1. INTRODUCTION
2. LITERATURE REVIEW
3. PROJECT MANAGEMENT
4. REQUIREMENT COLLECTION AND ANALYSIS
   1. Interview
   2. Persona
   3. USU
5. SYSTEM DESIGN
6. SYSTEM IMPLEMENTATION
7. TESTING
8. CONCLUSION AND FUTURE WORK

This design project attempts to create a mobile phone application that seamlessly integrates the digital world with the real world by allowing users to measure the distance between them and people around them. This application requires the integration of many different components to produce an optimal user experience. The main components of the proposed design include geolocation, image processing, user interface design, software design and implementation on the Android and a website.

Applications:

1. Live Stream, measure the distance based on 2D understanding
   1. Android
   2. Web Site (Flask)
2. Single Image, Measure distance based on 3D understanding
   1. Android

1 INTRODUCTION

We would like to apologize to whom might be negatively affected by opening this topic again. Our deepest condolences to the families and acquaintances of the Corona victims. During the sparid of the COVID-19 virus, 2019-2020, the Bahrain government ensured the importance of keeping a social distance as a healthy precautionary method applied to avoid transmission of the virus (dadada). Therefore, this project was developed with the purpose of helping people to apply social distance. Gratefully that the sparid this virus and social distance is no longer a major concern (dadada).

Although the project and the period of the virus spread are related. It would be unfair to think that the project would only be active during the critical period of its spread. what this project is trying to offer, in addition to solving the technical problem, is to contribute to the treatment of the psychological impact of what was left of that crisis. Using Augmented reality technologies that are related to entertaining experiences to solve such a problem will help to meet the project objectives.

We developed three applications. The first application is an Android application that works on a single image. It captures a single image using ARCore and passes it to the machine learning model provided by ML Kit to detect human pose, then render the captured image using OpenGL. If there is any person detected it will try to place an anchor on it, then this anchor will be used to calculate the distance between the camera device and the detected person.

The second application is an android application that works on a stream of images. It uses the CameraX analysis use case to pass the camera images to ML Kit human pose detector and renders landmarks over the detected person using ML Kit GraphicOverlay classes. We use the result of detected shoulders and hips for a person to measure how far from the camera the person is. We use a polynomial equation to get the actual distance based on the distance between the shoulders and hips.

The third application is a web application developed using Flask works on a stream of images. It uses the OpenCV API to get the camera images and pass them to Mvonet multi-pose detector. The Movnet model is loaded by TensorFlow-Hub API. Then rendering landmarks on detected persons using OpenCV. Measuring the distance is done in the same way that is used in the second application, using a polynomial equation.

**1.1 Problem Statement**

In the midst of the challenge to battle the COVID-19 virus from outbreaking is to stop its transmission. One of the healthy precautionary methods applied to avoid transmission of the virus is to keep a safe distance between people. Predicting the distance is a crucial process and it is not easy for everyone. Children are not aware of how they measure the distance until a certain age. Providing an application that helps in measuring the distance between people assists in applying healthy precautionary methods.

**1.2 Project Objectives**

The project aims to provide an application that uses the camera device to measure the distance between a mobile and any detected person.

The solution will:

* Access the device camera and get its data to pass this data to the Machine Learning model
* Implement Machine Learning (ML) to detect any person in the camera scene
* Provide feedback when a person got detected, a digital overlay on top of the detected person.
* Measure the distance between the mobile camera and the detected person
* Alert the user when any violation for predefined distance violated

**1.3 Relevance/Significance of the project**

The importance of this project lies in the ability to measure the distance between the device camera and a detected person. We applied two approaches to measuring the distance. The first approach is based on 3-dimensional understanding of the environment, the approach will be discussed in section 6.4. This approach by itself is not unique but applying it to measure the distance between the device camera and the detected person for a native Android application is what makes it unique. This application developed to work on a single image. The second approach is based on 2-dimensional understanding of the camera image, it will be explained in section 6.3.3. We developed two applications using this approach, one a native Android application and the other a web-based application. Both applications work on a stream of images.

**1.4 Report Outline**

The following report is divided into 8 chapters. In chapter 2, a literature review demonstrating similar systems and solutions is presented. In chapter 3, activities that were performed throughout the lifecycle of the project are presented. This includes an introduction and justification for the chosen Software Development Life Cycle model, identification of risks that might be encountered during the project’s lifecycle, and a breakdown of the project activities and the time that was spent on each activity. In Chapter 4, the functional and non-functional requirements of the solution are gathered and analyzed. This includes a discussion of how we collected the foundational characteristics of the solution and who will use it, and a list of the requirements that were determined necessary for the project, a section. In Chapter 5, a dive into the solution’s design is presented. This includes further research and analysis of the system retirements with prototypes and presentation of the solution’s various aspects such as diagrams concerning the solution’s architecture such as use case and sequence diagrams. In Chapter 6, implementation decisions and tools used are discussed. In Chapter 7, testing each system unit of the application. Finally in Chapter 8, a conclusion regarding the project is given along with limitations and potential for future work.

