, y;

    public Point(double x, double y)
    {
        this.x = x;
        this.y = y;
    }

    public double getX() {
        return x;
    }

    public void setX(double x) {
        this.x = x;
    }

    public double getY() {
        return y;
    }

    public void setY(double y) {
        this.y = y;
    }
}
```

C.5 backend/MLServer/src/tools/RectangleDB.java

```java
import java.io.Serializable;



public class RectangleDB implements Serializable{
    private Point lt,rt,lb,rb;

    public RectangleDB(Point lt, Point rt, Point lb, Point rb) {
        this.lt = lt;
        this.rt = rt;
        this.lb = lb;
        this.rb = rb;
    }

    public Point getLt() {
        return lt;
    }

    public void setLt(Point lt) {
        this.lt = lt;
    }

    public Point getRt() {
        return rt;
    }

    public void setRt(Point rt) {
        this.rt = rt;
    }

    public Point getLb() {
        return lb;
    }

    public void setLb(Point lb) {
        this.lb = lb;
    }

    public Point getRb() {
        return rb;
    }

    public void setRb(Point rb) {
        this.rb = rb;
    }

    public boolean isNeighbour (RectangleDB r){
        if(this.getRt() == r.getLt() && this.getRb() == r.getLb() )
            return true;
        if(this.getLt() == r.getRt() && this.getLb() == r.getRb())
            return true;
        if(this.getLt() == r.getLb() && this.getRt() == r.getRb())
            return true;
        if(this.getLb() == r.getLt() && this.getRb() == this.getRt())
            return true;

        return true;



    }
}
```

C.6 backend/MLServer/src/models/Room.java

```
import tools.Point;
import tools.RectangleDB;
import tools.Rectangle;

import java.io.Serializable;

public class Room implements Serializable{
   private String roomName;
   private String id;
   private String building_id;
   private Rectangle roomRectangle;
   private RectangleDB rectangleDB;
   private String roomDescription;
   private double width, length;
   private double est_time;
   private int excitement;



   public Room(String id, String building_id, String roomName, RectangleDB
       rectangleDB,
             double width, double length, double est_time, int excitement){
      this.roomName = roomName;
      this.id = id;
      this.width = width;
      this.length = length;
      this.rectangleDB = rectangleDB;
      this.building_id = building_id;
      this.roomRectangle = new Rectangle(new Point(0,0),0,0); //for testing!!!
      this.est_time = est_time;
      this.excitement = excitement;
   }

   public String getBuilding_id() {
        return building_id;
    }

    public Room(String building_id, String roomName, RectangleDB rectangleDB, double
         width, double length){
        this.building_id = building_id;
        this.roomName = roomName;
        this.width = width;
        this.length = length;
        this.roomDescription = "";
        this.rectangleDB = rectangleDB;
        this.roomRectangle = new Rectangle(new Point(0,0),0,0); //for testing!!!
    }

    public Room(Rectangle r){
        this.roomRectangle = r;
        roomName = "";
        roomDescription = "";
    }

    public String getRoomName() {
        return roomName;
    }

    public void setRoomName(String roomName) {
        this.roomName = roomName;
    }

    public Rectangle getRoomRectangle() {
        return roomRectangle;
    }

    public void setRoomRectangle(Rectangle roomRectangle) {
        this.roomRectangle = roomRectangle;
    }
```

```
public String getRoomDescription() {
    return roomDescription;
}

public void setRoomDescription(String roomDescription) {
    this.roomDescription = roomDescription;
}

public boolean isNeighbour(Room room){

    if(this == room)
        return false;

    if(this.getRoomRectangle().getCoordinates().getX() +
         this.getRoomRectangle().getWidth()
            == room.getRoomRectangle().getCoordinates().getX()
            || this.getRoomRectangle().getCoordinates().getY() +
                this.getRoomRectangle().getLength()
            == room.getRoomRectangle().getCoordinates().getY()
            || room.getRoomRectangle().getCoordinates().getX() +
                room.getRoomRectangle().getWidth()
            == this.getRoomRectangle().getCoordinates().getX()
            || room.getRoomRectangle().getCoordinates().getY() +
                room.getRoomRectangle().getLength()
            == this.getRoomRectangle().getCoordinates().getY())
        return true;

    return false;

}

public String getId() {
    return id;
}

public RectangleDB getRectangleDB() {
        return rectangleDB;
    }


    public double getWidth() {
        return width;
    }


    public double getLength() {
        return length;
    }


    public void setEst_time(double est_time) {
        this.est_time = est_time;
    }


    public int getExcitement(){return excitement; }


    public double getEst_time() {


        return est_time;
    }


}
```

C.7 backend/MLServer/src/models/Floor.java

```
import java.io.Serializable;
import java.util.ArrayList;


public class Floor implements Serializable{
    Room[] rooms;

    public Floor(Room[] rooms)
    {
        this.rooms = rooms;
    }

    public Room[] getRooms() {
        return rooms;
    }

    public Room[] getNeighbours(Room room) {
        ArrayList<Room> neighbours = new ArrayList<>();
        for(Room r:rooms){
            if(r.isNeighbour(room))
                neighbours.add(r);
        }
        return neighbours.toArray(new Room[neighbours.size()]);
    }

    public void setRooms(Room[] rooms) {
        this.rooms = rooms;
    }

}
```

C.8 backend/MLServer/src/models/Building.java

```java
import tools.RectangleDB;
import java.io.Serializable;


public class Building implements Serializable{
   private RectangleDB rectangle;
   private String name;
   private double width;
   private double length;
   private String id;
   private Room[] rooms;



   public Building(String id, RectangleDB rectangle, String name, double width,
       double length, Room[] rooms) {
      this.rectangle = rectangle;
      this.id = id;
      this.name = name;
      this.width = width;
      this.length = length;
      this.rooms = rooms;
   }

   public Building(RectangleDB rectangle, String name, double width, double length) {
      this.rectangle = rectangle;
      this.id = "";
      this.name = name;
      this.width = width;
      this.length = length;
   }

```

```
public void setRectangle(RectangleDB rectangle) {
    this.rectangle = rectangle;
}

public void setName(String name) {
    this.name = name;
}

public void setWidth(double width) {
    this.width = width;
}

public void setLength(double length) {
    this.length = length;
}

public RectangleDB getRectangle() {
    return rectangle;
}

public String getName() {
    return name;
}

public double getWidth() {
    return width;
}

public double getLength() {
    return length;
}

public String getId() {
    return id;
}
```

```
    public void setId(String id) {
        this.id = id;
    }

    public Room[] getRooms() {
        return rooms;
    }

    public void setRooms(Room[] rooms) {
        this.rooms = rooms;
    }
}
```

C.9 backend/MLServer/src/com/Planner.java

```java
import models.indoormapping.Room;
import tools.Point;
import org.json.JSONArray;
import org.json.JSONObject;
import tools.RectangleDB;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.nio.charset.Charset;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.ArrayList;
import java.util.Arrays;

public class Planner {
    private static final String[] pddlIntro = {
        "(define (problem simplemuseum)",
        "(:domain museum)",
        "(:objects",
        "    visitor - person"
    };

    public static String route(JSONArray jsonArray, int deadline, String building_id){
        ArrayList<Room> rooms = getRoomsFromJSON(jsonArray);
        makePDDL(rooms,deadline);
        building_id = building_id.replaceAll("[\\s&;]+","");
        ArrayList<String> toExecute = new ArrayList<>();
        toExecute.add("#!/bin/bash\n" +
                "ulimit -t 1\n");
        toExecute.add( "./planner " + "--optimise " + "domain.pddl
            "+building_id+"_temp.pddl");
        Path execPath = Paths.get("five-seconds");
        try {
            Files.write(execPath,toExecute, Charset.forName("UTF-8"));
        } catch (IOException e) {
            e.printStackTrace();
        }
        String command = "./five-seconds";
        Process process = null;
        try {
            process = Runtime.getRuntime().exec(command);
        } catch (IOException e) {
            e.printStackTrace();
        }
        BufferedReader reader = new BufferedReader(new InputStreamReader(
                process.getInputStream()));
        ArrayList<String> output = new ArrayList<>();
        String s;
        try {
            while ((s = reader.readLine()) != null) {
                System.out.println("Script output: " + s);
                output.add(s);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }

        return makeFinalResult(parseOutput(output),rooms);

    }
    public static ArrayList<Room> getRoomsFromJSON(JSONArray jsonArray){

        ArrayList<Room> toReturn = new ArrayList<>();
        for(Object o:jsonArray){
            JSONObject room = (JSONObject) o;
            Point lt = new
                 Point(room.getJSONObject("rectangle").getJSONObject("lt").getDouble("x"),
                room.getJSONObject("rectangle").getJSONObject("lt").getDouble("y"));
        Point rt = new
            Point(room.getJSONObject("rectangle").getJSONObject("rt").getDouble("x"),
                room.getJSONObject("rectangle").getJSONObject("rt").getDouble("y"));
        Point lb = new
            Point(room.getJSONObject("rectangle").getJSONObject("lb").getDouble("x"),
                room.getJSONObject("rectangle").getJSONObject("lb").getDouble("y"));
        Point rb = new
            Point(room.getJSONObject("rectangle").getJSONObject("rb").getDouble("x"),
                room.getJSONObject("rectangle").getJSONObject("rb").getDouble("y"));
        RectangleDB r = new RectangleDB(lt,rt,lb,rb);
        Room toAdd = new
            Room(room.getString("_id"),room.getString("building_id"),room.getString("name"),r,
                room.getDouble("width"),room.getDouble("length"),
                room.getDouble("est_time"),room.getInt("excitement"));
        toReturn.add(toAdd);
    }
    return toReturn;
}

private static void makePDDL(ArrayList<Room> rooms, int deadline){

    Path tempPath = Paths.get(rooms.get(0).getBuilding_id()+"_temp.pddl");
    ArrayList<String> toWrite = new ArrayList<>();
    toWrite.addAll(Arrays.asList(pddlIntro));
    toWrite.add("   ");
    for(Room r:rooms){
        toWrite.set(toWrite.size()-1,toWrite.get(toWrite.size()-1)+"e"+rooms.indexOf(r)+"
            ");
        System.out.println("ROOM CREATED: "+"\nINDEX: "+rooms.indexOf(r)+"
            "+r.getRoomName());
    }
    toWrite.set(toWrite.size()-1,toWrite.get(toWrite.size()-1)+"- exhibit");
    toWrite.add(")");
    toWrite.add("(:init");
    for(Room r:rooms){
        toWrite.add("(want-to-see "+"e"+rooms.indexOf(r)+")");
    }

    toWrite.addAll(getBuildingLayout(rooms));
    for(Room r:rooms){
        toWrite.add("(= (time-to-see "+"e"+rooms.indexOf(r)+") "+(int)
            r.getEst_time()+")");
    }
    toWrite.add("(at visitor "+"e0"+")");
    toWrite.add("(= (seen) 0)");
    for(Room r:rooms){
        toWrite.add("(= (excitement "+"e"+rooms.indexOf(r)+") "+
            r.getExcitement()+(")"));
    }
    toWrite.add("(open)");
    toWrite.add("(at "+deadline+" (not (open)))");
    toWrite.add(")");
    toWrite.add("(:goal (and");

    for(Room r:rooms){
        toWrite.add(";(visited "+"e"+rooms.indexOf(r)+")");
    }
    toWrite.add(")");
    toWrite.add(")");
    toWrite.add("(:metric maximize (seen))");
    toWrite.add(")");
    try {
        Files.write(tempPath,toWrite, Charset.forName("UTF-8"));
    } catch (IOException e) {
        e.printStackTrace();
    }
}

private static ArrayList<String> getBuildingLayout(ArrayList<Room> rooms){
    ArrayList<String> toReturn = new ArrayList<>();
    for(int i = 0; i<rooms.size()-1; ++i){
        for(int j=i+1; j<rooms.size();++j){
            if(rooms.get(i).getRectangleDB().isNeighbour(rooms.get(j).getRectangleDB())){
                toReturn.add("(path " + "e"+i + " "+"e"+ j + ") (path " +"e"+j+"
                    "+"e"+i+")");
                toReturn.add("(= (time-to-walk "+"e"+i+" "+"e"+j+") 0)");
                toReturn.add("(= (time-to-walk "+"e"+j+" "+"e"+i+") 0)");
            }
        }
    }
    return toReturn;
}

private static ArrayList<String> parseOutput(ArrayList<String> output){
    ArrayList<String> toReturn = new ArrayList<>();
    ArrayList<String> currentResult = new ArrayList<>();

    for(String s:output){
        if(s.equals(";;;; Solution Found")) {
            toReturn = currentResult;
            currentResult.clear();
        }
        if(!((s.startsWith(";")|| s.isEmpty()))) {
            if (s.contains("(") && s.contains(")")) {
                currentResult.add( s.substring(s.indexOf("(")+1, s.indexOf(")")));
            }
        }
        if(s.startsWith("..")){
            toReturn = currentResult;
        }
    }

   return toReturn;
}

private static String makeFinalResult(ArrayList<String> output, ArrayList<Room>
     rooms){
```

```
        String toReturn = "";
        for(String s:output){
            String[] tempSplit = s.split(" ");
            if(tempSplit[0].equals("view")){
                int roomIndex
                    =Integer.parseInt(tempSplit[2].substring(tempSplit[2].indexOf("e")+1));
                toReturn += "view:"+rooms.get(roomIndex).getId()+";";

            } else if(tempSplit[0].equals("walk")){
                int roomIndexOr =
                    Integer.parseInt(tempSplit[2].substring(tempSplit[2].indexOf("e")+1));
                int roomIndexDest =
                    Integer.parseInt(tempSplit[3].substring(tempSplit[3].indexOf("e")+1));

                toReturn +=
                    "walk:"+rooms.get(roomIndexOr).getId()+","+rooms.get(roomIndexDest).getId()+";";
            }
        }

        return toReturn.substring(0,toReturn.length()-1);
    }
}
```

C.10 backend/MLServer/src/com/Server.java

```java
import org.apache.commons.lang3.tuple.Pair;
import org.json.JSONObject;
import weka.classifiers.bayes.BayesNet;
import weka.classifiers.bayes.NaiveBayes;

import java.net.*;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.util.*;
import java.io.*;

public class Server
{
    String currentBID;
    ArrayList<Pair<String,BayesNet>> classifiersBN;
    ArrayList<Pair<String,NaiveBayes>> classifiersNB;
    public Server()
    {
        classifiersBN = new ArrayList<>();
        classifiersNB = new ArrayList<>();

        try {
            ServerSocket sSocket = new ServerSocket(5000);
            System.out.println("Server started!");
            currentBID = "";

            while(true) {
                Socket socket = sSocket.accept();
                ClientThread cT = new ClientThread(socket);
                new Thread(cT).start();
            }
        } catch(IOException exception) {
            System.out.println("Error: " + exception);
    }
}

class ClientThread implements Runnable
{
    Socket socket;
    public ClientThread(Socket socket)
    {
        this.socket = socket;
    }

    public void run()
    {
        try {
            PrintWriter pw = new PrintWriter(socket.getOutputStream(), true);
            BufferedReader br = new BufferedReader(new
                 InputStreamReader(socket.getInputStream()));
            String rec = br.readLine();
            while (rec == null) {
                rec = br.readLine();
            }
            System.out.println(rec);
            JSONObject recJSON = new JSONObject(rec);
            if(recJSON.getString("command").equals("learn")){
                BayesNet bn =
                    Learner.learnFromJSON_BN(recJSON.getString("building_id"),
                       recJSON.getJSONArray("learning_set"));
                NaiveBayes nb =
                    Learner.learnFromJSON_NB(recJSON.getString("building_id"),
                       recJSON.getJSONArray("learning_set"));
                int i =
                    buildingClassifierBNInitialised(recJSON.getString("building_id"));
                int j =
                    buildingClassifierNBInitialised(recJSON.getString("building_id"));
                    //not really needed.
                if(i>-1){
```

```
                classifiersBN.set(i,Pair.of(recJSON.getString("building_id"),bn));
            } else {
                classifiersBN.add(Pair.of(recJSON.getString("building_id"),bn));
            }

            if(i>-1){
                classifiersNB.set(i,Pair.of(recJSON.getString("building_id"),nb));
            } else {
                classifiersNB.add(Pair.of(recJSON.getString("building_id"),nb));
            }
            pw.write("Done!");
        } else if(recJSON.getString("command").equals("classify")){
            int i =
                buildingClassifierBNInitialised(recJSON.getString("building_id"));
            int j =
                buildingClassifierNBInitialised(recJSON.getString("building_id"));
                //same

            if(i>-1){
                String res
                    ="BN:"+Learner.classify_BN(recJSON.getString("building_id"),
                       recJSON.getJSONArray("learning_set"),classifiersBN.get(i).getRight());
                res +=
                    ",NB:"+Learner.classify_NB(recJSON.getString("building_id"),
                       recJSON.getJSONArray("learning_set"),classifiersNB.get(j).getRight());
                pw.write(res);
                System.out.println(res);
            } else
                if(Files.exists(Paths.get(recJSON.getString("building_id")+".arff"))){
                BayesNet bn =
                    Learner.getClassifierBN(recJSON.getString("building_id"));
                NaiveBayes nb =
                    Learner.getClassifierNB(recJSON.getString("building_id"));
                classifiersBN.add(Pair.of(recJSON.getString("building_id"),bn));
                classifiersNB.add(Pair.of(recJSON.getString("building_id"),nb));
```

