Project: UNCC_WORK

Author: Zachary Zaleski

Cover Letter of Transmittal

The artificial intelligence for powered augmented reality for workers project will ensure the safety of road workers through the usage of a high speed communication system to send alert messages to multiple client devices inside of a work zone. A vehicle detection model will be leveraged to generate the risk scores sent to the clients. The risk scores will be generated based on a few criteria such as size of the vehicle and distance from the workzone. In order to ensure that these risk scores and alerts are reaching workers at the fastest possible speeds, a multithreaded communication approach was implemented. This would ensure that high priority alerts and messages are being communicated to all people at the work zone simultaneously and quickly.

The team has also developed a mobile application for android devices and a smartwatch application on the Samsung Galaxy Watch Active. These devices, along with the Vuzix Blade smart goggles, are what workers will be interfacing with in the system. The mobile application will act as a digital twin and will track location of the workers inside the workzone in real-time to generate warnings should a worker exit the border of the workzone. The digital twin uses a Google Maps API to display the current location of all workers in a user-defined work zone that will send alerts when workers leave the site. The smart watch device can track the heart rate of workers and also generate warnings when the heart rate reaches unhealthy levels. This data can be stored on the digital twin to track changes in the work environment. Furthermore, the digital twin will be receiving the risk score that is generated by the server, and displaying either the current risk score, or potentially an averaged risk score over a period of time. The use of these metrics combined with the early warning system for vehicles provided by the AI will allow for safer and more efficient work zones.

To optimize the data that is collected the team worked closely with project supporters to create an efficient UI. The design of the UI consists of an alert banner that will display the risk score and other information needed for it, the real-time map with current location and any above average heart rates.

During the final weeks, the team shifted its focus to developing the vehicle detection and re-identification. The team was able to successfully detect vehicles and re-identify them, however, retrieving the size and distance of the vehicle for risk score generation is still in development and should be the starting point for future senior design teams potentially working on this project.

Project: UNCC_WORK

Author: Zachary Zaleski

A directory containing all relevant project documentation has been uploaded to the team's shared Google Drive folder. This folder has been zipped and uploaded to the senior design Canvas page, emailed to ISL management, and emailed to the team's faculty supporters. The zipped folder is titled UNCC_WORK_Comprehensive_Submission and the parent SD2_UNCC_WORK_S21 folder contains subdirectories to all of the files within.

Deliverable	File Name
Progress Report #1	UNCC_WORK_Progress Report1.pdf
Progress Report #2	UNCC_WORK_ProgressReport2.pdf
Prototype Status Review Presentation	UNCC_WORK_PSRSlides.pptx
Performance Specifications	UNCC_WORK_Performance_Specifications_RevA.pdf
Prototype Review Presentation	UNCC_WORK_PRPSlides.pptx
BOM and Budget	UNCC_WORK_BOM.xlsx
Statement of Work	UNCC_WORK_SOW_RevA.pdf
EXPO Poster	UNCC_WORK_Poster.pptx
Final Timesheet	UNCC_WORK_Timesheet6.xlsx
Project Plan	UNCC_WORK_ProjectPlan_RevO.mpp
Latency Testing Results	Folder Name: Latency Tests
System Communication Flow	CommunicationFlow.png
Project Progress Schematic	Progress.jpg
Initial Multithreaded Server Design	Initial_Multi-Threaded_Server_Design.png
Final Multithreaded Server Design	Theorized_Nested_Multi-Threaded_Server_Design.png
Single-Threaded Server Design	Single-Threaded_Latency_Testing_Server_Design.png
Multithreaded System Example	MultithreadedExample
System Overview Schematic	SystemOverhead.png
Multithreaded Server Code	MultithreadedServer.py
Digital Twin Android Application Code	DigitalTwinAndroidApplication.zip
Client Android Application Code	ClientAndroidApplication.zip
EXPO Poster Voiceover	UNCC_WORK_Poster.mp4
Project Video	UNCC_WORK_Video.mp4
Vehicle Tracking Video Demo	CarTracking.mp4
YOLO.v4 Custom Model Output	CarDetection.mp4
Goggle Video Output	VID_20210426_154605.mp4
Server Output Example	ZachWalkingWithGoggle.mp4
Client Application Risk Score Interface	ClientRiskScore.mp4
One client updating with RT-GPS	JustZachThenGoggleDT.mp4
Digital Twin GPS with all Clients Video	AllClientsDT.mp4
AI Model (YOLO.v4 and DeepSORT) Source Code	DeepSortYOLOv4.zip
Final Design Package Report	UNCC_WORK_Final_Report_S21.pdf

Project: UNCC_WORK

Author: Zachary Zaleski

Division of Duties Summary Table

	Team Member #1 Damian Hupka	Team Member #2 Zachary Zaleski	Team Member #3 Duncan Tennant	Team Member #4 William Clampett	Team Member # 5 Nathan Pecoraro	Total (should = 100%)
Digital Twin Android Application UI Design	10%	10%	35%	35%	10%	100%
Digital Twin Android Application Logic/Communication Design	20%	20%	20%	20%	20%	100%
Android Client Application UI Design	10%	10%	35%	35%	10%	100%
Android Client Application Logic/Communication Design	20%	20%	20%	20%	20%	100%
AI Detection Implementation	30%	30%	5%	5%	30%	100%
AI Tracking Implementation	30%	30%	5%	5%	30%	100%

UNCC_WORK Project – Final Project Report – Senior Design II

Date	Revision	Author	Comments
2021-05-04	1	Zachary Zaleski	

Table of Contents

1 Overview of this Document

This document describes the design of the UNCC_WORK project and of its end-product for Senior Design. This project is to develop a scalable backend communication system which will send warning messages to multiple clients in a work zone based on data received from a camera that will detect, track, and re-identify vehicles. The expected clients will be the Vuzix Blade goggles, an application developed on the Samsung Galaxy Watch Active, and a mobile Android application. These devices detect the heart rate of the worker as well as track their location in the work zone. The Android application will be deployed onto an Android OS tablet; wherein, it will act as a digital twin which aggregates the data from the clients and the server and displays such information in a meaningful, summarized manner.

This document will describe all of the work completed in Senior Design II as well as discussing the team's recommendations to improve and continue the project. The impact of the completed work and future development on society is also evaluated to determine the efficacy of the product.

Zachary Zaleski, who is identified as the project lead, will be responsible for any and all statements made on this report

2 Project Overview / Statement of Work Summary

The goal of the project is to develop a scalable backend communication system that will alert road workers of potential threats based on a risk score that will be generated from an AI model that uses object detection to identify/re-identify vehicles based on size and determine their trajectory. The risk scores generated by the server will be communicated to the Vuzix Blade goggles and displayed to clients in real-time. A mobile application that is also developed by the team will act as a digital twin to display GPS location of workers, risk score history (as generated from the server), and heart-rate history (as generated from the smart watch). Figure 1 below shows the overview of the system in a mocked work zone environment with each of the clients in view.

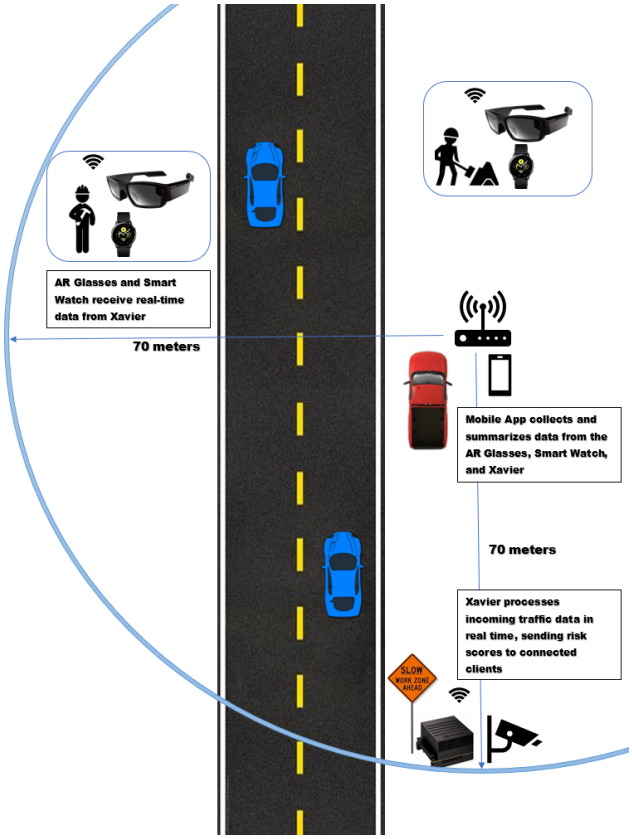

Figure 1: System Overhead

Real-time GPS location services will be available for each client in the workzone using the goggles or a smart watch and displayed on the digital twin application using a Google Maps API. The mobile application will allow for variable boundaries to be set based on the size of the work zone and warnings will be displayed when workers move near or beyonds those boundaries. Certain data such as risk score history and GPS information will be aggregated from the mobile application and stored in a local database for managerial observation and usage for future versions of the system.

3 Design Narrative

The project consisted of three major components: the server, clients, and digital twin application. Regarding the server, the team placed an emphasis on the ability to concurrently communicate with multiple connected devices. The reason for this being the team wanted to get the generated risk scores out to each client in a quick and efficient manner. Aside from this, the server was structured to handle each client connection in a separate thread. Each client thread would in turn start its own send and receive threads. Starting these threads is important because it allows for individual processing of each of the connections, which enables the server to forego certain processes that are taking too much time. Figure 4 below shows the general flow of the server operation.

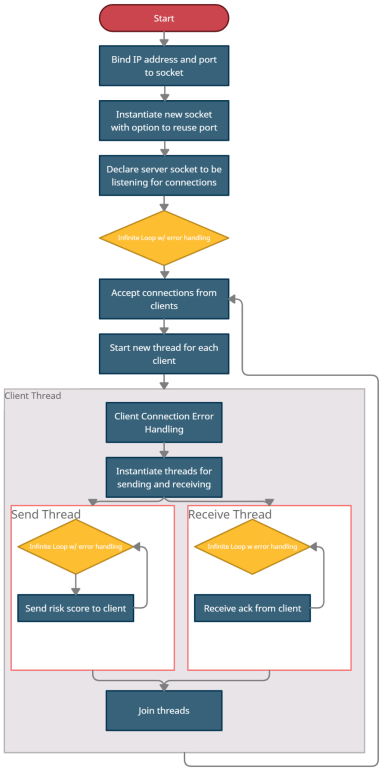

Figure 4: Final Multithreaded Server Flow

The server was developed using Python, this decision was made rather early on in the development process of the project. This language was chosen for the server design based on the rationale that further into implementation an AI model will be leveraged, and implementing such an AI model would prove to be most accessible through Python. This was proven when the team shifted focus into developing the AI model to be the key driving factor in risk score generation. However, from a communication standpoint, utilizing socket programming with Python was a more challenging task as compared to implementation using a language built for communication protocols such as GO lang as Python does this implementation from an OS level.

The clients are structured to expect a risk score from the server, and will return specific information (such as GPS location or heart-rate) as an acknowledgement that the risk score was received. The team discovered that sending this acknowledgement was important, as the method used to send the risk-score through a socket simply passed the information to a buffer. As a result, the server had no way of knowing if the messages were being sent correctly. The team believes that sending the GPS or heart-rate as an acknowledgement to be the most concise and efficient way to pass information from the clients to the server. Figure 6 below shows the communication flow between the devices in the system and the various messages that are being transmitted.

Figure 6: Communication Flow

The digital twin application should be considered a conglomerate of all the information the system has to offer. The information that was gathered on the digital twin application consisted of three primary categories: real-time client GPS location, real-time heart rate of clients, and the current or averaged risk score which clients were receiving. Having access to information such as when risk scores were high, the stress levels of workers, and environmental information such as location of clients would be useful to a workzone supervisor when making decisions about workplace safety or operation. The digital twin application was made in Android Studios as the integration of the APIs required for the application was more seamless to our design as competency in front-end application development was at a primitive level for a team without an extensive background in front-end development.

4 Test Results

The team initially conducted latency tests with the implementation of the initial single-threaded server scheme to gain an understanding of the baseline latency that was measured within the system. The goal of the performance specifications was to reach a latency of approximately less than 10ms. Overall, the system performed in this regard; however, once reaching greater distances (nearing 70 meters) the average latency of the system approached the realm of 30-70ms**.** With these measurements of the latency, the team pivoted to a multithreaded server communication approach to deliver messages to clients. Currently, gaining an accurate measurement of what these latencies are is not feasible, but it can be concluded that they will fall in the range similar to what was measured with the single-threaded approach, or perhaps even lower.

Lastly, the team believed that implementing an AI model within the system was not going to be practical within the amount of time that was available. However, with assistance from UNCC faculty and graduate mentors, the team was able to implement an AI model which completed two goals: detect a subset of vehicles (car, truck, bus, etc.) in real time, and similarly track/re-identify these vehicles as they enter and exit the frame. The goal of this AI model was to reach a FPS inference time of greater than 10FPS while executing real-time, live video inference on the NVIDIA Xavier AGX. This performance specification was met when only utilizing the YOLO.v4 [1] custom trained detection model; however, when adding another model layer (DeepSORT) to complete the vehicle tracking and re-identification, the performance dropped to approximately 7.5FPS.

