**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.

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 |

**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** 4

**2 Project Overview / Statement of Work Summary** 5

**3 Design Narrative** 6

**4 Test Results** 8

**5 Evaluation of Prototype/ Model/ System as Compared to Project Performance and Requirements Document** 8

**6 Recommendations for Further Development** 9

**7 Impact** 9

**8 Bill of Materials (BOM)** 10

**9 Budget** 10

**10 Conclusions** 11

**11 References** 11

**12 Appendices** 12

# Overview of this Document

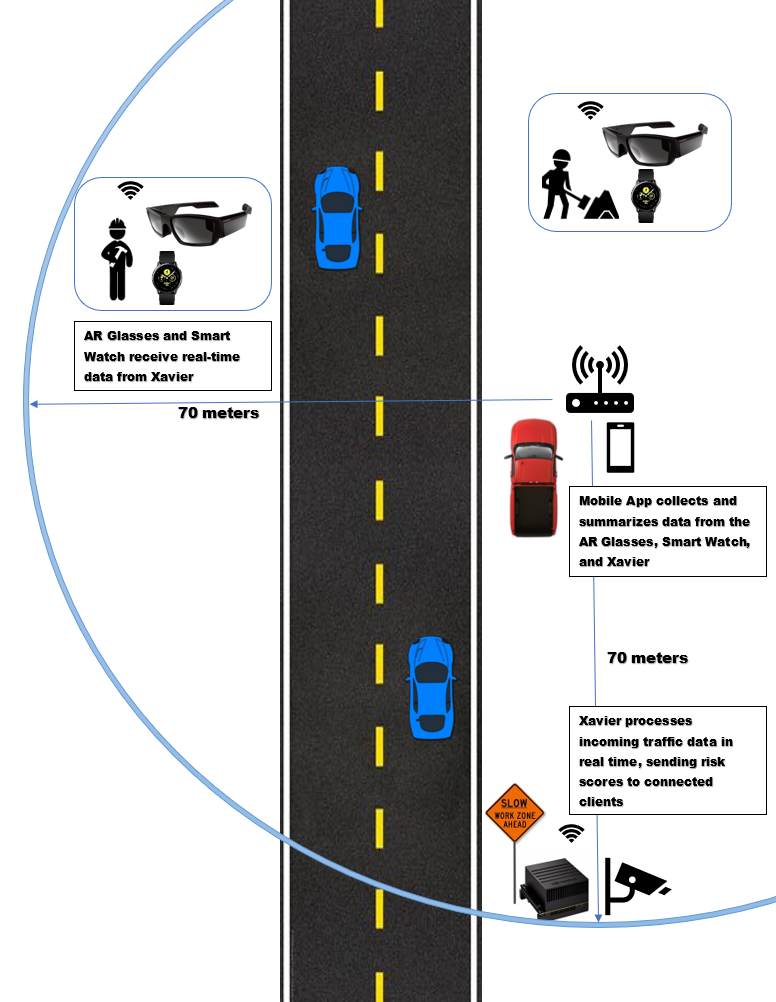
This document describes the design of the UNCC\_WORK project and of its end-product for Senior Design. This project is to develop a scalable backend communication system which will send warning messages to multiple clients in a work zone based on data received from a camera that will detect, track, and 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