2 LITERATURE REVIEW

The process of examining and assessing literature related to this project is defined as the literature review. The literature review consists of analyzing the literature, summarizing the information found, evaluating the gathered information by focusing on the features, what the system aims in the current state of these systems, theories, and opinions, and finally presenting the results (McCombes, 2019). As a result, this chapter will demonstrate two systems that are comparable to the system we are implementing. The chapter contains a description of the system's characteristics, purposes, and services.

2.1 Related Systems

2.2.1 AR Measurement

The application intends to recognize the 3-dimensional space using augmented reality (AR) technology. This will allow the application to measure distances of space. The application measures three distances:

1. Distance from the camera

2. Distance between two points

3. Distance between multiple points

Background pattern

Description automatically generated Background pattern

Description automatically generated Background pattern

Description automatically generated

This is an android application written in Kotlin. It uses ARCore and SceneForm APIs to provide the AR characteristics for the application. In this application, distance measurement is triggered by an onClick listener, which means that we have to click on the device screen to specify the points that we want to measure distances between or form.

2.2.2 ML Kit Sample App

The application aims to provide usage and integration examples for various vision-based ML Kit capabilities. It is an android application written in Kotlin and Java. The user has the choice to choose which language is used to function the application. This application is provided by Google as one of the samples for demonstrating ML Kit features and use cases (googlesamples, 2022). The application has nine use cases:

googlesamples (2022). *mlkit/android/vision-quickstart at master · googlesamples/mlkit*. [online] GitHub. Available at: https://github.com/googlesamples/mlkit/tree/master/android/vision-quickstart [Accessed 9 Dec. 2022].

1. Object Detection

2. Face Detection

3. Face Mesh Detection

4. Pose Detection

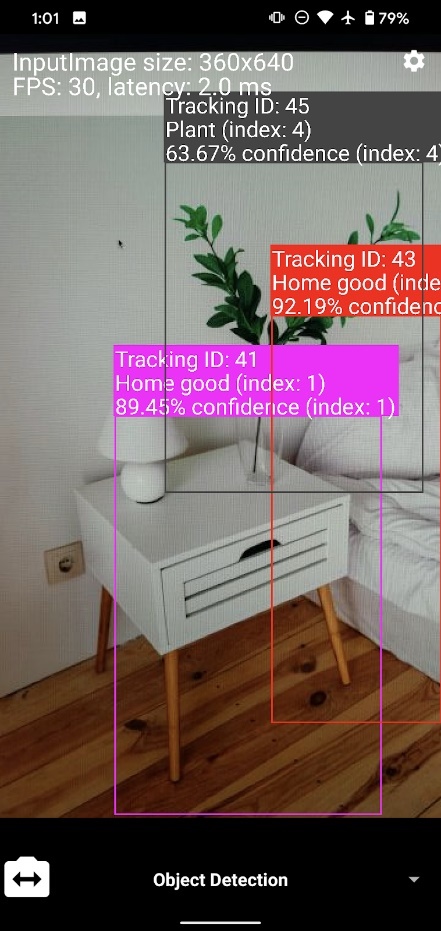
5. Text Recognition

6. Barcode Scanning

7. Custom Image Labeling - Birds

8. Image Labeling

9. Selfie Segmentation

CHAPTER 3 PROJECT MANAGEMENT

3.1 Process Model

The project was proposed and given a total of 6 months to be implemented. As such, a Spiral Software Development Life Cycle model was chosen to be followed throughout the project’s development. Each cycle of the Spiral Model is divided into four phases: Planning, Design, construction and evaluation. The reason for adopting the spiral model is that the tools for implementing this project were unclear and the spiral approach, is a risk-driven process model (Boehm and Hansen, 2000).

Boehm, B. and Hansen, W. (2000). *Spiral Development: Experience, Principles, and Refinements Spiral Development Workshop*. [online] Available at: https://resources.sei.cmu.edu/asset\_files/SpecialReport/2000\_003\_001\_13655.pdf.

‌We implemented the project over five cycles of the spiral model. The first cycle covered the low-fidelity prototype, and the second cycle covered the high-fidelity prototype, both low and high-fidelity prototypes will be discussed in chapter 4. The third cycle covered an application that works on a single image, this application will be discussed in section 6.4. The last cycle covers applications that works on image stream.

3.2 Risk Management

Software risk management (RM) is the process of identifying, addressing, and removing software risk factors before they either cause serious risks to the working software operation or result in costly software rework (Boehm, 1989).

Boehm, B. (1989). Software risk management. *Lecture Notes in Computer Science*, [online] pp.1–19. doi:10.1007/3-540-51635-2\_29.

|  |  |  |
| --- | --- | --- |
| Risk type | Possible risk | Description |
| Project | Data loss | The project data lost |
| Project | Unmet requirements | The application functionality does not meet the project objectives |
| Estimation | Not on schedule | Underestimating the time required to implement the project |
| Estimation | Not enough time to learn tools | Underestimating the time required to learn the needed tools |
| Tool | Tools are not supported | Tools are not compatible with the other tools versions or not supported by the current versions of the integrated development environment (IDE). |

|  |  |  |  |
| --- | --- | --- | --- |
| Rank | Risk | Probability | Effect |
| 1 | Unmet requirements | Medium | Catastrophic |
| 2 | Data loss | Low | Catastrophic |
| 3 | Not on schedule | Low | Catastrophic |
| 4 | Not enough time to learn tools | High | Serious |
| 5 | Tools are not supported | Medium | Tolerable |

|  |  |
| --- | --- |
| Risk | Strategy |
| Unmet requirements | Keep the supervisor updated with potential met requirements |
| Data loss | Using code version control repository e.g., GitHub. |
| Not on schedule | Breaking tasks into smaller ones, planning and providing alternatives. |
| Not enough time to learn tools | Use time boundaries for each tool |
| Tools are not supported | Sufficient research about the tool |

