Bilkent University

Department of Computer Engineering

Senior Design Project

Final Report

VANNY

Project Members

A A M Jubaeid Hasan Chowdhury - 21503356

Abdul Razak Daher Khatib - 21801340

Esad Ismail Tök - 21801679

Elifnur Alsaç - 21601098

Mohamed Wasim Mohamed Akram - 21801103

Supervisor

Can Alkan

Jury Members

Erhan Dolak

Tağmaç Topal

Innovation Expert

Faik Berk Güler

Table of Contents

1. Introduction

This report is the High-Level Design report of the Vanny project. In this report, the aim and design goals of the project will be mentioned. Afterward, the proposed software architecture will be mentioned. In this section, the subjects of the project such as subsystem decomposition, hardware/software mapping, security and control issues, and boundary conditions will be discussed. Following this, subsystem services and its layers will be explained. Afterward, the test cases of the project will be included. After various engineering design considerations, the dynamics and roles of teamwork will be discussed under the title of Teamwork Details.

Babies and toddlers are the main focus, care, and concern of all parents. From providing the best food and medical care to paying babysitters to set up a nanny cam, parents will stop at nothing to make sure their babies and toddlers are healthy and safe. Nanny cams specifically play an increasingly important role in helping parents extend their love and care beyond their immediate physical vicinity, allowing the parents to have a reliable and transparent tool to watch their loved babies and toddlers. The commonly available nanny cams, however, can be optimized to push this reliability and transparency further if the newest and most capable technological tools were integrated in said systems. Furthermore, by making these systems more intelligent and able to communicate with the parents the current nanny cam systems can be extended and transformed into an additional caring eye that is always present in the babies' and toddlers' bedrooms or playrooms.

After conducting market research we observed that most of the current nanny cam systems don't utilize some very useful Artificial Intelligence and Machine Learning algorithms efficiently. Alfred Home Security Camera, for example, is the most popular similar system on Apple's AppStore with over 40 million users, detects intruders by detecting "any movement", with no other AI or ML capabilities utilized. It can also live stream, see in low-light environments, and act as a walkie-talkie, among other common features in security cams[1].

Another example "Annie Baby Monitor" acts as a basic nanny cam with motion detection and the ability to play lullabies[2]. Similarly, other examples of such systems have motion detection with other AI capabilities like detecting noise and crying and notifying the parent that the toddler is awake[3][4][5]. The most capable similar system is the "Mi Home Security Camera". The aforementioned system is a general security camera system. Moreover, it has an angle of 130 degrees, has infrared and night vision scopes, and acts as a two-way voice communication system. In terms of AI capabilities, it has a zoning functionality that makes the system more sensitive to movements in these zones and can detect family members in the camera's range, using Mi Bands or iPhones, and turn them off when
they are in the range[6].

Finally, other specialized systems that focus strictly on a specific problem exist. "Angel Eyes", as an example, watches the baby while sleeping in the crib. The system uses a camera to watch the baby and detects any issues, like objects in the crib, or if the baby is in an unsafe position, and uses sensors to monitor humidity and temperature in the room[7]. Another system detects the hunger of the baby using some cues[8].

A more optimized system will act as a general purpose more capable nanny cam that fits somewhere between “Mi Home Security Camera” and “Angel Eyes”, and is more capable than the other aforementioned nanny cams by utilizing the capabilities of AI and ML and focus on the toddler in their rooms. Ideally, this system will transform nanny cams by pushing them to the limits of possible capabilities. This document describes the specifications of a system that optimizes and transforms these systems into a virtual babysitter. Note that the age group of children that are the target of the system is 0 to 5 years old, the document will refer to this age group as toddlers.

2. Requirements Details

2.1 Functional Requirements

2.1.1 Detecting the Environment of the Toddler

- The system is able to monitor the baby or toddler 24/7 and enables the parents of those toddlers to observe if their baby is in confort or not.
- The application is able to detect the baby or toddler as well as any unfamiliar person, namely not his/her parent.
- The system can detect any moving object to present the baby from possible dangers and warn the parents of the baby if there is a need.
- In case of the toddler falling down, the system warns the parents by detecting the baby.
- Any sharp object that might possibly cause a danger for the toddler is detected by the system and the necessary warnings are displayed to the parents.
- The system can detect the open doors and windows to prevent the baby escaping from those and warns the parents of the toddler.
- The system can also detect any fire or smoke that can constitute a danger for the baby together with any source of light that can bother the baby and cause uncomfort for him/her.
- Facial expressions of the baby such as crying can also be detected to make the necessary analyses.

2.1.2 Detecting Noise in the Environment of the Toddler

- The system can detect any loud noises in the environment to ensure the comfort of the toddler and also to see if the toddler is crying and has any uncomfort.

2.1.3 Sending Voice to the Toddler

- In case the parents want to send any voice to the toddler, the system provides two opportunities to the parent.
- First of all the parent can record his/her voice to send to the toddler. It is important for the baby to feel safe since a familiar voice is comforting for the babies that are in discomfort.
- The parents of the baby can also send a lullaby to the baby that can be selected in the application’s lullaby list. It would calm the baby making him/her more comfortable.

2.1.4 Notifying the Parents Based on the Analysis

- The system is designed so that in case of any situations that can constitute any danger to the toddler. In case any hint is observed and analyzed that it can cause an uncomfort for the toddler, the parents are notified and warned accordingly regarding the situation of the toddler.
- To notify the parents, the short captured videos of the situations that should be known by the parents are sent to the parents specifically to prevent them from missing those records.
- The parents are also notified in case of a drastic change in the activity patterns of the toddler or the crying pattern and duration of the toddler.

2.2 Nonfunctional Requirements

2.2.1 Security

- Since the data used in the application contains sensitive information, it needs to be secure in order to prevent any outside access .

2.2.2 Reliability

- The application will be to detect unusual activities without any anomalies.
- The application will perform well-being tests to report any partial failures.

2.2.3 Effectiveness

- The system will detect various types of possible dangers and report them to parents within 5 seconds, protecting the toddler continuously.

2.2.4 Memory Optimization

- Recording of the last 3 days will be saved locally in case the parents need them.
- Recordings that are older than 3 days will not be saved except when unusual activity or issues are detected. The recorded moments will be stored for only 7 days to increase the storage efficiency.

2.2.5 Privacy

- User data will be protected by encryption between the server and the client in the system.
- Recordings will be private with access from any third-party or individuals.
- Users can disable the recording whenever desired.
- Users can delete any form of data saved.

2.2.6 Reporting

- Any suspicious activity will be reported to the user immediately and saved for 7 days.
- Analysis of activity, sleep, and crying will be reported regularly to the parents.

2.2.7 Extensibility

- For the application to be extensible, its implementation will be modular and systematic. Thus, the application will be compatible with updates and features to be added or removed.

2.2.8 Robustness

- If a failure of the application occurs, the application will be able to restart automatically and continue from where it failed without losing information.
- The system will perform well-being tests and confirm that all components work and try to resolve any connectivity issues.

2.2.9 Usability

- The system will require a one-time setup afterwards users can use the system without directly interacting with the server side.
- The UI of the phone application will be designed so it is used by any user regardless of their tech knowledge.

2.2.10 Efficiency

- The system will be able to watch a frame-per-second rate of at least 10.
- The system detects any issues and reports in real-time therefore the notifications sent to the user should arrive within 5 seconds.
- The system will delete full day recordings in 3 days and highlights in 7 days, enabling efficient memory management.

2.2.11 Ethicality

- Parents will be made aware of what the system records and for how long.
- No third-party activity of any kind will have access to the data stored.
- Reports generated will only be shared with parents.
- Encryption and predefined password will be implemented to prevent any undesired access and undermining toddlers’ safety or privacy.