# 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.

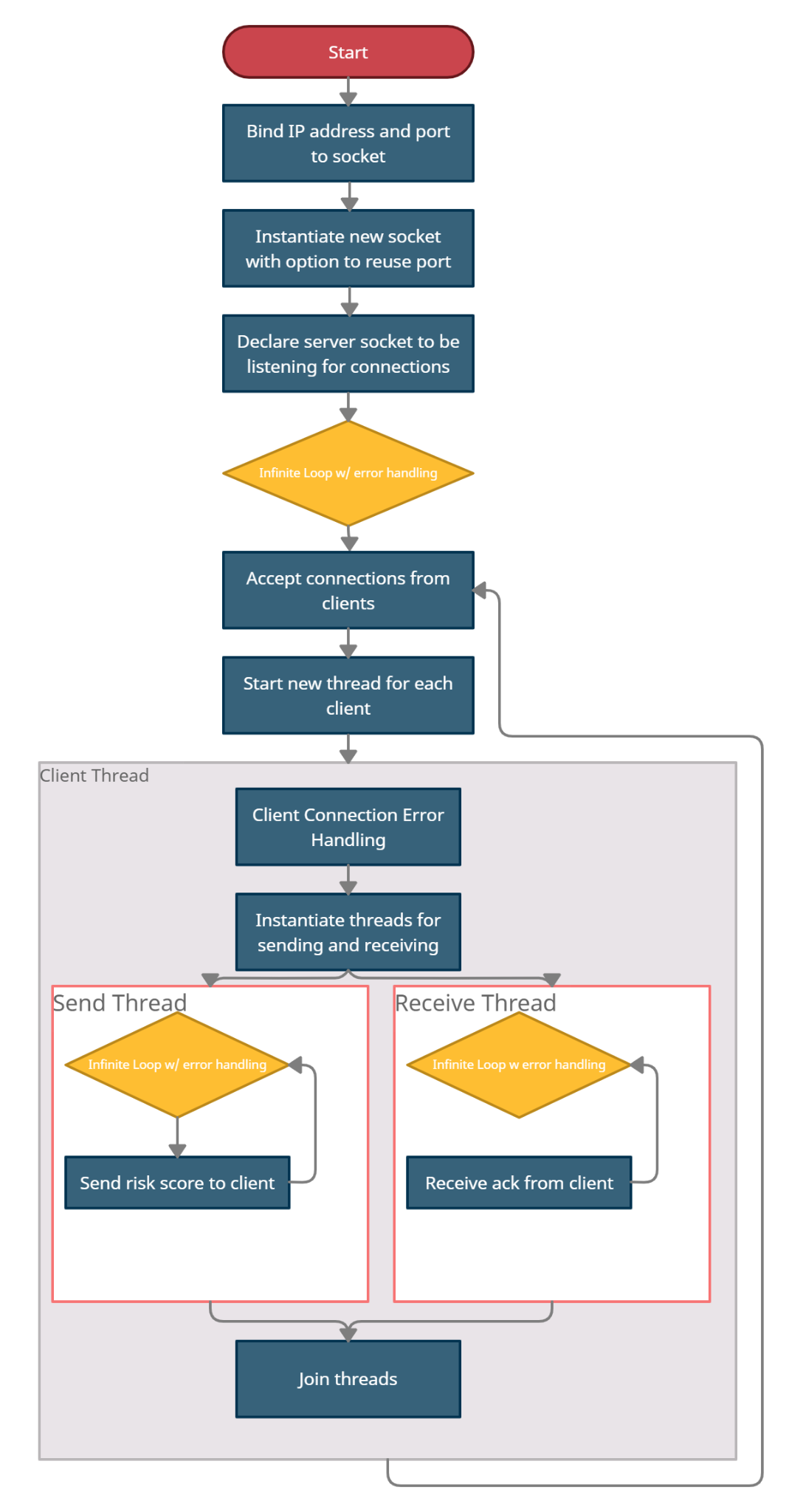


**Figure 1: System Overhead**

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.

# 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.

# 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.

# 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.

# 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.

# 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.

# Bill of Materials (BOM)



**Figure 31: Bill of Materials**

# 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.

# 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.

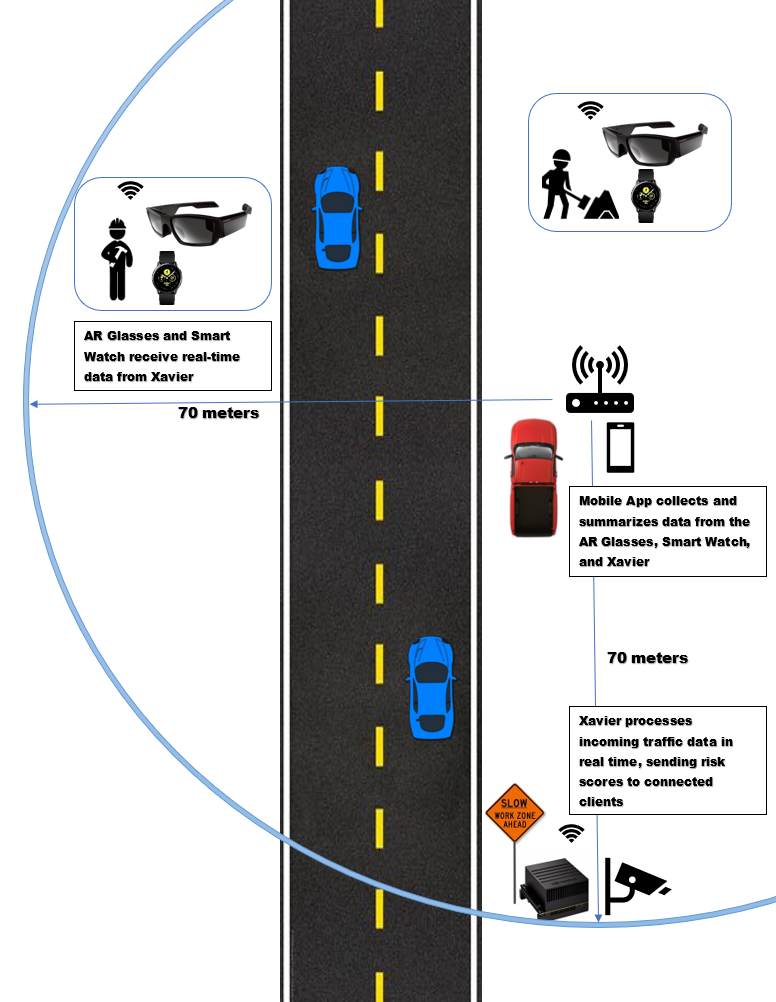
The development of the backend communication scheme through a multithreaded server was completed over the course of the year and the team gained significant insights into all of the various facets, challenges, and implementation specifics associated with using a socket based server in Python.

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.

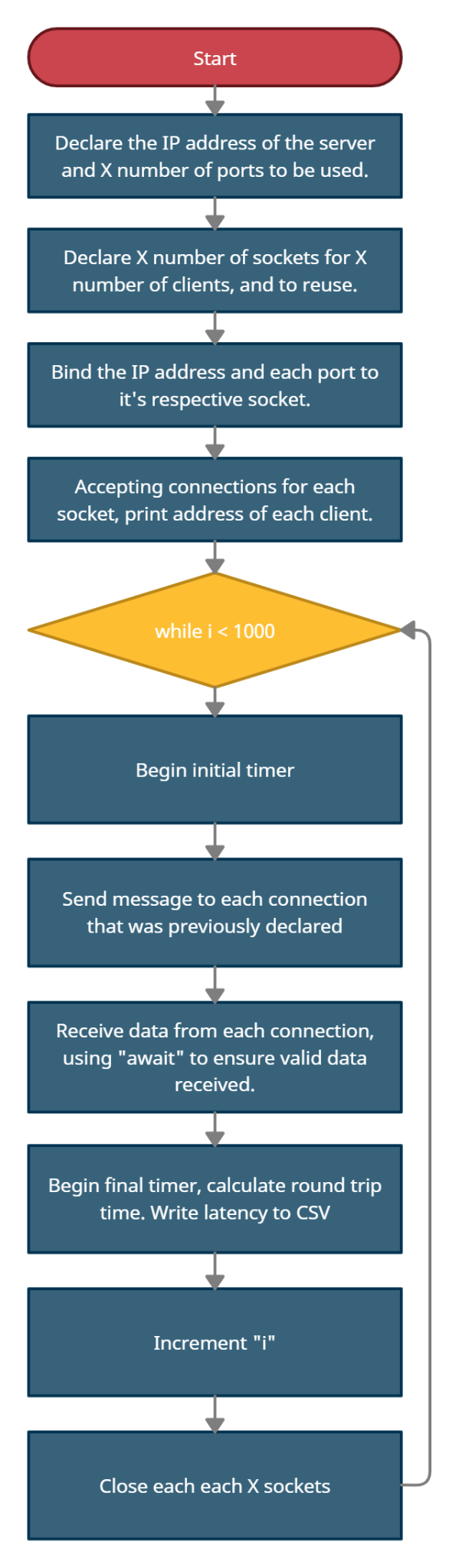
# References

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.

