**High level architecture diagram**

**A picture containing text, whiteboard

Description automatically generated**

Appendix

(Sensor Date Service Interface)

public interface SensorDataService extends Remote {

String getSensorData() throws RemoteException;

Boolean addSensor(int roomNo, int floor, String token) throws RemoteException;

}

(Login Service Interface)

public interface LoginService extends Remote{

String loginAsAdmin(String username, String password) throws RemoteException;

}

(Alert Service Interface)

public interface AlertService extends Remote {

void sendEmailAlert(String sensorId) throws RemoteException;

void sendSMSAlert(String sensorId) throws RemoteException;

}

(Sensor Data Service Implementation)

public class SensorDataServiceImpl implements SensorDataService {

@Override

public String getSensorData() {

URL url;

HttpURLConnection conn = null;

String response = "";

try {

url = new URL("http://localhost:3000/sensor");

conn = (HttpURLConnection) url.openConnection();

conn.setRequestMethod("GET");

conn.setRequestProperty("Accept", "application/json");

if(conn.getResponseCode() != 200) {

throw new RuntimeException("Failed: HTTP error code : " + conn.getResponseCode());

}

try (Scanner myScanner = new Scanner(url.openStream())) {

while(myScanner.hasNextLine()) {

response += myScanner.nextLine();

}

}

} catch (Exception e) {

e.printStackTrace();

}

return response;

}

@Override

public Boolean addSensor(int roomNo, int floor, String token) {

Boolean isSuccess = false;

String jsonToken = token; // TODO: json token should be applied here

URL url;

HttpURLConnection conn = null;

try {

url = new URL("http://localhost:3000/sensor");

conn = (HttpURLConnection) url.openConnection();

conn.setRequestMethod("POST");

conn.setRequestProperty("Content-Type", "application/json");

conn.setRequestProperty("Accept", "application/json");

conn.setRequestProperty("Authorization", jsonToken);

conn.setDoOutput(true);

JSONObject locationObj = new JSONObject();

locationObj.put("roomNo", roomNo);

locationObj.put("floor", floor);

JSONObject reqBody = new JSONObject();

reqBody.put("co2Level", 0);

reqBody.put("smokeLevel", 0);

reqBody.put("location", locationObj);

reqBody.put("isActive", true);

String jsonInputString = reqBody.toString();

try(OutputStream os = conn.getOutputStream()) {

byte[] input = jsonInputString.getBytes("utf-8");

os.write(input, 0, input.length);

}

try(BufferedReader br = new BufferedReader(

new InputStreamReader(conn.getInputStream(), "utf-8"))) {

StringBuilder response = new StringBuilder();

String responseLine = null;

while ((responseLine = br.readLine()) != null) {

response.append(responseLine.trim());

}

}

if(conn.getResponseCode() == 201) {

isSuccess = true;

}

} catch (Exception e) {

e.printStackTrace();

}

return isSuccess;

}

}

(Login Service Implementation)

public class LoginServiceImpl implements LoginService {

@Override

public String loginAsAdmin(String username, String password) {

URL url;

HttpURLConnection conn = null;

String result = "";

try {

url = new URL("http://localhost:3000/user/login");

conn = (HttpURLConnection) url.openConnection();

conn.setRequestMethod("POST");

conn.setRequestProperty("Content-Type", "application/json");

conn.setRequestProperty("Accept", "application/json");

conn.setDoOutput(true);;

String jsonInputString = "{ \"email\": \"" + username + "\", \"password\": \"" + password + "\" }"; // TODO: build request json body here

try(OutputStream os = conn.getOutputStream()) {

byte[] input = jsonInputString.getBytes("utf-8");

os.write(input, 0, input.length);

}

try(BufferedReader br = new BufferedReader(new InputStreamReader(conn.getInputStream(), "utf-8"))) {

StringBuilder response = new StringBuilder();

String responseLine = null;

while((responseLine = br.readLine()) != null) {

response.append(responseLine.trim());

}

result = response.toString();

}

} catch (Exception e) {

e.printStackTrace();

}

return result;