2.3 Pseudo Requirements

- The accuracy of the trained models is an important factor of the application. Therefore the accuracy of the trained models needs to be considered carefully.
- The application will be a mobile application and therefore the client side of the application needs to be carefully implemented to be in the minimum storage requirements for the users.
- If there will be any need to use a third-party API or a library then the licence agreements of those third-parties will be considered and applied if necessary.
- The system should enable the user to securely pair his/her mobile device with the system.
- The user should have access to manage all kinds of his/her recorded data.
- The user should be able to enable or disable the camera anytime he/she wants.

3. Final Architecture and Design Details

In the below class diagrams the system's architecture is shown. The system consists of the major packages, these three packages represent the main components of the system based on functionality and hardware. These three packages are also subsystems in our system. In a way, Vanny represents a standard client-server architecture, as will be shown below, with the Raspberry Pi being the server, and the Android phone being the client.

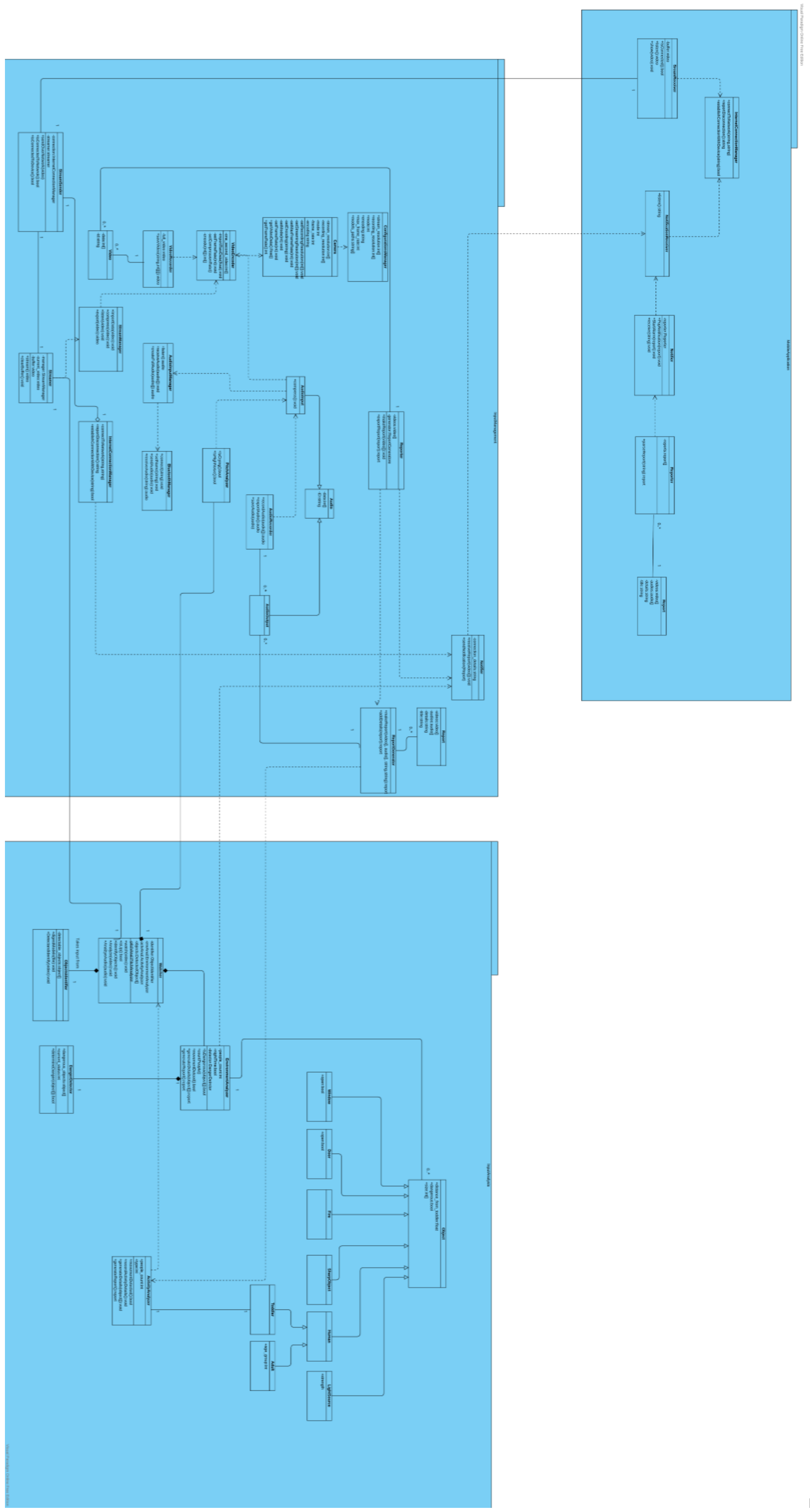

Figure 1: UML Class diagram.

Input Management Subsystem:

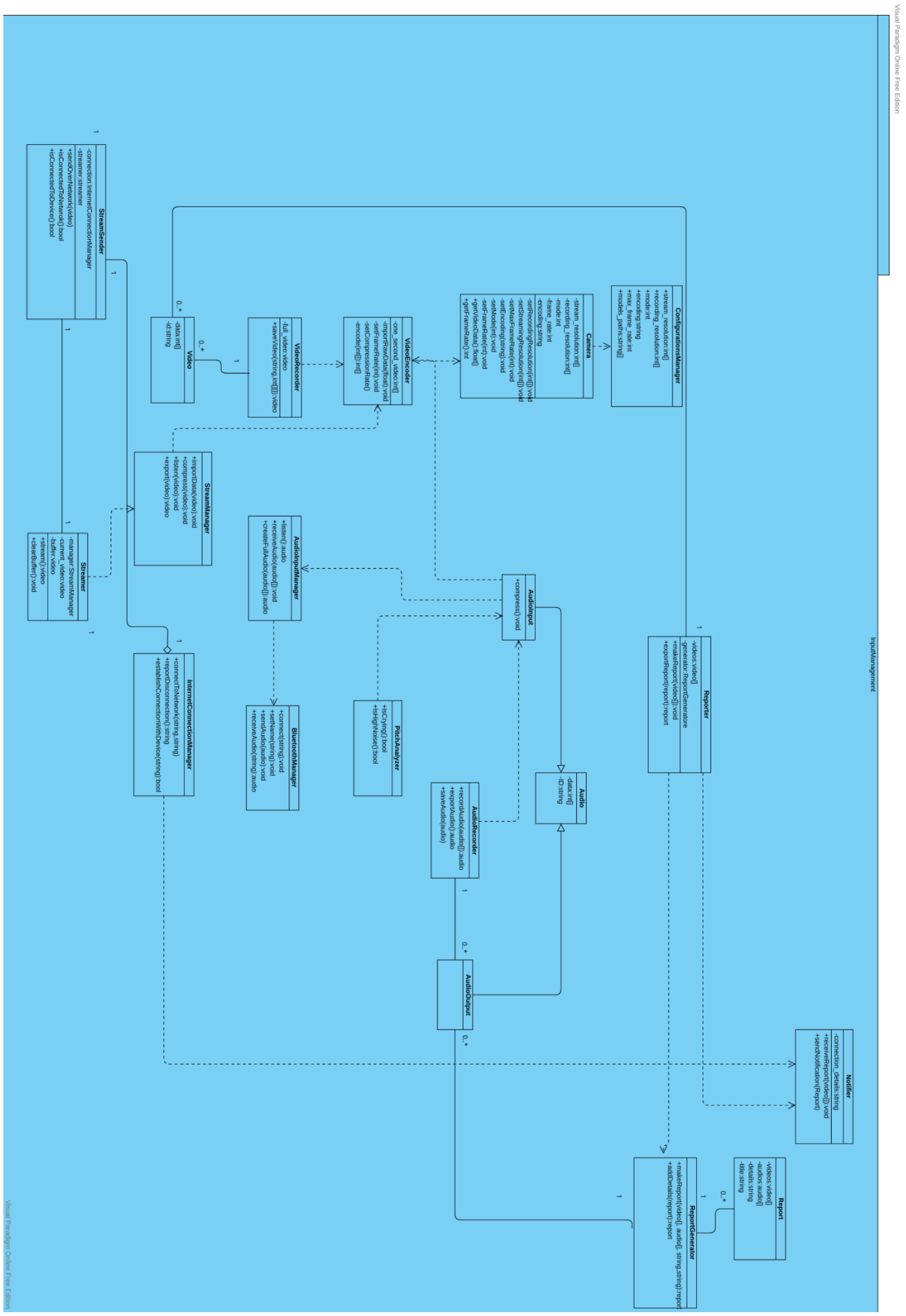

Figure 2: “Input Management” package.