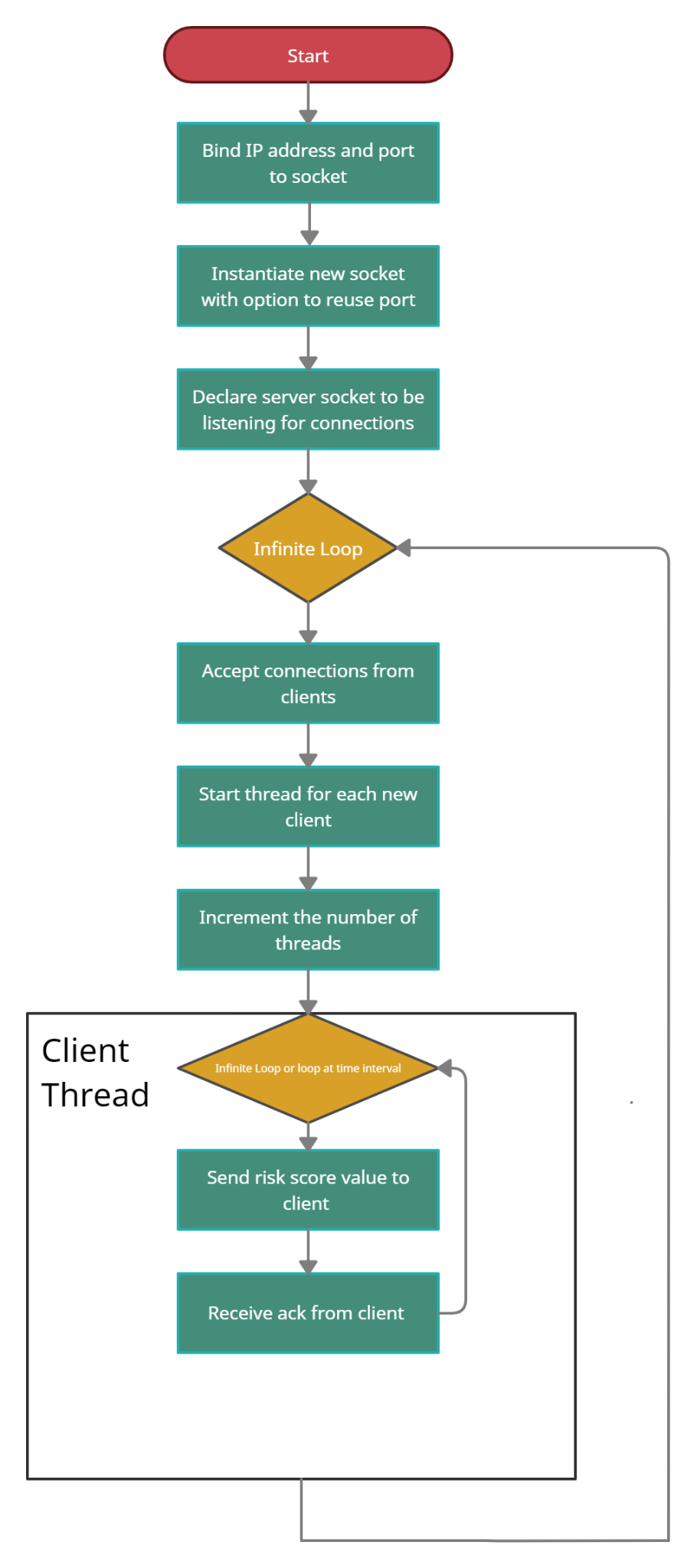
# Appendices



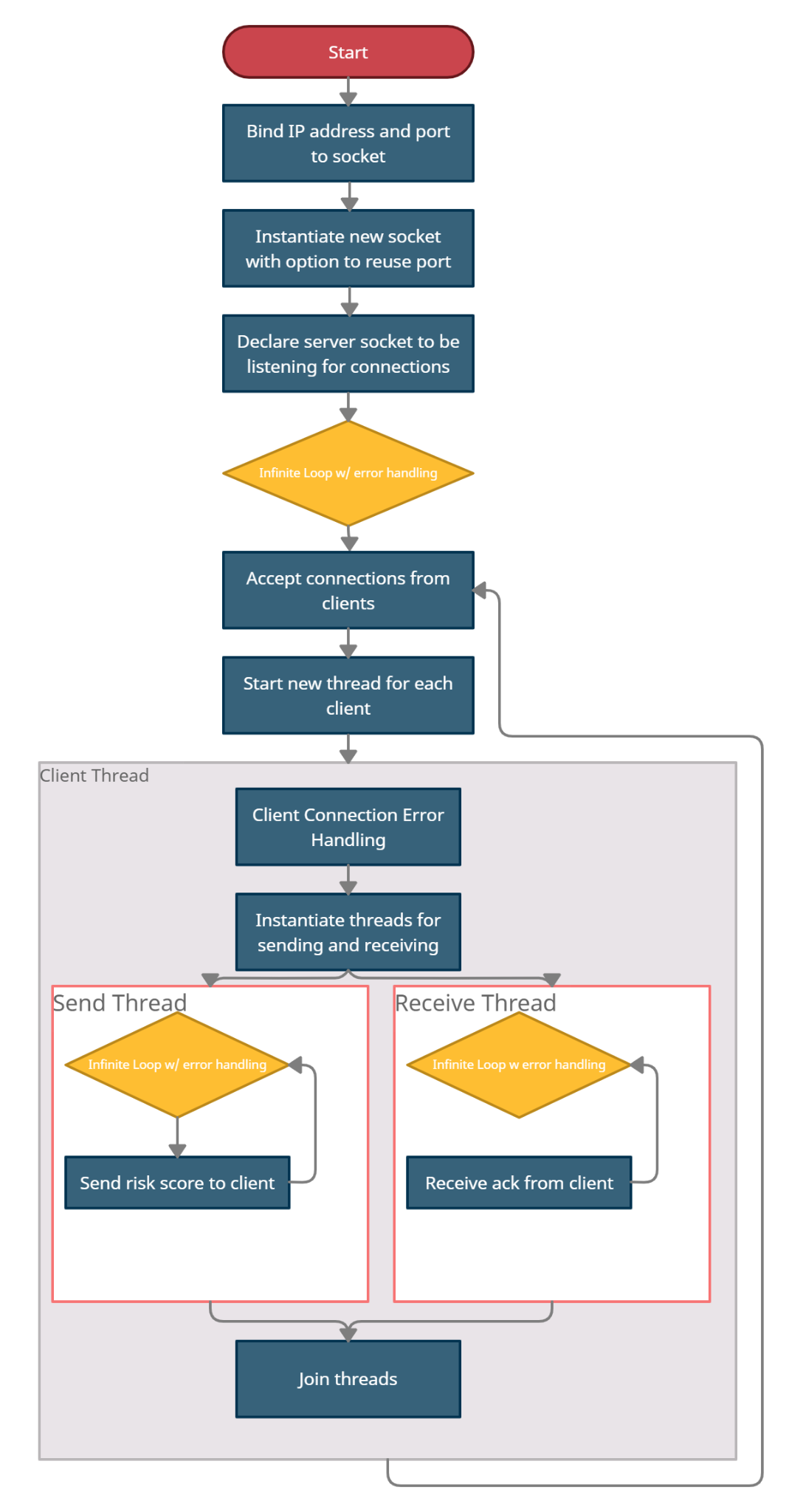
**Figure 1: System Overhead**

****

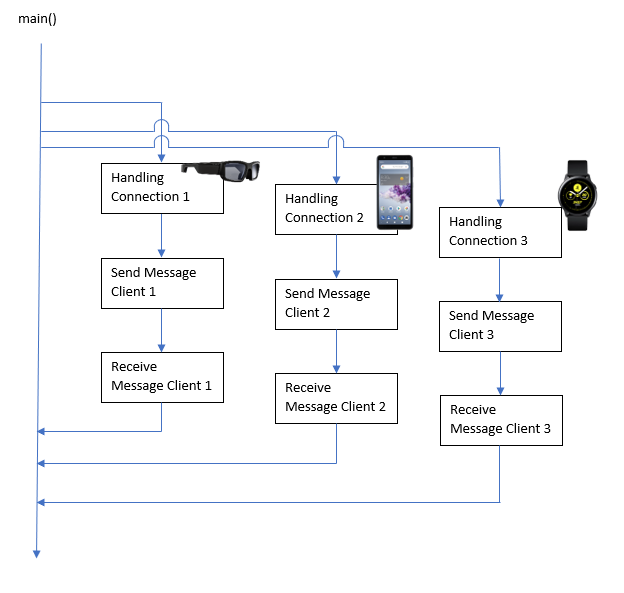
**Figure 2: Single-Threaded Server Flow**

****