5 Evaluation of Prototype/ Model/ System as Compared to Project Performance Specification Document

The project mentor stated two performance specifications as a goal for the project. The first being that the system should run at no less than 10FPS. This endeavor was only partially successful. As stated in the previous section, these specifications were only met when only utilizing the YOLO.v4 [1] custom trained model. Adding a layer with DeepSort [2] for tracking and re-identification caused the framerate to drop below 10FPS.

The second specification was to keep the communication latency below 10ms. During initial testing this seemed to be a plausible goal for the system, however, such a small latency was unable to be reached when testing in larger ranges. It was deemed that such a low latency was unable to be reached should the system be implemented in a real work zone. Reasons for this were that communication takes longer at larger distances as well as the inference on the model for vehicle detection and tracking would add extra latency time to the system as a whole. The vehicle re-identification and tracking has not been tested with the system, only individually.

Initially the project requirements were subject to change, so the team decided to take on the task of developing a mobile application on an Android device which could track the GPS location of workers in the work zone in real time as well as track the heart rate of said workers. The final

prototype has a work zone boundary specification which is declared constant for prototypical testing (this will be user-defined in later iterations of the application) on a localized map that will send alerts when workers leave that area. These devices are all successfully able to communicate concurrently with the server and receive risk score data, outputting that information to the users.

6 Recommendations for Further Development

Going forward, more time should be devoted to the artificial intelligence side of the project. The team has begun to implement some basic detection and re-identification algorithms, with the hope of performing a basic trajectory analysis and generating a risk score based on the output of the model. Trajectory analysis will not be able to be completed, but providing detection, re-identification, and basic distance analysis will pave the way for future development teams to continue where UNCC_WORK ended.

The digital twin application UI and user experience should be improved. As of now, most of the backend functionality is working as intended, and it serves as a good prototype. The overall look of a final product should be more refined, which will improve the general feel of the application, and will demonstrate enhanced user accessibility. In order to improve the risk analysis, the use of a weather API that could track changing weather conditions and incoming storms would help to provide a more comprehensive and realistic risk score for the setting. The team believes that implementing a risk alert banner on the homepage along with the real-time heart rate, map and weather will provide a functional and sleek homepage UI for the digital twin. Outside the homepage the team thought that having graphs and charts from the stored and tracked data available on a more detailed view would allow managerial staff to track workplace trends so they can make adjustments to any issues that arise from the data. Implementing these changes should allow the digital twin to act as a useful data aggregation tool that also helps the AI processes determine accurate and thorough risk scores.

7 Impact

The implementation of the UNCC_WORK safety system would impact the well-being of laborers on highway and roadside work environments. The alert system provides workers with more sense of security as there is a rapid response to inform them of any danger. Drivers and other pedestrians would also benefit as less injury to employees and workers would mean less overall damage for the responsible parties to be liable for. The client application that is implemented on the Vuzix Blade goggles and the smart-watch allow for two-fold risk transmittal through visual and haptic feedback. However, the development of the digital twin application can allow for work zone management to better analyze the safety of workers by understanding the potentially high risk situations that clients may be exposing themselves to dangerous situations.

The impact of UNCC_WORK on a global scale will be other states besides NC adopting the safety warning system because it will save lives and add productivity to the construction community. The AI involvement in culture increases due to people thinking it is safer for everyday life. The success of this project could help to demonstrate the useful applications of AI research and development. Furthermore, through ethical data gathering and processing of data

locally it can be ensured that the AI model is operating in a data conscious manner.

The societal impact of this project includes benefits to the lives of the construction workers and their families. A safer work environment will help to reduce worker stress and fatigue which could improve life at home and in the workplace. An increase in the use of this technology could impact traffic norms and behavior in a positive way which would benefit society as well. This could also impact the environment as a reduction in accidents would mean less resources are being spent on emergency services.

The environmental footprint of this project is minimal, thus it will not be discussed in detail.

The economic impact can be a growth in the market due to an increase in demand on AI products such as the Jetson board used in this project. Construction companies and contractors could use this product as a marketing tool when making offers for projects as a way to help reduce cost or have shorter timelines. The safety benefits would cause workers to be more efficient due to feeling secure which increases production and could impact costs. Employers could use the increase in safety as a hiring tool as well for any prospective employees.

8 Bill of Materials (BOM)

Qty	Name	Description	Cost	Website
1	NVIDIA Jetson AGX Xavier Developer Kit	AI Enabled Embedded Device	$ 699.00	https://www.amazon.com/NVIDIA-Jetson-Xavier-Developer-32GB/dp/B083ZL3X5B/
1	Vuzix Blade	Augmented Reality Goggles	$ 899.99	https://www.vuzix.com/products/blade-smart-glasses-upgraded
1	Samsung Galaxy Watch Active	Smartwatch	$ 199.99	https://www.samsung.com/us/mobile/wearables/smartwatches/galaxy-watch-active-40mm-black-sm-r500nzkaxar/
1	TP-Link AX1500 Next-Gen Wi-Fi 6 Router	Cutting Edge Access Point	$ 79.99	https://www.amazon.com/TP-Link-Wireless-AX1500-Wifi-Router/dp/B07ZSDR49S
1	Fire HD 10 Tablet	Amazon Android Tablet Device	$ 94.99	https://www.amazon.com/Fire-HD-10/dp/B07K1RZWMC
1	Logitech C922 Webcam	USB Connected	$ 99.99	https://www.logitech.com/en-us/products/webcams/c922-pro-stream-webcam.960-001087.html0.3
6			$ 2,073.95	

Figure 31: Bill of Materials

9 Budget

The budget plan for completing the project is included as in the Bill of Materials as above. However, this Bill of materials consists of a one-scale system for an individual person on the worksite. For expanding this into a workzone with more workers the system will include one (1) pair of the Vuzix blade and one (1) Samsung Galaxy Watch Active with each new worker. Furthermore, additional Amazon Fire HD 10 Tablets could be purchased and utilized within the system as this device is acting as the digital twin which is congregating information within the system and as such, many managerial role workers may wish to inspect the various metrics that are contained within the system.

No additional non-freeware software was necessary in completing this project, also no third-party labor is required for completion. However, AI models may need to be licensed for use in a production sense.

10 Conclusions

In conclusion, the team was able to complete this project nearly at the scope which was declared within the statement of work in the Fall 2020 semester. Throughout Senior Design I, it was

expected that the team would not be able to reach the implementation of the AI model within the system and was going to strictly develop the necessary backend communication framework for the risk score generation by the AI model to take place at a later date as well as developing the digital twin application to aggregate various system metrics. However, with completion of the backend communication scheme, implementation of the AI model's occurred rather seamlessly. Although the results of the model have yet to be implemented within the server to be completing the risk score generation that was planned, this is only a matter of simple logic within the server. Furthermore, if progress is to continue in this project, gaining an even more accurate risk score generation for improved worker safety through way of trajectory analysis is similarly only a few steps away.

The development of the backend communication scheme through a multithreaded server was completed over the course of the year and the team gained significant insights into all of the various facets, challenges, and implementation specifics associated with using a socket based server in Python.

The initial project outline did not require a mobile application but implementation of one has allowed the system to usefully display all the data that is being transferred and used. The smart watch heart rate can be easily displayed and used for risk score generation on the homepage of the app while also using the Google Maps API to show a current map of the worksite. Also the current design takes the real time GPS location and displays it on a user defined work area on the map interface. In this interface, the application will also give alerts when workers leave the defined area which can be used in the risk score as well. While there are still improvements that could be made to the digital twin, the current implementation lays the framework for more complex operations and still displays the data that is being used in our system.

11 References

[1] Tianxiaomo. "Tianxiaomo/Pytorch-YOLOv4." pytorch-YOLOV4. Accessed May 7, 2021. https://github.com/Tianxiaomo/pytorch-YOLOv4.

[2] Nwojke. "Nwojke/deep_sort." DeepSORT. Accessed May 7, 2021. https://github.com/nwojke/deep_sort.