```
                    i =
                        buildingClassifierBNInitialised(recJSON.getString("building_id"));
                    j =
                        buildingClassifierNBInitialised(recJSON.getString("building_id"));
                        //same
                    String res
                        ="BN:"+Learner.classify_BN(recJSON.getString("building_id"),
                           recJSON.getJSONArray("learning_set"),classifiersBN.get(i).getRight());
                    res +=
                        ",NB:"+Learner.classify_NB(recJSON.getString("building_id"),
                           recJSON.getJSONArray("learning_set"),classifiersNB.get(j).getRight());
                    pw.write(res);
                } else {
                    pw.write("Weird classify request!");
                }
            }  else if(recJSON.getString("command").equals("route")){
                pw.write(Planner.route(recJSON.getJSONArray("request_set"),recJSON.getInt("deadline"),
                       recJSON.getString("building_id")));
            }
            pw.flush();
        } catch(IOException exception) {
            System.out.println("Error: " + exception);
        }
    }
}

public int buildingClassifierBNInitialised(String building_id){
    for(Pair<String,BayesNet> p:classifiersBN){
        if(p.getLeft().equals(building_id))
            return classifiersBN.indexOf(p);
    }
    return -1;
}

public int buildingClassifierNBInitialised(String building_id){
    for(Pair<String,NaiveBayes> p:classifiersNB){
            if(p.getLeft().equals(building_id))
                return classifiersNB.indexOf(p);
        }
        return -1;
    }
}
```

C.11 backend/MLServer/src/com/Main.java

```
public class Main {

    public static void main(String[] args) {
        new Server();
    }
}
```

C.12 backend/MLServer/src/com/Learner.java

```
import org.json.JSONArray;
import org.json.JSONObject;
import weka.classifiers.bayes.BayesNet;
import weka.classifiers.bayes.NaiveBayes;
import weka.core.Instances;

import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.nio.charset.Charset;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.ArrayList;
import java.util.LinkedHashSet;
import java.util.Set;


public class Learner {

   public static NaiveBayes learnFromJSON_NB(String building_id, JSONArray JSONData){
      makeLearnerARFFfromJSON(building_id,JSONData);
      NaiveBayes nb = new NaiveBayes();
      try {
         Instances instances = new Instances(new BufferedReader(new
             FileReader(building_id+".arff")));
         instances.setClassIndex(instances.numAttributes()-1);
         nb.buildClassifier(instances);
      } catch (Exception e) {
         e.printStackTrace();
      }

      return nb;
```

```
}

public static BayesNet learnFromJSON_BN(String building_id, JSONArray JSONData){
    makeLearnerARFFfromJSON(building_id,JSONData);
    BayesNet bn = new BayesNet();
    try {
        Instances instances = new Instances(new BufferedReader(new
            FileReader(building_id+".arff")));
        instances.setClassIndex(instances.numAttributes()-1);
        bn.buildClassifier(instances);
    } catch (Exception e) {
        e.printStackTrace();
    }

    return bn;
}

public static BayesNet getClassifierBN(String building_id){
    BayesNet bn = new BayesNet();
    try {
        Instances instances = new Instances(new BufferedReader(new
            FileReader(building_id+".arff")));
        instances.setClassIndex(instances.numAttributes()-1);
        bn.buildClassifier(instances);
    } catch (Exception e) {
        e.printStackTrace();
    }

    return bn;
}

public static NaiveBayes getClassifierNB(String building_id){
    NaiveBayes nb = new NaiveBayes();
    try {
        Instances instances = new Instances(new BufferedReader(new
            FileReader(building_id+".arff")));
        instances.setClassIndex(instances.numAttributes()-1);
        nb.buildClassifier(instances);
    } catch (Exception e) {
        e.printStackTrace();
    }

    return nb;
}

public static String classify_NB(String building_id, JSONArray JSONdata,
     NaiveBayes nb){
    makeClassifierARFF(building_id,JSONdata);
    Instances unlabeled;
    Instances labeled = null;
    double clsLabel = 0;
    try {
        unlabeled = new Instances(new BufferedReader(new
            FileReader(building_id+"_temp.arff")));
        unlabeled.setClassIndex(unlabeled.numAttributes()-1);
        labeled = new Instances(unlabeled);
        clsLabel = nb.classifyInstance(unlabeled.firstInstance());
        labeled.firstInstance().setClassValue(clsLabel);
        Files.deleteIfExists(Paths.get(building_id+"_temp.arff"));
    } catch (Exception e) {
        e.printStackTrace();
    }
    if (labeled != null) {
        return labeled.instance(0).classAttribute().value((int)clsLabel);
    }
    return "na";

}

public static String classify_BN(String building_id, JSONArray JSONdata, BayesNet
     bn){
    makeClassifierARFF(building_id,JSONdata);
```

```
        Instances unlabeled;
        Instances labeled = null;
        double clsLabel = 0;
        try {
            unlabeled = new Instances(new BufferedReader(new
                FileReader(building_id+"_temp.arff")));
            unlabeled.setClassIndex(unlabeled.numAttributes()-1);
            labeled = new Instances(unlabeled);
            clsLabel = bn.classifyInstance(unlabeled.firstInstance());
            labeled.firstInstance().setClassValue(clsLabel);
            Files.deleteIfExists(Paths.get(building_id+"_temp.arff"));
        } catch (Exception e) {
            e.printStackTrace();
        }
        if (labeled != null) {
            return labeled.instance(0).classAttribute().value((int)clsLabel);
        }
        return "na";
    }

    public static void makeLearnerARFFfromJSON(String building_id, JSONArray JSONData){
        Path arffPath = Paths.get(building_id+".arff");
        Path roomsPath = Paths.get(building_id+"_rooms.data");
        Path rpPath = Paths.get(building_id+"_RPs.data");

        ArrayList<String> arffData = new ArrayList<>();
        ArrayList<String> rpids = getRPs(JSONData);
        ArrayList<String> rooms = getRooms(JSONData);

        arffData.add("@relation room"); arffData.add("");
        for(String s:rpids){
            arffData.add("@attribute "+s+" NUMERIC");
        }
        arffData.add("@attribute class {");
        for(int i=0;i<rooms.size()-1;++i){
            arffData.set(arffData.size()-1,arffData.get(arffData.size()-1)+rooms.get(i)+",");
        }
        arffData.set(arffData.size()-1,arffData.get(arffData.size()-1)+rooms.get(rooms.size()-1)+"}");
        arffData.add("@data");
        arffData.addAll(getOrderedReadings(JSONData,rpids));
        try {
            Files.write(arffPath,arffData, Charset.forName("UTF-8"));
            Files.write(roomsPath,rooms, Charset.forName("UTF-8"));
            Files.write(rpPath,rpids, Charset.forName("UTF-8"));

        } catch (IOException e) {
            e.printStackTrace();
        }

    }

    public static void makeClassifierARFF(String building_id,JSONArray JSONData){

        Path tempPath = Paths.get(building_id+"_temp.arff");
        Path roomsPath = Paths.get(building_id+"_rooms.data");
        Path rpPath = Paths.get(building_id+"_RPs.data");

        ArrayList<String> arffData = new ArrayList<>();
        ArrayList<String> rpids = new ArrayList<>();
        ArrayList<String> rooms = new ArrayList<>();

        try {
            BufferedReader br =
                Files.newBufferedReader(roomsPath,Charset.forName("UTF-8"));
            String line;
            while((line = br.readLine()) != null){
                rooms.add(line);
            }
            br = Files.newBufferedReader(rpPath,Charset.forName("UTF-8"));
            while((line = br.readLine()) != null){
                rpids.add(line);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
        arffData.add("@relation room"); arffData.add("");
        for(String s:rpids){
            arffData.add("@attribute "+s+" NUMERIC");
        }
        arffData.add("@attribute class {");
        for(int i=0;i<rooms.size()-1;++i){
            arffData.set(arffData.size()-1,arffData.get(arffData.size()-1)+rooms.get(i)+",");
        }
        arffData.set(arffData.size()-1,arffData.get(arffData.size()-1)+rooms.get(rooms.size()-1)+"}");
        arffData.add("@data");
        arffData.addAll(getOrderedReadings(JSONData,rpids));
        try {
            Files.write(tempPath,arffData, Charset.forName("UTF-8"));
        } catch (IOException e) {
            e.printStackTrace();
        }

    }

    public static ArrayList<String> getRPs(JSONArray JSONdata){
        ArrayList<String> rpids = new ArrayList<>();
        for(Object reading: JSONdata){
            for(Object pair: ((JSONObject)reading).getJSONArray("rpv_pair")){
                rpids.add(((JSONObject) pair).getString("RPID"));
            }
        }
        Set<String> noDuplicates = new LinkedHashSet<>(rpids);
        rpids.clear();
        rpids.addAll(noDuplicates);
        return rpids;
    }

    public static ArrayList<String> getRooms(JSONArray JSONdata){
        ArrayList<String> rooms = new ArrayList<>();
        for(Object reading: JSONdata){
            rooms.add(((JSONObject) reading).getString("room_id"));
        }
        Set<String> noDuplicates = new LinkedHashSet<>(rooms);
        rooms.clear();
        rooms.addAll(noDuplicates);
        return rooms;
    }

    public static ArrayList<String> getOrderedReadings(JSONArray jsonData,
         ArrayList<String> rpids){
        ArrayList<String> toReturn= new ArrayList<>();
        for(Object o:jsonData){
            String curLine = "";
            for(String rp:rpids){
                curLine += Double.toString(getValueByRPID(((JSONObject) o),rp))+",";

            }
            curLine += ((JSONObject) o).getString("room_id");
            toReturn.add(curLine);
        }

        return toReturn;
    }

    public static double getValueByRPID(JSONObject reading, String RPID){
        JSONArray pairs = reading.getJSONArray("rpv_pair");
        for(Object pair:pairs){
            if(RPID.equals(((JSONObject) pair).getString("RPID")))
                return ((JSONObject) pair).getDouble("value");
        }

        return 0;
    }
}
```

C.13 backend/nodeserver/server.js

```
// https://codeforgeek.com/2015/08/restful-api-node-mongodb/
// https://nodejs.org/api/
// https://docs.mongodb.org/manual/
// http://mongoosejs.com/docs/guide.html
// http://expressjs.com/en/guide/routing.html

var express    =   require("express");
var app        =   express();
var bodyParser =   require("body-parser");
var router     =   express.Router();
var Building   =   require('./models/building');
var Room       =   require('./models/room');
var RP         =   require('./models/RP');
var ERP        =   require('./models/ExhibitRP');
var RPMeasurement = require('./models/RPMeasurement');

app.use(bodyParser.json());
app.use(bodyParser.urlencoded({"extended" : false}));

router.get("/",function(req,res){
   res.json({"error" : false,"message" : "Hello World"});



});

app.use('/', router);

router.route("/buildings")
   .get(function(req,res){
      var response = {};
      Building.find({},function(err,data){
         if(err) {
            response = {"error" : true,"message" : "Error fetching data"};
         } else {
            response = data;
        }
        res.json(response);
    });
}).post(function(req,res){
    var db = new Building();
    var response = {};
    console.log("Post");
    console.log(req);
    db.rectangle.lt = req.body.rectangle.lt;
    db.rectangle.rt = req.body.rectangle.rt;
    db.rectangle.lb = req.body.rectangle.lb;
    db.rectangle.rb = req.body.rectangle.rb;
    db.name = req.body.name;
    db.width = req.body.width;
    db.length = req.body.length;
    db.save(function(err){
        if(err) {
            response = {"error" : true,"message" : "Failed!"};
            console.log(err);
        } else {
            response = {"error" : false,"message" : "Data added"};
        }
        res.json(response);
    });
});

router.route("/buildings/:id")
.get(function(req,res){
    var response = {};
    Building.findById(req.params.id,function(err,data){
        if(err) {
            response = {"error" : true,"message" : "Error fetching data"};
        } else {
            response = data;
        }
```

```
            res.json(response);
        });

    })
    .put(function(req,res){
        var response = {};
        Building.findById(req.params.id,function(err,data){
            if(err) {
                response = {"error" : true,"message" : "Error fetching data"};
            } else {
                if(req.body.rectangle !== undefined){
                    data.rectangle.lt = req.body.rectangle.lt;
                    data.rectangle.rt = req.body.rectangle.rt;
                    data.rectangle.lb = req.body.rectangle.lb;
                    data.rectangle.rb = req.body.rectangle.rb;
                }
                if(req.body.name !== undefined){
                    data.name = req.body.name;
                }
                if(req.body.width !== undefined){
                    data.width = req.body.width;
                }
                if(req.body.length !== undefined){
                    data.length = req.body.length;
                }

                data.save(function(err){
                    if(err) {
                        response = {"error" : true,"message" : "Error updating data"};
                        console.log(err);
                    } else {
                        response = {"error" : false,"message" : "Data is updated for
                            "+req.params.id};
                    }
                    res.json(response);
                })
            }
        });
    })
    .delete(function(req,res){
        var response = {};
        Building.findById(req.params.id,function(err,data){
            if(err) {
                response = {"error" : true,"message" : "Error fetching data"};
            } else {
                Room.remove({'building_id' : req.params.id},function(err){
                    if(err) {
                        response = {"error" : true,"message" : "Error deleting data"};
                    } else {
                        response = {"error" : true,"message" : "Data associated with
                            "+req.params.id+"is deleted"};
                    }
                });
                Building.remove({_id : req.params.id},function(err){
                    if(err) {
                        response = {"error" : true,"message" : "Error deleting data"};
                    } else {
                        response = {"error" : true,"message" : "Data associated with
                            "+req.params.id+"is deleted"};
                    }
                    res.json(response);
                });
            }
        });
    })

    router.route("/rooms/:id")
    .get(function(req,res){
        var response = {};
        Room.find({'building_id': req.params.id}, function(err,data){
            if(err){
                response = {"error" : true,"message" : "Error!"};
            } else {
                response = data;
            }
            res.json(response);
        });
    })

    router.route("/rooms/:id")
    .post(function(req,res){
        var response = {};
        var db = new Room();
        db.rectangle.lt = req.body.rectangle.lt;
        db.rectangle.rt = req.body.rectangle.rt;
        db.rectangle.lb = req.body.rectangle.lb;
        db.rectangle.rb = req.body.rectangle.rb;
        db.name = req.body.name;
        db.width = req.body.width;
        db.length = req.body.length;
        db.est_time = req.body.est_time;
        db.N_avg = 0;
        db.building_id = req.params.id;
        db.save(function(err){
            if(err) {
                response = {"error" : true,"message" : "Failed!"};
                console.log(err);
            } else {
                response = {"error" : false,"message" : "Data added"};
            }
            res.json(response);
        });

    })

    router.route("/measurements/")
    .get(function(req,res){
        var response = {};
        RPMeasurement.find({}, function(err,data){
            if(err){
                response = {"error" : true,"message" : "Error!"};
            } else {
                response = data;
            }
            res.json(response);
        });
    })
    .delete(function(req,res){
        var response = {};
        RPMeasurement.remove({},function(err){
                    if(err) {
                        response = {"error" : true,"message" : "Error deleting data"};
                    } else {
                        response = {"error" : true,"message" : "Data associated with
                             "+req.params.id+"is deleted"};
                    }
                    res.json(response);
        });
    })

    router.route("/measurements/:id")
    .get(function(req,res){
        var response = {};
        RPMeasurement.find({'room_id': req.params.id}, function(err,data){
            if(err){
                response = {"error" : true,"message" : "Error!"};
            } else {
                response = data;
            }
            res.json(response);
        });
    })
    .post(function(req,res){
        var response = {};
        var db = new RPMeasurement();

        db.rpv_pair = req.body.rpv_pair;

        db.room_id = req.params.id;

        db.save(function(err){

            if(err) {

                response = {"error" : true,"message" : "Failed!"};

                console.log(err);

            } else {

                response = {"error" : false,"message" : "Data added"};

            }

            res.json(response);

        });


    })


    router.route("/learn/:id")

    .get(function(req,res){

        var response = {};

        var net = require('net');

        var client = net.connect(5000, 'localhost');

        var request = {};

        var count = 0

        request.command = 'learn';

        request.building_id = req.params.id;

        request.learning_set = [];

        var rooms;

        Room.find({'building_id': req.params.id}, function(err,data){

            rooms = data.map(function(data) {return data._id;});

            RPMeasurement.find({room_id: {$in: rooms}}, function(err,data){

                request.learning_set = data;

                console.log(JSON.stringify(request));

                client.write(JSON.stringify(request));

                client.end();

                count ++;

            });

        });
    client.on('data', (data) => {
        console.log(data.toString());
        res.send(data.toString());
        count++;
    });
    })
.post(function(req,res){
    var response = {};
    var net = require('net');
    var client = net.connect(5000, 'localhost');
    var request = {};
    var count = 0
    request.command = 'learn';
    request.building_id = req.params.id;
    request.learning_set = [];
    var rooms;
    Room.find({'building_id': req.params.id}, function(err,data){
        rooms = data.map(function(data) {return data._id;});
        console.log(rooms);

        RPMeasurement.find({room_id: {$in: rooms}}, function(err,data){
            request.learning_set = data;
            console.log(JSON.stringify(request));
            client.write(JSON.stringify(request));
            count ++;
            if(count == 2)
                client.end();
        });
    });

    client.on('data', (data) => {
        console.log(data.toString());
        res.send(data.toString());
        count++;
        if(count == 2)
```