**Figure 3: Initial Multithreaded Server Flow**

****

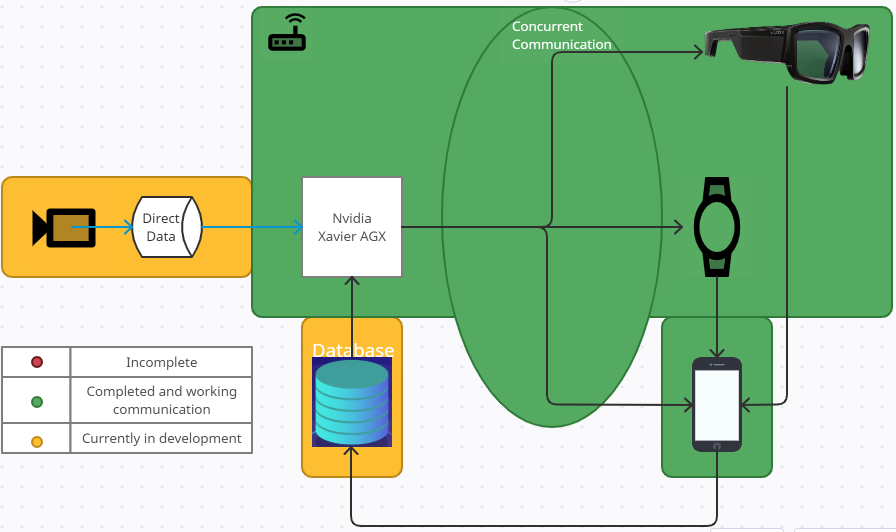
**Figure 4: Final Multithreaded Server Flow**

****

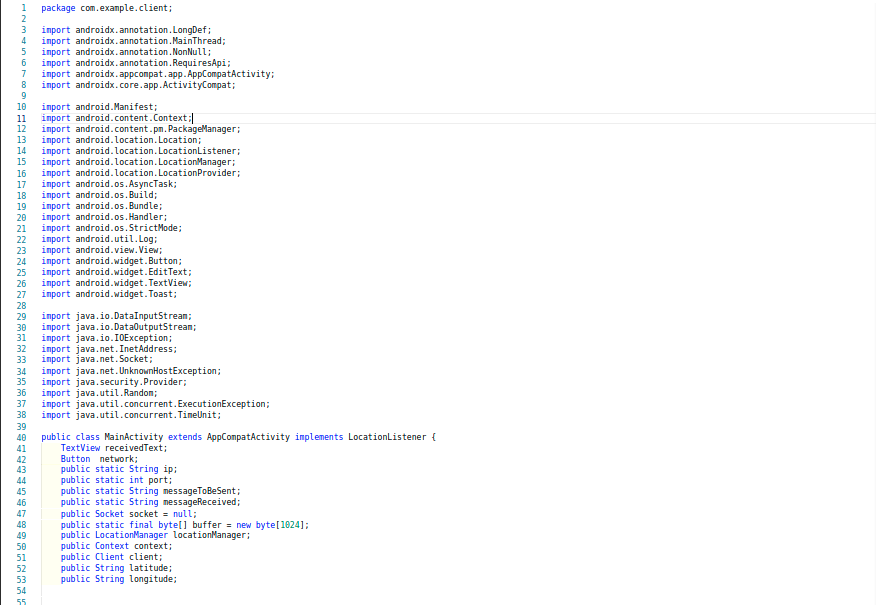
**Figure 5: Multithreaded Server Example**