The first sub-system is the Input Manager subsystem is the core subsystem in the system where the majority of the analysis and decision-making will be handled. In this subsystem, we have the Camera as the main source of input and provider of the data to be analyzed and the images to detect objects. The camera will use the configuration manager at the beginning of the launch of the system, these configurations include stream and recording resolution, the encoding of them, and frame rates, among other configurations. Another key component in this subsystem is the Stream Manager. The Stream Manager class will handle the process of streaming the live preview to the Android device. This component uses the streamer class to manage the internet connection and send the stream over the internet, as well as give live feedback on the detected dangers. The third main components in the system are the recorders, both for audio and video. In both of these components, the system will record, compress, encode, and store the recordings for later use. The fourth main component is the Reporter, the reporter component is responsible for generating the reports. This includes setting proper explanatory titles for reports, saving videos, and preparing weekly reports for activity analysis, sleep analysis, and for incidents recorder. Lastly, a small but critical part of this subsystem is the Notifier. The Notifier component is responsible for sending all types of notifications to the Android device, as well as sending the reports.

Input Analysis Subsystem:

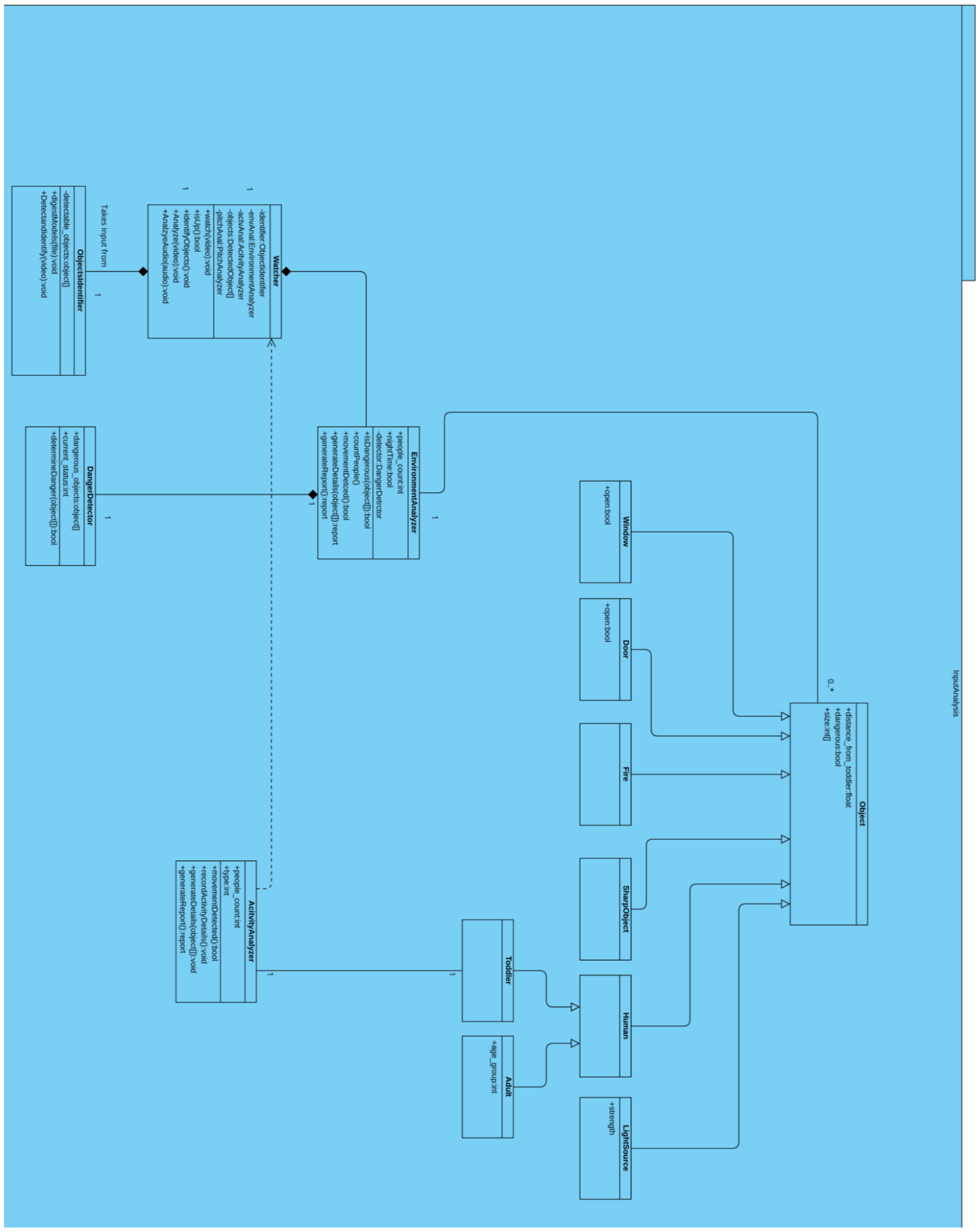

Figure 3: “Input Analysis” package.

The Second subsystem is the Input Analysis. While the Input Manager processes the input, the processing in that system is in the form of data management; no “Intelligent” decisions are made in that subsystem. In the Input Analysis, however, the ML and AI components are used to analyze the input obtained by the Input Manager subsystem.

The core of this subsystem is the Watcher components. This component uses the Object Identifier component, which is used for detecting all forms of objects in the area being watched, as well as digesting and loading the models that the system will detect, decisions will not be made here regarding whether there are any dangers around, or regarding the movement analysis, this component will solely identify the objects and notify, continuously the Watcher. Another component used by the Watcher is the Environment Analyzer. In this component, the major decisions are made regarding whether objects are dangerous, whether any movement is detected when there is supposed to be none, and keeping count of people in the area. It also keeps track of these elements for the report generator to collect and generate its reports. Activity Analyzer is one more component that the Watcher component uses. In this component, movement is recorded along with its details, like duration and time, and the report details are prepared here for the weekly reports that are generated by the reporter. Finally, with all of these components being used by the Watcher component, the Watcher component acts as a manager and an intermediary between all of these components and uses them to evaluate the situation in the toddler's environment and command the analyzer components to run the analysis on the feedback provided by the objects identifier.