```
            client.end();
    });
    });

router.route("/locate/:id")
.get(function(req,res){
    var response = "NOTHING";
    var net = require('net');
    var client = net.connect(5000, 'localhost');
    var request = {};
    request.command = 'classify';
    request.building_id = req.params.id;
    request.learning_set = [];
    RPMeasurement.find({}, function(err,data){
        response = data;
        request.learning_set = response;
    });
    client.write(JSON.stringify(request));
    client.on('data', (data) => {
        console.log(data.toString());
        response = data;
    });
    client.end;
    res.send(response);
})
.post(function(req,res){
    var response = {};
    var net = require('net');
    var client = net.connect(5000, 'localhost');
    var request = {};
    request.command = 'classify';
    request.building_id = req.params.id;
    request.learning_set = [];
    request.learning_set.push(req.body);
    client.write(JSON.stringify(request));
    client.on('data', (data) => {
        console.log(data.toString());
        client.end();
        response = data;
        res.send(response);
    });
    client.end();
    });

router.route("/locationdata/:id")
.post(function(req,res){
    var response = {};
    console.log(req.body);
    var duration = req.body.duration;
    Room.findById(req.params.id,function(err,data){
        if(err) {
            response = {"error" : true,"message" : "Error fetching data"};
        } else {
            var est_time;
            var N_avg;
            if(!isFinite(data.est_time)){
                data.est_time = 1;
                est_time = 1;
            } else {
                est_time = data.est_time;
            }
            if(!isFinite(data.N_avg)){
                data.N_avg = 1;
                N_avg = 1;
            } else {
                N_avg = data.N_avg;
            }
            data.est_time = (est_time*N_avg+duration)/N_avg+1;
            data.N_avg++;
            data.save(function(err){
                if(err) {
                    response = {"error" : true,"message" : "Error updating data"};
                console.log(err);
            } else {
                response = {"error" : false,"message" : "Data is updated for
                    "+req.params.id};
                console.log("Updated:"+data.name+" "+data.est_time);
            }
            res.json(response);
        })
    }
  });
})


router.route("/route/:id")
.post(function(req,res){
    var response = {};
    var net = require('net');
    var client = net.connect(5000, 'localhost');
    var request = {};
    var selectedRooms = [];
    request.command = 'route';
    request.building_id = req.params.id;
    request.deadline = req.body.deadline;
    request.request_set = [];
    selectedRooms = req.body.selected_rooms;
    var roomIDs = selectedRooms.map(function(selectedRooms) {return
        selectedRooms.id;});
    Room.find({_id: {$in: roomIDs}}, function(err,data){
        var dataObj = [];

        for(var i = 0; i< data.length;i++) {
            var temp = data[i].toObject();
            for(var j = 0; j<selectedRooms.length;j++){
                if(selectedRooms[j].id == temp._id){
                    temp.excitement = selectedRooms[j].excitement;
                    dataObj.push(temp);
                    }

                }

            }
            request.request_set = dataObj;
            console.log(request.request_set);

            client.write(JSON.stringify(request));
            client.end();
        });
        client.on('data', (data) => {
            console.log(data.toString());
            response = data;
            res.send(response);
        });
        });


router.route("/exhibits/:id")
    .get(function(req,res){
        var response = {};
        ERP.find({'room_id': req.params.id}, function(err,data){
            if(err){
                response = {"error" : true,"message" : "Error!"};
            } else {
                response = data;
            }
            res.json(response);
        });
    })
    .post(function(req,res){
        var db = new ERP();
        var response = {};
        db.rpid = req.body.rpid;
        db.name = req.body.name;
        db.room_id = req.params.id;
        db.save(function(err){
            if(err) {
                response = {"error" : true,"message" : "Failed!"};
                console.log(err);
            } else {
                response = {"error" : false,"message" : "Data added"};
            }
            res.json(response);
        });
    });

app.use('/',router);
app.listen(3000);
console.log("Listening to PORT 3000");
```

C.14 backend/nodeserver/db.js

```
var mongoose = require('mongoose');


var dbURL = 'mongodb://localhost:27017/test';




mongoose.connect(dbURL, function(){
   console.log('db connected');
});
module.exports = mongoose;
```

C.15 backend/nodeserver/models/room.js

```
var Rectangle = {
      lt:{
         x: Number,
         y: Number,
      },
      rt:{
         x: Number,
         y: Number,
      },
      lb:{
         x: Number,
         y: Number,
      },
      rb:{
         x: Number,
         y: Number,
      }
};

var schema = new db.Schema({
   rectangle: Rectangle,
   name: {type: String, required: true},
   width: {type: Number, required: true},
   length:{type: Number, required: true},
   floor: Number,
   est_time: {type: Number, default: 1},
   N_avg: {type: Number,default:1},
   building_id: {type: String, required: true}
});

var room = db.model('room', schema);
module.exports = room;
```

C.16 backend/nodeserver/models/ExhibitRP.js

```
var schema = new db.Schema({
  rpid: {type: String, required: true},
  room_id: {type: String, required: true},
  name: {type: String, required: true}
});

var erp = db.model('erp', schema);
module.exports = erp;
```

C.17 backend/nodeserver/models/RPMeasurement.js

```
var schema = new db.Schema({
   rpv_pair: [{ RPID: String, value: Number }],
   room_id: {type: String, required: true}
});

var RPMeasurement = db.model('RPMeasurement', schema);
module.exports = RPMeasurement;
```

C.18 backend/nodeserver/models/building.js

```
var Rectangle = {
     lt:{
        x: Number,
        y: Number,
     },
     rt:{
        x: Number,
        y: Number,
     },
     lb:{
        x: Number,
        y: Number,
     },
     rb:{
        x: Number,
        y: Number,
     }
};

var schema = new db.Schema({
   rectangle: Rectangle,
   name: {type: String, required: true},
   width: {type: Number, required: true},
   length:{type: Number, required: true}
})

var buildings = db.model('building', schema);

module.exports = buildings;
```

C.19 backend/nodeserver/models/RP.js

```
var schema = new db.Schema({
   rpid: {type: String, required: true},
   building_id: {type: String, required: true},
   coordinate: {
      x: Number,
      y: Number
   }
});

var rp = db.model('rp', schema);
module.exports = rp;
```

C.20 backend/nodeserver/package.json

```
{
  "name": "node-server",
  "description": "Server for api",
  "version": "0.0.1",
  "private": true,
  "dependencies": {
    "body-parser": "^1.15.0",
    "express": "^4.x",
    "mongodb": "^2.1.7",
    "mongoose": "^4.4.5"
  }
}
```

C.21 android/IPS_Admin/build.gradle

```
android {
    compileSdkVersion 23
    buildToolsVersion "23.0.2"

    defaultConfig {
        applicationId "com.example.sari.ips_admin"
        minSdkVersion 15
        targetSdkVersion 23
        versionCode 1
        versionName "1.0"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'),
                 'proguard-rules.pro'
        }
    }
}

dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    testCompile 'junit:junit:4.12'
    compile 'com.android.support:appcompat-v7:23.1.1'
    compile 'com.android.support:design:23.1.1'
    compile 'org.apache.directory.studio:org.apache.commons.io:2.4'
}
```

C.22 android/IPS_Admin/src/res/layout/content_edit_building.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="10dp"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context="com.example.sari.ips_admin.activities.EditBuildingActivity"
    tools:showIn="@layout/activity_edit_building">

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal">
            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="Name"/>
            <EditText
                android:layout_width="200dp"
                android:layout_height="wrap_content" />
            <Button
                android:layout_width="wrap_content"
                android:layout_height="wrap_content" />
        </LinearLayout>

```

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Width"/>
    <EditText
        android:layout_width="200dp"
        android:layout_height="wrap_content" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
</LinearLayout>



<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Length"/>
    <EditText
        android:layout_width="200dp"
        android:layout_height="wrap_content" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</LinearLayout>

<!-- RECTANGLE -->
<LinearLayout
```

```
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:orientation="horizontal">
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginRight="10dp"
    android:text="LT"
    android:textSize="20sp"/>
<LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="vertical">
<LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content">
     <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="X:"
        android:textSize="15sp"/>


    <EditText
        android:layout_width="75dp"
        android:inputType="numberDecimal"
        android:layout_height="wrap_content" />
    <Button
        android:layout_width="50dp"
        android:layout_height="wrap_content" />
</LinearLayout>
<LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
            android:text="Y:"
            android:textSize="15sp"/>

        <EditText
            android:layout_width="75dp"
            android:inputType="numberDecimal"
            android:layout_height="wrap_content" />
        <Button
            android:layout_width="50dp"
            android:layout_height="wrap_content" />
    </LinearLayout>
    </LinearLayout>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="RT"
        android:layout_marginRight="10dp"
        android:textSize="20sp"/>
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical">
        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content">
            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="X:"
                android:textSize="15sp"/>

            <EditText
                android:layout_width="75dp"
                android:inputType="numberDecimal"
                android:layout_height="wrap_content" />
            <Button
```

```
                android:layout_width="50dp"
                android:layout_height="wrap_content" />
        </LinearLayout>
        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content">
            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="Y:"
                android:textSize="15sp"/>

            <EditText
                android:layout_width="75dp"
                android:inputType="numberDecimal"
                android:layout_height="wrap_content" />
            <Button
                android:layout_width="50dp"
                android:layout_height="wrap_content" />
        </LinearLayout>
    </LinearLayout>

</LinearLayout>
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginRight="10dp"
        android:text="LB"
        android:textSize="20sp"/>
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
android:orientation="vertical">
<LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="X:"
        android:textSize="15sp"/>

    <EditText
        android:layout_width="75dp"
        android:inputType="numberDecimal"
        android:layout_height="wrap_content" />
    <Button
        android:layout_width="50dp"
        android:layout_height="wrap_content" />
</LinearLayout>
<LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Y:"
        android:textSize="15sp"/>

    <EditText
        android:layout_width="75dp"
        android:inputType="numberDecimal"
        android:layout_height="wrap_content" />
    <Button
        android:layout_width="50dp"
        android:layout_height="wrap_content" />

</LinearLayout>
```

```
</LinearLayout>
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="RB"
    android:layout_marginRight="10dp"
    android:textSize="20sp"/>
<LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="vertical">
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content">
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="X:"
            android:textSize="15sp"/>

        <EditText
            android:layout_width="75dp"
            android:inputType="numberDecimal"
            android:layout_height="wrap_content" />
        <Button
            android:layout_width="50dp"
            android:layout_height="wrap_content" />
    </LinearLayout>
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content">
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Y:"
            android:textSize="15sp"/>
```

```
                    <EditText
                        android:layout_width="75dp"
                        android:inputType="numberDecimal"
                        android:layout_height="wrap_content" />
                    <Button
                        android:layout_width="50dp"
                        android:layout_height="wrap_content" />
                </LinearLayout>
            </LinearLayout>


        </LinearLayout>
    </LinearLayout>
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="horizontal">
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="LEARN"
        android:id="@+id/building_learn_button"
        android:onClick="learnBuilding"/>

    </LinearLayout>
    <ListView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/room_list_edit_building"/>
</LinearLayout>


</RelativeLayout>
```

C.23 android/IPS_Admin/src/res/layout/content_add_building.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="10dp"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context="com.example.sari.ips_admin.activities.AddBuildingActivity"
    tools:showIn="@layout/activity_add_building">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:orientation="vertical">
            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:orientation="horizontal">
                <TextView
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:text="Name"/>
                <EditText
                    android:layout_width="200dp"
                    android:layout_height="wrap_content"
                    android:id="@+id/addbuilding_text_name"/>

            </LinearLayout>
```

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Width"/>
    <EditText
        android:layout_width="200dp"
        android:layout_height="wrap_content"
        android:inputType="numberDecimal"
        android:id="@+id/addbuilding_text_width"/>

</LinearLayout>



<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Length"/>
    <EditText
        android:layout_width="200dp"
        android:layout_height="wrap_content"
        android:inputType="numberDecimal"
        android:id="@+id/addbuilding_text_length"/>



</LinearLayout>


<!-- RECTANGLE -->
<LinearLayout
```

```
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:orientation="horizontal">
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginRight="10dp"
    android:text="LT"
    android:textSize="20sp"/>
<LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="vertical">
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content">
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="X:"
            android:textSize="15sp"/>

        <EditText
            android:layout_width="75dp"
            android:inputType="numberDecimal"
            android:layout_height="wrap_content"
            android:id="@+id/addbuilding_text_ltx"/>

    </LinearLayout>
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content">
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Y:"
```

```
                android:textSize="15sp"/>

            <EditText
                android:layout_width="75dp"
                android:inputType="numberDecimal"
                android:layout_height="wrap_content"
                android:id="@+id/addbuilding_text_lty"/>

        </LinearLayout>
    </LinearLayout>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="RT"
        android:layout_marginRight="10dp"
        android:textSize="20sp"/>
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical">
        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content">
            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="X:"
                android:textSize="15sp"/>

            <EditText
                android:layout_width="75dp"
                android:inputType="numberDecimal"
                android:layout_height="wrap_content"
                android:id="@+id/addbuilding_text_rtx"/>

        </LinearLayout>
```

```
            <LinearLayout
                android:layout_width="wrap_content"
                android:layout_height="wrap_content">
                <TextView
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:text="Y:"
                    android:textSize="15sp"/>

                <EditText
                    android:layout_width="75dp"
                    android:inputType="numberDecimal"
                    android:layout_height="wrap_content"
                    android:id="@+id/addbuilding_text_rty"/>

            </LinearLayout>
        </LinearLayout>

    </LinearLayout>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginRight="10dp"
            android:text="LB"
            android:textSize="20sp"/>
        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:orientation="vertical">
            <LinearLayout
                android:layout_width="wrap_content"
                android:layout_height="wrap_content">
            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="X:"
                android:textSize="15sp"/>

            <EditText
                android:layout_width="75dp"
                android:inputType="numberDecimal"
                android:layout_height="wrap_content"
                android:id="@+id/addbuilding_text_lbx"/>

        </LinearLayout>
        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content">
            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="Y:"
                android:textSize="15sp"/>

            <EditText
                android:layout_width="75dp"
                android:inputType="numberDecimal"
                android:layout_height="wrap_content"
                android:id="@+id/addbuilding_text_lby"/>


        </LinearLayout>
    </LinearLayout>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="RB"
        android:layout_marginRight="10dp"
        android:textSize="20sp"/>
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical">
        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content">
            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="X:"
                android:textSize="15sp"/>

            <EditText
                android:layout_width="75dp"
                android:inputType="numberDecimal"
                android:layout_height="wrap_content"
                android:id="@+id/addbuilding_text_rbx"/>

        </LinearLayout>
        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content">
            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="Y:"
                android:textSize="15sp"/>

            <EditText
                android:layout_width="75dp"
                android:inputType="numberDecimal"
                android:layout_height="wrap_content"
                android:id="@+id/addbuilding_text_rby"/>
```

```
                    </LinearLayout>

                </LinearLayout>


            </LinearLayout>

        </LinearLayout>

        <Button

            android:layout_width="wrap_content"

            android:layout_height="match_parent"

            android:text="SUBMIT"

            android:onClick="submitBuilding"/>


    </LinearLayout>

</RelativeLayout>
```

C.24 android/IPS_Admin/src/res/layout/activity_main.xml

```
<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"
    android:fitsSystemWindows="true"
    tools:context=".activities.MainActivity">

    <android.support.design.widget.AppBarLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:theme="@style/AppTheme.AppBarOverlay">

        <android.support.v7.widget.Toolbar
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr/actionBarSize"
            android:background="?attr/colorPrimary"
            app:popupTheme="@style/AppTheme.PopupOverlay" />

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

    <include layout="@layout/content_main" />

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

C.25 android/IPS_Admin/src/res/layout/content_edit_room.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context="com.example.sari.ips_admin.activities.EditRoomActivity"
    tools:showIn="@layout/activity_edit_room">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Measure"
        android:id="@+id/room_measure_button"/>
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/edit_room_results_textview"/>
    </LinearLayout>
</RelativeLayout>
```

C.26 android/IPS_Admin/src/res/layout/activity_edit_building.xml

```
<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"
    android:fitsSystemWindows="true"
    tools:context="com.example.sari.ips_admin.activities.EditBuildingActivity">

    <android.support.design.widget.AppBarLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:theme="@style/AppTheme.AppBarOverlay">

        <android.support.v7.widget.Toolbar
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr/actionBarSize"
            android:background="?attr/colorPrimary"
            app:popupTheme="@style/AppTheme.PopupOverlay" />

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

    <include layout="@layout/content_edit_building" />



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

C.27 android/IPS_Admin/src/res/layout/content_main.xml

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

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

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

    android:layout_width="match_parent"

    android:layout_height="match_parent"

    android:paddingBottom="@dimen/activity_vertical_margin"

    android:paddingLeft="@dimen/activity_horizontal_margin"

    android:paddingRight="@dimen/activity_horizontal_margin"

    android:paddingTop="@dimen/activity_vertical_margin"

    app:layout_behavior="@string/appbar_scrolling_view_behavior"

    tools:context=".activities.MainActivity"

    tools:showIn="@layout/activity_main">


    <Button

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"

        android:text="BUILDINGS"

        android:onClick="goToBuildingsActivity"/>




</RelativeLayout>
```

C.28 android/IPS_Admin/src/res/layout/activity_edit_room.xml

```xml
<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"
    android:fitsSystemWindows="true"
    tools:context="com.example.sari.ips_admin.activities.EditRoomActivity">

    <android.support.design.widget.AppBarLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:theme="@style/AppTheme.AppBarOverlay">

        <android.support.v7.widget.Toolbar
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr/actionBarSize"
            android:background="?attr/colorPrimary"
            app:popupTheme="@style/AppTheme.PopupOverlay" />

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

    <include layout="@layout/content_edit_room" />



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

C.29 android/IPS_Admin/src/res/layout/activity_buildings.xml

```
<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"
    android:fitsSystemWindows="true"
    tools:context=".activities.BuildingsActivity">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">
       <android.support.v7.widget.Toolbar
        android:id="@+id/my_toolbar"
        android:layout_width="match_parent"
        android:layout_height="?attr/actionBarSize"
        android:background="?attr/colorPrimary"
        android:elevation="4dp"
        android:theme="@style/ThemeOverlay.AppCompat.ActionBar"
        app:popupTheme="@style/ThemeOverlay.AppCompat.Light"/>

        <ListView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/museumListView"/>
    </LinearLayout>
</android.support.design.widget.CoordinatorLayout>
```