****

**Figure 6: Communication Flow**

****

**Figure 7: Project Progress**

****

**Figure 8: Client Application Main Activity Source Code**

****

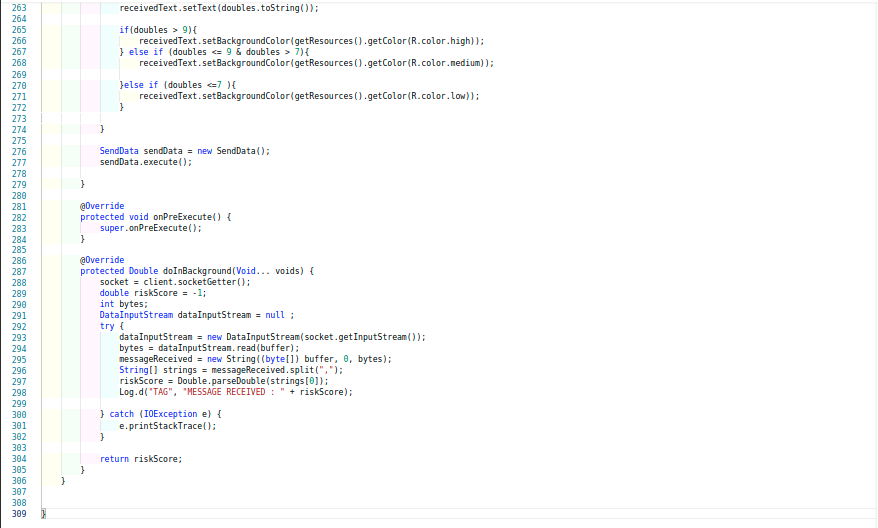
**Figure 9: Client Application Main Activity Source Code Continued**

****

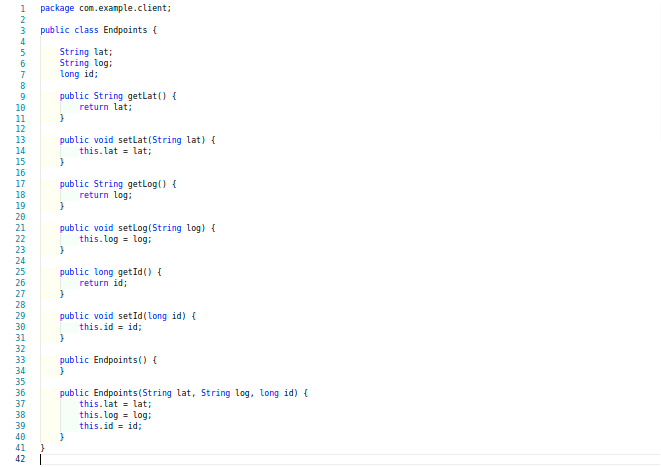
**Figure 10: Client Application Main Activity Source Code Continued**

****

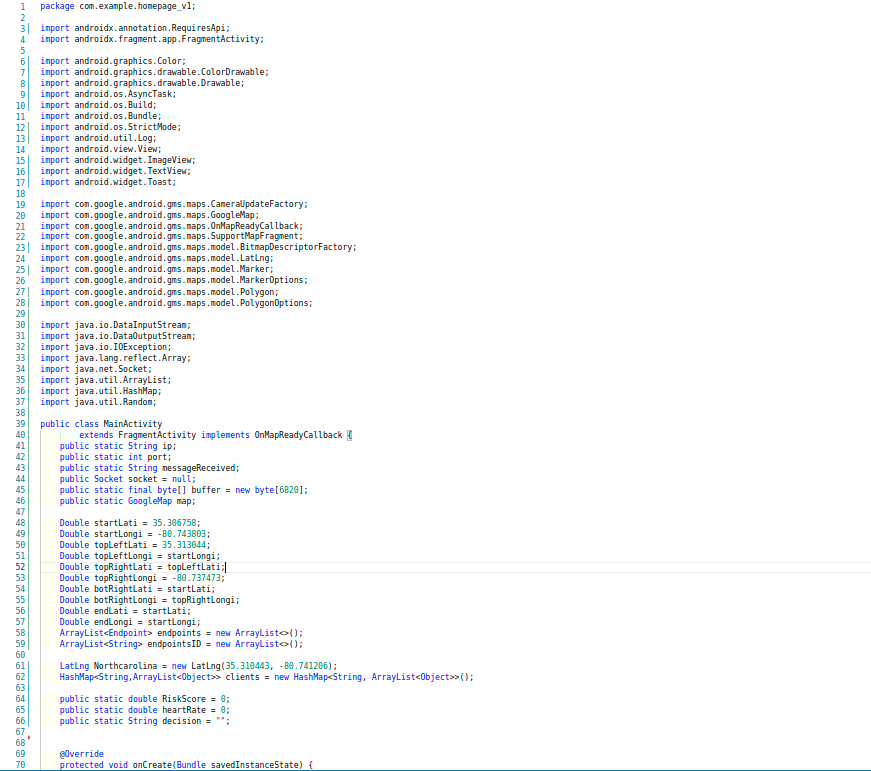
**Figure 11: Client Application Main Activity Source Code Continued**