12 Appendices

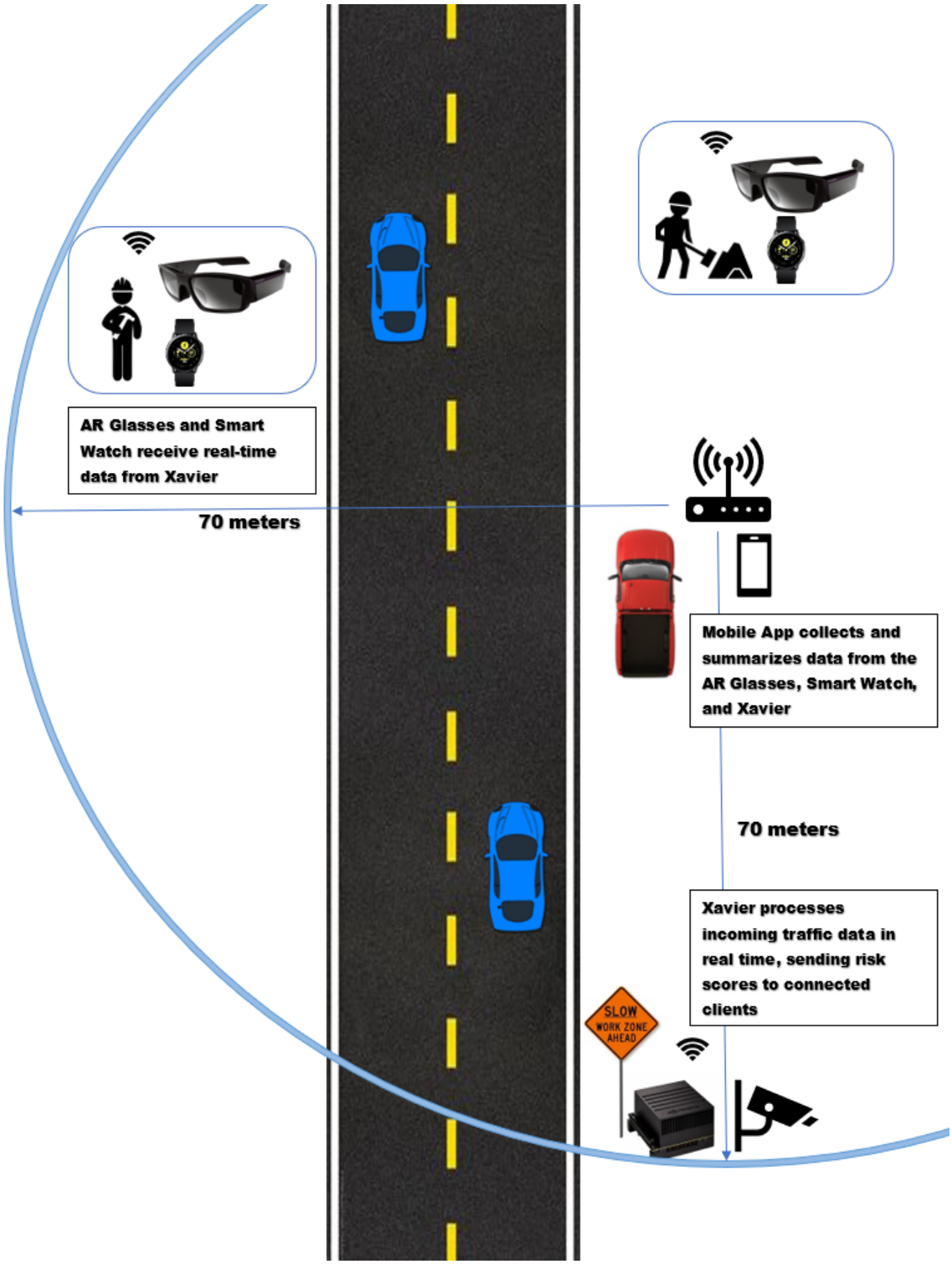

Figure 1: System Overhead

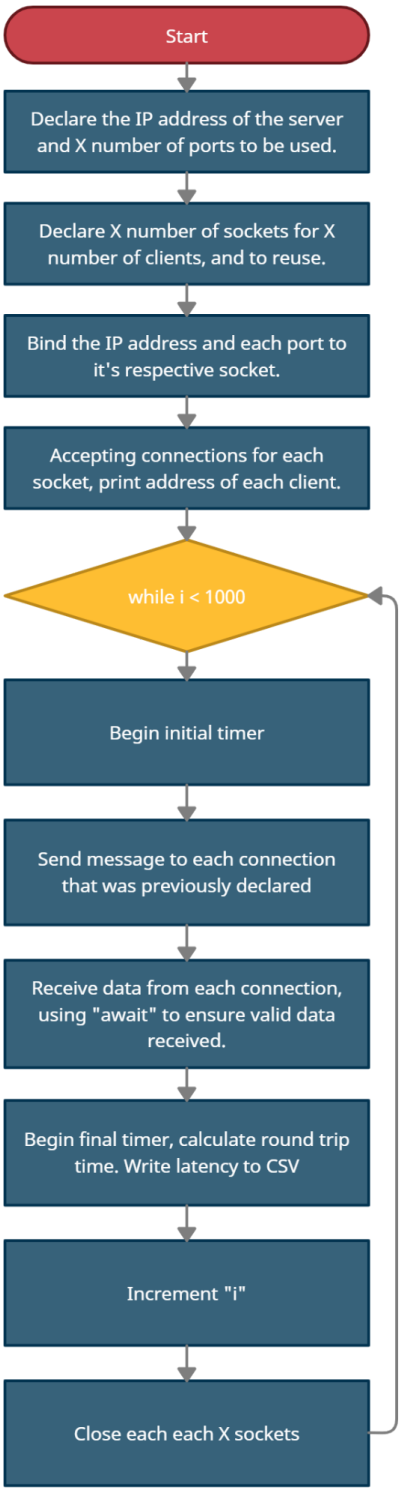

Figure 2: Single-Threaded Server Flow

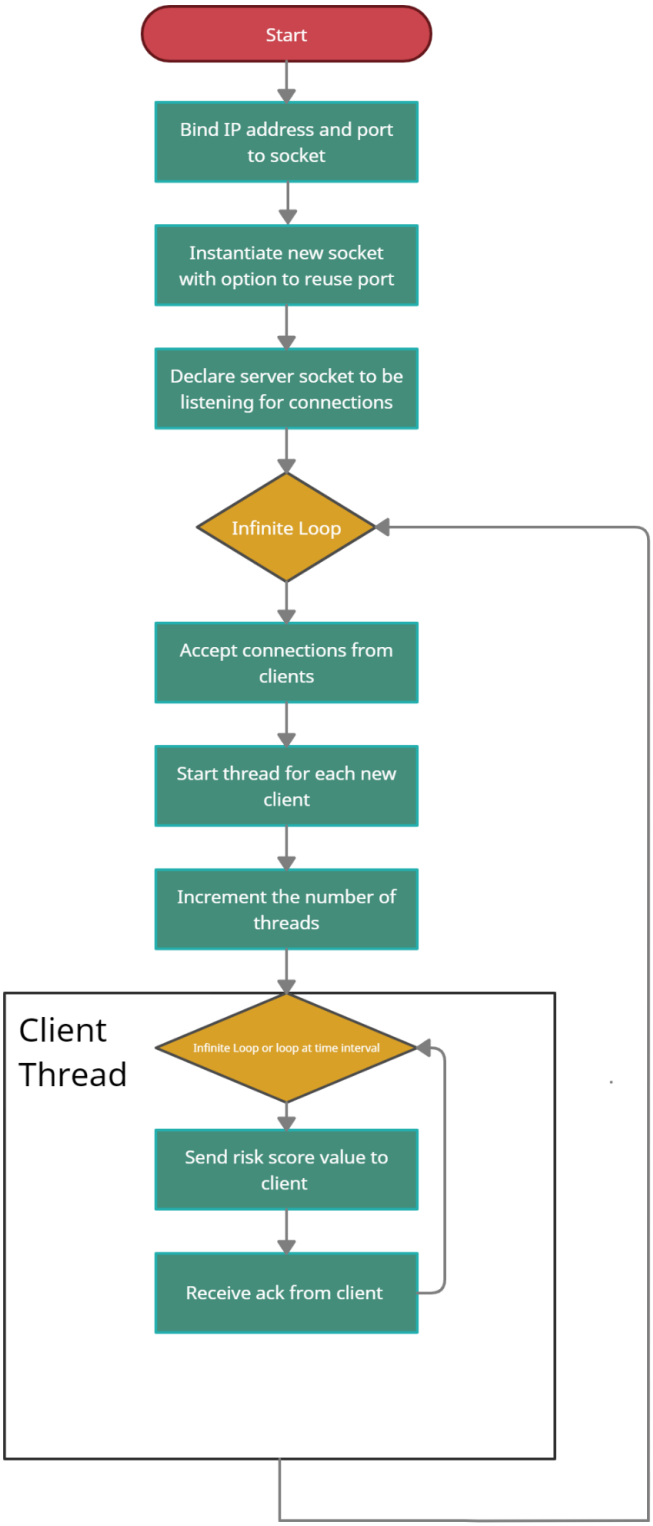

Figure 3: Initial Multithreaded Server Flow

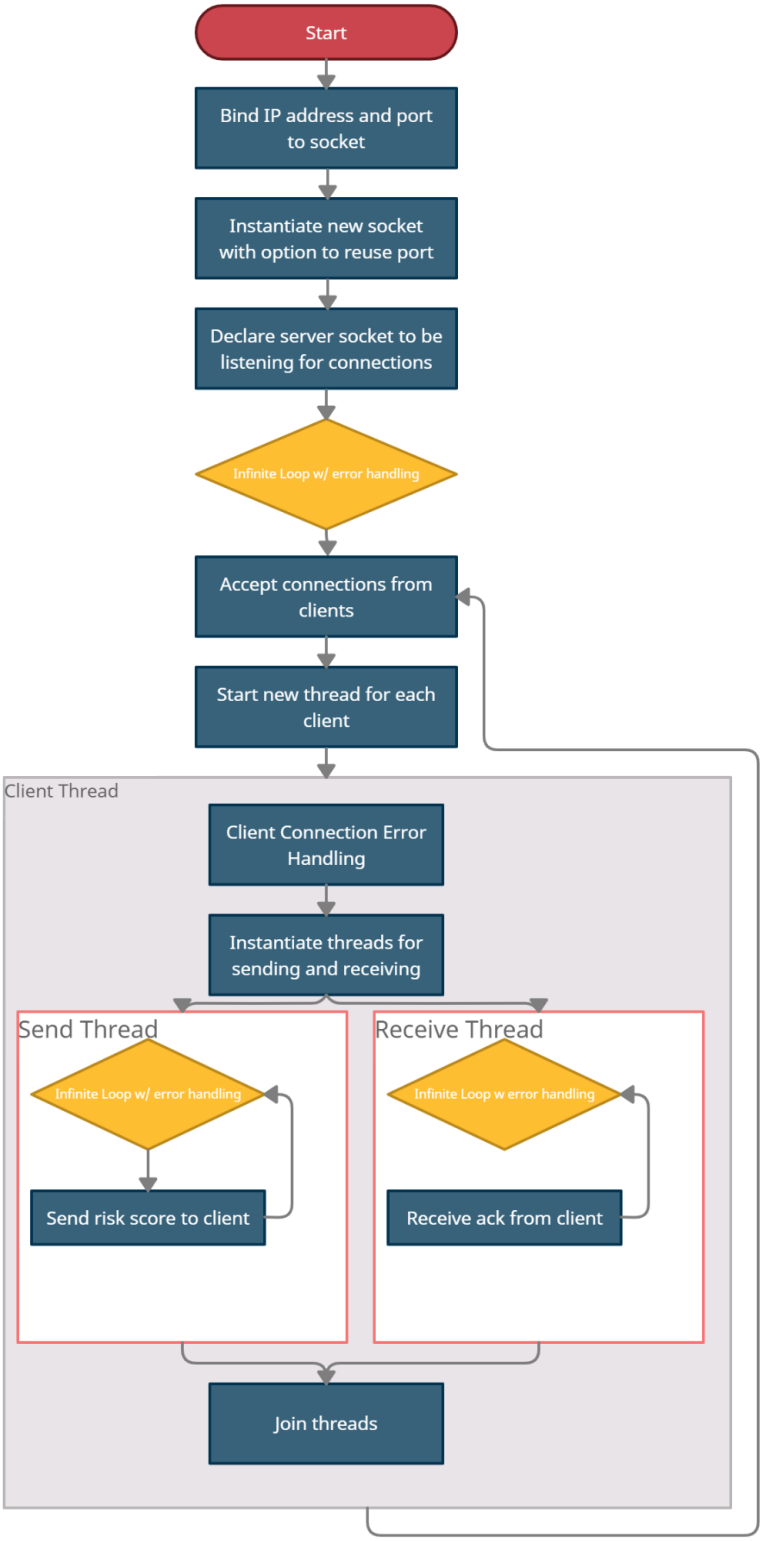

Figure 4: Final Multithreaded Server Flow

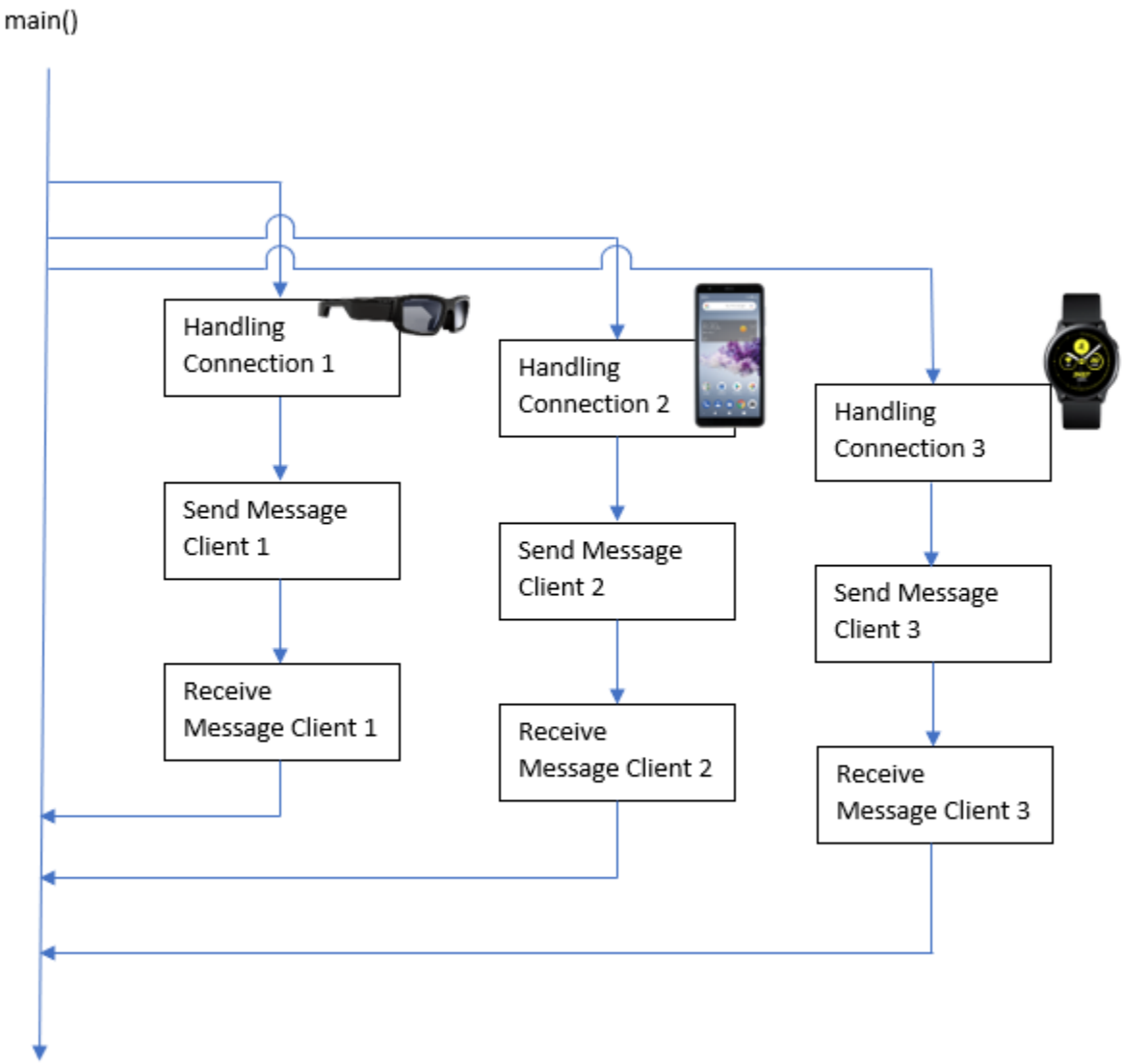

Figure 5: Multithreaded Server Example

Figure 6: Communication Flow

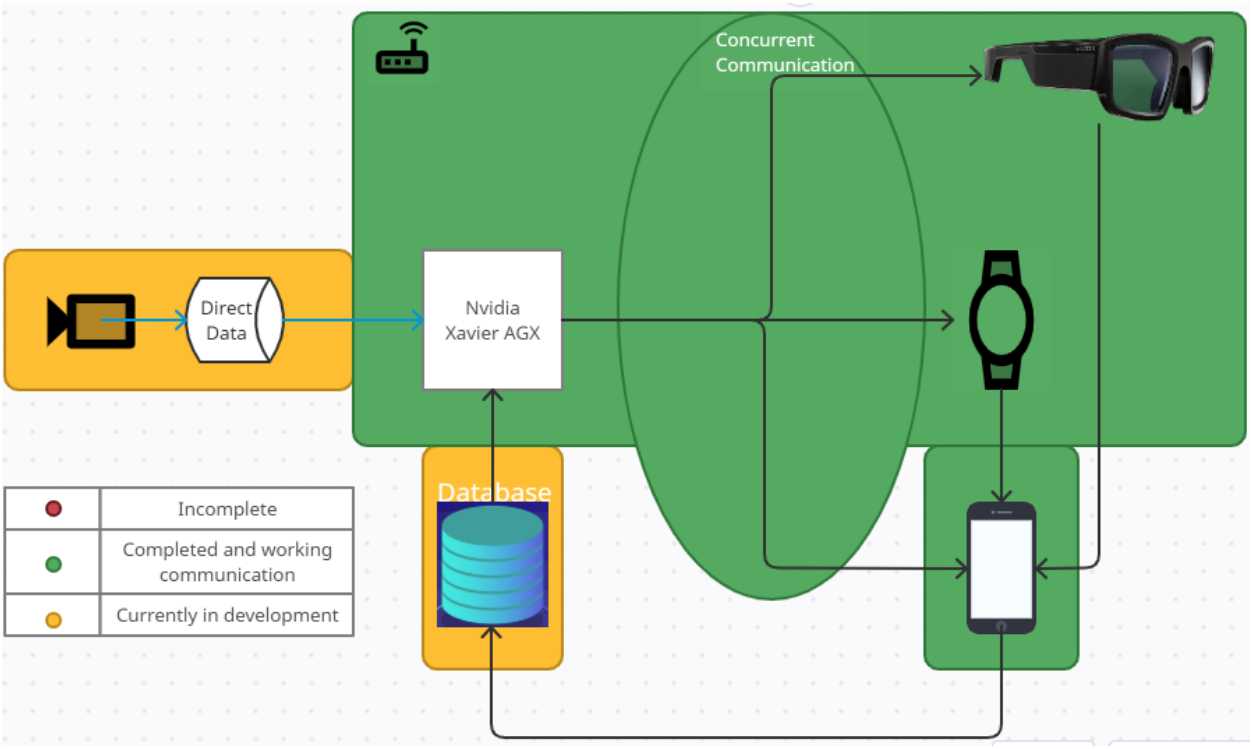

Figure 7: Project Progress

```java
package com.example.client;

import androidx.annotation.LongDef;
import androidx.annotation.MainThread;
import androidx.annotation.NonNull;
import androidx.annotation.RequiresApi;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;

import android.Manifest;
import android.content.Context;
import android.content.pm.PackageManager;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.location.LocationProvider;
import android.os.AsyncTask;
import android.os.Build;
import android.os.Bundle;
import android.os.Handler;
import android.os.StrictMode;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.InetAddress;
import java.net.Socket;
import java.net.UnknownHostException;
import java.security.Provider;
import java.util.Random;
import java.util.concurrent.ExecutionException;
import java.util.concurrent.TimeUnit;

public class MainActivity extends AppCompatActivity implements LocationListener {
    TextView receivedText;
    Button  network;
    public static String ip;
    public static int port;
    public static String messageToBeSent;
    public static String messageReceived;
    public Socket socket = null;
    public static final byte[] buffer = new byte[1024];
    public LocationManager locationManager;
    public Context context;
    public Client client;
    public String latitude;
    public String longitude;

```

Figure 8: Client Application Main Activity Source Code

```
56  @Override
57  protected void onCreate(Bundle savedInstanceState) {
58      super.onCreate(savedInstanceState);
59      setContentView(R.layout.activity_main);
60      receivedText = findViewById(R.id.receivedText);
61      network = findViewById(R.id.findNetwork);
62      final Handler handler = new Handler();
63
64      // We need to either create a new Thread OR use the below line
65      StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder().permitNetwork().build();
66      StrictMode.setThreadPolicy(policy);
67
68      setTitle("Client");
69
70      locationManager = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
71
72      if (ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION) !=
73      PackageManager.PERMISSION_GRANTED && ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_COARSE_LOCATION)
74      != PackageManager.PERMISSION_GRANTED) {
75          // TODO: Consider calling
76          //    ActivityCompat#requestPermissions
77          // here to request the missing permissions, and then overriding
78          //   public void onRequestPermissionsResult(int requestCode, String[] permissions,
79          //                                          int[] grantResults)
80          // to handle the case where the user grants the permission. See the documentation
81          // for ActivityCompat#requestPermissions for more details.
82          return;
83      }
84
85      locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 10,
86              (float) 0.5, this);
87
88
89      context = this;
90      network.setOnClickListener(new View.OnClickListener() {
91          //            @RequiresApi(api = Build.VERSION_CODES.P)
92          @Override
93          public void onClick(View v) {
94
95              try {
96
97                  network.setEnabled(false);
98
99                  port = 8000;
100                 ip = "192.168.0.167";
101                 client = new Client();
102                 client.execute();
103
104             } catch (Exception e) {
105                 e.printStackTrace();
106                 network.setEnabled(true);
107
108                 Toast.makeText(MainActivity.this,"Try again !",Toast.LENGTH_SHORT).show();
109             }
110         }
111
112     });
113
114 }
115
116 @Override
117 public void onLocationChanged(@NonNull Location location) {
118
119     latitude = String.valueOf(location.getLatitude());
120     longitude = String.valueOf(location.getLongitude());
121 }
122
123
124 class Client extends AsyncTask<Void,Void,Void>{
```

Figure 9: Client Application Main Activity Source Code Continued

```
    Socket socket;

    @Override
    protected Void doInBackground(Void... voids) {
        try {
            socket = new Socket(ip, port);
        } catch (IOException e) {
            e.printStackTrace();
            new Thread(new Runnable() {
                @Override
                public void run() {
                    Toast.makeText(MainActivity.this, "Task failed in creating a socket", Toast.LENGTH_SHORT).show();
                }
            });
        }
        return null;
    }

    @Override
    protected void onPreExecute() {
        super.onPreExecute();
        Toast.makeText(MainActivity.this, "Please wait", Toast.LENGTH_SHORT).show();
        network.setEnabled(false);
    }

    @Override
    protected void onPostExecute(Void aVoid) {
        super.onPostExecute(aVoid);
        Toast.makeText(MainActivity.this,"Socket created !",Toast.LENGTH_SHORT).show();

        ReceiveData receiveData = new ReceiveData();
        receiveData.execute();
    }

    public Socket socketGetter() {
        return this.socket;
    }
}


class SendData extends AsyncTask<Void, Void, Void> {
    Socket socket;

    @RequiresApi(api = Build.VERSION_CODES.P)
    @Override
    protected Void doInBackground(Void... voids) {

        socket = client.socketGetter();
        DataOutputStream dataOutputStream = null;
        messageToBeSent = null;
        try {
            dataOutputStream = new DataOutputStream(socket.getOutputStream());
        } catch (IOException e) {
            e.printStackTrace();
            Toast.makeText(MainActivity.this, "Something went wrong", Toast.LENGTH_SHORT).show();

        }
        try {

            if (ActivityCompat.checkSelfPermission(MainActivity.this, Manifest.permission.ACCESS_FINE_LOCATION) !=
            PackageManager.PERMISSION_GRANTED && ActivityCompat.checkSelfPermission(MainActivity.this, Manifest.permission.ACCESS_COARSE_LOCATION) !=
            PackageManager.PERMISSION_GRANTED) {
                // TODO: Consider calling
                //    ActivityCompat#requestPermissions
                // here to request the missing permissions, and then overriding
                //   public void onRequestPermissionsResult(int requestCode, String[] permissions,
                //                                          int[] grantResults)