C.30 android/IPS_Admin/src/res/layout/content_buildings.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context=".activities.BuildingsActivity"
    tools:showIn="@layout/activity_buildings">



</RelativeLayout>
```

C.31 android/IPS_Admin/src/res/layout/activity_add_building.xml

```
<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"
    android:fitsSystemWindows="true"
    tools:context="com.example.sari.ips_admin.activities.AddBuildingActivity">

    <android.support.design.widget.AppBarLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:theme="@style/AppTheme.AppBarOverlay">

        <android.support.v7.widget.Toolbar
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr/actionBarSize"
            android:background="?attr/colorPrimary"
            app:popupTheme="@style/AppTheme.PopupOverlay" />

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

    <include layout="@layout/content_add_building" />

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

C.32 android/IPS_Admin/src/res/layout/activity_add_room.xml

```
<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"

    android:fitsSystemWindows="true"

    tools:context="com.example.sari.ips_admin.activities.AddRoomActivity">


    <android.support.design.widget.AppBarLayout

        android:layout_width="match_parent"

        android:layout_height="wrap_content"

        android:theme="@style/AppTheme.AppBarOverlay">


        <android.support.v7.widget.Toolbar

            android:id="@+id/toolbar"

            android:layout_width="match_parent"

            android:layout_height="?attr/actionBarSize"

            android:background="?attr/colorPrimary"

            app:popupTheme="@style/AppTheme.PopupOverlay" />


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


    <include layout="@layout/content_add_room" />



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

C.33 android/IPS_Admin/src/res/layout/content_add_room.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="10dp"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context="com.example.sari.ips_admin.activities.AddRoomActivity"
    tools:showIn="@layout/activity_add_room">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:orientation="vertical">
            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:orientation="horizontal">
                <TextView
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:text="Name"/>
                <EditText
                    android:layout_width="200dp"
                    android:layout_height="wrap_content"
                    android:id="@+id/addroom_text_name"/>

            </LinearLayout>
```

```
<LinearLayout

    android:layout_width="match_parent"

    android:layout_height="wrap_content"

    android:orientation="horizontal">

    <TextView

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"

        android:text="Width"/>

    <EditText

        android:layout_width="200dp"

        android:layout_height="wrap_content"

        android:inputType="numberDecimal"

        android:id="@+id/addroom_text_width"/>


</LinearLayout>




<LinearLayout

    android:layout_width="match_parent"

    android:layout_height="wrap_content"

    android:orientation="horizontal">

    <TextView

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"

        android:text="Length"/>

    <EditText

        android:layout_width="200dp"

        android:layout_height="wrap_content"

        android:inputType="numberDecimal"

        android:id="@+id/addroom_text_length"/>




</LinearLayout>


<!-- RECTANGLE -->

<LinearLayout
```

```
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:orientation="horizontal">
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginRight="10dp"
    android:text="LT"
    android:textSize="20sp"/>
<LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="vertical">
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content">
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="X:"
            android:textSize="15sp"/>

        <EditText
            android:layout_width="75dp"
            android:inputType="numberDecimal"
            android:layout_height="wrap_content"
            android:id="@+id/addroom_text_ltx"/>

    </LinearLayout>
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content">
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Y:"
```

```
                android:textSize="15sp"/>

            <EditText
                android:layout_width="75dp"
                android:inputType="numberDecimal"
                android:layout_height="wrap_content"
                android:id="@+id/addroom_text_lty"/>

        </LinearLayout>
    </LinearLayout>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="RT"
        android:layout_marginRight="10dp"
        android:textSize="20sp"/>
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical">
        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content">
            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="X:"
                android:textSize="15sp"/>

            <EditText
                android:layout_width="75dp"
                android:inputType="numberDecimal"
                android:layout_height="wrap_content"
                android:id="@+id/addroom_text_rtx"/>

        </LinearLayout>
```

```
            <LinearLayout
                android:layout_width="wrap_content"
                android:layout_height="wrap_content">
                <TextView
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:text="Y:"
                    android:textSize="15sp"/>

                <EditText
                    android:layout_width="75dp"
                    android:inputType="numberDecimal"
                    android:layout_height="wrap_content"
                    android:id="@+id/addroom_text_rty"/>

            </LinearLayout>
        </LinearLayout>

    </LinearLayout>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginRight="10dp"
            android:text="LB"
            android:textSize="20sp"/>
        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:orientation="vertical">
            <LinearLayout
                android:layout_width="wrap_content"
                android:layout_height="wrap_content">
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="X:"
            android:textSize="15sp"/>

        <EditText
            android:layout_width="75dp"
            android:inputType="numberDecimal"
            android:layout_height="wrap_content"
            android:id="@+id/addroom_text_lbx"/>

    </LinearLayout>
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content">
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Y:"
            android:textSize="15sp"/>

        <EditText
            android:layout_width="75dp"
            android:inputType="numberDecimal"
            android:layout_height="wrap_content"
            android:id="@+id/addroom_text_lby"/>



    </LinearLayout>
</LinearLayout>
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="RB"
    android:layout_marginRight="10dp"
```

```
        android:textSize="20sp"/>
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical">
        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content">
            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="X:"
                android:textSize="15sp"/>

            <EditText
                android:layout_width="75dp"
                android:inputType="numberDecimal"
                android:layout_height="wrap_content"
                android:id="@+id/addroom_text_rbx"/>

        </LinearLayout>
        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content">
            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="Y:"
                android:textSize="15sp"/>

            <EditText
                android:layout_width="75dp"
                android:inputType="numberDecimal"
                android:layout_height="wrap_content"
                android:id="@+id/addroom_text_rby"/>
```

```
                    </LinearLayout>

                </LinearLayout>


            </LinearLayout>

        </LinearLayout>

        <Button

            android:layout_width="wrap_content"

            android:layout_height="match_parent"

            android:text="SUBMIT"

            android:onClick="submitRoom"/>


    </LinearLayout>

</RelativeLayout>
```

C.34 android/IPS_Admin/src/res/menu/menu_edit_building.xml

```
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <item
        android:id="@+id/action_deleteBuilding"
        android:title="Delete building"
        android:icon="@drawable/ic_action_discard"
        app:showAsAction="ifRoom"/>
    <item
        android:id="@+id/action_addRoom"
        android:title="Add room"
        android:icon="@drawable/ic_action_new"
        app:showAsAction="ifRoom"/>




</menu>
```

C.35 android/IPS_Admin/src/res/menu/menu_buildings.xml

```
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <item
        android:id="@+id/action_addBuilding"
        android:title="Add building"
        android:icon="@drawable/ic_action_new"
        app:showAsAction="ifRoom"/>

    <!-- Settings, should always be in the overflow -->
    <item android:id="@+id/action_settings"
        android:title="@string/action_settings"
        app:showAsAction="never"/>
</menu>
```

C.36 android/IPS_Admin/src/res/menu/menu_main.xml

```
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    tools:context=".activities.MainActivity">
    <item
        android:id="@+id/action_settings"
        android:orderInCategory="100"
        android:title="@string/action_settings"
        app:showAsAction="never" />
</menu>
```

C.37 android/IPS_Admin/src/AndroidManifest.xml

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.sari.ips_admin">

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
    <uses-permission android:name="android.permission.CHANGE_WIFI_STATE"/>
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".activities.MainActivity"
            android:label="@string/app_name"
            android:theme="@style/AppTheme.NoActionBar">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".activities.BuildingsActivity"
            android:label="@string/title_activity_buildings"
            android:parentActivityName=".activities.MainActivity"
            android:theme="@style/AppTheme.NoActionBar">
            <meta-data
                android:name="android.support.PARENT_ACTIVITY"
                android:value=".activities.MainActivity" />
        </activity>
        <activity
            android:name=".activities.EditBuildingActivity"
            android:label="@string/title_activity_edit_building"
            android:parentActivityName=".activities.BuildingsActivity"
            android:theme="@style/AppTheme.NoActionBar">
            <meta-data
                android:name="android.support.PARENT_ACTIVITY"
                android:value="com.example.sari.ips_admin.activities.BuildingsActivity"
                    />
        </activity>
        <activity
            android:name=".activities.AddBuildingActivity"
            android:label="@string/title_activity_add_building"
            android:parentActivityName=".activities.BuildingsActivity"
            android:theme="@style/AppTheme.NoActionBar">
            <meta-data
                android:name="android.support.PARENT_ACTIVITY"
                android:value="com.example.sari.ips_admin.activities.BuildingsActivity"
                    />
        </activity>
        <activity
            android:name=".activities.AddRoomActivity"
            android:label="@string/title_activity_add_room"
            android:parentActivityName=".activities.EditBuildingActivity"
            android:theme="@style/AppTheme.NoActionBar">
            <meta-data
                android:name="android.support.PARENT_ACTIVITY"
                android:value="com.example.sari.ips_admin.activities.EditBuildingActivity"
                    />
        </activity>
        <activity
            android:name=".activities.EditRoomActivity"
            android:label="@string/title_activity_edit_room"
            android:parentActivityName=".activities.EditBuildingActivity"
            android:theme="@style/AppTheme.NoActionBar">
            <meta-data
                android:name="android.support.PARENT_ACTIVITY"
                android:value="com.example.sari.ips_admin.activities.EditBuildingActivity"
                    />
        </activity>

    </application>


</manifest>
```

C.38 android/IPS_Admin/src/java/tools/Rectangle.java

```
import java.io.Serializable;

public class Rectangle implements Serializable{
    private Point coordinates;
    private double width, length;

    public Rectangle(Point coordinates, double width, double length){
        this.coordinates = coordinates;
        this.width = width;
        this.length = length;
    }

    public Point getCoordinates() {
        return coordinates;
    }

    public double getWidth() {
        return width;
    }

    public double getLength() {
        return length;
    }

    public void setCoordinates(Point coordinates) {
        this.coordinates = coordinates;
    }

    public void setWidth(double width) {
        this.width = width;
    }

    public void setLength(double length) {
        this.length = length;
    }


    public Point getCenter(){
        return new Point((width+coordinates.getX())/2,(length+coordinates.getY())/2);
    }


}
```

C.39 android/IPS_Admin/src/java/tools/Point.java

```java
import java.io.Serializable;

public class Point implements Serializable{
   private double x, y;

   public Point(double x, double y)
   {
      this.x = x;
      this.y = y;
   }

   public double getX() {
      return x;
   }

   public void setX(double x) {
      this.x = x;
   }

   public double getY() {
      return y;
   }

   public void setY(double y) {
      this.y = y;
   }
}
```

C.40 android/IPS_Admin/src/java/tools/RectangleDB.java

```java
import java.io.Serializable;


public class RectangleDB implements Serializable{
    private Point lt,rt,lb,rb;

    public RectangleDB(Point lt, Point rt, Point lb, Point rb) {
        this.lt = lt;
        this.rt = rt;
        this.lb = lb;
        this.rb = rb;
    }

    public Point getLt() {
        return lt;
    }

    public void setLt(Point lt) {
        this.lt = lt;
    }

    public Point getRt() {
        return rt;
    }

    public void setRt(Point rt) {
        this.rt = rt;
    }

    public Point getLb() {
        return lb;
    }
    public void setLb(Point lb) {
        this.lb = lb;
    }

    public Point getRb() {
        return rb;
    }

    public void setRb(Point rb) {
        this.rb = rb;
    }
}
```

C.41 android/IPS_Admin/src/java/database/Database.java

```java
import android.util.Log;
import com.example.sari.ips_admin.models.indoormapping.Building;
import com.example.sari.ips_admin.models.indoormapping.Floor;
import com.example.sari.ips_admin.models.indoormapping.Room;
import com.example.sari.ips_admin.models.positioning.RPMeasurement;
import com.example.sari.ips_admin.tools.Point;
import com.example.sari.ips_admin.tools.RectangleDB;
import org.apache.commons.io.IOUtils;
import org.json.JSONArray;
import org.json.JSONObject;
import java.io.BufferedInputStream;
import java.io.BufferedOutputStream;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.ArrayList;

/**
 * Class that connects to the api.
 */
public class Database {

    public final static String API_URL = "http://178.62.127.39:3000/";
    public static Building[] getBuildings(){
        String StringData = getData("buildings");
        ArrayList<Building> toReturn = new ArrayList<>();
        Log.d("BUILDINGS: ", StringData);
        try {
            JSONArray JSONData = new JSONArray(StringData);
            for(int i = 0; i<JSONData.length();++i){
                JSONObject building = JSONData.getJSONObject(i);
            Point lt = new
                Point(building.getJSONObject("rectangle").getJSONObject("lt").getDouble("x"),
                              building.getJSONObject("rectangle").getJSONObject("lt").getDouble("y"));
            Point rt = new
                Point(building.getJSONObject("rectangle").getJSONObject("rt").getDouble("x"),
                              building.getJSONObject("rectangle").getJSONObject("rt").getDouble("y"));
            Point lb = new
                Point(building.getJSONObject("rectangle").getJSONObject("lb").getDouble("x"),
                              building.getJSONObject("rectangle").getJSONObject("lb").getDouble("y"));
            Point rb = new
                Point(building.getJSONObject("rectangle").getJSONObject("rb").getDouble("x"),
                              building.getJSONObject("rectangle").getJSONObject("rb").getDouble("y"));
            RectangleDB r = new RectangleDB(lt,rt,lb,rb);
            Building toAdd = new
                Building(building.getString("_id"),r,building.getString("name"),
                   building.getDouble("width"),building.getDouble("length"),
                   getRooms(building.getString("_id")));
            toReturn.add(toAdd);
            getRooms(building.getString("_id"));
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
    return toReturn.toArray(new Building[toReturn.size()]);
}

public static boolean addBuilding(Building building){
    JSONObject buildingToAdd = new JSONObject();
    try {
        buildingToAdd.put("name",building.getName());
        buildingToAdd.put("width", building.getWidth());
        buildingToAdd.put("length",building.getLength());
        JSONObject rectangle = new JSONObject();
        rectangle.put("lt", new JSONObject().put("x",
            building.getRectangle().getLt().getX())
```

```
                                        .put("y",
                                            building.getRectangle().getLt().getY()))
                .put("rt", new JSONObject().put("x",
                    building.getRectangle().getRt().getX())
                                        .put("y",
                                            building.getRectangle().getRt().getY()))
                .put("lb", new JSONObject().put("x",
                    building.getRectangle().getLb().getX())
                                        .put("y",
                                            building.getRectangle().getLb().getY()))
                .put("rb", new JSONObject().put("x",
                    building.getRectangle().getRb().getX())
                                        .put("y",
                                            building.getRectangle().getRb().getY()));

        buildingToAdd.put("rectangle", rectangle);
        postData("buildings", buildingToAdd.toString());
    } catch (Exception e) {
        e.printStackTrace();
    }

    return true;
}

public static boolean postMeasurement(RPMeasurement measurement){

    JSONArray JSONAllM= new JSONArray();
    JSONObject toSend = new JSONObject();
    try {
        for(int i = 0; i<measurement.getReadings().size(); ++i){
            JSONAllM.put(new
                JSONObject().put("RPID",measurement.getReadings().get(i).first)
                    .put("value",measurement.getReadings().get(i).second));
        }

        toSend.put("rpv_pair", JSONAllM);
    } catch (Exception e) {
        e.printStackTrace();
    }
    postData("measurements", toSend.toString(), measurement.getRoom_id());

    return true;
}

public static Building getBuilding(String id){
    String StringData = getData("buildings",id);
    Building toReturn = null;
    try {
        JSONObject building = new JSONObject(StringData);
        Point lt = new
            Point(building.getJSONObject("rectangle").getJSONObject("lt").getDouble("x"),
                            building.getJSONObject("rectangle").getJSONObject("lt").getDouble("y"));
        Point rt = new
            Point(building.getJSONObject("rectangle").getJSONObject("rt").getDouble("x"),
                   building.getJSONObject("rectangle").getJSONObject("rt").getDouble("y"));
        Point lb = new
            Point(building.getJSONObject("rectangle").getJSONObject("lb").getDouble("x"),
                   building.getJSONObject("rectangle").getJSONObject("lb").getDouble("y"));
        Point rb = new
            Point(building.getJSONObject("rectangle").getJSONObject("rb").getDouble("x"),
                  building.getJSONObject("rectangle").getJSONObject("rb").getDouble("y"));
        RectangleDB r = new RectangleDB(lt,rt,lb,rb);
        toReturn = new
            Building(building.getString("_id"),r,building.getString("name"),
                   building.getDouble("width"),building.getDouble("length"),
                   getRooms(building.getString("_id")));

    } catch (Exception e) {
        e.printStackTrace();
    }
    return toReturn;
}
```