3.3 Project activities Plan

The project activity plan is how to complete a project in a certain timeframe, by dividing it into activities with approximated start and end dates (Lutkevich, 2021). The project activity plan is how to complete a project in a certain timeframe, by dividing it into activities with approximated start and end dates. The project activity plan worked as an indicator for any potential risk when an activity exceeded its estimated time. It helped balance between applying the main application functionality and enhancing an existing one.

Lutkevich, B. (2021). *Project planning: What is it and 5 steps to create a plan*. [online] CIO. Available at: https://www.techtarget.com/searchcio/definition/project-planning#:~:text=Project%20planning%20is%20a%20discipline,scheduling [Accessed 8 Dec. 2022].

‌

Table

Description automatically generated Graphical user interface, application

Description automatically generated

CHAPTER 4 REQUIREMENT COLLECTION AND ANALYSIS

4.1 Foundational Research:

Foundational research is done before start designing. Within the product development life cycle, foundational research happens during the first stage to help empathize with users, understand their needs, and inspire new design directions. During this stage, personas and interviews are made. An Interview with Dr. Ali Alsfar been done to collect in-depth information on opinions, thoughts, experiences, and feelings about the application.

<https://www.coursera.org/learn/start-ux-design-process/supplement/IMKD6/learn-more-about-ux-research>

4.1.1 Interview

4.1.2 Persona

|  |  |
| --- | --- |
| Characteristic | Name: Dana  Age: 7  Education: Primary School  Hometown: Bahrain  Occupation: Childhood |
| Goals | * Easy and enjoyable user interface * Measure the distance between the camera device and a detected person |
| Frustrations | Predicting and measuring the safe distance are default processes |
| Scenario | Dana is a primary school student. She lives in a family house. When she plays with her relatives, she would like to know if she is keeping a safe distance or not. She still does not know how to predict two meters, so she uses the safe distance application. She allows the application to access the camera device and adjust the distance to two meters. She opens the camera and checks if she is keeping a safe distance between her and her relatives. |

4.2 Application Requirements

The foundational research sets clear expectations that were then converted into concrete functional and non-functional requirements.

4.2.1 Functional Requirements

Functional requirements are described as the way the system will behave when certain input criteria are met (ReQtest, 2012).

1. Displaying the camera scene on the device screen: The user should be able to sees what the camera faces on real time on the device screen.
2. Detect human using phone camera: Whenever a person appears in the camera scene, the application should be able to detect it.
3. Overlay a digital text or graph on the detected person: The application should be able to provide the user with feedback about the detected person. This feedback might be a text or a graph overlay over the detected person on the device screen.
4. Measure the distance using phone camera: The application should be able to measure the distance between the camera device and the detected person
5. Ability to change the safe distance: The user should be able to change the safe distance, which is a predefined value that used to trigger the alert whenever it is violated
6. Show alert massage when the safe distance violated: A noticeable alert when the distance between a detected person and user are less than predefined value.

4.2.2 Non-functional Requirements

Non-functional requirements address all requirements that haven’t been included in the functional requirements. They refer to the criteria for judging a system's functionality rather than behaviors (ReQtest, 2012).

Non-functional requirements

|  |  |
| --- | --- |
| **Non-functional requirement** | **Description** |
| Reusability | The system should have the ability to store some of its building blocks in a library so that other systems and products can also use them |
| Usability | The system is easy to use and navigate for first-time users |
| Scalability | The system is easy to add more functions to for enhancement |

CHAPTER 5 SYSTEM DESIGN

5.1 Design Research:

Within the product development lifecycle, design research happens during the design stage to help inform the designs, to fit the needs of users, and to reduce risk. One of the common methods used to conduct design research is a usability study, which is a technique to evaluate a product by testing it on users. The goal of usability studies is to identify pain points that the user experiences with your prototypes, so the issues can be fixed before the product launches.

<https://www.coursera.org/learn/start-ux-design-process/supplement/IMKD6/learn-more-about-ux-research>

Two rounds of usability studies were conducted, the first was moving from digital wireframes to mockups, the designs were inspired by the results of the low-fidelity prototype. The second research showed which elements of the mockups needed to be improved through the usage of a high-fidelity prototype.