Mobile Application Subsystem:

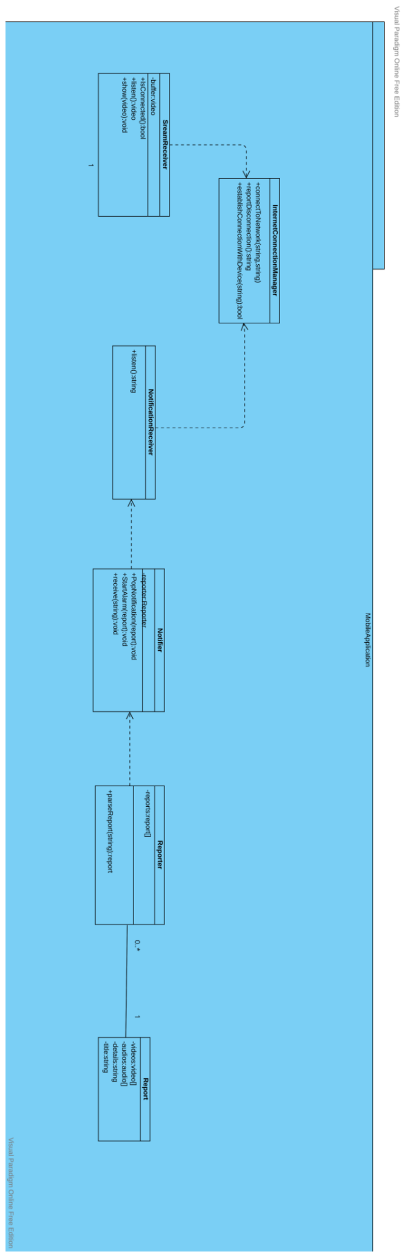

Figure 4: “Mobile Application” package.

The last subsystem is the Mobile Application subsystem. This subsystem is the simplest since the application will be used only for receiving the notifications, stream, and reports from the Input Manger’s Notifier and Streamer components. The two key components of the subsystem are Stream Receiver and the Notifications Receiver components. Both of these components act as listeners for any events from the Input Manager subsystem.

4. Development/Implementation Details

4.1 Raspberry Pi 4

In the Raspberry Pi 4, which works as a server and a manager of input we use the following libraries for the following tasks.

In the beginning Raspberry Pi 4 needs to be set up to work properly with the camera, this includes enabling it and using proper compatible OS version to work with the camera, in our case it is *Debian 10*, as well as installing packages like, *python-picamera*. Afterwards, Python and its virtual environment needs to be downloaded and set up. Then installing necessary video and audio packages to enable ffmpeg and recording of video and audio. In order to set up opencv the following packages were necessary:*pkg-config libjpeg-dev libtiff5-dev libjasper-dev libpng-dev libavcodec-dev libavformat-dev libswscale-dev libv4l-dev libxvidcore-dev libx264-dev libfontconfig1-dev libcairo2-dev libgdk-pixbuf2.0-dev libpango1.0-dev libgtk2.0-dev libgtk-3-dev libatlas-base-dev gfortran libhdf5-dev libhdf5-serial-dev libhdf5-103 python3-pyqt5 python3-dev.* Additionally, Testing for performance and benchmarking with different packages, models, sub processes, and order of execution was executed throughout the development cycle. Afterwards, to connect Pi to other device through Bluetooth, necessary Bluetooth manager packages need to be installed, like *bluetoothctl* and *pulseaudio-module-bluetooth*, then configured properly, finally, for managing these device packages like *bliuealsa* are needed to control audio devices, for example.

Pi first takes arguments about the info of captured images, and loading models. After it starts it initiates the sub processes for audio recording, streaming etc. Once the system is ready, the PiCamera model is used to capture the frames from the camera, then load them into openCV to prepare it to be fed to TFLite. After TFLite returns the results of detections, based on what is detected the system would report the client and start recording the incident to prepare a report, once the report is generated it is sent to the client. In the meanwhile, in a parallel process, the audio is recorded to include in the report. In another parallel process, queues are used to transmit the processed frames and then send them over to the client using the ZMQ messaging library. In order to encode the video properly before sending it over in the report, ffmpeg is used for that. ffmpeg enables the encoding of videos like, format, fps, audio source, codec, etc.

Library	Objective
OpenCV	Processing of the capture frames, like resizing, colors, etc, and writing the

	videos to files, as well as, reformatting the frames before steaming them and then re-formatting them after receiving, in order to show them. It also acts as the display for the received frames on the client device
PiCamera	Manager of the camera of the Pi device, it set resolution, size, frame rate, etc. for the camera
ffmpeg	Used for encoding the videos after writing them to files, as well as, merging them with audio once recording is done.
pyaudio	Manager of audio input while recording. Used to set bitrate, fps, and buffers for audio recording, then writing them to files.
Multiprocessing	Native python library used to manage sub-processes. It is used to create subprocesses for audio recording, streaming, transmitting reports, etc.
Multiprocessing.Queue	Queues were used to communicate between the processes, like to send frame from main analysis process to streaming process, to enable multi-concurrency.
ZMQ	ZeroMQ is a library for messages queues. The Zero part is related to the fact that it requires no broker, making it fast and similar to web sockets. It was used for streaming, and sending reports on the server side, and to receive these messages on the client side.
TFLite	A light-weight Tensor Flow library used for small and limited machines like Pi and smart phones to load the models and detect objects.

4.2 Client side

In the client device the stream is received and viewed, and by using multi-processing we enable waiting for the reports and then receive them, additionally we use run another sub-process to receive notifications and input from the user.

Library	Objective
OpenCV	Processing of the received frames, like reformatting, resizing, colors, etc, and writing the videos to files, as well as, viewing the frames live.
Multiprocessing	Native python library used to manage sub-processes. It is used to receive the stream, receive reports, as well as, managing notifications and input.
Multiprocessing.Queue	Queues were used to communicate between the processes, like to send input after receiving it.
ZMQ	ZeroMQ is used to receive stream and videos for reports.

4.3 Machine Learning

For Machine learning. First off we started by taking pictures and labeling them using LabelIMG. Afterwards, Google CoLab was used to feed the labeled picture and train the model. Then, on Google CoLab model is evaluated using the test images. After that, we freeze the graph of the model, and the frozen graph is converted into a TFLite friendly intermediate saved model. Finally, we convert the saved model into TFLite format and add metadata to the model and compress it.

5. Test Cases

Considering the complexity of using multiple hardware components and several meticulous elements in the system, like the Machine Learning component, the camera capture, the data management, and the streaming service; testing is crucial not just to make sure the developed functionalities work as intended, but also to ensure to help in the development process. Testing elements like Fakes, Mocks, and Stubs, as well as manual testing, will play a major role in the process of testing and development of our system.
In order to accomplish that we are using the following concepts to test our components. Below each concept, a list of specific tests and the mechanism of testing can be found.

Unit Tests

Unit tests test the functionality of each “unit” in the system, asserting that the unit does what it is expected to do. A “unit” can be defined in many ways depending on the system, conventions, and engineers. We will refer to each function in the system as a unit. Some unit tests are automated and others are manual, and therefore the tests in the first section below would be represented as test code in test suites in the project’s code, while the section afterward, would represent manual tests. Unit tests will follow the Arrange, Act and Assert (AAA) Pattern.

1. Test that the camera works. Mechanism: start it and capture a photo.
2. Test that the camera can capture videos. Mechanism: capture a short video of fewer than 3 seconds.
3. Test that the system can capture the last one minute before and after the incident. Mechanism: camera can be set to “preview” mode, and then capture 1 minute of content, introduce an incident, verify the length of the video by performing video size calculations, keeping in mind encoding, bitrate, resolution, and frame rate.
4. Test that the video captures are of specific size. Mechanism: define the bit-rate, resolution, frame size, length, and encoding, then calculating the expected size and then comparing it to the captured video.
5. Test the audio in the video captures are of a specific size. Mechanism: define the bit-rate, resolution, length, and encoding, then calculating the expected size and then comparing it to the captured video’s audio.
6. Test that the image compressor compresses the images to the specified ratio chosen. Mechanism: feed the systema a pre-made image with specific size, compress and calculate against the ratios. The ratios can vary depending on the implementation. Realistically, 0.8-0.5 is the aim.
7. Test that the incident videos are being deleted after a specified time. Mechanism: save a video, set periodic deletion time to 10 seconds, sleep for 10 seconds, verify that the file is deleted.
8. Test that the de-fisheye works. Mechanism: save a raw image of the camera, compare the resolutions and sizes of the images before and after.
9. Test that the object detection functionality works. Mechanism: feed the system a video of objects that should be detected and then logs the events of detection, compare them to predefined expected logs.
10. Test that the accuracy of the detection is above 85%. Mechanism: feed videos of objects that models were trained to detect and then check the logs values.
11. Test that the system can create proper notifications based on the events detected, each object and event will have its own tests. Mechanism: introduce a predefined set of events and compare the logs to predefined log values.
12. Test that the system can connect to other nearby Bluetooth devices. Mechanism: run the bluetooth devices before the start of the testing suites, run the tests, assert that Pi 4 is connected to the specific Bluetooth devices.