}

}

(Alert Service Implementation)

public class AlertServiceImpl implements AlertService {

@Override

public void sendEmailAlert(String sensorId) {

System.out.println("Sending email alert on sensor " + sensorId);

}

@Override

public void sendSMSAlert(String sensorId) {

System.out.println("Sending SMS alert on sensor " + sensorId + "\n");

}

}

(RMI Server)

public class Server {

public Server () {}

public static void main(String args[]) {

try {

LoginServiceImpl objOne = new LoginServiceImpl();

SensorDataServiceImpl objTwo = new SensorDataServiceImpl();

AlertServiceImpl objThree = new AlertServiceImpl();

LoginService stubOne = (LoginService) UnicastRemoteObject.exportObject(objOne, 0);

SensorDataService stubTwo = (SensorDataService) UnicastRemoteObject.exportObject(objTwo, 0);

AlertService stubThree = (AlertService) UnicastRemoteObject.exportObject(objThree, 0);

Registry registry = LocateRegistry.createRegistry(1099);

registry.bind("LoginService", stubOne);

registry.bind("SensorDataService", stubTwo);

registry.bind("AlertService", stubThree);

System.err.println("Server ready");

} catch (Exception e) {

e.printStackTrace();

}

}

}

(Get Data for Client Dashboard RMI)

private void getSensorData() {

try {

Registry registry = LocateRegistry.getRegistry(1099);

SensorDataService stub = (SensorDataService) registry.lookup("SensorDataService");

AlertService stub2 = (AlertService) registry.lookup("AlertService");

String response = stub.getSensorData();

response = "{ results: " + response + " }";

JSONObject responseObj = new JSONObject(response);

JSONArray responseArr = responseObj.getJSONArray("results");

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

JSONObject obj = responseArr.getJSONObject(i);

boolean isActive = obj.getBoolean("isActive");

int roomNo = obj.getJSONObject("location").getInt("roomNo");

int floor = obj.getJSONObject("location").getInt("floor");

int smokeLevel = obj.getInt("smokeLevel");

int co2Level = obj.getInt("co2Level");

String sensor = "Sensor " + (i + 1);

JTextArea sensorCard = new JTextArea(sensor);

sensorCard.append("\nActive: " + isActive);

sensorCard.append("\nRoom No: " + roomNo);

sensorCard.append("\nFloor: " + floor);

sensorCard.append("\nSmoke Level: " + smokeLevel);

sensorCard.append("\nCO2 Level: " + co2Level);

if(!isActive) {

sensorCard.setBackground(Color.DARK\_GRAY);

sensorCard.setForeground(Color.WHITE);

} else if(smokeLevel > 5 || co2Level > 5) {

sensorCard.setBackground(Color.red);

sensorCard.setForeground(Color.WHITE);

stub2.sendEmailAlert(obj.getString("\_id"));

stub2.sendSMSAlert(obj.getString("\_id"));

JOptionPane.showMessageDialog(null, "Fire Alert! " + sensor);

} else {

sensorCard.setBackground(Color.green);

}

jPanel2.add(sensorCard);

}

} catch (Exception e) {

e.printStackTrace();

}

}

(Timer in Dashboard)

Timer timer = new Timer();

timer.scheduleAtFixedRate(new TimerTask() {

@Override

public void run() {

System.out.println("Getting sensor data after 30 seconds...");

jPanel2.removeAll();

getSensorData();

jPanel2.revalidate();

jPanel2.repaint();

}

}, 30 \* 1000, 30 \* 1000);

(Admin Login Function)

private void tryLogin(String username, String password) {

try {

Registry registry = LocateRegistry.getRegistry(1099);

LoginService stub = (LoginService) registry.lookup("LoginService");

String response = stub.loginAsAdmin(username, password);

JSONObject responseObj = new JSONObject(response);

String message = responseObj.getString("message");

if(message.equalsIgnoreCase("Authentication successfull")) {

jwt\_token = responseObj.getString("token");

System.out.println(jwt\_token);

setVisible(false);

AdminPanel adminPanel = new AdminPanel();

adminPanel.setVisible(true);

}

} catch (Exception e) {

e.printStackTrace();

}

}

**There are more to add. But then the report will exceed the 10 pages.**