                // to handle the case where the user grants the permission. See the documentation
```

Figure 10: Client Application Main Activity Source Code Continued

```
                // for ActivityCompat#requestPermissions for more details.
                return null;
            }

            if(locationManager.isLocationEnabled()){

                Location location = locationManager.getLastKnownLocation(LocationManager.GPS_PROVIDER);

                if(location != null){
                    double randLat = location.getLatitude();
                    double randLongi = location.getLongitude();
                    messageToBeSent = "ID1" + "," + randLat +","+ randLongi;
                    dataOutputStream.writeUTF(messageToBeSent);
                    Log.d("TAG", "MSG SENT Actual: " + messageToBeSent);

                } else {
                    Random rd = new Random();

                    float randLat = (float) (35.3072 + rd.nextFloat() * (0.007));
                    float randLongi = (float) (-80.7373 + rd.nextFloat() * (0.007));
                    messageToBeSent = "ID1" + "," + randLat +","+ randLongi;
                    dataOutputStream.writeUTF(messageToBeSent);
                    Log.d("TAG", "MSG SENT Random: " + messageToBeSent);

                }
                }
                else {

                    Random rd = new Random();
                    double randLat =  (35.3072 + rd.nextDouble()*(0.007));
                     double randLongi =  (-80.7373 ) + rd.nextDouble() * (-0.007);
                    messageToBeSent = "ID1" + "," + randLat +","+ randLongi;
                    dataOutputStream.writeUTF(messageToBeSent);
                    Log.d("TAG", "MSG SENT Random: " + messageToBeSent);

                }

            } catch (IOException e) {
                e.printStackTrace();
                Toast.makeText(MainActivity.this, "Something went wrong", Toast.LENGTH_SHORT).show();;
            }

            return null;
        }

        @Override
        protected void onPostExecute(Void aVoid) {
            super.onPostExecute(aVoid);

            ReceiveData receiveData = new ReceiveData();
            receiveData.execute();
        }
    }




    class ReceiveData extends AsyncTask<Void,Void,Double>{

        Socket socket;

        @Override
        protected void onPostExecute(Double doubles) {
            super.onPostExecute(doubles);

            if(doubles != -1){
```

Figure 11: Client Application Main Activity Source Code Continued

```
263                 receivedText.setText(doubles.toString());
264
265                 if(doubles > 9){
266                     receivedText.setBackgroundColor(getResources().getColor(R.color.high));
267                 } else if (doubles <= 9 & doubles > 7){
268                     receivedText.setBackgroundColor(getResources().getColor(R.color.medium));
269
270                 }else if (doubles <=7 ){
271                     receivedText.setBackgroundColor(getResources().getColor(R.color.low));
272                 }
273
274             }
275
276             SendData sendData = new SendData();
277             sendData.execute();
278
279         }
280
281         @Override
282         protected void onPreExecute() {
283             super.onPreExecute();
284         }
285
286         @Override
287         protected Double doInBackground(Void... voids) {
288             socket = client.socketGetter();
289             double riskScore = -1;
290             int bytes;
291             DataInputStream dataInputStream = null ;
292             try {
293                 dataInputStream = new DataInputStream(socket.getInputStream());
294                 bytes = dataInputStream.read(buffer);
295                 messageReceived = new String((byte[]) buffer, 0, bytes);
296                 String[] strings = messageReceived.split(",");
297                 riskScore = Double.parseDouble(strings[0]);
298                 Log.d("TAG", "MESSAGE RECEIVED : " + riskScore);
299
300             } catch (IOException e) {
301                 e.printStackTrace();
302             }
303
304             return riskScore;
305         }
306     }
307
308
309 }
```

Figure 12: Client Application Main Activity Source Code Continued

```java
package com.example.client;

public class Endpoints {

    String lat;
    String log;
    long id;

    public String getLat() {
        return lat;
    }

    public void setLat(String lat) {
        this.lat = lat;
    }

    public String getLog() {
        return log;
    }

    public void setLog(String log) {
        this.log = log;
    }

    public long getId() {
        return id;
    }

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

    public Endpoints() {
    }

    public Endpoints(String lat, String log, long id) {
        this.lat = lat;
        this.log = log;
        this.id = id;
    }
}
```

Figure 13: Client Application "Endpoints" Class

```java
package com.example.homepage_v1;

import androidx.annotation.RequiresApi;
import androidx.fragment.app.FragmentActivity;

import android.graphics.Color;
import android.graphics.drawable.ColorDrawable;
import android.graphics.drawable.Drawable;
import android.os.AsyncTask;
import android.os.Build;
import android.os.Bundle;
import android.os.StrictMode;
import android.util.Log;
import android.view.View;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

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.BitmapDescriptorFactory;
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.Polygon;
import com.google.android.gms.maps.model.PolygonOptions;

import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.lang.reflect.Array;
import java.net.Socket;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Random;

public class MainActivity
        extends FragmentActivity implements OnMapReadyCallback {
    public static String ip;
    public static int port;
    public static String messageReceived;
    public Socket socket = null;
    public static final byte[] buffer = new byte[6820];
    public static GoogleMap map;

    Double startLati = 35.306758;
    Double startLongi = -80.743803;
    Double topLeftLati = 35.313044;
    Double topLeftLongi = startLongi;
    Double topRightLati = topLeftLati;
    Double topRightLongi = -80.737473;
    Double botRightLati = startLati;
    Double botRightLongi = topRightLongi;
    Double endLati = startLati;
    Double endLongi = startLongi;
    ArrayList<Endpoint> endpoints = new ArrayList<>();
    ArrayList<String> endpointsID = new ArrayList<>();

    LatLng Northcarolina = new LatLng(35.310443, -80.741206);
    HashMap<String,ArrayList<Object>> clients = new HashMap<String, ArrayList<Object>>();

    public static double RiskScore = 0;
    public static double heartRate = 0;
    public static String decision = "";


    @Override
    protected void onCreate(Bundle savedInstanceState) {
```