13. Test that the system can disconnect from Bluetooth devices already connected. Mechanism: On success of the last test, disconnect from these devices and assert that the system is no longer connected to these devices.
14. Test that the system can connect to the internet. Mechanism: make sure WiFi connection is available before the test suite is run, run script to connect to the internet, ping Google and confirm a received response.
15. Test that the system can perform a WiFi connection. Mechanism: Set up Android device for connections, run script to connect the Pi 4 devices, send a message and receive a reply from Android device.
16. Test that the system can send a notification through the internet. Mechanism: upon success of previous two tests, use stub to create a notification, feed it to the notifier and send the notification, on the Android device feed the same stub, receive, assert that they are identical.
17. Test that the system can "pair" with another device through the internet. Mechanism: Upon success of internet connections of the devices, create sockets, connect them and then bind each device to the other, send multiple messages in both ways, verify reception of each message.
18. Test that the system can stream the camera's content over the internet. Mechanism: launch the Pi 4 device with the camera on, connect to the internet, pair the devices, capture video preview and in real-time, compress send over the internet, receive on the Android device, decompress, compare details. Stop after 1 minute, compare the total sent data v against the total received.
19. Test that the Android device can receive a notification from the internet. Mechanism: Connect the device to the internet, create a testing fake, connect it to it, or connect to Pi 4 if the connection implementation is over. Create a stub of a notification, send it from the testing fake, or Pi, then confirm that the notification is identical to the pre-defined stub.
20. Test that the Android device can "pair" with the system through the internet. Mechanism: similar to the Pi test, create sockets, connect, then pair. Ping the devices to be sure their connection is stable. A testing fake can be used if the Pi 4 device cannot pair yet.
21. Test that the system generates reports of incidents automatically. Mechanism: Launch the system, feed it a pre-recorded video with a detected danger, run the identification on it, logs the results, compare them to pre-defined logs.
22. Test that the system can receive responses from the Android device for the notifications. Mechanism: Use a testing mock to invoke a notification, send a stub response, receive on Pi 4 and compare it to the predefined stub.
23. Test that the system changes the mode as it receives no response from the Android device. Mechanism: Launch the system, bind to Android device, feed it video with a dangerous objects, send notification, confirm reception, don't respond, assert that the status of the system after 30 seconds is "Panic Mode"
24. Test that the system would refuse to connect to devices without the predefined password. Mechanism: Launch device connect to the internet, set up its IP and port to predefined values, attempt connection from Android device, assert the request was sent, then assert that Pi 4 is not connected to any device.

25. Test that the system can compress the live stream data before sending it over the internet. Mechanism: Launch the system, pair with a mobile, capture video of 10 seconds, store the size, compress, calculate the size, decompress, compare the size to confirm it is identical. Assert that the video is of the same size before compression and after decompression.
26. Test that the Android device can decompress the data before viewing it. Mechanism: Send compressed using Pi 4 or through a testing fake, receive, decompress, compare size, and verify that the file is not corrupted.

The following tests are manual:

27. Test that the de-fisheye image is not limiting the view of the important elements of the room. Mechanism: capture a pic, de-fisheye, visually confirm no important details were removed from the picture after the cropping, and resizing.
28. Test that the infra-red sensor is working properly. Mechanism: Launch the camera, turn lights off, visually assert that the night vision is working.
29. Test that the speakers are producing audio with proper quality. Mechanism: Launch the system, play a high quality audio file, in person listen to the audio and confirm the quality is proper.
30. Test that the microphone can detect noises and record with proper quality. Mechanism: Launch the system, go near the microphone, make noises, play music, and talk, then save the file and listen to it, assert that the audio quality is proper.
31. Test that the audio and video content is of high quality (no choppiness, or electronic noise). Mechanism: Launch the system, go near the microphone, make noises, play music, and talk, then save the file and listen to it, assert that the audio quality is proper.
32. Test that the system can identify movement of the toddler and record it. Mechanism: Run the system, bring a toddler, let the toddler move, check the logs of the system to confirm the detection of movement.
33. Test that the system can identify noises in the night and record it. Mechanism: Launch system and produce loud noises, check the incidents and logs to confirm detection.
34. Test that the system can detect high brightness light in the night. Mechanism: Turn light off, run the system, bring a monitor or laptop set to maximum brightness, confirm from logs that the system detects it.
35. Test that the system generates correct analysis for the time of movement. Mechanism: run the system, bring a toddler, let the toddler move, check the logs of the system to confirm the detection of movement of specified length.
36. Test that the login page functions responsively. Mechanism: run the Android application, enter valid login credentials, assert the success.
37. Test that the details stored in the login page are retrieved and logs the user in. Mechanism: Upon success of previous test, close application, and open it again, assert that no login was required.
38. Test that the user is able to pair the device once logged in. Mechanism: Login, pair to Pi 4, enter the password, then open streaming window, assert live stream by being in the room seeing yourself live in the video.

39. Test that the Android device calibrates the video and streams the content. Mechanism: Login, pair to Pi 4, enter the password, then open streaming window, assert live stream by being in the room seeing yourself live in the video with no delay or choppiness.
40. Test that the Android device displays the identified objects through the stream. Mechanism: Login, pair to Pi 4, enter the password, then open streaming window, assert live stream by being in the room seeing yourself live in the video, bring a dangerous object in hand, check notifications on the phone.
41. Test that the Android device will store the necessary recorded information. Mechanism: Login, pair to Pi 4, enter the password, then open streaming window, assert live stream by being in the room seeing yourself live in the video, bring a dangerous object in hand, check the reports on the phone
42. Test that the Android device will allow the user to toggle between different safety options. Mechanism: Login, pair to Pi 4, enter the password, then open streaming window, set to max alertness, assert live stream by being in the room seeing yourself live in the video, assert notification of intruder, change to low alert, assert no notification, bring a dangerous object in hand, check notifications on the phone.
43. Test that the Android device is able to set lullabies and/or alarms. Mechanism: Login, pair to Pi 4, enter the password, then open streaming window, assert live stream by being in the room seeing yourself live in the video, play a lullaby confirm the audio being played
44. Test if the frozen object detection saved model can be converted to tflite format or not.

Integrations Tests

When working in a team of several developers, each using his/her own style and logic to program the elements they are responsible for, new problems arise, some problems are regarding integration. Different software, or even hardware, components need to communicate with each other. This can be a challenging task, that is why Integration tests are the second level of testing that Vanny will go through to ensure that all components not just work, as tested in unit tests, but also are capable of communicating with each other in the desired and expected way. The following tests will be performed to ensure the proper integration of all elements in the system. The main method of integration testing used will be Functional Incremental Testing. In this method of testing the components' integration will be tested incrementally, piece-by-piece based on their functionality.

1. Test the interface link between the Watcher class and the streamer class.
2. Test the interface link between the Report Generator class and the Activity Analyzer class.
3. Test the interface link between the Watcher and the Environment Analyzer class.
4. Test the interface link between the Watcher and the object identifier class.
5. Test the interface link between the Report Generator and the Activity Analyzer class.
6. Test the format and encoding compatibility between the video encoder and the Stream Manager.