|  |  |
| --- | --- |
| **Introduction** | * **Title:** Safe distance tracker * **Date**: 6/20/2022 * **Project background**: We’re creating a new app to help people keep a safe distance during COVID or in any situation might need to track their safe distance. * **Research goals**: Track the safe distance of the user, how will the results of the research affect your design decisions? Help to provide a more enjoyable and interactive experience for the users |
| **Research**  **questions** | * How long does it take for a user to check if he/she keeps a safe distance * What can we learn from the steps that users take to check the safe distance? * Are there any parts where users are getting stuck? |
| **Key Performance Indicators**  **(KPIs)** | * User error rate: if there any issues with the AR application * System usability scale: if app perform as expected |
| **Methodology** | * Unmoderated usability study * **Location**: Bahrain, remote (each participant will complete the study in their own home) * **Date**: Sessions will take place on March 12 (normal business hours) and March 13 (after hours) * **Length**: Each session will last 5 to 10 minutes, based on a list of prompts * 6 participants will use the app. Each participant will then complete a SUS on their experience. |
| **Participants** | * Participants should include children between 6 to 16 years old * Participants need to be interested in keeping a safe distance during COVID or any situation similar to that. * Participants should include:   + 2 girls   + 2 boys   + 2 adults (male and female) |
| **Script** | **During the unmoderated usability study**  A list of prompts appears on the device screen   * **Prompt 1:** Prepare your device to start the app   + Prompt 1 Follow-Up: How easy or difficult was it to do the task? Is there anything you would change about the process? * **Prompt 2:** Set your safe distance that you want to keep.   + Prompt 2 Follow-Up: How easy or difficult was it to do the task? Is there anything you would change about the process? * **Prompt 3:** Follow the guides to help you to be in safe distance.   + Prompt 3 Follow-Up: How easy or difficult was it to do the task? Is there anything you would change about the process? What do you think about the alert? * **Prompt 4:** Close the app.   + Prompt 4 Follow-Up: How easy or difficult was it to do the task? Is there anything you would change about the process?   **After the unmoderated usability study**  Participants will complete the System Usability Scale   * Participants will score the following ten statements by selecting one of five responses that range from “Strongly Disagree” to “Strongly Agree.”   + I think starting the camera was straight-forward.   + I find it clear when the app detects a person.   + I think accessing the settings was easy and clear.   + I think changing the safe distance was easy and clear.   + I find the alert obvious and clear.   + I find the app easy to navigate.   + There is inconsistency within the app.   + I think closing the camera was straight-forward. |

5.1.1 Low-Fidelity Prototype

Low-fidelity (lo-fi) prototyping is a quick and easy way to translate high-level design concepts into tangible and testable artifacts. It helps in checking and test functionality rather than the visual appearance of the product

Graphical user interface, application

Description automatically generated

Click path is a record of what path the participant took to complete the task. Observations are note about behaviors, opinions, and attitudes along with any errors, issues, or areas of confusion. Quotes are any significant quotes (positive and negative). In task completion: **1** stands for easy to complete, **2** stands for completed but with difficulty and **3** stands for not completed.

Example of the low-fidelity usability study notes.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Task** | **Click Path** | **Observations** | **Quotes** | **Task Completion** |
| Prompt 1: open the camera from the app | click on the button | the participant finished the task fast and smoothly | "It is the only button I can see" | 1 |
| Prompt 2: track your safe distance | moving around, while the camera is open | The participant got confused since the prototype does not provide a live camera scene | N/A | 2 |
| Prompt 3: change the safe distance | the options icon> change the safe distance | the participant did not understand the meaning of the icons | "if I can specify the unit would be better" | 2 |
| prompt4: violate your safe distance to active the alert | letting a person enter in the indicated safe zone | the participant understood the task clearly and implemented it smoothly | N/A | 1 |

5.1.2 Results of the lo-fi prototyping

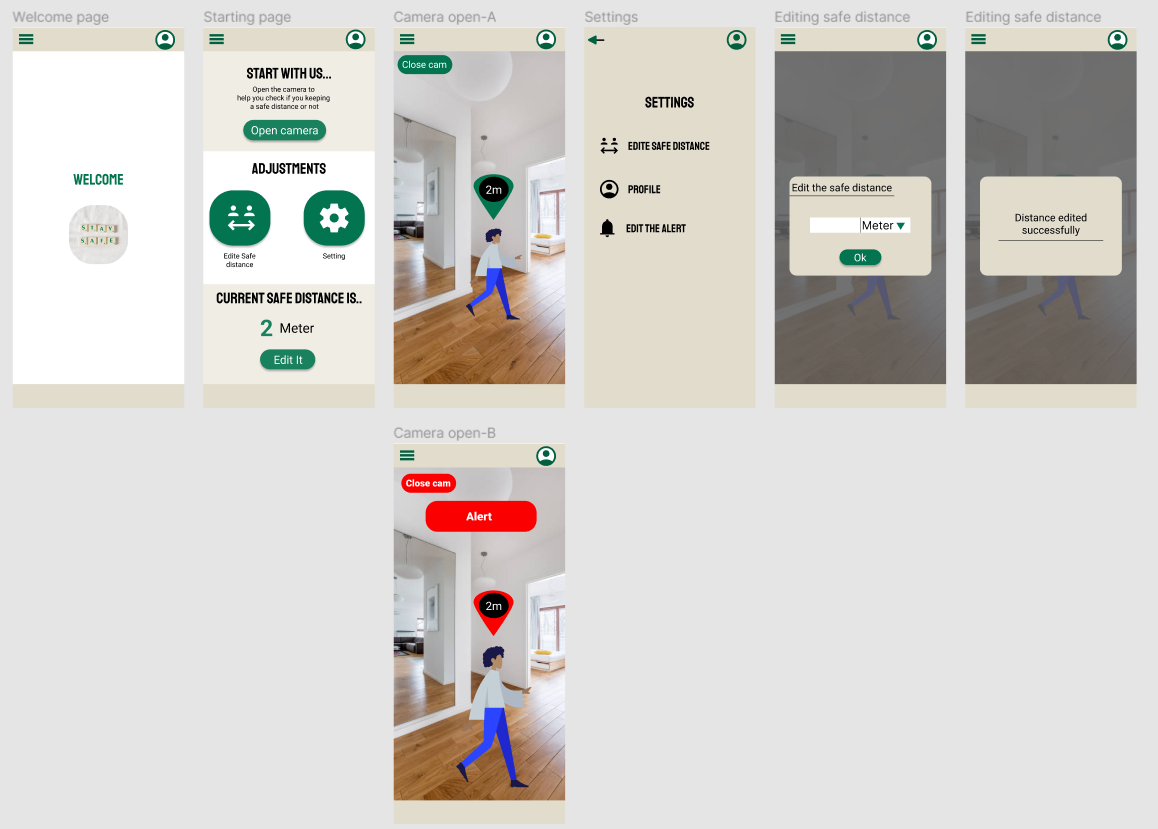
After conducting two round of usability studies, three main insights were derived and needed changes was applied on the prototype. The insights are:

1. Ability to change the measuring unit: Users should be able to recognize the measuring unit used in the application and should be able to change it.
2. A noticeable alert: Users need more interactive and noticeable ways to be alerted when safe distance violation happens
3. Know the current safe distance before opining the camera: Users need to know the current safe distance and access setting to change the safe distance before they start the camera.

Those insights considered and added to the functional requirement of the system to increase the usability of the application.

5.1.3 High-Fidelity Prototype

high-fidelity prototypes are highly functional and interactive. They are very close to the final product, with most of the necessary design assets and components developed and integrated. It helps in test usability and identify issues in the workflow.



Example of the high-fidelity usability study notes.

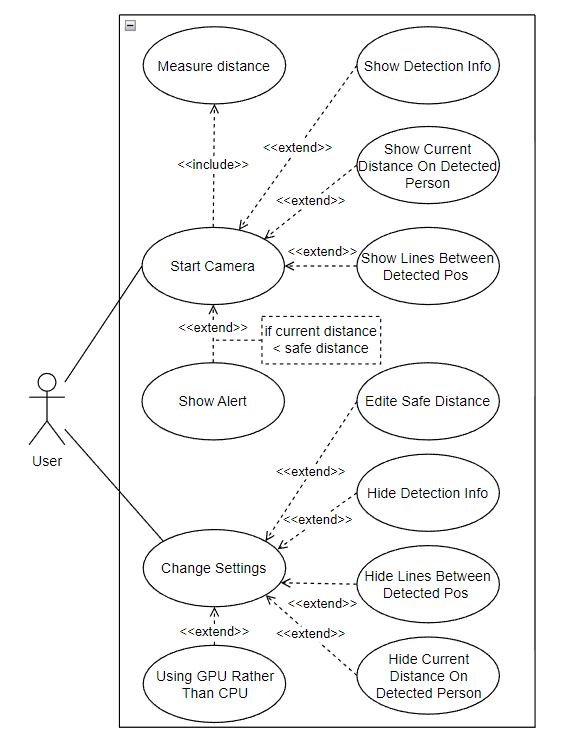
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Task** | **Click Path** | **Observations** | **Quotes** | **Task Completion** |
| Prompt 1: open the camera from the app | click on the button | the participant hovered the button then clicked on it | "It is not clear do I hover or I click" | 2 |
| Prompt 2: trach your safe distance | moving around, while the camera is open | The participant got confused since the prototype does not provide a live camera scene | N/A | 2 |
| Prompt 3: change the safe distance | N/A | the participant did not understand the meaning of the icons the participant was pressing on the line and the number that shows the current safe distance | "I can not see any setting or options" | 3 |
| prompt4: violate your safe distance to active the alert | letting a person enter in the indicated safe zone | the participant understood the task clearly and implemented it smoothly | N/A | 1 |

5.2 UML Diagrams

Unified Modeling Language (UML) is a standardized modeling language enabling developers to specify, visualize, construct, and document artifacts of a software system.

**5.2.1 Use case diagram**

Use case diagrams are used to show the usage requirements for a system. UML Use Cases are used for actual development to provide significantly more value because they describe the core of the requirements.



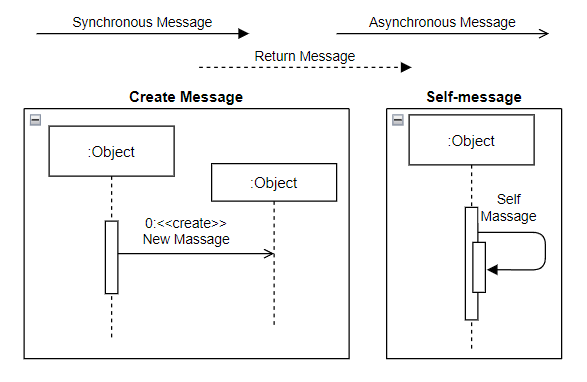
Start Camera: The user starts the camera device to measure the distance between the detected object and the camera. The user can show lines between person poses and the current distance between the detected person on this person. The user can show the detection information, like frame rate, image size, and detection latency. If the current distance between the detected person and the camera is less than the safe distance, an alert will be shown.