Figure 14: Digital Twin Application Main Activity Source Code

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // We need to either create a new Thread OR use the below line
        StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder().permitNetwork().build();
        StrictMode.setThreadPolicy(policy);
        setTitle("SafeWorkZone");
        findViewById(R.id.HR).setBackgroundResource(R.drawable.hr);
        findViewById(R.id.alertLevel).setBackgroundResource(R.drawable.alert);

        SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()
                .findFragmentById(R.id.map);
        mapFragment.getMapAsync(this);

    }


    class Client extends AsyncTask<Void,Void,Socket>{

        @Override
        protected void onPreExecute() {
            super.onPreExecute();
            Toast.makeText(MainActivity.this, "Please Wait", Toast.LENGTH_SHORT).show();

        }

        @Override
        protected void onPostExecute(Socket values) {
            super.onPostExecute(values);
            Toast.makeText(MainActivity.this, "Created the socket", Toast.LENGTH_SHORT).show();
            socket = values;
            ReceiveData receiveData = new ReceiveData();
            receiveData.execute();
        }

        @Override
        protected Socket doInBackground(Void... voids) {
            port = 8000;
            ip = "192.168.0.167";

            try {
                socket = new Socket(ip, port);
            } catch (IOException e) {
                e.printStackTrace();
                runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        Toast.makeText(MainActivity.this, "Something went wrong ", Toast.LENGTH_SHORT).show();
                    }
                });
                Client client = new Client();
                client.execute();
            }

            return socket;
        }


    }


    class SendData extends AsyncTask<Void,Void,Void> {

        @Override
        protected void onPreExecute() {
            super.onPreExecute();
```

Figure 15: Digital Twin Application Main Activity Source Code Continued

```
140
141
142         }
143
144         @Override
145         protected void onPostExecute(Void aVoid) {
146             super.onPostExecute(aVoid);
147
148             ReceiveData receiveData = new ReceiveData();
149             receiveData.execute();
150         }
151
152         @Override
153         protected Void doInBackground(Void... voids) {
154
155                 DataOutputStream dataOutputStream = null;
156                 try {
157                     dataOutputStream = new DataOutputStream(socket.getOutputStream());
158                     dataOutputStream.writeUTF("");
159                 } catch (IOException e) {
160                     e.printStackTrace();
161                     ReceiveData receiveData = new ReceiveData();
162                     receiveData.execute();
163             }
164             return null;
165         }
166     }
167
168
169     class ReceiveData extends AsyncTask<Void, Endpoint, ArrayList<Endpoint>> {
170
171         @Override
172         protected void onPreExecute() {
173             super.onPreExecute();
174             findViewById(R.id.HR).setBackgroundResource(0);
175             findViewById(R.id.alertLevel).setBackgroundResource(0);
176
177         }
178
179         @Override
180         protected void onPostExecute( ArrayList<Endpoint> values) {
181             super.onPostExecute(values);
182
183             runOnUiThread(new Runnable() {
184                 @RequiresApi(api = Build.VERSION_CODES.N)
185                 @Override
186                 public void run() {
187                     Toast.makeText(MainActivity.this, "Number of Active Workers in Work Zone " + values.size(), Toast.LENGTH_SHORT).show();
188
189                     TextView textView = findViewById(R.id.alertLevel);
190                     textView.setText(String.valueOf(RiskScore));
191
192                     TextView textView1 = findViewById(R.id.Alerts);
193                     findViewById(R.id.alertLevel).setBackgroundResource(R.drawable.alert);
194
195                     if(RiskScore > 9){
196 //                      findViewById(R.id.alertLevel).setBackgroundColor(getResources().getColor(R.color.high));
197                         textView.setTextColor(getResources().getColor(R.color.high));
198                         findViewById(R.id.alertLevel).setBackgroundResource(R.drawable.alert);
199
200                         if(!decision.equals("")){
201 //                          findViewById(R.id.Alerts).setBackgroundColor(getResources().getColor(R.color.high));
202                             textView1.setTextColor(getResources().getColor(R.color.high));
203                             textView1.setText(decision);
204                         } else {
205                             findViewById(R.id.Alerts).setBackgroundColor(getResources().getColor(R.color.white));
206                             textView1.setText(decision);
207                         }
208
```

Figure 16: Digital Twin Application Main Activity Source Code Continued

```
209                 } else if(RiskScore > 7 & RiskScore <= 9){
210 //                      findViewById(R.id.alertLevel).setBackgroundColor(getResources().getColor(R.color.medium));
211                     textView.setTextColor(getResources().getColor(R.color.medium));
212                     findViewById(R.id.alertLevel).setBackgroundResource(R.drawable.alert);
213
214                     if(!decision.equals("")){
215 //                      findViewById(R.id.Alerts).setBackgroundColor(getResources().getColor(R.color.medium));
216                         textView1.setTextColor(getResources().getColor(R.color.medium));
217                         textView1.setText(decision);
218
219                     }else {
220                         findViewById(R.id.Alerts).setBackgroundColor(getResources().getColor(R.color.white));
221                         textView1.setText(decision);
222                     }
223
224                 } else if(RiskScore > 0 & RiskScore <= 7){
225 //                      findViewById(R.id.alertLevel).setBackgroundColor(getResources().getColor(R.color.low));
226                     textView.setTextColor(getResources().getColor(R.color.low));
227
228 //                          findViewById(R.id.Alerts).setBackgroundColor(getResources().getColor(R.color.white));
229                     textView1.setTextColor(getResources().getColor(R.color.medium));
230
231                     textView1.setText(decision);
232
233                 }
234
235
236                 findViewById(R.id.HR).setBackgroundResource(R.drawable.hr);
237
238                 TextView textView2 = findViewById(R.id.HR);
239                 textView2.setText(String.valueOf((int)heartRate));
240
241
242 //                  if(heartRate <= 0){
243 //
244 //                      findViewById(R.id.HR).setBackgroundColor(getResources().getColor(R.color.low));
245 //                      textView2.setText(String.valueOf(0));
246 //
247 //                  }
248 //
249 //                  else if(heartRate > 0 & heartRate < 50){
250 //                      findViewById(R.id.HR).setBackgroundColor(getResources().getColor(R.color.low));
251 //                      textView2.setText(String.valueOf(heartRate));
252 //
253 //
254 //                  } else if(heartRate >= 50 & heartRate < 90) {
255 //                      findViewById(R.id.HR).setBackgroundColor(getResources().getColor(R.color.medium));
256 //                      textView2.setText(String.valueOf(heartRate));
257 //
258 //
259 //                  } else if(heartRate >= 90) {
260 //                      findViewById(R.id.HR).setBackgroundColor(getResources().getColor(R.color.high));
261 //                      textView2.setText(String.valueOf(heartRate));
262 //
263 //                  }
264
265                 ArrayList<String> activeTags = new ArrayList<>();
266
267                 try{
268                     if(values.size() > 0 & clients.size() > 0 ) {
269                         for(Endpoint endpoint:values){
270                             activeTags.add(endpoint.getId());
271                             if(clients.get(endpoint.getId()).size() == 1 ){
272
273                                 Marker marker =  map.addMarker(new MarkerOptions().icon(BitmapDescriptorFactory.defaultMarker((Float) clients.get(endpoint.getId()).get
274                                         .position(new LatLng(Double.parseDouble(endpoint.getLatitude()),Double.parseDouble(endpoint.getLongitude()))))
275                                 );
276                                 marker.setTag(endpoint.getId());
277                                 ArrayList<Object> arrayList = clients.get(endpoint.getId());
```

Figure 17: Digital Twin Application Main Activity Source Code Continued

```
                    arrayList.add(marker);
                    clients.put(endpoint.getId(),arrayList);
                    marker.setPosition(new LatLng(Double.parseDouble(endpoint.getLatitude()),Double.parseDouble(endpoint.getLongitude())));

                } else if(clients.get(endpoint.getId()).size() == 2){

                    ArrayList<Object> arrayList = clients.get(endpoint.getId());
                    Marker marker = (Marker) arrayList.get(1);
                    marker.setPosition(new LatLng(Double.parseDouble(endpoint.getLatitude()),Double.parseDouble(endpoint.getLongitude())));
                }

            }

        } else if(values.size() == 0 ){

            map.clear();
            map.moveCamera(CameraUpdateFactory.newLatLngZoom(Northcarolina, 16));
            Polygon polygon1 = map.addPolygon(new PolygonOptions()
                    .clickable(true)
                    .add(
                            new LatLng(startLati, startLongi),
                            new LatLng(topLeftLati, topLeftLongi),
                            new LatLng(topRightLati, topRightLongi),
                            new LatLng(botRightLati, botRightLongi),
                            new LatLng(endLati, endLongi))
                    .strokeColor(Color.RED)
                    .fillColor(Color.argb(50, 255, 0, 0)));
        }

        ArrayList<Marker> activeMarkers = new ArrayList<>();

        for(String string : clients.keySet()){
            if(clients.get(string).size() == 2){
                activeMarkers.add((Marker) clients.get(string).get(1));
            }
        }

        for(Marker marker : activeMarkers){
            if(!activeTags.contains(marker.getTag())){
                ArrayList<Object> newArray = clients.get(marker.getTag());
                newArray.remove(1);
                clients.put((String) marker.getTag(),newArray);
                marker.remove();
            }
        }

        SendData sendData = new SendData();
        sendData.execute();

    } catch (Exception e){

        map.clear();
        map.moveCamera(CameraUpdateFactory.newLatLngZoom(Northcarolina, 16));
        Polygon polygon1 = map.addPolygon(new PolygonOptions()
                .clickable(true)
                .add(
                        new LatLng(startLati, startLongi),
                        new LatLng(topLeftLati, topLeftLongi),
                        new LatLng(topRightLati, topRightLongi),
                        new LatLng(botRightLati, botRightLongi),
                        new LatLng(endLati, endLongi))
                .strokeColor(Color.RED)
                .fillColor(Color.argb(50, 255, 0, 0)));


        SendData sendData = new SendData();
        sendData.execute();

    }
```

Figure 18: Digital Twin Application Main Activity Source Code Continued

```
348
349             }
350
351         });}
352
353
354     @Override
355     protected ArrayList<Endpoint> doInBackground(Void... values) {
356
357         int bytes;
358         DataInputStream dataInputStream ;
359         Random random = new Random();
360
361         endpointsID.clear();
362         endpoints.clear();
363
364             try {
365                 dataInputStream =  new DataInputStream(socket.getInputStream());
366                 bytes = dataInputStream.read(buffer);
367                 messageReceived = new String((byte[]) buffer, 0, bytes);
368                 String[] strings = messageReceived.split(",");
369
370                 RiskScore = Double.parseDouble(strings[0]);
371                 if(RiskScore > 7){
372                     decision = strings[1];
373                 } else {
374                     decision = "";
375                 }
376
377                 for (int i = 0; i < strings.length; i++){
378
379                     if(strings[i].contains("ID") & !endpointsID.contains(strings[i])){
380                         heartRate = Double.parseDouble(strings[i+3]);
381                         if(heartRate < 0){
382                             heartRate = 0;
383                         }
384
385                         if(!clients.keySet().contains(strings[i])){
386                             Log.d("TAG", "doInBackground: "+"here2");
387                             Endpoint endpoint = new Endpoint(strings[i+1],strings[i+2],strings[i]);
388                             float color = random.nextFloat()*250;
389                             ArrayList<Object> arrayList = new ArrayList<>();
390                             arrayList.add(color);
391                             clients.put(strings[i],arrayList);
392                             endpoints.add(endpoint);
393                             endpointsID.add(strings[i]);
394                             Log.d("TAG", "doInBackground: "+endpoint.toString());
395                         } else {
396                             Log.d("TAG", "doInBackground: "+"here3");
397                             Endpoint endpoint = new Endpoint(strings[i+1],strings[i+2],strings[i]);
398                             endpoints.add(endpoint);
399                             endpointsID.add(strings[i]);
400
401                             Log.d("TAG", "doInBackground: "+endpoint.toString());
402                         }
403
404                     }
405                 }
406                 return endpoints;
407
408
409             }
410             catch (IOException e) {
411                 e.printStackTrace();
412                 SendData sendData = new SendData();
413                 sendData.execute();
414             }
415
416         return null;
```

Figure 19: Digital Twin Application Main Activity Source Code Continued