7. Test the compatibility of data format, compression and encoding between the Stream Sender, on Pi 4, and the Stream Receiver, on the Android mobile.
8. Test the compatibility of the tflite object detection model with OpenCV.
9. Test the compatibility of the output tensor shape with the OpenCV

System Tests

System tests are performed relatively late in the development lifecycle. The main purpose of such tests is to validate the finished software product and evaluate the completeness of specifications. There are many types of system testing. The following tests will be performed as a part of Vanny's system tests.

1. Usability Testing: focuses mainly on the user's ease of use. Mechanism: bring a person unfamiliar with the system, give them a brief explanation and a simple guide on how to use the system then observe them using all the functionalities of the system, repeat with multiple people, take feedback and improve the product accordingly.
2. Load Testing: focuses on the behavior of the system under expected load. Mechanism: run the system with a toddler in a room, pre-recorded or live, introduce dangerous objects to the environment every 15 minutes and observe the reaction time of the system and the performance when such objects are introduced. Additionally, at the time of the reports' generation an object will be introduced while a live stream is being performed. This is considered an expected load as all of these events can indeed happen at the same time.
3. Stress testing: focuses on the behavior of the system as the load exceeds the expected maximum load. The aim is to help us understand the behavior of Vanny in such incidents. Mechanism: introduce detectable objects and people one by one and keep them in the scope of the camera, use multiple devices for streaming and generate multiple reports at the same time. Keep increasing the number of objects, devices, reports, and people until the system gets overloaded; observe the behavior. Detect bottlenecks and accommodate and optimize accordingly, if needed.
4. Hardware/Software Testing: focus on the interactions between hardware and software. Many types of tests can be performed under this category. The focus for our project would be Performance testing, since the Pi is limited in terms of performance and that it lacks a capable GPU. Considering the large amount of graphic data processed in the system this could become a choking point for the system. Mechanism: Set the frames rate to an acceptable range, like 20 fps. Perform load tests on the system, observe the behavior and the performance, if the system is slow or sluggish, drop the frames rate. Keep repeating until the system becomes stable with a specific frames rate. Perform a stress test, observe the system. Throughout the test keep logs of the systems' element usage rate, as well as memory writing rate.

6. Maintenance Plan and Details

Maintenance Plan:

1. Regular Hardware Check: Conduct periodic inspections of the Raspberry Pi, camera, speaker, and other components to ensure they are functioning properly. Check for any physical damage or loose connections.

2. Software Updates: Stay up to date with the latest software updates for the Raspberry Pi operating system and any required libraries or frameworks used in the application. Regularly apply necessary updates to improve performance, security, and compatibility.

3. Data Backup: Implement a backup system to regularly save the recorded audio and video data captured by VANNY. This ensures that the data is protected in case of system failures or data loss.

4. Performance Monitoring: Monitor the performance of the VANNY system to identify any potential bottlenecks, resource limitations, or anomalies. Monitor system resource usage, network connectivity, and response times to maintain optimal performance.

5. Security Measures: Implement security measures to protect the VANNY system from potential threats and unauthorized access. This includes using secure protocols for data transmission, implementing access controls, and regular security audits.

6. Testing and Validation: Conduct regular testing and validation of the VANNY system to ensure that all components are working as expected. Test the real-time streaming functionality, danger detection algorithms, and notification system to validate their accuracy and reliability.

Details Information:

VANNY (Virtual nANNY) is an ML, AI, and IoT solution designed to provide parents with a comprehensive monitoring system for the safety of their toddlers. The system leverages a Raspberry Pi 4, a camera, speaker, and a client application to actively scan the environment and detect potential dangers.

Using real-time image analysis on the Raspberry Pi, VANNY identifies sharp objects, fire, and smoke in the toddler's surroundings. The system records audio and video to capture potential dangers and processes the data locally. Alerts are then sent through the internet to the client application.

The client application offers real-time streaming, allowing parents to remotely monitor their child. They can securely access the live video and audio feed from VANNY's camera and speaker.

Additionally, the application provides instant notifications when dangers are detected, enabling swift action to ensure the child's safety.

To ensure the smooth operation of VANNY, a maintenance plan is implemented. This plan includes regular hardware checks to inspect the Raspberry Pi, camera, speaker, and other components for any physical damage or loose connections. Software updates are applied regularly to keep the system up to date with the latest improvements, security patches, and compatibility enhancements.

Data backup is an essential part of the maintenance plan, ensuring that the recorded audio and video data captured by VANNY is securely backed up to prevent data loss in the event of system failures. Performance monitoring is conducted to identify any bottlenecks or anomalies that could affect the system's performance.

Security measures are implemented to protect the VANNY system from potential threats and unauthorized access. Secure protocols are used for data transmission, access controls are implemented, and regular security audits are conducted to maintain a high level of security.

Regular testing and validation are performed to ensure that all components of the VANNY system are functioning as expected. Testing includes validating the accuracy and reliability of the real-time streaming functionality, danger detection algorithms, and notification system.

Overall, VANNY combines the power of ML, AI, and IoT to prioritize child safety, providing parents with peace of mind by offering comprehensive monitoring and swift alerts in case of potential dangers.

7. Other Project Elements

7.1 Consideration of Various Factors in Engineering Design

7.1.1 Public Health and Safety

Vanny protects babies from any harm. We detect any sharp object near the baby which might cause any harm and warn the parents. We also detect any stranger who might be a baby. It also detects any irregular behavior of the baby which might be due to illness and warns the parents early. The application also ensures the baby has a healthy sleeping pattern.

7.1.2 Public Welfare

Our system is much cheaper than most other available systems in the market as it only uses commodity hardware. Also, Vanny only has a one-time cost as the system doesn't use any remote backend.

7.1.3 Public Information Security

Every data is encrypted and communicated to the client through a secure channel. The application doesn't store any permanent user data. It deletes everything after a maximum of a week. The application is also not connected to a remote backend. Hence, the maker of the system doesn't have access to any user data. This was designed like that intentionally to increase the consumer's confidence in using the system. Vanny provides information about a baby who is alone in his/her home. Hence, privacy issues need to be taken very seriously to prevent any access to user data by any third party including the maker of the system.

7.1.4 Global Factors

The application is available in many countries around the world. The application will be trained in multiple languages. Hence, it can be used in different parts of the world.

7.1.5 Cultural Factors

The application needs audio processing. Although a child might only utter the simplest words, it will still be different for different languages. Hence, we need to train our audio ML model specific to the language the child speaks. This way we may be able to understand the child's words and provide more nuanced information to the parents.

7.1.6 Environmental Factors

No Environmental factors. Vanny doesn't harm the environment in any way.

7.1.7 Economic Factors

Vanny saves the salary that the parents had to pay the real nanny. Parents can work longer hours in their job without much worry because Vanny is always there to look after their children. Hence, they contribute to the economy more. Vanny also costs less than other systems in the market as it uses commodity hardware only. This also saves money for the parents.

7.1.8 Social Factors

Vanny creates a social distance between the baby and the parents as there will be fewer real-life interactions. However, we have taken some steps to ensure that this social distance is reduced to a minimum. Vanny can play the voices of the parents to the baby. It is also possible for the parents to interact with the baby in real-time with live video chat which can be played on the monitor in front of the baby. However, this feature will only be available in future versions.

	Effect Level (out of 10)	Effect
Public Health and Safety	8	Protects babies from any harm.
Public Welfare	3	Cheaper than other available options.
Public Information Security	9	Encrypts everything, has no remote backend and deletes everything after a week.
Global Factors	2	Applicable in any country
Cultural Factors	4	Language differences in audio

		processing are considered.
Environmental Factors	0	None
Economic Factors	8	Saves money for the parents as they don’t need real money.
Social Factors	4	Creates a distance between the baby and the parents.

