UOW MALAYSIA KDU

SCHOOL OF COMPUTING & CREATIVE MEDIA

SCCM/CSForm/Rev. Ver 2.4

Please tick ✓ or click if using MS WORD

☐FOUNDATION ☐DIPLOMA ☒ DEGREE ☐ MASTER

Assignment Coversheet

Please complete all details required clearly. For softcopy submissions, please ensure this cover sheet is included at the start of your document or in the file folder.

Assignment & Course Details:

Subject Code: *(e.g. XCAT1234)* **XBMC2024N**	**Subject Name** *(e.g. Fundamentals of Computing)*: **Final Year Project 2**
Course *(e.g. Bachelor in Computing)* : **Bachelor of Computer Science (Hons)**	
Supervisor Name: **Ms. Bridget Merliza** **Lecturer Name:** **Ms. Siti Fazilah Shamsudin**	

Assessment Due Date: *(dd/mm/yy)*	24/03/2024	**Assessment Title:**	FYP2 Documentation

I/We declare that:

- *This assignment is my/our own original work, except where I/we have appropriately cited the original source.*
- *This assignment or parts of it has not previously been submitted for assessment in this or any other subject.*
- *I/We allow the assessor of this assignment to test any work submitted by me/us, using text comparison software for plagiarism.*
- ***(For more information, Please read the Academic Integrity Guidelines)***

Name : Nicholas Dylan
Student ID: 0133646
Email : 0133646@uow.student.edu.my
Mobile No: 01111385378 **Signature:**
Date: 23/03/2024

For office use only – Lecturer comments (if applicable)

Marks Breakdown

PART OF THE UNIVERSITY
OF WOLLONGONG AUSTRALIA
GLOBAL NETWORK

AR Enabled Assembly Guidance System

Nicholas Dylan

(0133646)

A project submitted in partial fulfillment of the requirements for the award of the Bachelor of Computer Science (Hons)

Supervised by **Ms. Bridget Merliza**

School of Computing and Creative Media

University of Wollongong Malaysia

2024

Table of Contents

List of Tables

List of Images

Chapter 1: Introduction

1.1 Background

In order to ensure the continuity of small and medium manufacturing industries, it is crucial that they adopt and implement technology in their manufacturing process, particularly in the assembly phase, in order to improve productivity, flexibility, and quality. Equally important is providing adequate training to workers to keep up with the changing demands, as their skills directly impact their performance and the company’s performance. Additionally, the rise of Industry 4.0 has made manufacturing more complex, leading to an increased need for higher skilled workers.

Regrettably, the training process for assembling nowadays is mostly reliant on text-based guidelines, documented in the hard copy form, which may be bulky and cumbersome to carry around, hence hindering worker’s performance (Kühn-Kauffeldt & Böttcher, 2020). One of the ways manufacturing companies may increase efficiency in training but still provide adequate training at the same time is by implementing AR during the production process, especially the training process.

Together, both AR may overcome the downside of text-based training, which is the lack of visualization, which leads to misinterpretation, especially on more complex processes. On top of that, AR also may enable significant savings of time (Fast-Berglund et. Al., 2018) and expenses in areas such as error rectification.

This paper will attempt to develop and design an AR based system for assembling training purposes and, subsequently, reviewing the effectiveness of AR in comparison towards text-based guidelines.

1.2 List of Problems

1.2.1 Training guidelines can be bulky

Printed training guidelines consist of pages which make it bulky. Carrying it around may be cumbersome to the workers.

(Kühn-Kauffeldt & Böttcher, 2020)

1.2.2 Text-based only guide can be confusing

Without the help of additional visualization such as images and videos, it might be hard for the workers to comprehend the steps, especially for complex processes. This might cause more errors to happen.

(Lai et al., 2020)

1.2.3 Printed guidelines may increase expenses and waste resource

The increasing number of errors also means more expenses and resources are required in order to repair the errors.