```
public static Floor[] getFloors(String building_id){
    return null;
}

public static Room[] getRooms(String building_id){
    String StringData = getData("rooms", building_id);
    Log.d("Rooms",StringData);
    ArrayList<Room> toReturn = new ArrayList<>();
    try {
        JSONArray JSONData = new JSONArray(StringData);
        for(int i = 0; i<JSONData.length();++i){
            JSONObject room = JSONData.getJSONObject(i);
            Point lt = new
                Point(room.getJSONObject("rectangle").getJSONObject("lt").getDouble("x"),
                               room.getJSONObject("rectangle").getJSONObject("lt").getDouble("y"));
            Point rt = new
                Point(room.getJSONObject("rectangle").getJSONObject("rt").getDouble("x"),
                               room.getJSONObject("rectangle").getJSONObject("rt").getDouble("y"));
            Point lb = new
                Point(room.getJSONObject("rectangle").getJSONObject("lb").getDouble("x"),
                               room.getJSONObject("rectangle").getJSONObject("lb").getDouble("y"));
            Point rb = new
                Point(room.getJSONObject("rectangle").getJSONObject("rb").getDouble("x"),
                               room.getJSONObject("rectangle").getJSONObject("rb").getDouble("y"));
            RectangleDB r = new RectangleDB(lt,rt,lb,rb);
            Room toAdd = new Room(room.getString("_id"),building_id,
                    room.getString("name"),r,room.getDouble("width"),
                    room.getDouble("length"),room.getDouble("est_time"));
            toReturn.add(toAdd);
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
    return toReturn.toArray(new Room[toReturn.size()]);
}
```

```
public static boolean addRoom(String building_id, Room room){
    JSONObject roomToAdd = new JSONObject();
    try {
        roomToAdd.put("name", room.getRoomName());
        roomToAdd.put("width", room.getWidth());
        roomToAdd.put("length", room.getLength());
        JSONObject rectangle = new JSONObject();
        rectangle.put("lt", new JSONObject().put("x",
            room.getRectangleDB().getLt().getX())
                .put("y", room.getRectangleDB().getLt().getY()))
                .put("rt", new JSONObject().put("x",
                    room.getRectangleDB().getRt().getX())
                        .put("y", room.getRectangleDB().getRt().getY()))
                .put("lb", new JSONObject().put("x",
                    room.getRectangleDB().getLb().getX())
                        .put("y", room.getRectangleDB().getLb().getY()))
                .put("rb", new JSONObject().put("x",
                    room.getRectangleDB().getRb().getX())
                        .put("y", room.getRectangleDB().getRb().getY()));

        roomToAdd.put("rectangle", rectangle);
        postData("rooms", roomToAdd.toString(), building_id);

    } catch (Exception e) {
        e.printStackTrace();
    }

    return true;
}



public static String getData(String... params) {
    String data = "";
    String building_id = "";
    String request = params[0];
```

```
        InputStream inputStream;

        if (params.length >= 2){
            building_id = params[1];
        }

        try {
            URL url = new URL(API_URL+request+"/"+building_id);
            HttpURLConnection connection = (HttpURLConnection) url.openConnection();
            inputStream = new BufferedInputStream(connection.getInputStream());
            data = IOUtils.toString(inputStream,"UTF-8");
            connection.disconnect();
        } catch (Exception e) {
            e.printStackTrace();
        }
        return data;

    }

    public static String postData(String... params){
        String id = "";
        String request = params[0];
        String data = params[1];
        InputStream is;
        String dataRec = "";
        if (params.length > 2){
            id = params[2];
        }

        try {
            URL url = new URL(API_URL+request+"/"+id);
            HttpURLConnection connection = (HttpURLConnection) url.openConnection();
            Log.d("Error", "Ex " + url.toString());
            connection.setDoOutput(true);
            connection.setChunkedStreamingMode(0);
            connection.setRequestProperty("Content-Type", "application/json");
            OutputStream os = new BufferedOutputStream(connection.getOutputStream());
            os.write(data.getBytes());
            os.flush();
            is = new BufferedInputStream(connection.getInputStream());
            dataRec = IOUtils.toString(is,"UTF-8");
            Log.d("Found:", "Room:"+dataRec);
            if (connection.getResponseCode() != HttpURLConnection.HTTP_CREATED) {
                throw new RuntimeException("HTTP code : "
                        + connection.getResponseCode());
            }
            os.close();
            connection.disconnect();
        } catch (Exception e) {
            Log.d("Error","Ex "+e);
            e.printStackTrace();
        }

        return dataRec;
    }

    public static void deleteData(String... params){
        String building_id = "";
        String request = params[0];



        if (params.length == 2){
            building_id = params[1];
        }

        try {
            URL url = new URL(API_URL+request+"/"+building_id);
            HttpURLConnection connection = (HttpURLConnection) url.openConnection();
            connection.setDoOutput(true);
            connection.setRequestMethod("DELETE");
            connection.getContent();
            connection.disconnect();
        } catch (Exception e) {
            Log.d("Error","Ex "+e);
            e.printStackTrace();
        }
    }

    public static String classify(RPMeasurement measurement, String building_id){
        JSONArray JSONAllM= new JSONArray();
        JSONObject toSend = new JSONObject();
        JSONArray toSendArray = new JSONArray();
        try {
            for(int i = 0; i<measurement.getReadings().size(); ++i){
                JSONAllM.put(new
                    JSONObject().put("RPID",measurement.getReadings().get(i).first)
                        .put("value",measurement.getReadings().get(i).second));
            }

            toSend.put("rpv_pair", JSONAllM);
            toSend.put("room_id","?");
            toSendArray.put(toSend);
        } catch (Exception e) {
            e.printStackTrace();
        }
        return postData("locate", building_id, toSend.toString());
    }

}
```

C.42 android/IPS_Admin/src/java/models/RPMeasurement.java

```java
import android.util.Pair;
import java.util.ArrayList;


public class RPMeasurement {
   //String RPID; //this is a unique identifier for the reference point. (eg.: MAC
       ADDRESS)
   String room_id;
   ArrayList<Pair<String,Double>> readings = new ArrayList<>();

   public RPMeasurement(String[] RPIDs, Double[] values,String room_id) {
      for(int i = 0; i<RPIDs.length; ++i)
      {
         readings.add(new Pair<>(RPIDs[i],values[i]));
      }
      this.room_id = room_id;
   }

   public ArrayList<Pair<String, Double>> getReadings() {
      return readings;
   }



   public String getRoom_id() {
      return room_id;
   }

   public void setRoom_id(String room_id) {
      this.room_id = room_id;
   }
}
```

C.43 android/IPS_Admin/src/java/models/Room.java

```
import com.example.sari.ips_admin.tools.Point;
import com.example.sari.ips_admin.tools.Rectangle;
import com.example.sari.ips_admin.tools.RectangleDB;
import java.io.Serializable;
import java.util.ArrayList;

public class Room implements Serializable{
   private String roomName;
   private String id;
   private String building_id;
   private Rectangle roomRectangle;
   private RectangleDB rectangleDB;
   private String roomDescription;
   private double width,length;
   private double est_time;
   private ArrayList<String> exhibits;

   public Room(String id, String building_id, String roomName, RectangleDB
       rectangleDB,
             double width, double length, double est_time){
      this.roomName = roomName;
      this.id = id;
      this.width = width;
      this.length = length;
      this.rectangleDB = rectangleDB;
      this.building_id = building_id;
      this.roomRectangle = new Rectangle(new Point(0,0),0,0); //for testing!!!
      this.est_time = est_time;
      exhibits = new ArrayList<>();
   }

   public String getBuilding_id() {
      return building_id;
   }

   public Room(String building_id, String roomName, RectangleDB rectangleDB,
             double width, double length){
      this.building_id = building_id;

      this.roomName = roomName;
      this.width = width;
      this.length = length;
      this.roomDescription = "";
      this.rectangleDB = rectangleDB;
      exhibits = new ArrayList<>();
      this.roomRectangle = new Rectangle(new Point(0,0),0,0); //for testing!!!
   }

   public Room(Rectangle r){
      this.roomRectangle = r;
      roomName = "";
      roomDescription = "";
   }

   public String getRoomName() {
      return roomName;
   }

   public void setRoomName(String roomName) {
      this.roomName = roomName;
   }

   public Rectangle getRoomRectangle() {
      return roomRectangle;
   }

   public void setRoomRectangle(Rectangle roomRectangle) {
      this.roomRectangle = roomRectangle;
   }
```

```
public String getRoomDescription() {
    return roomDescription;
}

public void setRoomDescription(String roomDescription) {
    this.roomDescription = roomDescription;
}

public boolean isNeighbour(Room room){
    if(this == room)
        return false;

    if(this.getRoomRectangle().getCoordinates().getX() +
        this.getRoomRectangle().getWidth()
           == room.getRoomRectangle().getCoordinates().getX()
           || this.getRoomRectangle().getCoordinates().getY()
           + this.getRoomRectangle().getLength()
           == room.getRoomRectangle().getCoordinates().getY()
           || room.getRoomRectangle().getCoordinates().getX()
           + room.getRoomRectangle().getWidth()
           == this.getRoomRectangle().getCoordinates().getX()
           || room.getRoomRectangle().getCoordinates().getY()
           + room.getRoomRectangle().getLength()
           == this.getRoomRectangle().getCoordinates().getY())
        return true;

    return false;

}

public String getId() {
    return id;
}

public RectangleDB getRectangleDB() {
        return rectangleDB;
    }

    public double getWidth() {
        return width;
    }

    public double getLength() {
        return length;
    }

    public void addExhibit(String exhibit){
        exhibits.add(exhibit);
    }

    public ArrayList<String> getExhibits(){
        return exhibits;
    }

    public void setEst_time(double est_time) {
        this.est_time = est_time;
    }

    public double getEst_time() {

        return est_time;
    }
}
```

C.44 android/IPS_Admin/src/java/models/Position.java

```java
public class Position {
    RPMeasurement[] measurements;

    public Position(RPMeasurement[] measurements){
        this.measurements = measurements;
    }

    public RPMeasurement[] getMeasurements() {
        return measurements;
    }

    public void setMeasurements(RPMeasurement[] measurements) {
        this.measurements = measurements;
    }
}
```

C.45 android/IPS_Admin/src/java/models/Floor.java

```
import java.io.Serializable;
import java.util.ArrayList;

public class Floor implements Serializable{
    Room[] rooms;

    public Floor(Room[] rooms)
    {
        this.rooms = rooms;
    }

    public Room[] getRooms() {
        return rooms;
    }

    public Room[] getNeighbours(Room room) {
        ArrayList<Room> neighbours = new ArrayList<>();
        for(Room r:rooms){
            if(r.isNeighbour(room))
                neighbours.add(r);
        }
        return neighbours.toArray(new Room[neighbours.size()]);



    }

    public void setRooms(Room[] rooms) {
        this.rooms = rooms;
    }

}
```

C.46 android/IPS_Admin/src/java/models/Building.java

```
import com.example.sari.ips_admin.tools.RectangleDB;
import java.io.Serializable;


public class Building implements Serializable{
   private RectangleDB rectangle;
   private String name;
   private double width;
   private double length;
   private String id;
   private Room[] rooms;



   public Building(String id, RectangleDB rectangle, String name,
               double width, double length, Room[] rooms) {
      this.rectangle = rectangle;
      this.id = id;
      this.name = name;
      this.width = width;
      this.length = length;
      this.rooms = rooms;
   }

   public Building(RectangleDB rectangle, String name, double width, double length) {
      this.rectangle = rectangle;
      this.id = "";
      this.name = name;
      this.width = width;
      this.length = length;
   }

   public void setRectangle(RectangleDB rectangle) {
      this.rectangle = rectangle;
   }

   public void setName(String name) {
      this.name = name;
   }

   public void setWidth(double width) {
      this.width = width;
   }

   public void setLength(double length) {
      this.length = length;
   }

   public RectangleDB getRectangle() {

      return rectangle;
   }

   public String getName() {
      return name;
   }

   public double getWidth() {
      return width;
   }

   public double getLength() {
      return length;
   }

   public String getId() {
      return id;
   }
   public void setId(String id) {
      this.id = id;
   }

   public Room[] getRooms() {
      return rooms;
   }

   public void setRooms(Room[] rooms) {
      this.rooms = rooms;
   }
}
```

C.47 android/IPS_Admin/src/java/activities/EditRoomActivity.java

```
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.net.wifi.ScanResult;
import android.net.wifi.WifiManager;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;
import com.example.sari.ips_admin.R;
import com.example.sari.ips_admin.database.Database;
import com.example.sari.ips_admin.models.positioning.RPMeasurement;
import java.util.List;

public class EditRoomActivity extends AppCompatActivity {
    private String currentRoom = "";
    WifiManager wifiManager;
    private TextView resultsTextview;
    private boolean toMeasure = false;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_edit_room);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);
        resultsTextview = (TextView) findViewById(R.id.edit_room_results_textview);
        Button measureButton = (Button) findViewById(R.id.room_measure_button);
        getSupportActionBar().setDisplayHomeAsUpEnabled(false);
    Intent intent = getIntent();
    currentRoom = intent.getStringExtra("room_id");
    wifiManager = (WifiManager) getSystemService(Context.WIFI_SERVICE);
    registerReceiver(broadcastReceiver,
            new IntentFilter(WifiManager.SCAN_RESULTS_AVAILABLE_ACTION));
    measureButton.setOnClickListener(startMListener);

}

private void startMeasurement(){
    wifiManager.startScan();
    toMeasure = true;
}

private void stopMeasurement(){
    toMeasure = false;
}

@Override
protected void onDestroy() {
    super.onDestroy();
    unregisterReceiver(broadcastReceiver);
}
private View.OnClickListener startMListener = new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        ((Button)v).setText("STOP");
        startMeasurement();
        v.setOnClickListener(stopMListener);
    }
};

private View.OnClickListener stopMListener = new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        ((Button)v).setText("MEASURE");
```

```
            stopMeasurement();
            v.setOnClickListener(startMListener);
        }
    };

    private BroadcastReceiver broadcastReceiver = new BroadcastReceiver() {
        @Override
        public void onReceive(Context context, Intent intent) {
            if (toMeasure) {
                List<ScanResult> found = wifiManager.getScanResults();
                resultsTextview.setText("");
                String[] RPIDs = new String[found.size()];
                Double[] values = new Double[found.size()];

                for (ScanResult sr : found) {
                    Log.d("FOUND: ", "" + sr.BSSID + ": " + sr.level);
                    resultsTextview.append(sr.BSSID + " = " + sr.level + "\n");
                    RPIDs[found.indexOf(sr)] = sr.BSSID;
                    values[found.indexOf(sr)] = (double) sr.level;
                }
                Database.postMeasurement(new RPMeasurement(RPIDs, values, currentRoom));
                Toast.makeText(getApplicationContext(),"SENT",Toast.LENGTH_SHORT).show();
                wifiManager.startScan();
            }
        }
    };
}
```

C.48 android/IPS_Admin/src/java/activities/MainActivity.java

```
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.view.View;
import android.view.Menu;
import android.view.MenuItem;
import com.example.sari.ips_admin.R;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.menu_main, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        int id = item.getItemId();

        return id == R.id.action_settings || super.onOptionsItemSelected(item);

    }
```

```java

    public void goToBuildingsActivity(View view) {

        Intent intent = new Intent(this, BuildingsActivity.class);

        startActivity(intent);

    }

}
```

C.49 android/IPS_Admin/src/java/activities/BuildingsActivity.java

```
import android.content.Intent;
import android.os.Bundle;
import android.os.StrictMode;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.util.Log;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import com.example.sari.ips_admin.R;
import com.example.sari.ips_admin.database.Database;
import com.example.sari.ips_admin.models.indoormapping.Building;
import java.util.ArrayList;

public class BuildingsActivity extends AppCompatActivity {
    Building[] buildings;
    private ListView museumListView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_buildings);
        Toolbar toolbar = (Toolbar) findViewById(R.id.my_toolbar);
        setSupportActionBar(toolbar);

        /*
        WARNING!!!
        Should not stay like this forever, fetching data must be done outside the UI
            thread.
        It is implemented like this only because of my laziness.
         */
        StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy
                                        .Builder().permitAll().build();
        StrictMode.setThreadPolicy(policy);
        museumListView = (ListView)findViewById(R.id.museumListView);
        museumListView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> parent, View view, int position,
                 long id) {
                Log.d("Item selected", "" + position);
                startEditBuildingActivity(position);

            }
        });
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.menu_buildings, menu);
        return super.onCreateOptionsMenu(menu);
    }

    @Override
    protected void onResume() {
        super.onResume();
        buildings = Database.getBuildings();
        ArrayList buildingNames = new ArrayList<>();
        for(Building b: buildings){
            buildingNames.add(b.getName());
        }
        ArrayAdapter museumListAdapter = new ArrayAdapter(this,
                android.R.layout.simple_expandable_list_item_1, buildingNames);
        museumListView.setAdapter(museumListAdapter);
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        if(item.getItemId() == R.id.action_addBuilding){
            startActivity(new Intent(this, AddBuildingActivity.class));
        }
        return super.onOptionsItemSelected(item);
    }

    private void startEditBuildingActivity(int position){
        Intent intent = new Intent(this, EditBuildingActivity.class);
        intent.putExtra("building",buildings[position]);
        startActivity(intent);
    }
}
```

C.50 android/IPS_Admin/src/java/activities/AddRoomActivity.java