Figure 5. Evaluation of factors in engineering design

7.2 Ethics and Professional Responsibilities

- In order to avoid license and copyright issues that might be encountered during the development, the open-source libraries will be preferred during the development. Similarly, all software tools that will be used will be free or properly licensed for our usage. All code excerpts copied from the internet or other sources will be referenced properly.
- The application will record the babies and toddlers with the parents consent. It will not be used in any public area which might cause some ethical and social issues. The application will run on a local network and only the devices that are authorized with a predefined password will be able to connect to the camera. Therefore, if any kind of security problem occurs in the local network, the data cannot not be obtained by the attackers.

- When processing the voice of the parent of the toddlers in case of sending the voice to babies, the record is only stored locally therefore it will not cause any problem regarding the sharing of the data with third parties.
- User data will be protected according to the GDPR (General Data Protection Regulation) and KVKK (Kişisel Verilerin Korunması Kanunu) regulations. Any recorded data that is unnecessary for the client and for the performance of the application will not be stored. It is planned to store the highlights and important records with approximately three minute periods will be stored instead of storing the whole record. Moreover, the stored data will be permanently deleted after a period of time such as 3 days unless the user chooses to keep it. Only the parents of the babies and toddlers will have access to the stored data and the live records and that data will not be shared with any third party organization by any means.
- As team Vanny we also have ethical responsibilities towards the environment and most of the time, calculation requirements constitute considerable amounts of power consumption in the environment. Therefore in our project and calculations, we aim to implement our functionalities in a way that creates a minimum amount of energy consumption making our process sustainable.

7.3 Teamwork Details

Teamwork is one of the most essential parts of any project. Each team member must have nearly equal responsibilities and each member needs to participate in each deliverable. In order to ensure those critical aspects of teamwork, we needed to make necessary measurements and adjustments in order to create the environment that is needed for effective teamwork. In this section, the contribution of the team members, the environment that enables and increases those contributions, and the leading roles and shared leadership status among the team are described.

7.3.1 Contributing and Functioning Effectively on the Team

Our project is a comprehensive one that requires knowledge about both hardware and software systems including the knowledge about setting the communication links between those systems. Each team member has a distinct and unique specialty that he/she gains throughout the university or work life. So in order to achieve the best possible contribution and functioning in the team, we decided which member would be the most knowledgeable one in a specific task. According to those specifications, we formed subgroups that work together to finalize the subtasks of the project. That's how everyone works on the parts that he/she is most suitable and effective with.

As another measurement, we know that each member has several classes and other kinds of heavy work that he/she needs to complete other than the project. So we also need to consider those special situations to ensure that everyone can contribute without being overwhelmed. For example, if some team members have exams or other urgent assignments, we do not schedule any meetings in those weeks to ease the workload of the team members.

7.3.2 Helping to Create a Collaborative and Inclusive Environment

There are several methods to create a collaborative environment for efficient development. In this section of the report, some of those measurements are listed in detail.

- **Regular Meetings:** There need to be regular meetings that are scheduled and synchronous. Only being in communication using chatting platforms such as WhatsApp or Jira is not enough for efficient communication, rather there need to be synchronous meetings where each member can ask questions regarding any aspect of the project to the others that may have the answers to those questions. Those meetings should not be on arbitrary dates but they should be scheduled regularly sufficiently before the meeting day so that each member can be better prepared for the meeting.

- **GitHub:** GitHub is a great platform that we can manage version control over our project. It is also possible to see the commits and progress of the other team members via Github so that we can see what the other members are doing and who needs additional help in his/her current work. GitHub issues also create a great environment where we can note the required changes, implementations, or bug fixes that are necessary for the project. Using Pull Requests from different branches, we are able to review the coded features of team members and suggest changes to those implementations when needed. Overall GitHub is a tool that helps us to create a collaborative environment that every team member can make use of.

- **Slack:** Rather than using WhatsApp as the asynchronous messaging platform for our project, we prefer to use Slack. Using Slack feels more professional in terms of development and it reduces the distraction that comes with the usage of WhatsApp. Being able to create channels in Slack enables us to divide each subtask into channels on Slack and keep the communication in the relevant channel which ensures abstraction. Using the Huddle feature of Slack, we are also able to communicate via voice without spending much time scheduling a zoom meeting or a regular meeting. Instead, we are able to resolve small bugs or small misunderstandings using a quick huddle between the related team members about the issue.

- **Logbooks:** One of the helping regulations of the Senior Project is that each team member needs to have a logbook in which he/she documents his/her progress regarding the project. Keeping a logbook is an encouraging activity that increases productivity since it is valuable to see the work done regularly in the logbook. To see and increment the progress made, each team member is eager to update his/her logbook on new progress. Therefore logbooks are also one of the most valuable things that help create a productive work environment.

7.3.3 Taking Lead Role and Sharing Leadership on the Team

As a group we believe that leadership should be handled in each different subtask in the development. There should not be only one leader that is leading the whole process but rather whoever is capable of a subtask he/she needs to lead the group that is responsible for the relevant subtask. So while we are dividing the team into subgroups for different implementation stages such as machine learning, frontend, and hardware sides, we consider the most eligible participant in the corresponding subtask as the leader of that group. In case the leader thinks that additional help is needed in a specific task he/she can request help from different subgroups and according to the decision of the leader in the other subgroup, the workload can be divided with balance.

Therefore, the leadership is shared among the team and whoever is eligible for the leadership of the relevant task, he/she is eager to take the lead in that task and manage the rest of the team members.

7.3.4 Meeting Objectives

Our system consisted of six main elements: 1- Hardware Management and Image Processing. 2- Toddler Detection 3- Objects detection. 4- Audio analysis. 5- Mobile application. 6- Optimization and streaming. We will go through each element and whether they were implemented or not.

7.3.4.1 Hardware Management and Image Processing

This element was the first and most critical element. To accomplish this we needed to set up the Raspberry Pi 4 device and make the attached camera work and record. After setting up the proper OS and configuring the camera this element was accomplished successfully. The Pi device can take pictures, process them on the fly and then output them in the preferred way.

7.3.4.2 Toddler Detection

This ML element was the most critical. The detection of toddlers was accomplished in December. However, Because of the fact that we are using a Raspberry Pi device the ML models needed to be configured so we did not consider this task over until the following tasks were implemented: 1- Creation of Tensor Flow model. 2- Ability to detect toddlers with high accuracy. 3- Conversion of the model to TFLite model, in order to make it possible to run on Pi. 4- Testing the performance and improving accordingly. Once these tasks were over, we considered this element over.

7.3.4.3 Objects Detection

Similar to the previous element this could not be considered done until the model was accurate and could be loaded on the Pi machine. The only difference here is that for objects detection we had to detect some objects that were small, while TFLie processes larger chunks of the picture at a time. This was challenging, however, we managed to accomplish this by detecting doors, windows, scissors, knives, large pens, for a reasonable distance, up to around 3 meters for the small objects.

7.3.4.5 Mobile Application

The mobile application was going pretty well. We implemented the UI, lists of report, ability to play videos, etc. However, we severely underestimated the streaming side of the application. After countless attempts using MQTT, ZMQ, Web Socket, Google Cloud, HLS streaming without any success, we had to give up this element and use another client Python-powered device. Streaming from one small device, Pi, to another, Android, is no trivial task. In addition to the fact that the servers that were used to support streaming. Each device, and method and consequently server, requires some configurations and format to work smoothly with live video streaming, however, setting up such option for one components always meant that another component will stop working, After an endless chase and countless tests, where two Python-powered application could stream to each seamlessly, and after testing Google Cloud and getting “Connection Reset” meaning that we hit the limit of requests and that the server would close for some minutes, after performing a frame-by-frame streaming and after realizing how slow the stream was, and consequently how it would shut down, we finally decided to give it up, and move to another computer that receives the stream live and shows it, as well as manage reports and receive them as soon as they are available.