Change Settings**:** The user can hide detection information, lines between person poses, and the current distance that is rendered on the detected person. The user can use the device's GPU rather than CPU.

5.2.2 Sequence diagram

Sequence diagrams describe interactions among classes in terms of an exchange of messages over time. They're also called event diagrams. A sequence diagram is a good way to visualize and validate various runtime scenarios.

Sequence diagram components



**Synchronous Message**  
A synchronous message requires a response before the interaction can continue. It's usually drawn using a line with a solid arrowhead pointing from one object to another.

**Asynchronous Message**  
Asynchronous messages don't need a reply for interaction to continue. Like synchronous messages, they are drawn with an arrow connecting two lifelines; however, the arrowhead is usually open and there's no return message depicted.

**Return Message**  
A reply message is drawn with a dotted line and an open arrowhead pointing back to the original lifeline.

**Self-Message**  
A message an object sends to itself, usually shown as a U-shaped arrow pointing back to itself.

**Create Message**  
This is a message that creates a new object. Similar to a return message, it's depicted with a dashed line and an open arrowhead that points to the rectangle representing the object created.

<https://www.smartdraw.com/sequence-diagram/>

<https://www.researchgate.net/figure/Sequence-diagram-of-face-recognition-system_fig4_321166919>

Application Sequence diagram

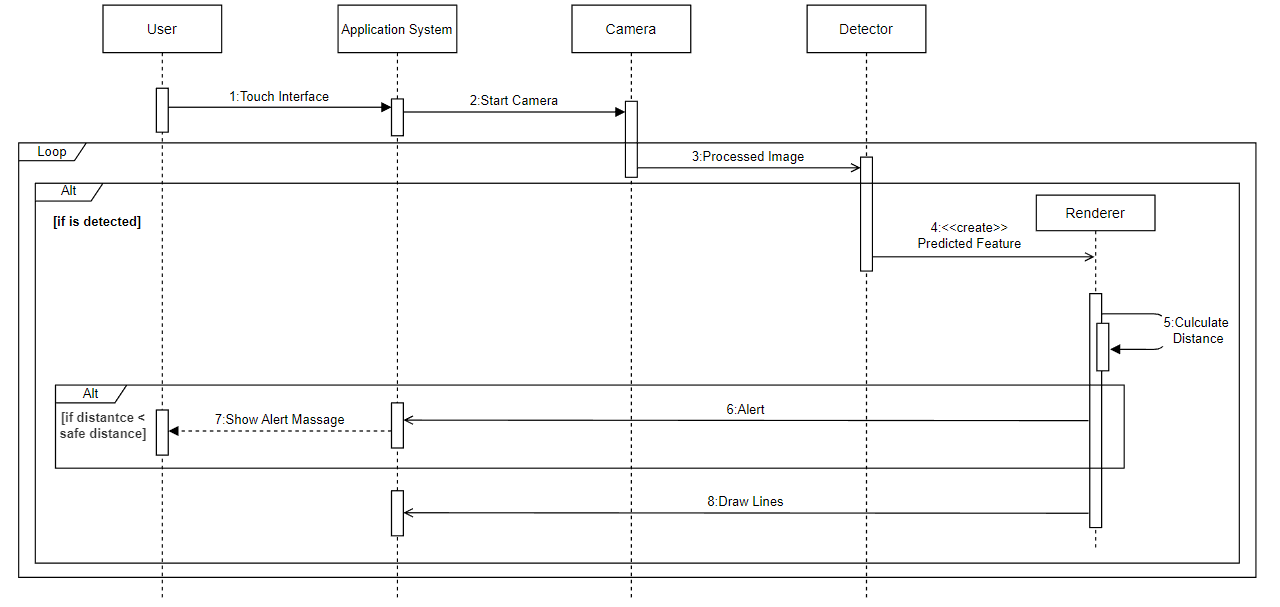


Figure X shows the four main objects used in the Measuring distance application. All four objects depend on user events. The application system the main logic that controls the application settings, permissions, and what the user sees. The camera object is responsible for accessing and communicating with the device camera. The detector object is responsible for detecting extracted features and getting their orientation in the image. The renderer object is responsible for calculating the distance between the past orientations and drawing lines between detected features.

Step 1: The user touches the screen to start the application.

Step 2: The application system opens the device camera.

Step 3: The camera gets live streaming data and processes them to match the detector input.

Step 4: If there is a detected object, the renderer gets created and receives the orientations of the detected object.

Step 5: The renderer calculates the distance between the detected object and the camera device.

Step 6: In case the calculated distance (in step 5) is less than the safe distance, the renderer will send an alert to the application system.

Step 7: the application system shows the user an alert message.

Step 8: If there is a detected object, The renderer draws lines on the detected person.