```
417
418         }
419
420
421         }
422
423
424     @Override
425     public void onMapReady(GoogleMap googleMap) {
426         map = googleMap;
427
428         map.setOnMapClickListener(new GoogleMap.OnMapClickListener() {
429             @Override
430             public void onMapClick(LatLng latLng) {
431                 Log.d("TAG", "onMapClick: " + latLng.toString());
432
433                 Client client = new Client();
434                 client.execute();
435
436             }});
437
438                 map.moveCamera(CameraUpdateFactory.newLatLngZoom(Northcarolina, 16));
439                 Polygon polygon1 = map.addPolygon(new PolygonOptions()
440                         .clickable(true)
441                         .add(
442                                 new LatLng(startLati, startLongi),
443                                 new LatLng(topLeftLati, topLeftLongi),
444                                 new LatLng(topRightLati, topRightLongi),
445                                 new LatLng(botRightLati, botRightLongi),
446                                 new LatLng(endLati, endLongi))
447                         .strokeColor(Color.RED)
448                         .fillColor(Color.argb(50, 255, 0, 0)));
449
450
451
452         };
453
454
455     }
```

Figure 20: Digital Twin Application Main Activity Source Code

```java
package com.example.homepage_v1;

import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.FragmentActivity;

import android.content.Intent;
import android.graphics.Color;
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.Handler;
import android.os.StrictMode;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

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.internal.ICameraUpdateFactoryDelegate;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
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.Polygon;
import com.google.android.gms.maps.model.PolygonOptions;

import java.io.DataInputStream;
import java.io.IOException;
import java.net.Socket;

public class LoginActivity extends AppCompatActivity {
    private Button button;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_connnection_test);

        button = (Button) findViewById(R.id.Test);
        ImageView imageView = findViewById(R.id.logo);
        imageView.setImageResource(R.drawable.logo);
        button.setText("Click here !");

        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {



                openMainActivity();
            }

        });
    }

    public void openMainActivity() {
        Intent intent = new Intent(this, MainActivity.class);
        startActivity(intent);
    }


}
```

Figure 21: Digital Twin Application Login Activity Source Code

```java
package com.example.homepage_v1;

import java.util.Random;

public class Endpoint {

    String latitude;
    String longitude;
    String id;


    public float getColor() {
        return color;
    }

    public void setColor(float color) {
        this.color = color;
    }

    float color;

    public String getLatitude() {
        return latitude;
    }

    public void setLatitude(String latitude) {
        this.latitude = latitude;
    }

    public String getLongitude() {
        return longitude;
    }

    public void setLongitude(String longitude) {
        this.longitude = longitude;
    }

    public String getId() {
        return id;
    }

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

    public Endpoint() {
    }

    public Endpoint(String latitude, String longitude, String id) {
        this.latitude = latitude;
        this.longitude = longitude;
        this.id = id;
    }

    @Override
    public String toString() {
        return "Endpoint{" +
                "latitude='" + latitude + '\'' +
                ", longitude='" + longitude + '\'' +
                ", id='" + id + '\'' +
                '}';
    }
}

```

Figure 22: Digital Twin Application “Endpoints” Class Source Code

```python
import socket
from _thread import *
import threading
import sys
import csv
import time
import random
import datetime

HOST = '192.168.0.167' # IP address of server
PORT = 8000
threadCount = 0
i = 0
messagePackage = []
serverSocket = socket.socket()
serverSocket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR,1)
serverSocket.settimeout(1)
serverSocket.bind((HOST,PORT))


def threadedSendMessage(arrayOfCones , messagePackage,sentMessage):
    for cons in arrayOfCones:
        message = sentMessage
        for i in messagePackage:
            message = message + i

        messageSplit = message.split(",")
        for characters in messageSplit:
            if '('  in characters or ')' in characters:
                message = sentMessage
        try:
            cons.sendall(message.encode())
        except socket.error:
            cons.close()
    print("Final Message:" +  message)

    messagePackage.clear()

def threadedClientIndi(conn , messagePackage,e):
    e.clear()
    try:
        data = conn.recv(1024)
        recvMessage = data.decode('utf-8' , 'replace')
        recvMessage = recvMessage[2:]
        print(recvMessage)
        messagePackage.append(recvMessage)
        e.set()
    except socket.error:
        e.set()

def threadedClient(arrayOfCones , messagePackage , e):

    for conn in arrayOfCones:
        start_new_thread(threadedClientIndi, ((conn,messagePackage,e)))
    print('Done with this iteration')



print('Server started')
print('Socket Listening...')
serverSocket.listen(5) # Queue of connections

allClient = []

iteration = 0

e = threading.Event()
messages = ["Intrusion - 2 Seconds - Run," , "Barrier Removed - Fix It - Now,"]

while True:
```

Figure 23: Multithreaded Socket Server Source Code

```
    try:
        client , address = serverSocket.accept()
        allClient.append(client)

    except socket.timeout:
        e.set()
        sentMessage = ""

        if iteration < 30:
            sentMessage = "{:.2f}".format(random.random()*6.99)
        elif 30 <= iteration and iteration < 35:
            sentMessage = "{:.2f}".format(7.0 + random.random())
        elif iteration >= 35 and iteration < 40:
            sentMessage = "{:.2f}".format(random.random()*6.99)
        elif iteration >= 40 and iteration < 45:
            sentMessage = "{:.2f}".format(9.99)

        else:
            sentMessage = "{:.2f}".format(random.random()*6.99)


        if(float(sentMessage) > 9):
            sentMessage = sentMessage +","+ messages[0]
        elif (float(sentMessage) <=9 and float(sentMessage) > 7 ):
            sentMessage = sentMessage +","+ messages[1]
        else :
            sentMessage = sentMessage +","

        if(len(allClient) != 0 and e.isSet()):
            start_new_thread(threadedSendMessage, ((allClient,messagePackage,sentMessage)))
            start_new_thread(threadedClient, ((allClient,messagePackage,e)))
            time.sleep(1)
            iteration = iteration + 1