```java
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.view.View;
import android.widget.EditText;
import com.example.sari.ips_admin.R;
import com.example.sari.ips_admin.database.Database;
import com.example.sari.ips_admin.models.indoormapping.Room;
import com.example.sari.ips_admin.tools.Point;
import com.example.sari.ips_admin.tools.RectangleDB;

public class AddRoomActivity extends AppCompatActivity {
    String building_id;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_add_room);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);
        getSupportActionBar().setDisplayHomeAsUpEnabled(false);
        building_id = getIntent().getStringExtra("building_id");
    }

    public void submitRoom(View v){
        String lbx   = ((EditText)
            findViewById(R.id.addroom_text_lbx)).getText().toString();
        String lby   = ((EditText)
            findViewById(R.id.addroom_text_lby)).getText().toString();
        String rbx   = ((EditText)
            findViewById(R.id.addroom_text_rbx)).getText().toString();
        String rby   = ((EditText)
            findViewById(R.id.addroom_text_rby)).getText().toString();
        String ltx   = ((EditText)
            findViewById(R.id.addroom_text_ltx)).getText().toString();
        String lty   = ((EditText)
            findViewById(R.id.addroom_text_lty)).getText().toString();
        String rtx   = ((EditText)
            findViewById(R.id.addroom_text_rtx)).getText().toString();
        String rty   = ((EditText)
            findViewById(R.id.addroom_text_rty)).getText().toString();
        String name  = ((EditText)
            findViewById(R.id.addroom_text_name)).getText().toString();
        String width = ((EditText)
            findViewById(R.id.addroom_text_width)).getText().toString();
        String length = ((EditText)
            findViewById(R.id.addroom_text_length)).getText().toString();



        if(!lbx.equals("") && !lby.equals("") && !rbx.equals("") &&
               !rby.equals("") && !ltx.equals("") && !lty.equals("") &&
               !rtx.equals("") && !rty.equals("") && !name.equals("") &&
               !width.equals("") && !length.equals("")){
            Point lb = new Point(Double.parseDouble(lbx),Double.parseDouble(lby));
            Point rb = new Point(Double.parseDouble(rbx),Double.parseDouble(rby));
            Point lt = new Point(Double.parseDouble(ltx),Double.parseDouble(lty));
            Point rt = new Point(Double.parseDouble(rtx),Double.parseDouble(rty));
            Room r = new Room(building_id, name, new RectangleDB(lt,rt,lb,rb),
                   Double.parseDouble(width), Double.parseDouble(length));
            Database.addRoom(building_id,r);
            this.finish();
        }

    }
}
```

C.51 android/IPS_Admin/src/java/activities/AddBuildingActivity.java

```
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.view.View;
import android.widget.EditText;
import com.example.sari.ips_admin.R;
import com.example.sari.ips_admin.database.Database;
import com.example.sari.ips_admin.models.indoormapping.Building;
import com.example.sari.ips_admin.tools.Point;
import com.example.sari.ips_admin.tools.RectangleDB;

public class AddBuildingActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_add_building);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

        getSupportActionBar().setDisplayHomeAsUpEnabled(false);
    }



    public void submitBuilding(View v){
        String lbx   = ((EditText) findViewById(R.id.addbuilding_text_lbx))
                       .getText().toString();
        String lby   = ((EditText) findViewById(R.id.addbuilding_text_lby))
                       .getText().toString();
        String rbx   = ((EditText) findViewById(R.id.addbuilding_text_rbx))
                       .getText().toString();
        String rby   = ((EditText) findViewById(R.id.addbuilding_text_rby))
                       .getText().toString();
        String ltx   = ((EditText) findViewById(R.id.addbuilding_text_ltx))
                              .getText().toString();
        String lty   = ((EditText) findViewById(R.id.addbuilding_text_lty))
                              .getText().toString();
        String rtx   = ((EditText) findViewById(R.id.addbuilding_text_rtx))
                              .getText().toString();
        String rty   = ((EditText) findViewById(R.id.addbuilding_text_rty))
                              .getText().toString();
        String name  = ((EditText) findViewById(R.id.addbuilding_text_name))
                              .getText().toString();
        String width = ((EditText) findViewById(R.id.addbuilding_text_width))
                              .getText().toString();
        String length = ((EditText) findViewById(R.id.addbuilding_text_length))
                              .getText().toString();

        if(!lbx.equals("") && !lby.equals("") && !rbx.equals("") &&
           !rby.equals("") && !ltx.equals("") && !lty.equals("") &&
           !rtx.equals("") && !rty.equals("") && !name.equals("") &&
           !width.equals("") && !length.equals("")){
            Point lb = new Point(Double.parseDouble(lbx),Double.parseDouble(lby));
            Point rb = new Point(Double.parseDouble(rbx),Double.parseDouble(rby));
            Point lt = new Point(Double.parseDouble(ltx),Double.parseDouble(lty));
            Point rt = new Point(Double.parseDouble(rtx),Double.parseDouble(rty));
            Building b = new Building(new RectangleDB(lt,rt,lb,rb),name
                    ,Double.parseDouble(width), Double.parseDouble(length));
            Database.addBuilding(b);
            this.finish();
        }
    }
}
```

C.52 android/IPS_Admin/src/java/activities/EditBuildingActivity.java

```java
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.util.Log;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import com.example.sari.ips_admin.R;
import com.example.sari.ips_admin.database.Database;
import com.example.sari.ips_admin.models.indoormapping.Building;
import com.example.sari.ips_admin.models.indoormapping.Room;
import java.util.ArrayList;


//CHECK IF GETTING BUILDINGS HERE IS ACTUALLY NEEDED.
public class EditBuildingActivity extends AppCompatActivity {
    private ListView museumListView;
    private Building b;
    private String building_id;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_edit_building);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);
        Log.d("ON CREATE", "EDIT BUILDING");
        b =(Building) getIntent().getSerializableExtra("building");
        building_id = b.getId();
    museumListView = (ListView)findViewById(R.id.room_list_edit_building);
    museumListView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
        @Override
        public void onItemClick(AdapterView<?> parent, View view, int position,
             long id) {
            startEditRoomActivity(position);
        }
    });
    getSupportActionBar().setDisplayHomeAsUpEnabled(false);
}

@Override
protected void onResume() {
    super.onResume();
    ArrayList roomNames = new ArrayList<>();
    b = Database.getBuilding(building_id);
    for(Room r: b.getRooms()){
        roomNames.add(r.getRoomName());
    }
    ArrayAdapter museumListAdapter = new ArrayAdapter(this,
            android.R.layout.simple_expandable_list_item_1, roomNames);
    museumListView.setAdapter(museumListAdapter);

}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.menu_edit_building, menu);
    return super.onCreateOptionsMenu(menu);
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    if(item.getItemId() == R.id.action_deleteBuilding){
        Database.deleteData("buildings",building_id);
```

```
            this.finish();
        }

        if(item.getItemId() == R.id.action_addRoom){
            Intent intent = new Intent(this,AddRoomActivity.class);
            intent.putExtra("building_id",building_id);
            startActivity(intent);
        }
        return super.onOptionsItemSelected(item);
    }

    private void startEditRoomActivity(int position){
        Intent intent = new Intent(this, EditRoomActivity.class);
        intent.putExtra("room_id",b.getRooms()[position].getId());
        startActivity(intent);
    }

    public void learnBuilding(View v){
        Database.getData("learn", building_id);
    }

}
```

C.53 android/MuseumGuide/build.gradle

```
android {
    compileSdkVersion 23
    buildToolsVersion "23.0.2"

    defaultConfig {
        applicationId "com.example.sari.museumguide"
        minSdkVersion 15
        targetSdkVersion 23
        versionCode 1
        versionName "1.0"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'),
                 'proguard-rules.pro'
        }
    }
}

dependencies {
    compile fileTree(include: ['*.jar'], dir: 'libs')
    testCompile 'junit:junit:4.12'
    compile 'com.android.support:appcompat-v7:23.1.1'
    compile 'com.android.support:design:23.1.1'
    compile 'com.google.android.gms:play-services:8.4.0'
    compile 'org.apache.directory.studio:org.apache.commons.io:2.4'
    compile files('src/main/libs/mysqlconn.jar')
}
```

C.54 android/MuseumGuide/src/res/layout/activity_building.xml

```
<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"
   android:fitsSystemWindows="true"
   tools:context="com.example.sari.museumguide.activities.BuildingActivity">

   <android.support.design.widget.AppBarLayout
       android:layout_width="match_parent"
       android:layout_height="wrap_content"
       android:theme="@style/AppTheme.AppBarOverlay">

       <android.support.v7.widget.Toolbar
           android:id="@+id/toolbar"
           android:layout_width="match_parent"
           android:layout_height="?attr/actionBarSize"
           android:background="?attr/colorPrimary"
           app:popupTheme="@style/AppTheme.PopupOverlay" />

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

   <include layout="@layout/content_building" />
</android.support.design.widget.CoordinatorLayout>
```

C.55 android/MuseumGuide/src/res/layout/content_guide.xml

```xml
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context="com.example.sari.museumguide.activities.GuideActivity"
    tools:showIn="@layout/activity_guide"
    android:orientation="vertical">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/current_room_textview"/>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/visitor_route_textview"/>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/visitor_exhibit_textview"/>

</LinearLayout>
```

C.56 android/MuseumGuide/src/res/layout/activity_guide.xml

```
<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"
    android:fitsSystemWindows="true"
    tools:context="com.example.sari.museumguide.activities.GuideActivity">

    <android.support.design.widget.AppBarLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:theme="@style/AppTheme.AppBarOverlay">

        <android.support.v7.widget.Toolbar
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr/actionBarSize"
            android:background="?attr/colorPrimary"
            app:popupTheme="@style/AppTheme.PopupOverlay" />

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

    <include layout="@layout/content_guide" />

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

C.57 android/MuseumGuide/src/res/layout/activity_main.xml

```
<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"

   android:fitsSystemWindows="true"

   tools:context=".activities.MainActivity">



   <include layout="@layout/content_main" />


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

C.58 android/MuseumGuide/src/res/layout/content_main.xml

```xml
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context=".activities.MainActivity"
    tools:showIn="@layout/activity_main">

    <ListView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/museumListView"/>



</RelativeLayout>
```

C.59 android/MuseumGuide/src/res/layout/content_building.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingBottom="@dimen/activity_vertical_margin"
   android:paddingLeft="@dimen/activity_horizontal_margin"
   android:paddingRight="@dimen/activity_horizontal_margin"
   android:paddingTop="@dimen/activity_vertical_margin"
   app:layout_behavior="@string/appbar_scrolling_view_behavior"
   tools:context="com.example.sari.museumguide.activities.BuildingActivity"
   tools:showIn="@layout/activity_building">


      <ListView
         android:layout_width="match_parent"
         android:layout_height="300dp"
         android:id="@+id/room_list_building"/>
      <LinearLayout
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         android:orientation="vertical"
         android:layout_below="@id/room_list_building">
         <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:orientation="horizontal">
            <TextView
               android:layout_width="wrap_content"
               android:layout_height="wrap_content"
               android:text="Deadline: "/>
            <EditText
               android:layout_width="50dp"
               android:layout_height="wrap_content"
               android:inputType="number"
```

```
                    android:id="@+id/deadline_edittext"/>

            </LinearLayout>
            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:id="@+id/estimated_time_textview"
                android:text="Estimated Time:"/>

            <Button
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="GO"
                android:onClick="startGuideActivity"/>
        </LinearLayout>

</RelativeLayout>
```

C.60 android/MuseumGuide/src/res/layout/custom_listview_item.xml

```xml
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/room_name"/>
    <CheckBox
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/chkBox"/>

    <SeekBar
        android:focusable="false"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:id="@+id/seekBar"
        android:layout_margin="10dp"
        android:progress="0"
        android:max="8"
        android:secondaryProgress="0"
        android:indeterminate="false" />



</LinearLayout>
```

C.61 android/MuseumGuide/src/res/layout/activity_maps.xml

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

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

android:id="@+id/map"

android:name="com.google.android.gms.maps.SupportMapFragment"

android:layout_width="match_parent"

android:layout_height="match_parent"

tools:context="com.example.sari.museumguide.activities.MapsActivity" />
```

C.62 android/MuseumGuide/src/res/menu/menu_main.xml

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

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

    tools:context=".activities.MainActivity">

    <item

        android:id="@+id/action_settings"

        android:orderInCategory="100"

        android:title="@string/action_settings"

        app:showAsAction="never" />

</menu>
```

C.63 android/MuseumGuide/src/AndroidManifest.xml

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="com.example.sari.museumguide">


   <!--
       The ACCESS_COARSE/FINE_LOCATION permissions are not required to use
       Google Maps Android API v2, but you must specify either coarse or fine
       location permissions for the 'MyLocation' functionality.
   -->
   <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
   <uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
   <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
   <uses-permission android:name="android.permission.CHANGE_WIFI_STATE"/>
   <uses-permission android:name="android.permission.BLUETOOTH"/>
   <application
       android:allowBackup="true"
       android:icon="@mipmap/ic_launcher"
       android:label="@string/app_name"
       android:supportsRtl="true"
       android:theme="@style/AppTheme">
       <activity
           android:name=".activities.MainActivity"
           android:label="@string/app_name"
           android:screenOrientation="portrait"
           android:theme="@style/AppTheme.NoActionBar">
           <intent-filter>
               <action android:name="android.intent.action.MAIN" />


               <category android:name="android.intent.category.LAUNCHER" />
           </intent-filter>
       </activity>
       <!--
            The API key for Google Maps-based APIs is defined as a string resource.
            (See the file "res/values/google_maps_api.xml").
            Note that the API key is linked to the encryption key used to sign the APK.
```

```
     You need a different API key for each encryption key, including the
         release key that is used to
     sign the APK for publishing.
     You can define the keys for the debug and release targets in src/debug/
         and src/release/.
-->
<meta-data
    android:name="com.google.android.geo.API_KEY"
    android:value="@string/google_maps_key" />

<activity
    android:name=".activities.MapsActivity"
    android:label="@string/title_activity_maps">
    <meta-data
        android:name="android.support.PARENT_ACTIVITY"
        android:value="com.example.sari.museumguide.activities.MainActivity" />
</activity>
<activity
    android:name=".activities.BuildingActivity"
    android:label="@string/title_activity_building"
    android:parentActivityName=".activities.MainActivity"
    android:screenOrientation="portrait"
    android:launchMode="singleTask"
    android:theme="@style/AppTheme.NoActionBar">
    <meta-data
        android:name="android.support.PARENT_ACTIVITY"
        android:value="com.example.sari.museumguide.activities.MainActivity" />
</activity>
<activity
    android:name=".activities.GuideActivity"
    android:label="@string/title_activity_guide"
    android:screenOrientation="portrait"
    android:parentActivityName=".activities.BuildingActivity"
    android:theme="@style/AppTheme.NoActionBar">
    <meta-data
        android:name="android.support.PARENT_ACTIVITY"
```

```
                android:value="com.example.sari.museumguide.activities.BuildingActivity"
                    />
        </activity>
    </application>


</manifest>
```

C.64 android/MuseumGuide/src/java/test/ModelTest.java

```
import com.example.sari.museumguide.models.indoormapping.Building;
import com.example.sari.museumguide.models.indoormapping.Floor;
import com.example.sari.museumguide.models.indoormapping.Room;
import com.example.sari.museumguide.tools.Point;
import com.example.sari.museumguide.tools.Rectangle;

import org.junit.BeforeClass;
import org.junit.Test;

import static org.junit.Assert.*;

/**
 * To work on unit tests, switch the Test Artifact in the Build Variants view.
 */

/* Test for basic building

   0___________10
   |     |     |
   |  1  |  2  |
   |     |     |
   |_____|_____|Floor 0
   10

   0___________10
   |  3  |     |
   |_____|  4  |
   |  5  |     |
   |_____|_____|Floor 1
   10

   0___________10
   |  6  |  7  |
   |_____|_____|
   |  8  |  9  |
   |_____|_____|Floor 2
   10

 */
public class ModelTest {
   static Room r1, r2, r3, r4, r5, r6, r7, r8, r9;
   static Floor f0, f1, f2;
   static Building testBuilding;

    @BeforeClass
    public static void setUp() {
        r1 = new Room(new Rectangle(new Point(0,0),5,10));
        r2 = new Room(new Rectangle(new Point(5,0),5,10));

        r3 = new Room(new Rectangle(new Point(0,0),5,5));
        r4 = new Room(new Rectangle(new Point(5,0),5,10));
        r5 = new Room(new Rectangle(new Point(0,5),5,5));

        r6 = new Room(new Rectangle(new Point(0,0),5,5));
        r7 = new Room(new Rectangle(new Point(5,0),5,5));
        r8= new Room(new Rectangle(new Point(0,5),5,5));
        r9 = new Room(new Rectangle(new Point(5,5),5,5));
        Room[] f0Array = {r1,r2};
        Room[] f1Array = {r3,r4,r5};
        Room[] f2Array = {r6,r7,r8,r9};

        f0 = new Floor(f0Array);
        f1 = new Floor(f1Array);
        f2 = new Floor(f2Array);

        Floor[] floors = {f0,f1,f2};

        testBuilding = new Building(floors);
    }

    @Test
    public void addition_isCorrect() throws Exception {
        assertEquals(4, 2 + 2);
    }


    @Test
    public void buildingTest(){
        assertEquals(testBuilding.getFloors().length, 3);
        System.out.print(testBuilding.getFloors()[1].getNeighbours(r8).length);


    }
}
```

C.65 android/MuseumGuide/src/java/tools/Rectangle.java

```
import java.io.Serializable;

public class Rectangle implements Serializable{
    private Point coordinates;
    private double width, length;

    public Rectangle(Point coordinates, double width, double length){
        this.coordinates = coordinates;
        this.width = width;
        this.length = length;
    }

    public Point getCoordinates() {
        return coordinates;
    }

    public double getWidth() {
        return width;
    }

    public double getLength() {
        return length;
    }

    public void setCoordinates(Point coordinates) {
        this.coordinates = coordinates;
    }

    public void setWidth(double width) {
        this.width = width;
    }

    public void setLength(double length) {
```

```java
        this.length = length;
    }

    public Point getCenter(){
        return new Point((width+coordinates.getX())/2,(length+coordinates.getY())/2);
    }

}
```

C.66 android/MuseumGuide/src/java/tools/Point.java

```
import java.io.Serializable;



public class Point implements Serializable {
    private double x, y;

    public Point(double x, double y)
    {
        this.x = x;
        this.y = y;
    }

    public double getX() {
        return x;
    }

    public void setX(double x) {
        this.x = x;
    }

    public double getY() {
        return y;
    }

    public void setY(double y) {
        this.y = y;
    }
}
```

C.67 android/MuseumGuide/src/java/tools/RectangleDB.java