(Botto et al., 2020)

1.3 Problem Statement

There is a high demand for manufacturer workers everywhere, even Kedah, needs 25 thousand workers for its Kulim Hi-Tech Park alone. Unfortunately, the predominantly training methods currently used, text-based guidelines, are inefficient and prone to misinterpretation. This seems to be the custom in the industry, especially small-scale industry, despite the fact that it slows down the training process whenever a certain page is needed. Furthermore, it tends to be bulky and cumbersome to carry around (Kühn-Kauffeldt & Böttcher, 2020). Moreover, text alone may not communicate the process effectively enough (Douglas, Kimbaris, & Stein, 2021), causing errors, leading to wastage of resources for rectification (Lai, et al., 2020; Botto, et al., 2020). With the widespread availability of smartphones, Augmented Reality (AR) offers a potential solution to improve the training process by helping to visualize the steps of the process with media like images or videos. A possible cause for the lack of implementation of AR in the manufacturing industry to train new assemblers are due to the cost and the resources to develop. Perhaps a study which investigates how AR may help the manufacturing industry in the assembly training phase could help to resolve this.

1.4 Problem Solution

In order to alleviate the problem, an AR-enabled guiding system for assembling and training processes can be implemented. It offers extra visualization for guidelines, enabling workers to learn how to assemble products more effectively. Resulting in enhanced understanding, reduced errors and expenses at the same time. By adopting AR-enabled guiding system, manufacturing industries can streamline the onboarding process for new assemblers and upskill existing assemblers. With the availability of smartphones nowadays this system can be easily accessible by the assemblers.

1.5 Research Question

1. What is the current method of training new assemblers?
2. How can the AR system help the process of assemblers training?
3. Can the effectiveness and efficiency of AR improve the training?

1.6 Research Objective

1. To investigate the current method and techniques used to train new assemblers.
2. To design and develop AR applications for the manufacturing industry for assemblers training.
3. To validate the effectiveness and efficiency of AR application for manufacturing industry for assemblers training.

1.7 Project Scope

Level	**Function**	**Description**
Basic	The system should be able to be accessed from the client's devices.	Hosting a server on a LAN (Local Area Network) allows devices within the same network to connect and access the services provided by the server.
Intermediate	The system should be able to tell the users the steps required.	Show pop up which tells the user the next steps necessary in order to finish the manufacturing process.
	The phone and the server are able to make a connection to send and receive media streams from the phone's camera.	The assembler's phone should be able to make a connection to send a media stream to the server, where it can be processed, and sent back to the phone.
Advanced	The system should be able to run model inference on the media stream received and detect the components.	Once the server receives the media stream or frames from the phone's camera, the server will run model inference on it and segment out the components.

Table 1.7.1

1.8 Conceptual Framework

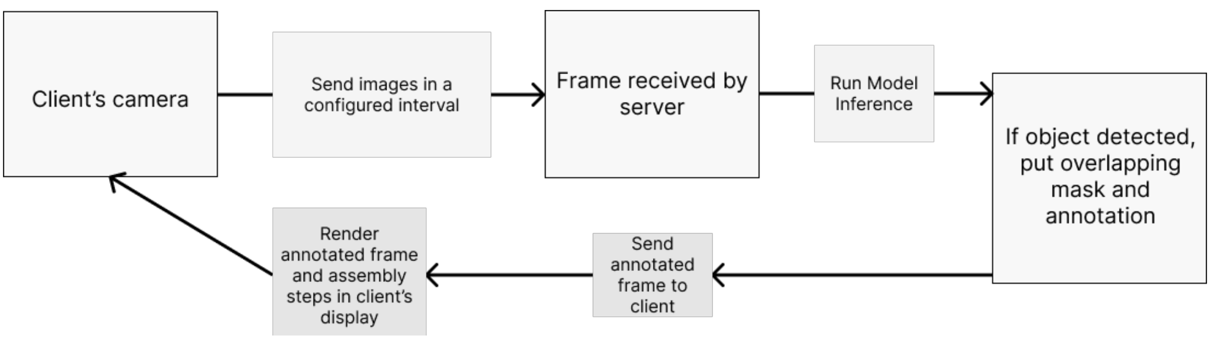


Figure 1.8.1

1.9 Project Significance

1. Manufacturers employing the assemblers

This project can benefit manufacturers as implementing AR might reduce the errors made by new assemblers, hence reducing overall expenses.