serverSocket.close()
serverSocket.shutdown(socket.SHUT_RDWR)
```

Figure 24: Multithreaded Socket Server Source Code Continued

Project UNCC_WORK SD2	**93.33 days**	**Wed 1/20/21**	**Fri 5/7/21**		**2**	**216 hrs**	**239**
Project Management	**93.33 days**	**Wed 1/20/21**	**Fri 5/7/21**		**2.1**	**19 hrs**	**241**
Revised Final Design Package	0.13 days?	Mon 2/1/21	Mon 2/1/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro,William Clampett,Zach Zaleski	DOCUMENT - ADMIN	2 hrs	248
Project Plan	**93.33 days**	**Wed 1/20/21**	**Fri 5/7/21**		**PLAN**	**7 hrs**	**242**
Initial Project Plan Updating	0.33 days?	Wed 1/20/21	Wed 1/20/21	Damian Hupka	PLAN	1 hr	240
Updating Project Plan	0.33 days?	Mon 2/8/21	Mon 2/8/21	Damian Hupka	PLAN	1 hr	263
Updating Project Plan	0.33 days?	Thu 2/25/21	Thu 2/25/21	Damian Hupka	PLAN	1 hr	283
Updating Project Plan	0.33 days?	Mon 3/1/21	Mon 3/1/21	Damian Hupka	PLAN	1 hr	284
Updating Project Plan	0.33 days?	Mon 3/22/21	Mon 3/22/21	Damian Hupka	PLAN	1 hr	290
Updating Project Plan	0.33 days?	Mon 4/26/21	Mon 4/26/21	Damian Hupka	PLAN	1 hr	311
Updating Project Plan	0.33 days?	Thu 5/6/21	Thu 5/6/21	Damian Hupka	PLAN	1 hr	334
Timesheets	**93.33 days**	**Wed 1/20/21**	**Fri 5/7/21**		**2.1.3**	**6 hrs**	**243**
Timesheet #1	0.07 days?	Mon 2/8/21	Mon 2/8/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro,William Clampett,Zach Zaleski	DOCUMENT - ADMIN	1 hr	244
Timesheet #2	0.07 days?	Mon 3/1/21	Mon 3/1/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro,William Clampett,Zach Zaleski	DOCUMENT - ADMIN	1 hr	282
Timesheet #3	0.33 days?	Mon 3/22/21	Mon 3/22/21	Zach Zaleski	DOCUMENT - ADMIN	1 hr	289
Timesheet #4	1 day?	Mon 4/12/21	Mon 4/12/21	Zach Zaleski[33%]	DOCUMENT - ADMIN	1 hr	313
Timesheet #5	1 day?	Mon 4/26/21	Mon 4/26/21	Zach Zaleski[33%]	DOCUMENT - ADMIN	1 hr	312
Timesheet #6	0.33 days?	Thu 5/6/21	Thu 5/6/21	Zach Zaleski	DOCUMENT - ADMIN	1 hr	333
Progress Reports	**93.33 days**	**Wed 1/20/21**	**Fri 5/7/21**		**DOCUMENT - ADMIN**	**4 hrs**	**245**
Progress Report #1	0.13 days?	Mon 3/1/21	Mon 3/1/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro,William Clampett,Zach Zaleski	DOCUMENT - ADMIN	2 hrs	246
Progress Report #2	0.13 days?	Mon 4/5/21	Mon 4/5/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro,William Clampett,Zach Zaleski	DOCUMENT - ADMIN	2 hrs	299

Figure 25: Project Plan

Project Status Review Presentation (PSR)	**41.33 days**	**Wed 1/20/21**	**Sat 3/6/21**		**DESIGN**	**15 hrs**	**249**
PSR Presentation Preparation	0.27 days?	Thu 2/25/21	Thu 2/25/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro,William Clampett,Zach Zaleski	DESIGN	4 hrs	250
Initial PSR Meeting	0.2 days?	Fri 2/26/21	Fri 2/26/21	Damian Hupka[33%],Duncan Tennant[33%], Nathan Pecoraro[33%], William Clampett[33%],Zach Zaleski[33%]	DESIGN	1 hr	286
Various Schematic Design	0.2 days?	Mon 3/1/21	Mon 3/1/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro,William Clampett,Zach Zaleski	DESIGN	3 hrs	285
Re-do PSR Meeting	0.07 days?	Fri 3/5/21	Fri 3/5/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro,William Clampett,Zach Zaleski	DESIGN	1 hr	288
Initial Creation of Updated PSR	0.13 days?	Tue 3/2/21	Tue 3/2/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro,William Clampett,Zach Zaleski	DESIGN	2 hrs	291
Updated with feedback PSR	0.27 days?	Fri 3/5/21	Fri 3/5/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro,William Clampett,Zach Zaleski	DESIGN	4 hrs	287
Prototype Review Presentation (PRP)	**73.33 days**	**Wed 1/20/21**	**Wed 4/21/21**		**DESIGN**	**8 hrs**	**253**
Initial PRP Creation	0.2 days?	Mon 4/12/21	Mon 4/12/21	Damian Hupka[33%],Duncan Tennant[33%], Nathan Pecoraro[33%], William Clampett[33%],Zach Zaleski[33%]	DESIGN	1 hr	254
PRP Work	0.2 days?	Thu 4/15/21	Thu 4/15/21	Damian Hupka,Duncan Tennant,Nathan Pecorarc	DESIGN	3 hrs	321
Demo of app. For PRP	0.25 days?	Fri 4/16/21	Fri 4/16/21	Damian Hupka,Duncan Tennant,Nathan Pecorarc	DESIGN	3 hrs	322
PRP Presentation	0.07 days?	Wed 4/21/21	Wed 4/21/21	Damian Hupka,Duncan Tennant,Nathan Pecorarc	DESIGN	1 hr	323
Final Project Report	**93.33 days**	**Wed 1/20/21**	**Fri 5/7/21**		**DOCUMENT - ADMIN**	**10 hrs**	**255**
Compiling Design Package Submission and Final Report	0.67 days?	Fri 5/7/21	Fri 5/7/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro,William Clampett,Zach Zaleski	DOCUMENT - ADMIN	10 hrs	256

Figure 26: Project Plan Continued

Lab Meetings, Design, and Prototyping	93.33 days	Wed 1/20/21	Fri 5/7/21		2.5	151 hrs	257
Establishing Github Repository and Flashing New Desktop	0.33 days?	Mon 2/8/21	Mon 2/8/21	Damian Hupka,Nathan Pecoraro	DESIGN	2 hrs	258
Application Development Research	0.5 days?	Tue 2/2/21	Tue 2/2/21	Duncan Tennant[67%],William Clampett[67%]	DESIGN	2 hrs	266
Application "Stories" trimming	0.2 days?	Mon 2/1/21	Mon 2/1/21	Damian Hupka[67%],Duncan Tennant[67%], Nathan Pecoraro[67%], William Clampett[67%],Zach Zaleski[67%]	DESIGN	2 hrs	264
Multithreading Implementation Research	1 day?	Tue 2/9/21	Tue 2/9/21	Zach Zaleski[83%]	RESEARCH	2.5 hrs	267
Multi-socket code configuration	0.33 days?	Wed 2/10/21	Wed 2/10/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro,Zach Zaleski	DESIGN	4 hrs	273
End-User Application Development	1 day?	Wed 1/20/21	Wed 1/20/21	William Clampett	DESIGN	3 hrs	274
End-User Application Homepage	1 day?	Wed 2/10/21	Wed 2/10/21	Duncan Tennant	DESIGN	3 hrs	269
End-User Application Development	1 day?	Thu 2/11/21	Thu 2/11/21	William Clampett	DESIGN	3 hrs	272
Tizen Development and End-User Application Development	0.28 days?	Thu 2/11/21	Thu 2/11/21	Damian Hupka,Nathan Pecoraro,Zach Zaleski	DESIGN	2.5 hrs	271
2 Socket Server, tizen Development and End-User Application Development	0.61 days?	Fri 2/12/21	Fri 2/12/21	Damian Hupka,Nathan Pecoraro,Zach Zaleski	DESIGN	5.5 hrs	268
Outdoor Latency Testing (2 socket)	0.33 days?	Wed 2/17/21	Wed 2/17/21	Damian Hupka,Nathan Pecoraro,Zach Zaleski	DESIGN	3 hrs	279
Threaded Server Testing	0.33 days?	Sat 2/20/21	Sat 2/20/21	Damian Hupka,Nathan Pecoraro,Zach Zaleski	DESIGN	3 hrs	278
Testing watch latency and more clients	0.33 days?	Wed 2/24/21	Wed 2/24/21	Damian Hupka,Nathan Pecoraro,Zach Zaleski	DESIGN	3 hrs	277
Multi-Client Latency Testing	0.44 days?	Thu 2/25/21	Thu 2/25/21	Damian Hupka,Nathan Pecoraro, William Clampett	DESIGN	4 hrs	276
App. Redesign	0.83 days?	Sun 2/28/21	Sun 2/28/21	Duncan Tennant	DESIGN	2.5 hrs	293
App. Dev. Discussion with graduate mentor	1 day?	Mon 3/1/21	Mon 3/1/21		DESIGN	1 hr	292
Finished all latency test scenarios	0.78 days?	Fri 3/12/21	Fri 3/12/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro	DESIGN	7 hrs	295

Figure 27: Project Plan Continued

Fragment and Importing Map API	0.5 days?	Wed 3/17/21	Wed 3/17/21	Duncan Tennant,William Clampett	DESIGN	3 hrs	296
Map Working with Emulator	0.67 days?	Thu 3/18/21	Thu 3/18/21	Duncan Tennant,William Clampett	DESIGN	4 hrs	297
Multithreaded Server Implementation Work	0.44 days?	Fri 3/19/21	Fri 3/19/21	Damian Hupka,Nathan Pecoraro,Zach Zaleski	DESIGN	4 hrs	298
Multithreaded Discussion	0.33 days?	Mon 3/29/21	Mon 3/29/21	Damian Hupka,Nathan Pecoraro,Zach Zaleski	DESIGN	3 hrs	302
Application Development Discussion	0.17 days?	Wed 3/31/21	Wed 3/31/21	Duncan Tennant,William Clampett	DESIGN	1 hr	300
Application Development Research and Development	0.5 days?	Wed 3/31/21	Wed 3/31/21	Duncan Tennant,William Clampett	DESIGN	3 hrs	301
Multithreaded Server Development	0.25 days?	Mon 4/5/21	Mon 4/5/21	Damian Hupka,Nathan Pecoraro, William Clampett,Zach Zaleski	DESIGN	3 hrs	305
Multithreaded Server Dev.	0.33 days?	Wed 4/7/21	Wed 4/7/21	Damian Hupka,Nathan Pecoraro,Zach Zaleski	DESIGN	3 hrs	308
GPS on Digital Twin App.	0.2 days?	Fri 4/9/21	Fri 4/9/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro,William Clampett,Zach Zaleski	DESIGN	3 hrs	309
GPS Implementation	2 days?	Mon 4/12/21	Tue 4/13/21	Damian Hupka	DESIGN	6 hrs	307
GPS Implementation	0.44 days?	Mon 4/12/21	Mon 4/12/21	Duncan Tennant,Nathan Pecoraro,Zach Zaleski	DESIGN	4 hrs	306
GPS Implementation with Grad. Mentor Assistance	0.25 days?	Wed 4/14/21	Wed 4/14/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro,Zach Zaleski	DESIGN	3 hrs	314
App. Demo and further implementation	0.33 days?	Mon 4/19/21	Mon 4/19/21	Damian Hupka,Nathan Pecoraro, William Clampett,Zach Zaleski	DESIGN	4 hrs	318
Server Demo and fixing implementation	0.33 days?	Tue 4/20/21	Tue 4/20/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro	DESIGN	3 hrs	315
Preparing project video ,server implementation, client implementation, digital twin implementation	0.89 days?	Fri 4/23/21	Fri 4/23/21	Damian Hupka,Nathan Pecoraro,Zach Zaleski	DESIGN	8 hrs	316
Server upgrades for communication	0.89 days?	Sat 4/24/21	Sat 4/24/21	Damian Hupka,Nathan Pecoraro,Zach Zaleski	DESIGN	8 hrs	317
UI IX for Digital Twin	1 day?	Sat 4/24/21	Sat 4/24/21	Duncan Tennant,William Clampett	DESIGN	6 hrs	320
Finished server implementatino for Demo, Updating DT UI, updating DT, Updating Client	0.8 days?	Sun 4/25/21	Sun 4/25/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro, William Clampett,Zach Zaleski	DESIGN	12 hrs	319

Figure 28: Project Plan Continued

Finished implementation of Server for demo upgraded DT, recording demos	0.6 days?	Mon 4/26/21	Mon 4/26/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro, William Clampett,Zach Zaleski	DESIGN	9 hrs	325
Meeting to discuss AI implementation with ex-graduate researcher	0.13 days?	Fri 4/30/21	Fri 4/30/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro, William Clampett,Zach Zaleski	DESIGN	2 hrs	324
AI Dependencies on Xavier and Re-identification research	0.67 days?	Sat 5/1/21	Sat 5/1/21	Damian Hupka	DESIGN	2 hrs	328
Implementing DeepSORT (Re-ID) with YOLO.v4 Trained Model	0.27 days?	Tue 5/4/21	Tue 5/4/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro, William Clampett,Zach Zaleski	DESIGN	4 hrs	330

Figure 29: Project Plan Continued

Weekly Meetings	92.33 days	Thu 1/21/21	Fri 5/7/21		DESIGN	13 hrs	259
Weekly Meeting with Supporter/Graduate Mentor	0.2 days?	Thu 1/21/21	Thu 1/21/21	Damian Hupka[33%],Duncan Tennant[33%], Nathan Pecoraro[33%], William Clampett[33%],Zach Zaleski[33%]	DESIGN	1 hr	260
Weekly Meeting with Supporter/Graduate Mentor	0.2 days?	Fri 1/29/21	Fri 1/29/21	Damian Hupka[33%],Duncan Tennant[33%], Nathan Pecoraro[33%], William Clampett[33%],Zach Zaleski[33%]	DESIGN	1 hr	261
Weekly Meeting with Supporter/Graduate Mentor	0.2 days?	Fri 2/5/21	Fri 2/5/21	Damian Hupka[33%],Duncan Tennant[33%], Nathan Pecoraro[33%], William Clampett[33%],Zach Zaleski[33%]	DESIGN	1 hr	262
Weekly Meeting with Supporter/Graduate Mentor	0.2 days?	Fri 2/12/21	Fri 2/12/21	Damian Hupka[33%],Duncan Tennant[33%], Nathan Pecoraro[33%], William Clampett[33%],Zach Zaleski[33%]	DESIGN	1 hr	280
Weekly Meeting with Supporter/Graduate Mentor	0.07 days?	Fri 2/19/21	Fri 2/19/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro,William Clampett,Zach Zaleski	DESIGN	1 hr	281
Weekly Meeting with Supporter/Graduate Mentor	0.07 days?	Fri 3/19/21	Fri 3/19/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro,William Clampett,Zach Zaleski	DESIGN	1 hr	294
Weekly Meeting with Supporter/Graduate Mentor	0.07 days?	Fri 3/26/21	Fri 3/26/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro,William Clampett,Zach Zaleski	DESIGN	1 hr	303
Weekly Meeting with Supporter/Graduate Mentor	0.07 days?	Fri 4/2/21	Fri 4/2/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro,William Clampett,Zach Zaleski	DESIGN	1 hr	304
Weekly Meeting with Supporter/Graduate Mentor	0.25 days?	Fri 4/9/21	Fri 4/9/21	Damian Hupka[33%],Duncan Tennant[33%], Nathan Pecoraro[33%], Zach Zaleski[33%]	DESIGN	1 hr	310
Weekly Meeting with Supporter/Graduate Mentor	0.07 days?	Fri 4/16/21	Fri 4/16/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro,William Clampett,Zach Zaleski	DESIGN	1 hr	326
Weekly Meeting with Supporter/Graduate Mentor	0.07 days?	Fri 4/23/21	Fri 4/23/21	Damian Hupka,Duncan Tennant, Nathan Pecoraro,William Clampett,Zach Zaleski	DESIGN	1 hr	327
Weekly Meeting with Supporter/Graduate Mentor	0.2 days?	Fri 4/30/21	Fri 4/30/21	Damian Hupka[33%],Duncan Tennant[33%], Nathan Pecoraro[33%],William Clampett[33%],Z	DESIGN	1 hr	331
Weekly Meeting with Supporter/Graduate Mentor	0.2 days?	Fri 5/7/21	Fri 5/7/21	Damian Hupka[33%],Duncan Tennant[33%], Nathan Pecoraro[33%],William Clampett[33%],Z	2.6.13	1 hr	332

Figure 30: Project Plan Continued

Qty	Name	Description	Cost	
1	NVIDIA Jetson AGX Xavier Developer Kit	AI Enabled Embedded Device	$	699.00
1	Vuzix Blade	Augmented Reality Goggles	$	899.99
1	Samsung Galaxy Watch Active	Smartwatch	$	199.99
1	TP-Link AX1500 Next-Gen Wi-Fi 6 Router	Cutting Edge Access Point	$	79.99
1	Fire HD 10 Tablet	Amazon Android Tablet Device	$	94.99
1	Logitech C922 Webcam	USB Connected	$	99.99
6			$	2,073.95

Figure 31: Bill of Materials

```python
import argparse

from models import *  # set ONNX_EXPORT in models.py
from utils.datasets import *
from utils.utils import *
from deep_sort import DeepSort

deepsort = DeepSort("deep_sort/deep/checkpoint/ckpt.t7")
palette = (2 ** 11 - 1, 2 ** 15 - 1, 2 ** 20 - 1)

def bbox_rel(image_width, image_height, bbox_left, bbox_top, bbox_w, bbox_h):
    """" Calculates the relative bounding box from absolute pixel values. """
    x_c = (bbox_left + bbox_w / 2)
    y_c = (bbox_top + bbox_h / 2)
    w = bbox_w
    h = bbox_h
    return x_c, y_c, w, h


def compute_color_for_labels(label):
    """
    Simple function that adds fixed color depending on the class
    """
    color = [int((p * (label ** 2 - label + 1)) % 255) for p in palette]
    return tuple(color)


def draw_boxes(img, bbox, identities, offset=(0,0)):
    for i, box in enumerate(bbox):
        x1, y1, x2, y2 = [int(i) for i in box]

        x1 += offset[0]
        x2 += offset[0]
        y1 += offset[1]
        y2 += offset[1]
        # box text and bar
        id = int(identities[i]) if identities is not None else 0
        color = compute_color_for_labels(id)
        label = '{}{:d}'.format("", id)
        t_size = cv2.getTextSize(label, cv2.FONT_HERSHEY_PLAIN, 2 , 2)[0]
        cv2.rectangle(img, (x1, y1),(x2,y2), color, 3)
        cv2.rectangle(img, (x1, y1), (x1 + t_size[0] + 3, y1 + t_size[1] + 4), color, -1)
        cv2.putText(img, label, (x1, y1 + t_size[1] + 4), cv2.FONT_HERSHEY_PLAIN, 2, [255, 255, 255], 2)
    return img

def detect(save_img=False):
    imgsz = (320, 192) if ONNX_EXPORT else opt.img_size  # (320, 192) or (416, 256) or (608, 352) for (height, width)
    out, source, weights, view_img, save_txt = opt.output, opt.source, opt.weights, opt.view_img, opt.save_txt
    webcam = source == '0' or source.startswith('rtsp') or source.startswith('http') or source.endswith('.txt')

    # Initialize
    device = torch_utils.select_device(device='cpu' if ONNX_EXPORT else opt.device)
    if os.path.exists(out):
        shutil.rmtree(out)  # delete output folder
    os.makedirs(out)  # make new output folder

    # Initialize model
    model = Darknet(opt.cfg, imgsz, quantized=opt.quantized, quantizer_output=opt.quantizer_output, a_bit=opt.a_bit,
                    w_bit=opt.w_bit,
                    FPGA=opt.FPGA)

    # Load weights
    attempt_download(weights)
    if weights.endswith('.pt'):  # pytorch format
        model.load_state_dict(torch.load(weights, map_location=device)['model'], strict=False)
    else:  # darknet format
        load_darknet_weights(model, weights)
    ################]]Dmodel_list
    '''AWEIGHT = torch.load(weights, map_location=device)['model']
    for k,v in AWEIGHT.items():
        print(k)'''
```

Figure 32: YOLO and DeepSORT “detect.py”