```
import java.io.Serializable;

public class RectangleDB implements Serializable{
    private Point lt,rt,lb,rb;

    public RectangleDB(Point lt, Point rt, Point lb, Point rb) {
        this.lt = lt;
        this.rt = rt;
        this.lb = lb;
        this.rb = rb;
    }

    public Point getLt() {
        return lt;
    }

    public void setLt(Point lt) {
        this.lt = lt;
    }

    public Point getRt() {
        return rt;
    }

    public void setRt(Point rt) {
        this.rt = rt;
    }

    public Point getLb() {
        return lb;
    }

    public void setLb(Point lb) {
        this.lb = lb;
    }

    public Point getRb() {
        return rb;
    }

    public void setRb(Point rb) {
        this.rb = rb;
    }
}
```

C.68 android/MuseumGuide/src/java/tools/CustomAdapter.java

```
import android.content.Context;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.CheckBox;
import android.widget.SeekBar;
import android.widget.TextView;

import com.example.sari.museumguide.R;
import com.example.sari.museumguide.activities.BuildingActivity;

import java.util.ArrayList;



public class CustomAdapter extends ArrayAdapter<String> {
    boolean[] checked;
    int[] progresses;
    public CustomAdapter(Context context, ArrayList<String> rooms){
         super(context, R.layout.custom_listview_item, rooms);
         checked = new boolean[rooms.size()];
         progresses = new int[rooms.size()];
    }

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

        if(convertView == null){
            LayoutInflater layoutInflater = LayoutInflater.from(getContext());
            convertView =
                 layoutInflater.inflate(R.layout.custom_listview_item,parent,false);
        }
        String room = getItem(position);
        TextView textView = (TextView) convertView.findViewById(R.id.room_name);
        final CheckBox checkBox = (CheckBox) convertView.findViewById(R.id.chkBox);
        SeekBar seekBar = (SeekBar) convertView.findViewById(R.id.seekBar);
        textView.setText(room);
        checkBox.setTag(position);
        seekBar.setTag(position);
        checkBox.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Log.d("Clicked on: ", "" + v.getTag() + " " + checked[(int)
                    v.getTag()]);
                checked[(int) v.getTag()] = !checked[(int) v.getTag()];
                Log.d("Changed: ", "" + v.getTag() + " " + checked[(int) v.getTag()]);
                ((BuildingActivity) v.getContext()).updateEstTime();
            }
        });
        seekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
            @Override
            public void onProgressChanged(SeekBar seekBar, int progress, boolean
                 fromUser) {
                progresses[(int) seekBar.getTag()] = progress;
            }

            @Override
            public void onStartTrackingTouch(SeekBar seekBar) {


            }


            @Override
            public void onStopTrackingTouch(SeekBar seekBar) {


            }
        });
        return convertView;
    }
    public boolean isChecked(int position) {
        return checked[position];
    }

    public int getProgress(int position){
        return progresses[position]+1;
    }

    public void check(int position){
        checked[position] = !checked[position];
    }
}
```

C.69 android/MuseumGuide/src/java/database/Database.java

```java
import android.util.Log;
import com.example.sari.museumguide.models.indoormapping.Building;
import com.example.sari.museumguide.models.indoormapping.Floor;
import com.example.sari.museumguide.models.indoormapping.Room;
import com.example.sari.museumguide.models.positioning.RPMeasurement;
import com.example.sari.museumguide.tools.Point;
import com.example.sari.museumguide.tools.RectangleDB;
import org.apache.commons.io.IOUtils;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;
import java.io.BufferedInputStream;
import java.io.BufferedOutputStream;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.ArrayList;

/**
 * Class that connects to the api.
 */
public class Database {

    public final static String API_URL = "http://192.168.1.106:3000/";
    public static Building[] getBuildings(){
        String StringData = getData("buildings");
        ArrayList<Building> toReturn = new ArrayList<>();
        Log.d("BUILDINGS: ", StringData);
        try {
            JSONArray JSONData = new JSONArray(StringData);
            for(int i = 0; i<JSONData.length();++i){
                JSONObject building = JSONData.getJSONObject(i);
                Point lt = new Point(building.getJSONObject("rectangle")
                                          .getJSONObject("lt").getDouble("x"),
                                  building.getJSONObject("rectangle")
                                          .getJSONObject("lt").getDouble("y"));
                Point rt = new Point(building.getJSONObject("rectangle")
                                          .getJSONObject("rt").getDouble("x"),
                                  building.getJSONObject("rectangle")
                                          .getJSONObject("rt").getDouble("y"));
                Point lb = new Point(building.getJSONObject("rectangle")
                                          .getJSONObject("lb").getDouble("x"),
                                  building.getJSONObject("rectangle")
                                          .getJSONObject("lb").getDouble("y"));
                Point rb = new Point(building.getJSONObject("rectangle")
                                          .getJSONObject("rb").getDouble("x"),
                                  building.getJSONObject("rectangle")
                                          .getJSONObject("rb").getDouble("y"));
                RectangleDB r = new RectangleDB(lt,rt,lb,rb);
                Building toAdd = new Building(building.getString("_id"),r,
                                          building.getString("name"),
                                          building.getDouble("width"),
                                          building.getDouble("length"),
                                          getRooms(building.getString("_id")));
                toReturn.add(toAdd);
                getRooms(building.getString("_id"));
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        return toReturn.toArray(new Building[toReturn.size()]);
    }

    public static boolean addBuilding(Building building){
        JSONObject buildingToAdd = new JSONObject();
        try {
            buildingToAdd.put("name",building.getName());
            buildingToAdd.put("width", building.getWidth());
        buildingToAdd.put("length",building.getLength());
        JSONObject rectangle = new JSONObject();
        rectangle.put("lt", new JSONObject().put("x",
            building.getRectangle().getLt().getX())
                                .put("y",
                                    building.getRectangle().getLt().getY()))
                .put("rt", new JSONObject().put("x",
                    building.getRectangle().getRt().getX())
                                .put("y",
                                    building.getRectangle().getRt().getY()))
                .put("lb", new JSONObject().put("x",
                    building.getRectangle().getLb().getX())
                                .put("y",
                                    building.getRectangle().getLb().getY()))
                .put("rb", new JSONObject().put("x",
                    building.getRectangle().getRb().getX())
                                .put("y",
                                    building.getRectangle().getRb().getY()));

        buildingToAdd.put("rectangle", rectangle);
        postData("buildings", "", buildingToAdd.toString());
    } catch (Exception e) {
        e.printStackTrace();
    }

    return true;
}

public static boolean postMeasurement(RPMeasurement measurement){

    JSONArray JSONAllM= new JSONArray();
    JSONObject toSend = new JSONObject();
    try {
        for(int i = 0; i<measurement.getReadings().size(); ++i){
            JSONAllM.put(new
                JSONObject().put("RPID",measurement.getReadings().get(i).first)
```

```
                  .put("value",measurement.getReadings().get(i).second));
        }


        toSend.put("rpv_pair", JSONAllM);
    } catch (Exception e) {
        e.printStackTrace();
    }
    postData("measurements", measurement.getRoom_id(), toSend.toString());


    return true;
}




//Only storing duration so far
public static boolean postLocationData(String room_id, long duration){
   JSONObject toSend = new JSONObject();
    try {
        toSend.put("duration",duration);
    } catch (JSONException e) {
        e.printStackTrace();
    }
    postData("locationdata", room_id, toSend.toString());
    return true;
}


public static String getRoute(String building_id, String rooms_exc, int deadline){


    JSONObject toSend = new JSONObject();
    JSONArray selectedRooms = new JSONArray();
    try {
        String[] rooms = rooms_exc.split(",");
    for(String s:rooms){
        selectedRooms.put(new JSONObject().put("id", (s.split(":"))[0])
                                   .put("excitement", (s.split(":"))[1]));
    }
    toSend.put("deadline",deadline);
    toSend.put("selected_rooms",selectedRooms);
    } catch (JSONException e) {
        e.printStackTrace();
    }
    return postData("route",building_id,toSend.toString());
}

public static Building getBuilding(String id){
    String StringData = getData("buildings",id);
    Building toReturn = null;
    try {
        JSONObject building = new JSONObject(StringData);
        Point lt = new Point(building.getJSONObject("rectangle")
                              .getJSONObject("lt").getDouble("x"),
                      building.getJSONObject("rectangle")
                              .getJSONObject("lt").getDouble("y"));
        Point rt = new Point(building.getJSONObject("rectangle")
                              .getJSONObject("rt").getDouble("x"),
              building.getJSONObject("rectangle")
                              .getJSONObject("rt").getDouble("y"));
        Point lb = new Point(building.getJSONObject("rectangle")
                              .getJSONObject("lb").getDouble("x"),
              building.getJSONObject("rectangle")
                              .getJSONObject("lb").getDouble("y"));
        Point rb = new Point(building.getJSONObject("rectangle")
                               .getJSONObject("rb").getDouble("x"),
             building.getJSONObject("rectangle").getJSONObject("rb").getDouble("y"));
        RectangleDB r = new RectangleDB(lt,rt,lb,rb);
        toReturn = new
            Building(building.getString("_id"),r,building.getString("name"),
              building.getDouble("width"),building.getDouble("length"),
              getRooms(building.getString("_id")));

    } catch (Exception e) {
        e.printStackTrace();
    }
```

```
        return toReturn;
    }

    public static Floor[] getFloors(String building_id){
        return null;
    }

    public static Room[] getRooms(String building_id){
        String StringData = getData("rooms", building_id);
        Log.d("Rooms",StringData);
        ArrayList<Room> toReturn = new ArrayList<>();
        try {
            JSONArray JSONData = new JSONArray(StringData);
            for(int i = 0; i<JSONData.length();++i){
                JSONObject room = JSONData.getJSONObject(i);
                Point lt = new Point(room.getJSONObject("rectangle")
                                        .getJSONObject("lt").getDouble("x"),
                                    room.getJSONObject("rectangle")
                                        .getJSONObject("lt").getDouble("y"));
                Point rt = new Point(room.getJSONObject("rectangle")
                                        .getJSONObject("rt").getDouble("x"),
                                    room.getJSONObject("rectangle")
                                        .getJSONObject("rt").getDouble("y"));
                Point lb = new Point(room.getJSONObject("rectangle")
                                        .getJSONObject("lb").getDouble("x"),
                                    room.getJSONObject("rectangle")
                                        .getJSONObject("lb").getDouble("y"));
                Point rb = new Point(room.getJSONObject("rectangle")
                                        .getJSONObject("rb").getDouble("x"),
                                    room.getJSONObject("rectangle")
                                        .getJSONObject("rb").getDouble("y"));
                RectangleDB r = new RectangleDB(lt,rt,lb,rb);
                Room toAdd = new
                    Room(room.getString("_id"),building_id,room.getString("name"),
                       r,room.getDouble("width"),room.getDouble("length"),
                       room.getDouble("est_time"));
                toReturn.add(toAdd);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        for(Room room: toReturn){
            String exhibits = getData("exhibits", room.getId());
            try {
                JSONArray array = new JSONArray(exhibits);
                for(int i = 0; i<array.length(); ++i){
                    room.addExhibit(array.getJSONObject(i).getString("name")+","
                            +array.getJSONObject(i).getString("rpid"));
                    Log.d("ADDED EXHIBIT",array.getJSONObject(i).getString("name")+":"
                            +array.getJSONObject(i).getString("rpid"));
                }
            } catch (JSONException e) {
                e.printStackTrace();
            }
        }
        return toReturn.toArray(new Room[toReturn.size()]);
    }

    public static boolean addRoom(String building_id, Room room){
        JSONObject roomToAdd = new JSONObject();
        try {
            roomToAdd.put("name", room.getRoomName());
            roomToAdd.put("width", room.getWidth());
            roomToAdd.put("length", room.getLength());
            JSONObject rectangle = new JSONObject();
            rectangle.put("lt", new JSONObject().put("x",
                 room.getRectangleDB().getLt().getX())
                    .put("y", room.getRectangleDB().getLt().getY()))
                    .put("rt", new JSONObject().put("x",
                        room.getRectangleDB().getRt().getX())
                            .put("y", room.getRectangleDB().getRt().getY()))
                    .put("lb", new JSONObject().put("x",
                        room.getRectangleDB().getLb().getX())
                            .put("y", room.getRectangleDB().getLb().getY()))
                    .put("rb", new JSONObject().put("x",
                        room.getRectangleDB().getRb().getX())
                            .put("y", room.getRectangleDB().getRb().getY()));

            roomToAdd.put("rectangle", rectangle);
            postData("rooms", building_id, roomToAdd.toString());

        } catch (Exception e) {
            e.printStackTrace();
        }

        return true;
    }



    public static String getData(String... params) {
        String data = "";
        String building_id = "";
        String request = params[0];
        InputStream inputStream;

        if (params.length >= 2){
            building_id = params[1];
        }

        try {
            URL url = new URL(API_URL+request+"/"+building_id);
            HttpURLConnection connection = (HttpURLConnection) url.openConnection();
            inputStream = new BufferedInputStream(connection.getInputStream());
            data = IOUtils.toString(inputStream,"UTF-8");
            connection.disconnect();
        } catch (Exception e) {
            e.printStackTrace();
        }
        return data;

    }

    public static String postData(String... params){
        String data = params[2];
        Log.d("Data to send", data);
        String building_id = "";
        String request = params[0];
        InputStream is;
        String dataRec = "";
        if (params.length >= 2){
            building_id = params[1];
        }

        try {
            URL url = new URL(API_URL+request+"/"+building_id);
            HttpURLConnection connection = (HttpURLConnection) url.openConnection();
            Log.d("Error", "Ex " + url.toString());
            connection.setDoOutput(true);
            connection.setChunkedStreamingMode(0);
            connection.setRequestProperty("Content-Type", "application/json");
            OutputStream os = new BufferedOutputStream(connection.getOutputStream());

            os.write(data.getBytes());
            os.flush();
            is = new BufferedInputStream(connection.getInputStream());
            dataRec = IOUtils.toString(is,"UTF-8");
            Log.d("Found:", "Room:"+dataRec);
            if (connection.getResponseCode() != HttpURLConnection.HTTP_CREATED) {
                throw new RuntimeException("Failed : HTTP error code : "
                        + connection.getResponseCode());
            }
            os.close();
            connection.disconnect();
        } catch (Exception e) {
            Log.d("Error","Ex "+e);
            e.printStackTrace();
        }

        return dataRec;
    }

    public static void deleteData(String... params){
        String building_id = "";
        String request = params[0];



        if (params.length == 2){
            building_id = params[1];
        }

        try {
            URL url = new URL(API_URL+request+"/"+building_id);
            HttpURLConnection connection = (HttpURLConnection) url.openConnection();
            connection.setDoOutput(true);
            connection.setRequestMethod("DELETE");
            connection.getContent();
            connection.disconnect();
        } catch (Exception e) {
            Log.d("Error","Ex "+e);
            e.printStackTrace();
        }
    }

    public static String classify(RPMeasurement measurement, String building_id){
        JSONArray JSONAllM= new JSONArray();
        JSONObject toSend = new JSONObject();
        JSONArray toSendArray = new JSONArray();
        try {
            for(int i = 0; i<measurement.getReadings().size(); ++i){
                JSONAllM.put(new
                    JSONObject().put("RPID",measurement.getReadings().get(i).first)
                       .put("value",measurement.getReadings().get(i).second));
            }


            toSend.put("rpv_pair", JSONAllM);
            toSend.put("room_id","?");
            toSendArray.put(toSend);
        } catch (Exception e) {
            e.printStackTrace();
        }
        return postData("locate", building_id, toSend.toString());
    }


}
```

C.70 android/MuseumGuide/src/java/models/RPMeasurement.java

```
import android.util.Pair;

import java.util.ArrayList;


public class RPMeasurement {
    //String RPID; //this is a unique identifier for the reference point. (eg.: MAC
        ADDRESS)
    String room_id;
    ArrayList<Pair<String,Double>> readings = new ArrayList<>();

    public RPMeasurement(String[] RPIDs, Double[] values,String room_id) {
        for(int i = 0; i<RPIDs.length; ++i)
        {
            readings.add(new Pair<>(RPIDs[i],values[i]));
        }
        this.room_id = room_id;
    }

    public ArrayList<Pair<String, Double>> getReadings() {
        return readings;
    }



    public String getRoom_id() {
        return room_id;
    }

    public void setRoom_id(String room_id) {
        this.room_id = room_id;
    }
}
```

C.71 android/MuseumGuide/src/java/models/Room.java