2. New assemblers

By implementing AR, new assemblers may feel more assured and comprehend the steps easier with the help of additional visualization.

3. Augmented Reality (AR)

Nowadays, AR is more prominent in the gaming industry as we can see on games such as Pokémon GO, however, a successful implementation of this system can prove that AR can also be useful in other industries, to be specific manufacturing.

1.10 Hardware and Software

Hardware	For Assemblers: Mobile devices that are equipped with a camera and ability to connect to the internet For Admin: Desktop that can connect to the internet	Justification: Camera will be utilized to see the components and internet will be required to access the server hosted in within the LAN.
Software	1. Django Framework 2. Python 3. Visual Studio Code 4. MacOS Ventura 13.3 or Windows 10	Justification: These are the software that will be used in order to develop this system

Table 1.10.1

Chapter 2: Literature Review

2.1 Introduction

This literature review will explore significant aspects which are related to the training of new manufacturing workers. The points that will be discussed here will cover points including knowledge domain, technical domain, methodology, qualitative and quantitative analysis, research instruments, existing systems and also future improvements.

2.2 Knowledge Domain

2.2.1 Paper based guidelines

Paper based guidelines are often used in a training process. These text-based printed guidelines are often bulky, making it cumbersome to carry around. On top of it, paper-based guidelines lack interactivity and visualization, making it challenging for trainees to actively engage with the content or to understand it. Some information needs to be transferred orally and may need additional action from experienced assemblers.

2.2.2 Augmented Reality (AR)

Augmented Reality or AR is a technology that is able to combine virtual information with the real world, enabling the display of virtual objects, either 2D or 3D, in real time. This functionality enables AR to be able to improve the interaction between humans and machines by improving the visual perception of the actual physical environment (Bottani & Vignali, 2019). Even the European Union classified AR as one of the main technologies which will drive up the development of smart factories. Nowadays AR is also more easily accessible due to the availability of smartphones, hence making the adoption of AR easier.

2.2.3 Visual Learning

Visual learning can be defined as assimilation of information from visual formats. Visual information can be presented in different formats such as images, videos, diagrams, slide shows, games, flowcharts, or even a 3D object. Visual learning may also encourage visual thinking among students, where they become better in understanding by associating ideas with visualization.

2.2.4 User Experience (UX) Design

UX refers to how the user perceives and feels regarding the product's usability, system, or services (Fariyanto, Suaidah, & Ulum, 2021). The primary aim of UX Design is to make sure that each user experiences a positive interaction with the system.

2.2.5 Workplace Training

Training itself is a process of inculcating a specific skill or competencies in a person. Meanwhile, workplace training is a systemic approach to learning and development in order to improve an individual's, a team's, or an organization's effectiveness. It is a prime chance for assemblers to expand or improve their skill or knowledge. The learning that assemblers participate in to improve performance may be called workplace training. It refers to activities such as on-the-job instruction or classroom-based courses (Kraiger & Ford, 2021).

2.2.6 Review of Knowledge Domain

As reviewed above, paper-based guidelines are a hassle to carry around and it is hard to understand the process without any visualization aid, hence an AR-enabled assembly guidance system can be a viable solution for this issue as it can improve visual learning. Together with workplace training and UX design, an AR-enabled assembly guidance system may help reduce the expenses by reducing the cost by reducing errors needing rectification and the need for an experienced assembler to oversee the training.

2.3 Technical Domain

2.3.1 Integrated Development Environment (IDE)

2.3.1.1 Replit

Replit is an online integrated development environment which can be accessed at www.replit.com. It was founded in 2016 in San Francisco by Amjad Masad, Haya Odeh, and Faris Masad. It allows users to code in multiple programming languages including Python, C, C#, Node.js, Basic, and more. Replit also supports HTML, CSS, and even JavaScript webpage code. Replit offers a free tier and also a paid tier which starts at US$7.00/month.

2.3.1.2 Visual Studio Code

Visual Studio Code, also commonly known as VS Code, is an integrated development environment from Microsoft, and it is available for Windows, Linux and macOS. With it, developers can develop applications for Android, iOS, Windows, as well as web applications and cloud services.