7.3.4.6 Optimization and Streaming

Working with a small device like Pi 4 meant that every element that we add would be visible when the performance drops. Meaning we needed to be careful and utilize all the computing power of the device in the most efficient way. First decision that enabled this was storing only the video and the audio recordings that might be needed for the report, meaning if 1 minute will be recorded in case an incident happens, then no more than 1 minute will be in the buffer, the rest will be flushed, enabling us to get around the memory limitations of the SD memory card and the frequent, expensive, writings. Second, we know that Pi 4 has 4 cores, meaning concurrency is possible. Since Python does not support real multi-threading, because of the Global Interpreter Lock, we decided to move forward with multi-processing instead. We successfully accomplished multi-processing for video recording, audio recording, streaming, and report transmissions. Finally, we optimized the ML models to Lite so that it works with acceptable performance with the hardware at hand.

7.4 New Knowledge Acquired and Applied

The development of our project involves several elements that are new and unfamiliar to us. Therefore, it was crucial for the team members responsible for these areas to acquire and apply new knowledge effectively. Here are the key aspects we focused on and the learning strategies we employed:

1. Hardware Specifications: One of the significant aspects of the project that we were unfamiliar with was the hardware setup, particularly the Raspberry Pi 4, camera, speaker, and microphone integration. To tackle this, we utilized various learning strategies. Firstly, we made use of online resources, such as Stack Overflow and forums, where developers with similar experiences had shared their insights and solutions. Additionally, we had a team member who possessed knowledge about Raspberry Pi, so we leveraged peer advising by seeking guidance and advice within the team. Moreover, we approached instructors in our department for expert guidance in addressing hardware-related challenges.
2. Android Application Development: Developing an Android application introduced new concepts and tools to the team members. We chose Android Studio as our development environment and Java as the programming language. To acquire knowledge in this domain, we primarily relied on the official documentation and APIs provided by Android. By thoroughly studying the official documentation, we were able to gain a comprehensive understanding of the platform and utilize the available resources effectively. Furthermore, we capitalized on the expertise of a team member who had experience with Android development, seeking advice and guidance as needed.
3. Computer Vision and Machine Learning: Our project heavily relies on computer vision and machine learning concepts, which were unfamiliar to all team members. To address this knowledge gap, we adopted a multi-faceted approach. Firstly, we conducted extensive literature reviews to explore existing implementations and gain insights into best practices. This allowed us to understand the theoretical foundations and explore relevant solutions in the field. Additionally, we sought the expertise of our innovation

expert, who had a strong background in machine learning, computer vision, and the Internet of Things. Their guidance provided valuable domain-specific knowledge and helped us align our application with industry standards. To further enhance our understanding and practical skills, we utilized official documentations, such as OpenCV and TensorFlow, as our main resources. Additionally, we explored book reviews and online resources like W3Schools to supplement our learning.

In conclusion, the acquisition and application of new knowledge were essential for the successful development of our project. By implementing effective learning strategies such as literature reviews, official documentations, peer advising, expert opinions, and online resources, we were able to enhance our understanding of the new concepts and technologies. This approach allowed us to develop our application efficiently and fulfill the non-functional requirements we aimed to achieve.

8. Conclusion and Future Work

8.1 Conclusion

In conclusion, VANNY (Virtual nANNY) is a powerful ML, AI, and IoT solution that empowers parents and ensures the safety of their toddlers. By leveraging the capabilities of Raspberry Pi 4, a camera, speaker, and a client application, VANNY offers comprehensive monitoring and proactive danger detection.

VANNY actively scans the toddler's environment, recording audio and video to detect potential dangers. Through real-time image analysis on the Raspberry Pi, it can identify sharp objects, fire, and smoke. These alerts are processed locally and efficiently sent through the internet to the client application, providing parents with timely notifications.

The client application of VANNY enables real-time streaming, allowing parents to remotely monitor their child from anywhere. This feature gives them a sense of reassurance and the ability to take immediate action if any dangers are detected. The integration of ML, AI, and IoT technologies allows VANNY to prioritize child safety effectively.

8.2 Future Work

While VANNY presents a significant advancement in child safety monitoring, there are several avenues for future work and enhancements to further improve its capabilities:

1. Expand Danger Detection: Consider expanding the range of potential dangers that VANNY can detect. This may include identifying other hazardous objects, toxic substances, or potential risks specific to different environments.
2. Improve Accuracy: Continuously refine the ML and AI algorithms used by VANNY to enhance accuracy in danger detection. Explore techniques such as deep learning and neural networks to improve the system's ability to distinguish between different objects and environmental hazards.
3. Enhance User Interface: Further enhance the user interface of the client application to provide a more intuitive and user-friendly experience for parents. Incorporate additional features such as historical data analysis, customizable notifications, and personalized settings.
4. Integration with Smart Home Systems: Explore integration possibilities with smart home systems to extend VANNY's functionality and interoperability. This may involve integration with home security systems, voice assistants, or other IoT devices for seamless interaction and expanded safety capabilities.
5. Continuous Performance Monitoring: Implement continuous performance monitoring mechanisms to ensure the VANNY system is operating optimally. Monitor system resource usage, network connectivity, and response times to identify and address any potential performance bottlenecks.
6. User Feedback and Iterative Improvements: Collect feedback from parents and users of VANNY to gather insights and understand their needs better. Utilize this feedback to drive iterative improvements, addressing any usability issues or feature requests that arise.

In summary, VANNY has the potential for further enhancements and future development to continue empowering parents and ensuring the safety of their toddlers. By leveraging ML, AI, and IoT technologies, VANNY combines innovation and child safety, offering parents peace of mind while their children are being monitored.

9. References

[1] A. S. Inc., “Alfred Home Security camera,” *App Store*, 02-May-2015. [Online]. Available: https://apps.apple.com/us/app/alfred-home-security-camera/id966460837. [Accessed: 09-March-2023].

[2] A. B. A. s.r.o., “Annie Baby Monitor: Nanny cam,” *App Store*, 30-Mar-2015. [Online]. Available: https://apps.apple.com/us/app/annie-baby-monitor-nanny-cam/id944269894. [Accessed: 09-March-2023].

[3] T. LLC, “Baby Monitor Saby + Cry Alarm,” *App Store*, 05-Aug-2020. [Online]. Available: https://apps.apple.com/us/app/baby-monitor-saby-cry-alarm/id1503992310. [Accessed: 09-March-2023].

[4] A. Shabunia, “Camy - live video monitoring,” *App Store*, 09-Sep-2019. [Online]. Available: https://apps.apple.com/us/app/camy-live-video-monitoring/id1475627437. [Accessed: 09-March-2023].

[5] M. V. Solutions, “Video surveillance ivideon,” *App Store*, 16-Sep-2011. [Online]. Available: https://apps.apple.com/us/app/video-surveillance-ivideon/id464234175. [Accessed: 09-March-2023].

[6] “Mi home security camera1080p,” *Mi Home Security Camera Wireless IP Surveillance System 1080p HD Night Vision (US Version with Warranty) - Xiaomi United States*. [Online]. Available: https://www.mi.com/us/mi-home-security-camera. [Accessed: 09-March-2023].

[7] “Angel Eyes Ai - Microsoft AI Lab,” *Angel Eyes AI - Microsoft AI Lab*. [Online]. Available: https://www.microsoft.com/en-us/ai/ai-lab-angel-eyes. [Accessed: 09-March-2023].

[8] D. Maloney, “Machine learning baby monitor prevents the hunger games,” *Hackaday*, 23-July-2022. [Online]. Available: https://hackaday.com/2022/07/23/machine-learning-baby-monitor-prevents-the-hunger-game s/. [Accessed: 09-March-2023].