****

**Figure 12: Client Application Main Activity Source Code Continued**

****

**Figure 13: Client Application “Endpoints” Class**

****

**Figure 14: Digital Twin Application Main Activity Source Code**

****

**Figure 15: Digital Twin Application Main Activity Source Code Continued**

****

**Figure 16: Digital Twin Application Main Activity Source Code Continued**

****

**Figure 17: Digital Twin Application Main Activity Source Code Continued**

****

**Figure 18: Digital Twin Application Main Activity Source Code Continued**

****

**Figure 19: Digital Twin Application Main Activity Source Code Continued**

****

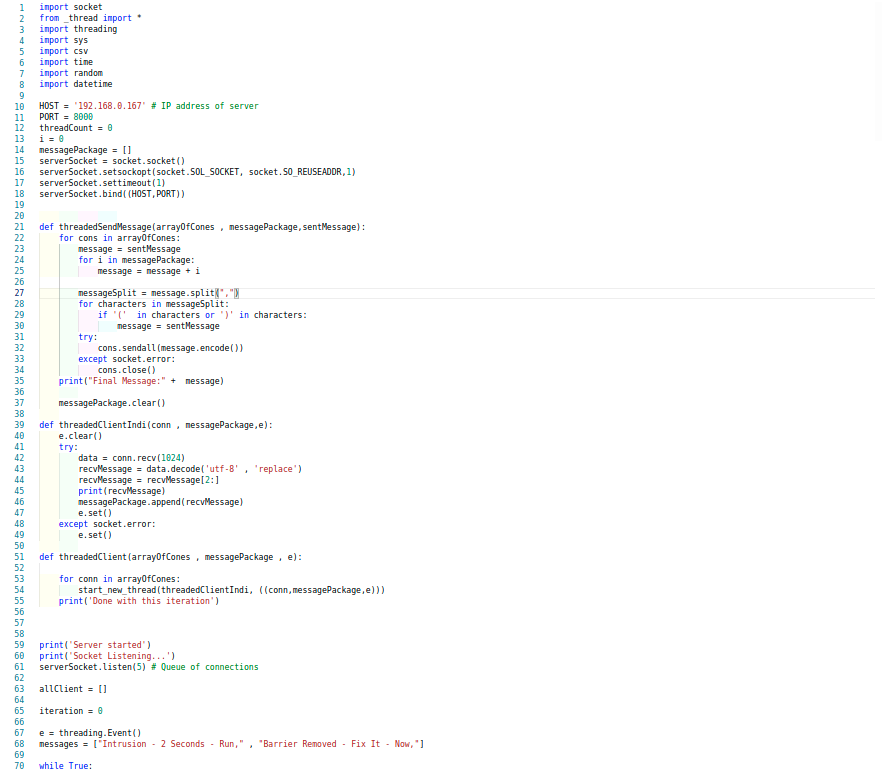
**Figure 20: Digital Twin Application Main Activity Source Code**

****

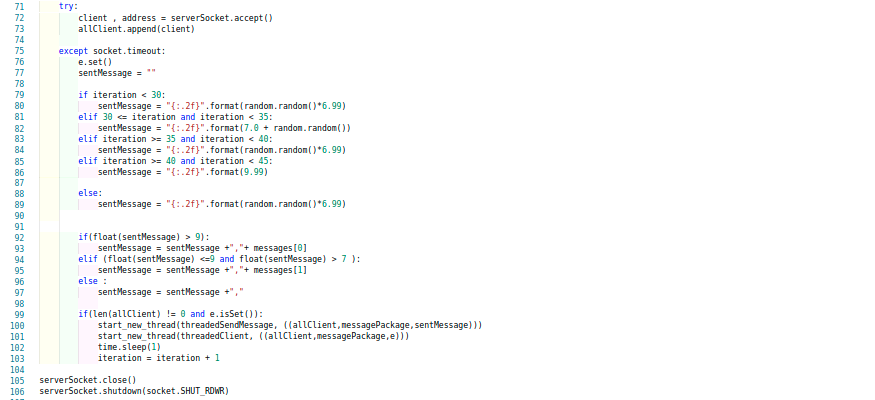
**Figure 21: Digital Twin Application Login Activity Source Code**