```
72
73      # Eval mode
74      model.to(device).eval()
75
76      # Set Dataloader
77      vid_path, vid_writer = None, None
78      if webcam:
79          view_img = True
80          torch.backends.cudnn.benchmark = True  # set True to speed up constant image size inference
81          dataset = LoadStreams(source, img_size=imgsz)
82      else:
83          save_img = True
84          dataset = LoadImages(source, img_size=imgsz)
85
86      # Get names and colors
87      names = load_classes(opt.names)
88      colors = [[random.randint(0, 255) for _ in range(3)] for _ in range(len(names))]
89
90      # Run inference
91      t0 = time.time()
92      img = torch.zeros((1, 3, imgsz, imgsz), device=device)  # init img
93      # _ = model(img.float()) if device.type != 'cpu' else None  # run once
94      for path, img, im0s, vid_cap in dataset:
95          img = torch.from_numpy(img).to(device)
96          img = img.float()  # uint8 to fp16/32
97          img /= 255.0  # 0 - 255 to 0.0 - 1.0
98          if img.ndimension() == 3:
99              img = img.unsqueeze(0)
100
101         # Inference
102         t1 = torch_utils.time_synchronized()
103         pred = model(img, augment=opt.augment)[0]
104         t2 = torch_utils.time_synchronized()
105
106         # Apply NMS
107         pred = non_max_suppression(pred, opt.conf_thres, opt.iou_thres,
108                                    multi_label=False, classes=opt.classes, agnostic=opt.agnostic_nms)
109
110         # Process detections
111         for i, det in enumerate(pred):  # detections for image i
112             if webcam:  # batch_size >= 1
113                 p, s, im0 = path[i], '%g: ' % i, im0s[i].copy()
114             else:
115                 p, s, im0 = path, '', im0s
116
117             save_path = str(Path(out) / Path(p).name)
118             s += '%gx%g ' % img.shape[2:]  # print string
119             gn = torch.tensor(im0.shape)[[1, 0, 1, 0]]  #  normalization gain whwh
120             if det is not None and len(det):
121                 # Rescale boxes from imgsz to im0 size
122                 det[:, :4] = scale_coords(img.shape[2:], det[:, :4], im0.shape).round()
123
124                 # Print results
125                 for c in det[:, -1].unique():
126                     n = (det[:, -1] == c).sum()  # detections per class
127                     s += '%g %ss, ' % (n, names[int(c)])  # add to string
128
129                 bbox_xywh = []
130                 confs = []
131
132                 # #Write results
133                 # for *xyxy, conf, cls in det:
134                 #     if save_txt:  # Write to file
135                 #         xywh = (xyxy2xywh(torch.tensor(xyxy).view(1, 4)) / gn).view(-1).tolist()  # normalized xywh
136                 #         with open(save_path[:save_path.rfind('.')] + '.txt', 'a') as file:
137                 #             file.write(('%g ' * 5 + '\n') % (cls, *xywh))  # label format
138
139                 #     if save_img or view_img:  # Add bbox to image
140                 #         label = '%s %.2f' % (names[int(cls)], conf)
141                 #         plot_one_box(xyxy, im0, label=label, color=colors[int(cls)])
```

Figure 32: YOLO and DeepSORT “detect.py” continued

```
                for *xyxy, conf, cls in det:
                    img_h, img_w, _ = im0.shape  # get image shape
                    bbox_left = min([xyxy[0].item(), xyxy[2].item()])
                    bbox_top = min([xyxy[1].item(), xyxy[3].item()])
                    bbox_w = abs(xyxy[0].item() - xyxy[2].item())
                    bbox_h = abs(xyxy[1].item() - xyxy[3].item())
                    x_c, y_c, bbox_w, bbox_h = bbox_rel(img_w, img_h, bbox_left, bbox_top, bbox_w, bbox_h)
                    #print(x_c, y_c, bbox_w, bbox_h)
                    obj = [x_c, y_c, bbox_w, bbox_h]
                    bbox_xywh.append(obj)
                    confs.append([conf.item()])
                    label = '%s %.2f' % (names[int(cls)], conf)
                    #
                    #print('bboxes')
                    #print(torch.Tensor(bbox_xywh))
                    #print('confs')
                    #print(torch.Tensor(confs))
                    outputs = deepsort.update((torch.Tensor(bbox_xywh)), (torch.Tensor(confs)) , im0)
                    if len(outputs) > 0:
                        bbox_xyxy = outputs[:, :4]
                        identities = outputs[:, -1]
                        draw_boxes(im0, bbox_xyxy,identities)
                    #print('\n\n\t\ttracked objects')
                    #print(outputs)
            # Print time (inference + NMS)
            print('%sDone. (%.3fs)' % (s, t2 - t1))

            # Stream results
            if view_img:
                cv2.imshow(p, im0)
                if cv2.waitKey(1) == ord('q'):  # q to quit
                    raise StopIteration

            # Save results (image with detections)
            if save_img:
                if dataset.mode == 'images':
                    cv2.imwrite(save_path, im0)
                else:
                    if vid_path != save_path:  # new video
                        vid_path = save_path
                        if isinstance(vid_writer, cv2.VideoWriter):
                            vid_writer.release()  # release previous video writer

                        fps = vid_cap.get(cv2.CAP_PROP_FPS)
                        w = int(vid_cap.get(cv2.CAP_PROP_FRAME_WIDTH))
                        h = int(vid_cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
                        vid_writer = cv2.VideoWriter(save_path, cv2.VideoWriter_fourcc(*opt.fourcc), fps, (w, h))
                    vid_writer.write(im0)

    if save_txt or save_img:
        print('Results saved to %s' % os.getcwd() + os.sep + out)
        if platform == 'darwin':  # MacOS
            os.system('open ' + save_path)

    print('Done. (%.3fs)' % (time.time() - t0))


if __name__ == '__main__':
    parser = argparse.ArgumentParser()
    parser.add_argument('--cfg', type=str, default='cfg/yolov3-spp.cfg', help='*.cfg path')
    parser.add_argument('--names', type=str, default='data/coco.names', help='*.names path')
    parser.add_argument('--weights', type=str, default='weights/yolov3-spp-ultralytics.pt', help='weights path')
    parser.add_argument('--source', type=str, default='data/samples', help='source')  # input file/folder, 0 for webcam
    parser.add_argument('--output', type=str, default='output', help='output folder')  # output folder
    parser.add_argument('--img-size', type=int, default=512, help='inference size (pixels)')
    parser.add_argument('--conf-thres', type=float, default=0.3, help='object confidence threshold')
    parser.add_argument('--iou-thres', type=float, default=0.6, help='IOU threshold for NMS')
    parser.add_argument('--fourcc', type=str, default='mp4v', help='output video codec (verify ffmpeg support)')
    parser.add_argument('--device', default='', help='device id (i.e. 0 or 0,1) or cpu')
    parser.add_argument('--view-img', action='store_true', help='display results')
    parser.add_argument('--save-txt', action='store_true', help='save results to *.txt')
    parser.add_argument('--classes', nargs='+', type=int, help='filter by class')
```

Figure 33: YOLO and DeepSORT “detect.py” continued

```
    parser.add_argument('--agnostic-nms', action='store_true', help='class-agnostic NMS')
    parser.add_argument('--augment', action='store_true', help='augmented inference')
    parser.add_argument('--quantized', type=int, default=0,
                        help='0:quantization way one Ternarized weight and 8bit activation')
    parser.add_argument('--a-bit', type=int, default=8,
                        help='a-bit')
    parser.add_argument('--w-bit', type=int, default=8,
                        help='w-bit')
    parser.add_argument('--FPGA', action='store_true', help='FPGA')
    parser.add_argument('--quantizer_output', type=bool, default=False, help='output')
    opt = parser.parse_args()
    opt.cfg = list(glob.iglob('./**/' + opt.cfg, recursive=True))[0]  # find file
    opt.names = list(glob.iglob('./**/' + opt.names, recursive=True))[0]  # find file
    print(opt)

    with torch.no_grad():
        detect()

if opt.quantizer_output == True:

    path = './quantier_output/q_bias_out'
    i = 1
    for file in os.listdir(path):
        if os.path.isfile(os.path.join(path, file)) == True:
            new_name = file.replace(file, "q_bias-modulelist_Conv2d_%d.txt" % (76 - i))
            os.rename(os.path.join(path, file), os.path.join(path, new_name))
            i += 1

    path = './quantier_output/q_weight_out'
    i = 1
    for file in os.listdir(path):
        if os.path.isfile(os.path.join(path, file)) == True:
            new_name = file.replace(file, "q_weight-modulelist_Conv2d_%d.txt" % (76 - i))
            os.rename(os.path.join(path, file), os.path.join(path, new_name))
            i += 1

    path = './quantier_output/q_activation_out'
    i = 1
    for file in os.listdir(path):
        if os.path.isfile(os.path.join(path, file)) == True:
            new_name = file.replace(file, "q_activation-modulelist_Conv2d_%d.txt" % (76 - i))
            os.rename(os.path.join(path, file), os.path.join(path, new_name))
            i += 1

    path = './quantier_output/b_scale_out'
    i = 1
    for file in os.listdir(path):
        if os.path.isfile(os.path.join(path, file)) == True:
            new_name = file.replace(file, "scale_bias-modulelist_Conv2d_%d.txt" % (76 - i))
            os.rename(os.path.join(path, file), os.path.join(path, new_name))
            i += 1

    path = './quantier_output/w_scale_out'
    i = 1
    for file in os.listdir(path):
        if os.path.isfile(os.path.join(path, file)) == True:
            new_name = file.replace(file, "scale_weight-modulelist_Conv2d_%d.txt" % (76 - i))
            os.rename(os.path.join(path, file), os.path.join(path, new_name))
            i += 1

    path = './quantier_output/a_scale_out'
    i = 1
    for file in os.listdir(path):
        if os.path.isfile(os.path.join(path, file)) == True:
            new_name = file.replace(file, "scale_activation-modulelist_Conv2d_%d.txt" % (76 - i))
            os.rename(os.path.join(path, file), os.path.join(path, new_name))
            i += 1

    ################输出每一层量化后的最大权值
    path = './quantier_output/q_weight_max'
    i = 1
    for file in os.listdir(path):
```

Figure 34: YOLO and DeepSORT “detect.py” continued

```
        if os.path.isfile(os.path.join(path, file)) == True:
            new_name = file.replace(file, "max_weight-modulelist_Conv2d_%d.txt" % (76 - i))
            os.rename(os.path.join(path, file), os.path.join(path, new_name))
            file = open(os.path.join(path, new_name), "r", encoding="utf-8")
            mystrl = file.readline()  # 表示一次读取一行
            file_max = open('./quantier_output/q_weight_max/q_weight_max.txt', "a", encoding="utf-8")
            file_max.write(mystrl[:-1] + '\n')
            file_max.close()
            file.close()
            i += 1
    ################输出每一层量化后的最大激活
    path = './quantier_output/q_activation_max'
    i = 1
    for file in os.listdir(path):
        if os.path.isfile(os.path.join(path, file)) == True:
            new_name = file.replace(file, "max_activation-modulelist_Conv2d_%d.txt" % (76 - i))
            os.rename(os.path.join(path, file), os.path.join(path, new_name))
            # 合并最大值文档
            file = open(os.path.join(path, new_name), "r", encoding="utf-8", errors="ignore")
            mystrl = file.readline()  # 表示一次读取一行
            file_max = open('./quantier_output/q_activation_max/q_activation_max.txt', "a", encoding="utf-8",
                            errors="ignore")
            file_max.write(mystrl[:-1] + '\n')
            file_max.close()
            file.close()
            i += 1

    #########从这一行开始合并count文件
    path = './quantier_output/max_weight_count'
    i = 1
    for file in os.listdir(path):
        if os.path.isfile(os.path.join(path, file)) == True:
            new_name = file.replace(file, "max_weight_count-modulelist_Conv2d_%d.txt" % (76 - i))
            os.rename(os.path.join(path, file), os.path.join(path, new_name))
            file = open(os.path.join(path, new_name), "r", encoding="utf-8")
            mystrl = file.readline()  # 表示一次读取一行
            file_max = open('./quantier_output/max_weight_count/max_weight_count.txt', "a", encoding="utf-8")
            file_max.write(mystrl[:-1] + '\n')
            file_max.close()
            file.close()
            i += 1

    path = './quantier_output/max_activation_count'
    i = 1
    for file in os.listdir(path):
        if os.path.isfile(os.path.join(path, file)) == True:
            new_name = file.replace(file, "max_activation_count-modulelist_Conv2d_%d.txt" % (76 - i))
            os.rename(os.path.join(path, file), os.path.join(path, new_name))
            file = open(os.path.join(path, new_name), "r", encoding="utf-8", errors="ignore")
            mystrl = file.readline()  # 表示一次读取一行
            file_max = open('./quantier_output/max_activation_count/max_activation_count.txt', "a", encoding="utf-8",
                            errors="ignore")
            file_max.write(mystrl[:-1] + '\n')
            file_max.close()
            file.close()
            i += 1
```

Figure 35: YOLO and DeepSORT “detect.py” continued