```
import com.example.sari.museumguide.tools.Point;
import com.example.sari.museumguide.tools.Rectangle;
import com.example.sari.museumguide.tools.RectangleDB;

import java.io.Serializable;
import java.util.ArrayList;

public class Room implements Serializable{
   private String roomName;
   private String id;
   private String building_id;
   private Rectangle roomRectangle;
   private RectangleDB rectangleDB;
   private String roomDescription;
   private double width,length;
   private double est_time;
   private ArrayList<String> exhibits;



   public Room(String id, String building_id, String roomName,
            RectangleDB rectangleDB, double width, double length, double est_time){
      this.roomName = roomName;
      this.id = id;
      this.width = width;
      this.length = length;
      this.rectangleDB = rectangleDB;
      this.building_id = building_id;
      this.roomRectangle = new Rectangle(new Point(0,0),0,0); //for testing!!!
      this.est_time = est_time;
      exhibits = new ArrayList<>();
   }

   public String getBuilding_id() {
      return building_id;
   }

   public Room(String building_id, String roomName,
            RectangleDB rectangleDB, double width, double length){
      this.building_id = building_id;

      this.roomName = roomName;
      this.width = width;
      this.length = length;
      this.roomDescription = "";
      this.rectangleDB = rectangleDB;
      exhibits = new ArrayList<>();
      this.roomRectangle = new Rectangle(new Point(0,0),0,0); //for testing!!!
   }

   public Room(Rectangle r){
      this.roomRectangle = r;
      roomName = "";
      roomDescription = "";
      exhibits = new ArrayList<>();
   }

   public String getRoomName() {
      return roomName;
   }

   public void setRoomName(String roomName) {
      this.roomName = roomName;
   }

   public Rectangle getRoomRectangle() {
      return roomRectangle;
   }

   public void setRoomRectangle(Rectangle roomRectangle) {
```

```
    this.roomRectangle = roomRectangle;
}

public String getRoomDescription() {
    return roomDescription;
}

public void setRoomDescription(String roomDescription) {
    this.roomDescription = roomDescription;
}

public boolean isNeighbour(Room room){
    if(this == room)
        return false;

    if(this.getRoomRectangle().getCoordinates().getX()
            + this.getRoomRectangle().getWidth()
            == room.getRoomRectangle().getCoordinates().getX()
            || this.getRoomRectangle().getCoordinates().getY()
            + this.getRoomRectangle().getLength()
            == room.getRoomRectangle().getCoordinates().getY()
            || room.getRoomRectangle().getCoordinates().getX()
            + room.getRoomRectangle().getWidth()
            == this.getRoomRectangle().getCoordinates().getX()
            || room.getRoomRectangle().getCoordinates().getY()
            + room.getRoomRectangle().getLength()
            == this.getRoomRectangle().getCoordinates().getY())
        return true;

    return false;

}

public String getId() {
    return id;
}
```

```
public RectangleDB getRectangleDB() {
    return rectangleDB;
}

public double getWidth() {
    return width;
}

public double getLength() {
    return length;
}

public void setEst_time(double est_time) {
    this.est_time = est_time;
}

public void addExhibit(String exhibit){
    exhibits.add(exhibit);
}

public ArrayList<String> getExhibits(){
    return exhibits;
}

public String hasExhibit(String exhibit_id){
    for(String e: exhibits){
        if(e.split(",")[1].equals(exhibit_id)){
            return e;
        }
    }
    return null;
}

public double getEst_time() {
        return est_time;
    }


}
```

C.72 android/MuseumGuide/src/java/models/Position.java

```
public class Position {
    RPMeasurement[] measurements;

    public Position(RPMeasurement[] measurements){
        this.measurements = measurements;
    }

    public RPMeasurement[] getMeasurements() {
        return measurements;
    }

    public void setMeasurements(RPMeasurement[] measurements) {
        this.measurements = measurements;
    }
}
```

C.73 android/MuseumGuide/src/java/models/Floor.java

```
import java.io.Serializable;
import java.util.ArrayList;



public class Floor implements Serializable{
    Room[] rooms;

    public Floor(Room[] rooms)
    {
        this.rooms = rooms;
    }

    public Room[] getRooms() {
        return rooms;
    }



    public Room[] getNeighbours(Room room) {
        ArrayList<Room> neighbours = new ArrayList<>();
        for(Room r:rooms){
            if(r.isNeighbour(room))
                neighbours.add(r);
        }
        return neighbours.toArray(new Room[neighbours.size()]);



    }

    public void setRooms(Room[] rooms) {
        this.rooms = rooms;
    }

}
```

C.74 android/MuseumGuide/src/java/models/Building.java

```
import com.example.sari.museumguide.tools.RectangleDB;

import java.io.Serializable;


public class Building implements Serializable{
   private RectangleDB rectangle;
   private String name;
   private double width;
   private double length;
   private String id;
   private Room[] rooms;



   public Building(String id, RectangleDB rectangle,
                 String name, double width, double length, Room[] rooms) {
      this.rectangle = rectangle;
      this.id = id;
      this.name = name;
      this.width = width;
      this.length = length;
      this.rooms = rooms;
   }

   public Building(RectangleDB rectangle, String name, double width, double length) {
      this.rectangle = rectangle;
      this.id = "";
      this.name = name;
      this.width = width;
      this.length = length;
   }
```

```
public void setRectangle(RectangleDB rectangle) {
    this.rectangle = rectangle;
}

public void setName(String name) {
    this.name = name;
}

public void setWidth(double width) {
    this.width = width;
}

public void setLength(double length) {
    this.length = length;
}

public RectangleDB getRectangle() {

    return rectangle;
}

public String getName() {
    return name;
}

public double getWidth() {
    return width;
}

public double getLength() {
    return length;
}

public String getId() {
    return id;
    }

    public void setId(String id) {
        this.id = id;
    }

    public Room[] getRooms() {
        return rooms;
    }

    public void setRooms(Room[] rooms) {
        this.rooms = rooms;
    }

    public Room findRoomById(String room_id){
        for(Room r:rooms){
            if(r.getId().equals(room_id)){
                return r;
            }
        }
        return null;
    }
}
```

C.75 android/MuseumGuide/src/java/activities/MainActivity.java

```
import android.content.Intent;
import android.os.Bundle;
import android.os.StrictMode;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.view.Menu;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import com.example.sari.museumguide.R;
import com.example.sari.museumguide.database.Database;
import com.example.sari.museumguide.models.indoormapping.Building;
import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {
    Building[] buildings;
    private ListView museumListView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        StrictMode.ThreadPolicy policy = new StrictMode
                .ThreadPolicy.Builder().permitAll().build();
        StrictMode.setThreadPolicy(policy);
        museumListView = (ListView)findViewById(R.id.museumListView);
        museumListView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> parent, View view, int position,
                 long id) {
                startBuildingActivity(position);
            }
        });
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.menu_main, menu);
        return true;
    }

    @Override
    protected void onResume() {
        super.onResume();
        buildings = Database.getBuildings();
        ArrayList buildingNames = new ArrayList<String>();
        for(Building b: buildings){
            buildingNames.add(b.getName());
        }
        ArrayAdapter museumListAdapter =
                new ArrayAdapter(this, android.R.layout.simple_expandable_list_item_1,
                        buildingNames);
        museumListView.setAdapter(museumListAdapter);
    }

    private void startMapsActivity(int position){
        Intent intent = new Intent(this, MapsActivity.class);
        startActivity(intent);
    }

    private void startBuildingActivity(int position){
        Intent intent = new Intent(this, BuildingActivity.class);
        intent.putExtra("building",buildings[position]);
        startActivity(intent);
    }

}
```

C.76 android/MuseumGuide/src/java/activities/GuideActivity.java

```java
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.net.wifi.ScanResult;
import android.net.wifi.WifiManager;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.util.Log;
import android.widget.TextView;
import com.example.sari.museumguide.R;
import com.example.sari.museumguide.database.Database;
import com.example.sari.museumguide.models.indoormapping.Building;
import com.example.sari.museumguide.models.indoormapping.Room;
import com.example.sari.museumguide.models.positioning.RPMeasurement;
import java.util.List;
import java.util.Timer;
import java.util.TimerTask;
import java.util.concurrent.TimeUnit;

public class GuideActivity extends AppCompatActivity {
    private static final int BLUETOOTH_THRESHOLD = -40;
    private Building b;
    WifiManager wifiManager;
    BluetoothAdapter bluetoothAdapter;
    Room currentRoom;
    String selectedRooms;
    boolean btAvailable = false;
    long timeOfChange = System.currentTimeMillis();
    String currentExhibit;
```

```
TextView currentRoomView;

TextView routeTextView;

TextView exhibitTextView;

@Override

protected void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);

    setContentView(R.layout.activity_guide);

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

    setSupportActionBar(toolbar);

    getSupportActionBar().setDisplayHomeAsUpEnabled(true);

    bluetoothAdapter = BluetoothAdapter.getDefaultAdapter();

    if (bluetoothAdapter == null) {

        btAvailable = false;

    } else if (!bluetoothAdapter.isEnabled()) {

        Intent enableBtIntent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);

        startActivityForResult(enableBtIntent, 1);

    }

    wifiManager = (WifiManager) getSystemService(Context.WIFI_SERVICE);

    registerReceiver(wifiBroadcastReceiver,

            new IntentFilter(WifiManager.SCAN_RESULTS_AVAILABLE_ACTION));

    b =(Building) getIntent().getSerializableExtra("building");

    selectedRooms = getIntent().getStringExtra("selected_rooms");

    currentRoomView = (TextView)findViewById(R.id.current_room_textview);

    routeTextView = (TextView)findViewById(R.id.visitor_route_textview);

    exhibitTextView = (TextView) findViewById(R.id.visitor_exhibit_textview);

    String route = Database.getRoute(b.getId(),selectedRooms,

            getIntent().getIntExtra("deadline",10000));

    routeTextView.setText(parseRoute(route));

    currentRoom = null; //Entrance room?

    wifiManager.startScan();

    currentExhibit = "";

    exhibitTextView.setText("No exhibit nearby");

    registerReceiver(btBroadcastReceiver,

            new IntentFilter(BluetoothDevice.ACTION_FOUND));

    TimerTask tt = new TimerTask() {

        @Override
```

```
            public void run() {
                bluetoothAdapter.startDiscovery();
            }
        };
        new Timer().scheduleAtFixedRate(tt,0,5000);

    }

    @Override
    protected void onStop()
    {
        unregisterReceiver(wifiBroadcastReceiver);
        unregisterReceiver(btBroadcastReceiver);
        super.onStop();
    }

    private BroadcastReceiver wifiBroadcastReceiver = new BroadcastReceiver() {
        @Override
        public void onReceive(Context context, Intent intent) {

            List<ScanResult> found = wifiManager.getScanResults();
            String[] RPIDs = new String[found.size()];
            Double[] values = new Double[found.size()];

            currentRoomView.setText("");
            for (ScanResult sr : found) {
                Log.d("FOUND: ", "" + sr.BSSID + ": " + sr.level);
                RPIDs[found.indexOf(sr)] = sr.BSSID;
                values[found.indexOf(sr)] = (double) sr.level;
            }
            String foundRoomID = Database.classify(new RPMeasurement(RPIDs, values,
                 null),
                    b.getId());
            String[] results = foundRoomID.split(",");
            for(String result:results){
                for(Room room:b.getRooms()){
```

```
            if(room.getId().equals(result.split(":")[1])){
                currentRoomView.setText(currentRoomView.getText()
                        +result.split(":")[0]+":"+room.getRoomName()+"\n");
                shouldChangeRoom(room);
                break;
            }
        }
    }
    shouldChangeRoom(currentRoom);
    wifiManager.startScan();
    }
};

private final BroadcastReceiver btBroadcastReceiver = new BroadcastReceiver() {
    public void onReceive(Context context, Intent intent) {
        String action = intent.getAction();
        if(BluetoothDevice.ACTION_FOUND.equals(action)) {
            int  rssi = intent.getShortExtra(BluetoothDevice.EXTRA_RSSI,
                 Short.MIN_VALUE);
            BluetoothDevice device =
                 intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
            Log.d("BLUE:", device.getAddress()+" "+device.getName()+" "+rssi);
            if(currentExhibit.isEmpty())
            {
                if(device.getName() != null && currentRoom != null){
                    String exhibit = currentRoom.hasExhibit(device.getAddress());
                    if(exhibit !=null && rssi > BLUETOOTH_THRESHOLD){
                        bluetoothAdapter.cancelDiscovery();
                        currentExhibit = device.getAddress();
                        exhibitTextView.setText("You are viewing:
                            "+exhibit.split(",")[0]);
                    }
                }
            } else {
                if(device.getAddress().equals(currentExhibit)
                        && rssi >= BLUETOOTH_THRESHOLD){
```

```
                    bluetoothAdapter.cancelDiscovery();
                } else {
                    currentExhibit = "";
                    exhibitTextView.setText("No exhibit nearby");
                }
            }
        }
    }
};

public boolean shouldChangeRoom(Room r){
    boolean shouldChange = false;
    if(currentRoom == null){
        shouldChange = true;
        currentRoom = r;
    }
    if(!currentRoom.equals(r)){
        currentRoom = r;
        shouldChange = true;
    }
    if(shouldChange){
        long curTime = System.currentTimeMillis();
        long duration = curTime-timeOfChange;
        timeOfChange = curTime;
        Database.postLocationData(currentRoom.getId(),
                TimeUnit.MILLISECONDS.toSeconds(duration));
    }
    return true;
}

private String parseRoute(String route){
    String toReturn = "";
    String[] steps = route.split(";");
    for(String step:steps){
        if(step.split(":")[0].equals("view")){
            toReturn+="View room: "
```

```
                    +b.findRoomById(step.split(":")[1]).getRoomName()+"\n";
            } else {
                toReturn+="Go to room: "
                    +b.findRoomById(step.split(":")[1].split(",")[1])
                    .getRoomName()+"\n";
            }
        }
        return toReturn;
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        if(resultCode == RESULT_OK){
            btAvailable = true;
        } else
            btAvailable = false;
    }
}
```

C.77 android/MuseumGuide/src/java/activities/MapsActivity.java

```java
import android.graphics.Color;
import android.support.v4.app.FragmentActivity;
import android.os.Bundle;
import com.example.sari.museumguide.R;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.Marker;
import com.google.android.gms.maps.model.MarkerOptions;
import com.google.android.gms.maps.model.PolylineOptions;

public class MapsActivity extends FragmentActivity implements OnMapReadyCallback {

    private GoogleMap mMap;

    private Marker locationMarker;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_maps);
        // Obtain the SupportMapFragment and get notified when the map is ready to be
            used.
        SupportMapFragment mapFragment = (SupportMapFragment)
            getSupportFragmentManager()
               .findFragmentById(R.id.map);
        mapFragment.getMapAsync(this);
    }



    /**
```

```
     * Manipulates the map once available.
     * This callback is triggered when the map is ready to be used.
     * This is where we can add markers or lines, add listeners or move the camera. In
         this case,
     * we just add a marker near Sydney, Australia.
     * If Google Play services is not installed on the device, the user will be
         prompted to install
     * it inside the SupportMapFragment. This method will only be triggered once the
         user has
     * installed Google Play services and returned to the app.
     */
    @Override
    public void onMapReady(GoogleMap googleMap) {
        mMap = googleMap;
        LatLng scienceMuseum = new LatLng(51.4972,-0.1767);
        mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(scienceMuseum, 18));
        locationMarker = mMap.addMarker(new MarkerOptions().position(scienceMuseum)
                .title("You are here!"));

        PolylineOptions options = new
            PolylineOptions().color(Color.BLUE).width(5).visible(true);
        options.add(scienceMuseum);
        options.add(new LatLng(51.4972531,-0.175478));
        mMap.addPolyline(options);

    }

    public void updateLocationMarker(double latitude, double longitude){
        locationMarker.setPosition(new LatLng(latitude,longitude));
    }

    public void updateLocationMarker(LatLng latLng){
        locationMarker.setPosition(latLng);
    }
}
```

C.78 android/MuseumGuide/src/java/activities/BuildingActivity.java

```java
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.util.Log;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.TextView;
import com.example.sari.museumguide.R;
import com.example.sari.museumguide.database.Database;
import com.example.sari.museumguide.models.indoormapping.Building;
import com.example.sari.museumguide.models.indoormapping.Room;
import com.example.sari.museumguide.tools.CustomAdapter;
import java.util.ArrayList;


public class BuildingActivity extends AppCompatActivity {
    private ListView museumListView;
    private TextView estTimeView;
    private EditText deadlineText;
    private Building b;
    private String building_id;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_building);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);
        b =(Building) getIntent().getSerializableExtra("building");
        building_id = b.getId();
        museumListView = (ListView)findViewById(R.id.room_list_building);
        deadlineText = (EditText) findViewById(R.id.deadline_edittext);
    museumListView.setChoiceMode(ListView.CHOICE_MODE_MULTIPLE);
    estTimeView = (TextView)findViewById(R.id.estimated_time_textview);
    getSupportActionBar().setDisplayHomeAsUpEnabled(true);
}

@Override
protected void onResume() {
    super.onResume();
    final ArrayList roomNames = new ArrayList<String>();
    b = Database.getBuilding(building_id);
    for(Room r: b.getRooms()){
        roomNames.add(r.getRoomName());
    }
    ArrayAdapter museumListAdapter = new CustomAdapter(this, roomNames);
    museumListView.setAdapter(museumListAdapter);
    updateEstTime();
}

public void updateEstTime(){
    estTimeView.setText("Estimated time: " +(int) getEstTime());
}

public double getEstTime(){
    double toReturn = 0;
    for(int i = 0;i<b.getRooms().length;++i){
        if(((CustomAdapter) museumListView.getAdapter()).isChecked(i)){
            toReturn+=b.getRooms()[i].getEst_time();
        }
    }
    return toReturn;
}

public void startGuideActivity(View v){
    String selectedRooms = "";
    for(int i = 0;i<b.getRooms().length;++i){
        if(((CustomAdapter) museumListView.getAdapter()).isChecked(i)){
                selectedRooms+=b.getRooms()[i].getId()+":"
                        +((CustomAdapter) museumListView.getAdapter()).getProgress(i)
                        +",";
            }
        }
        Log.d("SELECTED ROOMS: ",selectedRooms);
        if(!selectedRooms.isEmpty() && !deadlineText.getText().toString().isEmpty()){
            Intent intent = new Intent(this, GuideActivity.class);
            intent.putExtra("building",b);
            intent.putExtra("selected_rooms",selectedRooms
                                .substring(0,selectedRooms.length()-1));
            intent.putExtra("deadline",Integer.parseInt(deadlineText.getText().toString()));
            startActivity(intent);
        }
    }

}
```