****

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

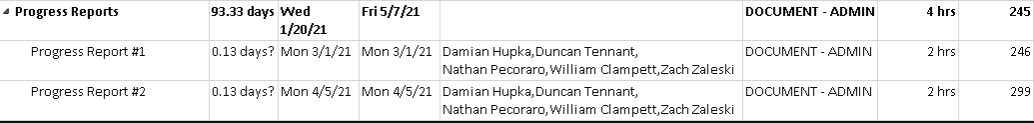
****

**Figure 23: Multithreaded Socket Server Source Code**

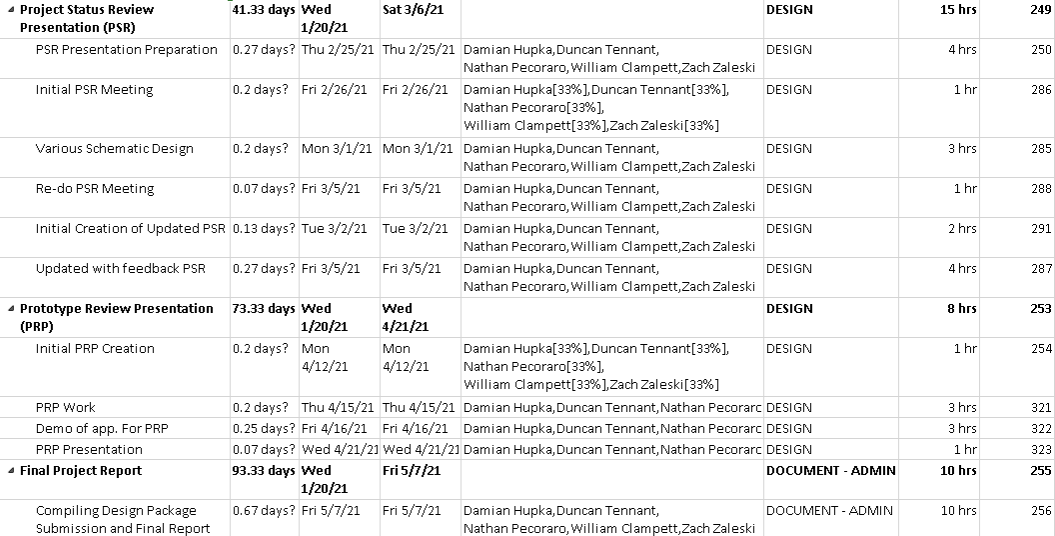
****

**Figure 24: Multithreaded Socket Server Source Code Continued**

****

****

**Figure 25: Project Plan**

****

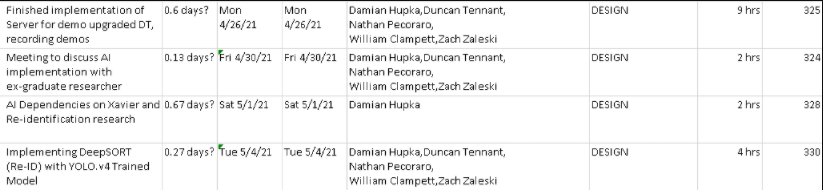
**Figure 26: Project Plan Continued**

****

**Figure 27: Project Plan Continued**

****

**Figure 28: Project Plan Continued**

****

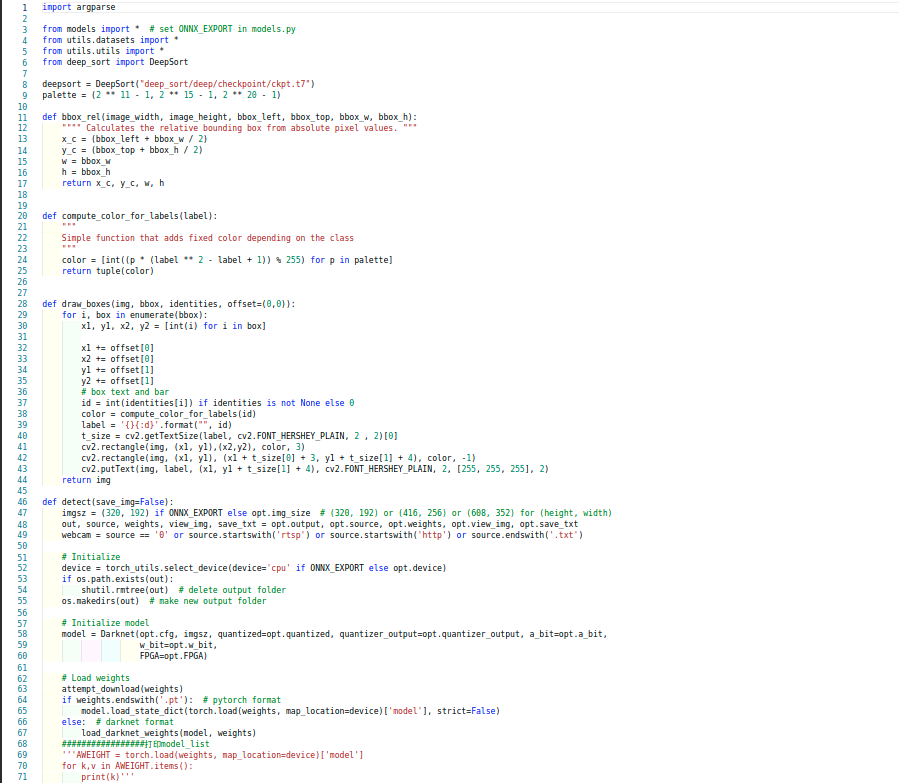