CHAPTER 6 SYSTEM IMPLEMENTATION AND TESTING

System implementation is the process of specifying how the information system should be developed, verifying that the information system is operational and being utilized, and ensuring that the information system fulfills quality standards.

We implemented three versions of the application. The first application differs from the other in the way that it understands the environment and measures the distance, which will be discussed in section 6.4. The difference between the other two applications is that one is a web-based application and the other is an Android application.

|  |  |  |
| --- | --- | --- |
| Application | Camera Tool | Detection Tool |
| Single Image Android (first application) | AR Core | ML Kit |
| Stream Android | CameraX | ML Kit |
| Web-based | OpenCV | TensorFlow model |

6.1 Software Tools

1. Visual Studio Code

Visual Studio Code, also known as VS Code, is a free and open-source editor. It is compatible on many platforms and operating systems including Linux, macOS, and Windows. It also supports numerous programming languages such as Python, HTML, PHP, GO and many more. VS Code allows you to add extensions such as debuggers which improve user experience (Mustafeez).

1. Figma

Figma is a free, online user interface design application. It helps developers better visualize the system and its flow by creating an interactive design. It runs entirely on a browser which means that projects will be easily accessible anytime, anywhere (themejunkie).

1. GitHub

GitHub is a version control and collaboration tool for programming. It allows people to easily collaborate on projects from any location (GithubDocs).

1. Android Studio

Android Studio is the official integrated development environment for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for 48 Android development. It is available for download on Windows, macOS, and Linux based operating systems. We used Android Studio to build our application into Android devices.

1. CameraX

CameraX is an Android library makes complex camera functionality available in an easy-to-use API. It emphasizes use cases, which allow you to focus on the task you need to get done instead of managing device-specific nuances. Most common camera use cases are supported, Image analysis to access a buffer seamlessly for use in your algorithms, such as to pass to ML Kit, Image capture to save images and Video capture to save video and audio. In our project, we used two of the supported use cases, preview and image analysis. Preview use case allows view an image on the display. Image analysis use case provides the access a buffer seamlessly for use in machine learning detector.

<https://android-developers.googleblog.com/2022/08/camerax-12-is-now-in-beta.html>

1. ML Kit

ML Kit is a mobile SDK that enables you to add powerful machine learning features to a mobile application. It supports both Android and iOS and offers the same features for both platforms. The SDK is part of Firebase and bundles together various machine learning technologies from Google. This SDK comes with a set of ready-to-use APIs for common mobile use cases such as face detection, text recognition and pose detection.

* Pose Detector

The ML Kit Pose Detection API is a solution to detect the pose of a subject's body in real time from a continuous video or static image. A pose describes the body's position at one moment in time with a set of skeletal landmark points. The landmarks correspond to different body parts such as the shoulders and hips. We used the distance between shoulders and hips to get the distance between the device camera and detected person. Pose detection can only detect one person in an image. If two people are in the image, the model will assign landmarks to the person detected with the highest confidence, this will cause an issue will be discussed in …

<https://auth0.com/blog/a-look-at-android-ml-kit-the-machine-learning-sdk/>

<https://developers.google.com/ml-kit/vision/pose-detection>

1. ARCore
2. SceneForm
3. OpenGL
4. OpenCV
5. TensorFlow-hub
6. Flusk

6.2 Programing Languages

1. Java

Java is a programming language and computing platform first released by Sun Microsystems in 1995. It has evolved from humble beginnings to power a large share of today’s digital world, by providing the reliable platform upon which many services and applications are built. New, innovative products and digital services designed for the future continue to rely on Java, as well.

<https://www.java.com/en/download/help/whatis_java.html>

1. Kotlin

Kotlin is a general purpose, free, open source, statically typed “pragmatic” programming language initially designed for the JVM (Java Virtual Machine) and Android. It combines object-oriented and functional programming features. Kotlin originated at JetBrains, the company behind IntelliJ IDEA, in 2010, and has been open source since 2012. It is focused on interoperability, safety, clarity, and tooling support.

<https://www.infoworld.com/article/3224868/what-is-kotlin-the-java-alternative-explained.html>

6.3 Development

6.3.1 Handling Camera

**CameraX Use Cases**

First, we create a CameraX instance, `cameraProvider`, and bind it to the lifecycle of the application. The `cameraProvider` will be used to bind the CameraX use cases to the application lifecycle. We use ViewModelProvider which provides ViewModels. ViewModel is a class that is responsible for preparing and managing the data for an Activity. Then, we use CameraXViewModel class which is responsible for interacting with CameraX to access processCameraProvider. The processCameraProvider is a method of the CameraXViewModel class that returns the LiveData that can be observed over the application lifecycle.

We create the builder that will be used to create the preview use case. We can attach some settings to the builder like the image resolution, that can be changed by the user. The previewUseCase is of type Preview which is a camera preview stream for displaying on-screen. Then we attach to the use case a Surface Provider. We are using previewView which is a custom View that displays the camera feed for CameraX's Preview use case. The previewView class manages the Surface lifecycle, as well as the preview aspect ratio and orientation. Internally, it uses either a TextureView or SurfaceView, classes in Open Graphics Library (OpenGL) renderer, to display the camera feed. Finally, we bind the use case to CameraX instance, cameraProvider.