**Figure 29: Project Plan Continued**

****

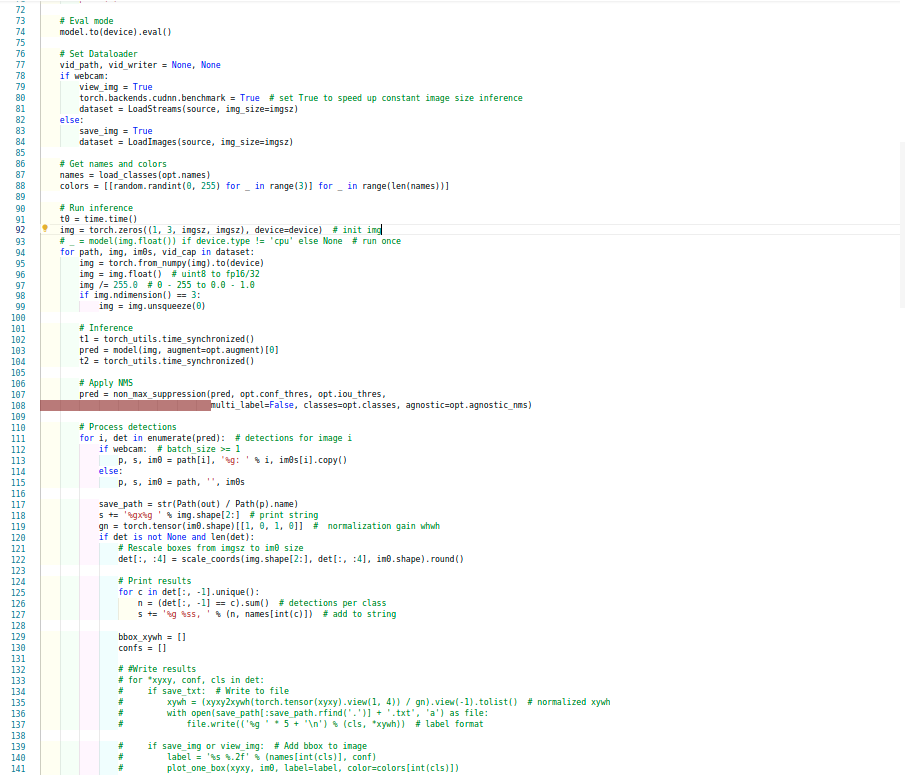
**Figure 30: Project Plan Continued**

****

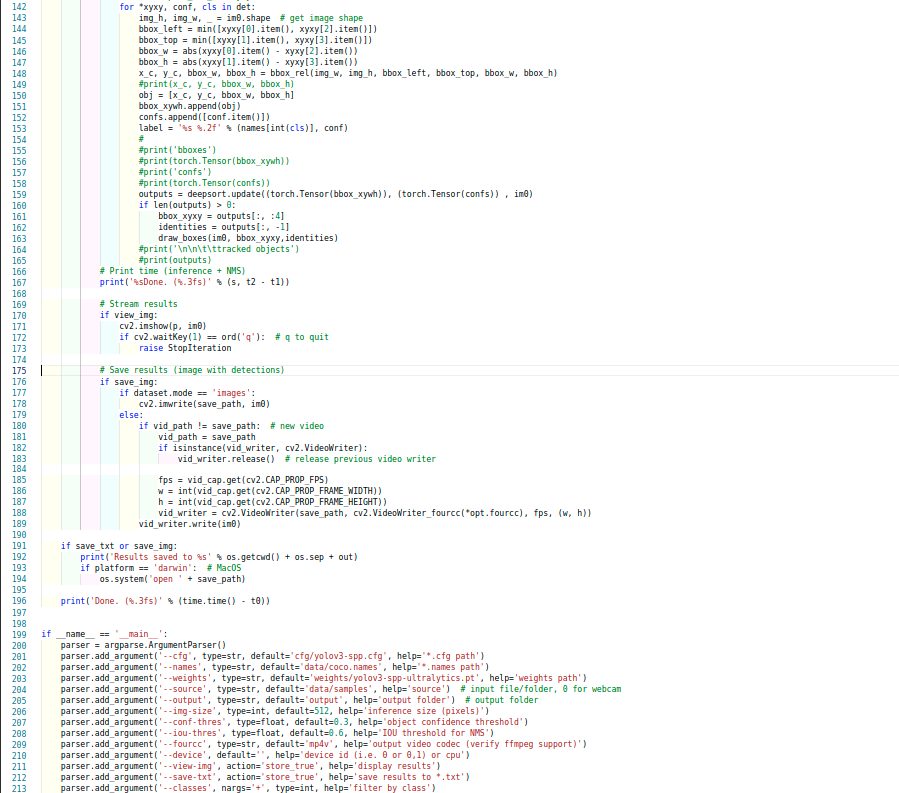
**Figure 31: Bill of Materials**

****

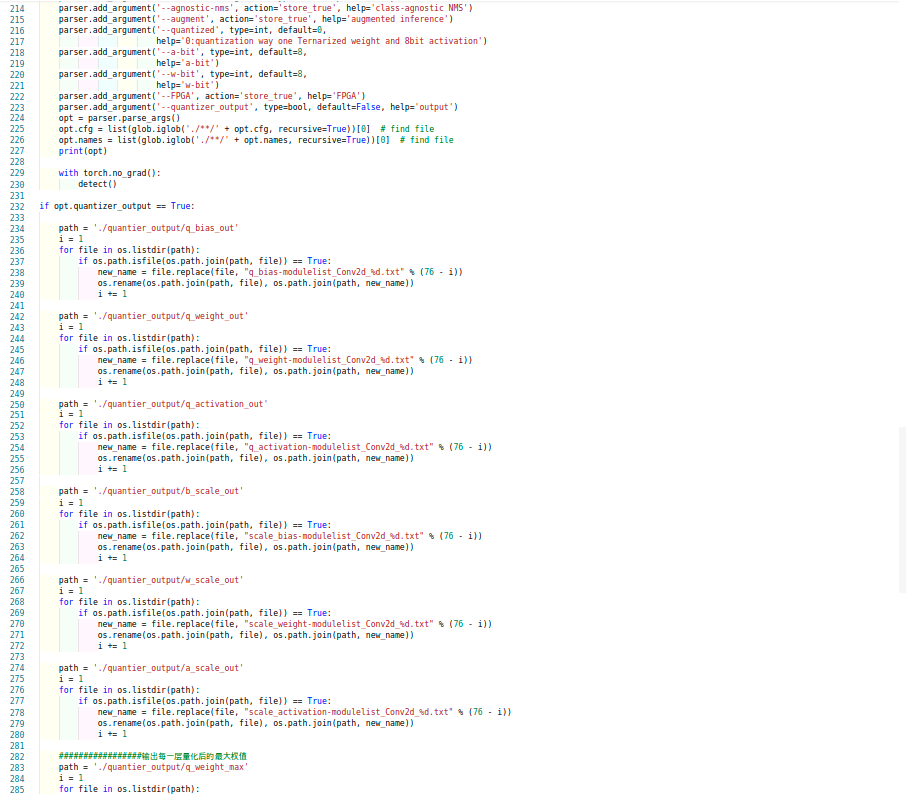
**Figure 32: YOLO and DeepSORT “detect.py”**

****

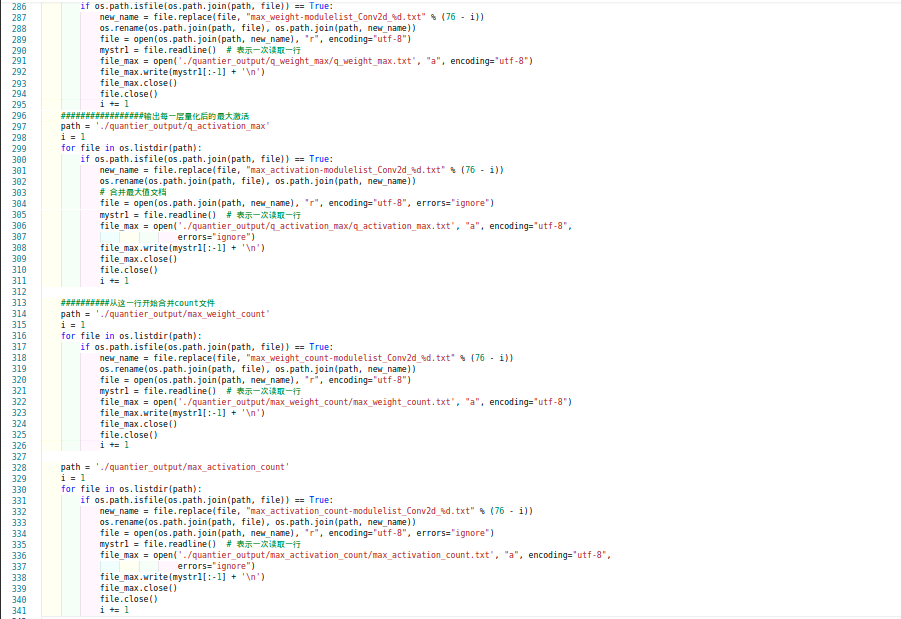
**Figure 32: YOLO and DeepSORT “detect.py” continued**

****

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

****

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

****

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