**OpenCV**

We create an instance of OpenCV with a specified camera lens that allows accessing the current frame and processing it. After processing the frame, we save it as a buffer and yield it to the front end. we use yield since we return a value continuously while executing the function and this value is only needed once.

6.3.2 Object detection and rendering

**ML Kit Pose Detector**

We set the options of the poseDetectorProcessor in the CameraXLivePreviewActivity, then pass them to PoseDetectorProcessor. PoseDetectorProcessor is an instance of VisionProcessorBase that extracts the result of the detected poses and classifies them. The pose detection and tracking process are done in PoseDetectorProcessor and its patent VisionProcessorBase. The result is of a type of Pose and it is passed to the PoseGraph file. The Pose class has a method called getAllLandmark that returns a list of all detected landmarks. The getAllLandmark method will be used to draw some landmarks between the poses that are used to measure the distance. All those tasks are done asynchronously, and their results are handled using listeners.

**TensorFlow-hub Movnet model**

We load the Movnet model using TensorFlow-hub API. Movenet Multi-pose Lightning model is a convolutional neural network model that predicts the human pose locations of people in the image. The model was designed to be run in the browse (Tfhub.dev, 2022). Then we process the image to fit the Movnet model input specification using TensorFlow API. After passing the frame to the Movnet detector we get the orientation of the detected poses for each person in the frame. We use those orientations to draw and render points and landmarks on the detected person using OpenCV API.

6.3.3 Distance measurement

We measure the distance based on 2D understanding. We build an understanding of the actual scene based on the distance between two points in an image. For example, the railroad ties appear to get smaller as they get further away from us. If we measured the apparent size of each railroad tie, their measured size would decrease in proportion to their distance from our eyes. The same thing happens to the distance between shoulders and hips.

A picture containing text

Description automatically generated

In order to calculate the distance between the camera and the detected person, we calculate the distance between the shoulders and hips. We could use the pupillary distance (PD), the distance between the centers of your pupils since it is fixed as the detected person moves, but it will be affected if the detected person did not face the camera. We used the distance between the shoulders and hips, considering if the detected person rotated.

We get the average of the distance right side and the left one, then using a polynomial equation we get how far the detected person is from the camera. The coefficients of the polynomial equation calculate by measuring the size of landmarks, the distance between shoulders and hips, to how far a person is from the device camera.

6.4 Single image version

This is the first version of the application. It is an Android application. We tried to measure the distance between the camera and any point in the scene using ARCore and SceneForm. In this approach measures the distance based on a 3D understanding of the environment. Using ARCore API we can get the position of the camera compared to the scene and the position of, approximately, any other point in the scene. The issue was with detecting an object or a person on a stream of images.

We used the ML Kit Pose detection to detect the human body, and it is the same tool that will be used in the other android versions of the application. Integrating the ARCore part with ML Kit part will require managing the ARcore inputs, images, and passing them to the ML Kit part. One of the approaches was to use OpenGL. We were able to implement this approach on a single image, but we could not implement it on a stream of images. One of the main characteristics of java language is that it uses a garbage collector, so we have to save the image buffer into the heap. It seems that applying the detection on a stream of images is possible but will require more understanding of OpenGL. Learning OpenGL exceeded the estimated time and considering the risks effects and strategies, we adopted measuring distance based on 2D understanding which was inspired by the same book used to learn the OpenGl[].

7 Testing

7.1 Unit testing

Unit Testing: Unit Testing the level of software testing where individual units/components of the software are tested. The purpose is to validate that each unit of the software performs as designed.

|  |  |  |
| --- | --- | --- |
| Test Case | Expected Result | Actual Result |
| Asks for permissions | The application asks for camera and internet permissions, it will show a pop-up window to get the user approval. |  |
| Start the camera | When the user clicks on start the application camera, the application will take the user to the camera activity. |  |
| Stream camera images | The camera activity will show a stream of images of the current scene. |  |
| detect a person | The application shows a message when a person got detected |  |
| Render landmarks | The application renders landmarks between the shoulders and hips of the detected person |  |
| Render the current distance on the detected person | The application renders the current distance between the detected person and the camera |  |
| Show the alert message | The application shows an alert message when the person violates the safe distance |  |
| Access settings | When the user clicks on the settings icon, the application will take the user to the settings activity |  |
| Edit the safe distance | The user should be able to change the safe distance |  |
| Stop rendering landmarks | The user should be able to control showing landmarks |  |
| Stop rendering the distance on the detected person | The user should be able to control showing the current distance on the detected person |  |
| Use GPU instead of CPU | The user should be able to control using the device’s GPU instead of CPU. |  |
| Close the camera | When the user clicks on close the camera, the application will take the user back to the entry activity |  |

8 CONCLUSION AND FUTURE WORK

Finally, three applications have been created during this project that can calculate the distance between a detected person and the device camera. The project aims to help users to keep a predefined distance between people around them. The project presented two approaches for measuring the distance using the device camera. The first approach is based on 3-dimensional understanding of the environment. The second approach is based on 2-dimensional understanding of the image. During this project, two applications were developed for Android devices, and another application was developed using the Flask framework. The possibility to use and contribute to a larger journey of a promising technology like augmented reality was a fantastic opportunity.