It can be used to write, edit, debug, and build code, and then deploy your app. It is free to use and it is able to support extensions. On top of that, a study stated that around half of StackOverflow's survey participants are using VS Code (Plainer, 2021).

2.3.1.3 Sublime Text

Sublime Text was launched on January 18, 2008. It is a source code and text editor which supports many programming languages and markup languages and it is available for Windows, macOS, and Linux. It can be used for free but users will need to pay for continued use as the website stated. Third-party plugins may be installed in order to expand the functionality. It includes features such as Python-based plugin API, extensive customizability via JSON settings files, etc.

2.3.1.4 Review of IDE

	Replit	Visual Studio	Sublime Text
Language support	Support for popular languages such as Python, JavaScript, etc.	Built-in support for popular languages such as JavaScript, CSS, and HTML	Supports popular programming languages such as Python and also front-end languages such as HTML
Platform supported	All devices with a browser and the ability to connect to the internet	• Windows 10 and 11 • OS X High Sierra • Linux (Debian): Ubuntu Desktop 16.04, Debian 9 • Linux (Red Hat): Red Hat Enterprise Linux 7, CentOS 7, Fedora 34	• Windows 10/11 • Windows 8/8.1 • Windows 7 • macOS 10.12+ • OS X 10.9+ • Linux (32 bit)
Pricing	Free tier available with limited resources (0.5 vCPU, 1GB RAM, storage) and code need to be public.	Free for both private and commercial use	Offers free trial, users may pay to make the prompt to pay go away for US$99.00.

	Paid tier is starting from US$7.00/month.		
Collaboration support	Yes	Yes	No
Extension support	Yes, but limited as it is just released in May 2023	Yes	Yes
Auto code completion	Yes	Yes, with IntelliSense	Yes
Others	1. More user friendly 2. All work is automaticatically saved in the cloud 3. Easy access to term	1. Feature-rich IDE 2. More extensions available 3. Customizable syntax highlighting 4. Offers debugging option 5. Easy access to the terminal	1. Lightweight 2. Low-memory consumption

Table 2.3.1.4.1

Based on the comparison above, Visual Studio was chosen to be the integrated development environment for this research. This was done as Visual Studio is fully free to use and it has a lot of extensions available to be used such as Prettier. Also, it offers easy access to the terminal, something that Sublime Text lacks. On top of that, around half of StackOverflow's users are using Visual Studio Code, this ensures the resource regarding Visual Studio Code can be found easily.

2.3.2 Object Detection Algorithms

In order to detect the components required, an object detection practice needs to be implemented inside the system.

2.3.2.1 Ultralytics YOLOv8

As of the writing of this documentation, the latest version of YOLO is Ultralytics YOLOv8. The main selling point of the YOLO model is its relatively high speed and accuracy. Originally launched in 2015, it has been improved and enhanced repeatedly. In 2022, YOLOv6 was then made open-source by the company Meituan. Currently, YOLOv8 supports a full range of vision AI tasks, including detection, segmentation, pose estimation, tracking, and classification. A study from Sweden even stated that YOLOv8 can recognize objects 25.6 times faster than humans (Sjöberg & Hyberg, 2023).

2.3.2.2 SSD

SSD, short for Single Shot Multibox Detector, is an object detection algorithm that achieves almost real-time processing capabilities while still maintaining high accuracy. It is based on the VGG-16 architecture, due to its strong performance in high quality image classification tasks (Xie, et al., 2017). However, the limitation of SSD is that it can only produce bounding boxes as its output, rendering segmentation impossible.

2.3.2.2 Mask R-CNN

Mask R-CNN is based on Feature Pyramid Network (FPN) and a ResNet101 backbone. According to a study (Cheng, et al., 2020), Mask R-CNN is flexible and effective, hence it serves as a state-of-the-art baseline. Moreover, it is capable of also doing image segmentation. However, a study done in 2018 suggested that a fairly large dataset is required in order to produce a model with good accuracy (Anantharaman, et al., 2018).

2.3.2.3 Review of Algorithms

	YOLOv8	SSD	Mask R-CNN
Segmentation support	Yes	No	Yes
Specialty	Fast and lightweight	Fast and lightweight	High accuracy

Table 2.3.2.3.1

A study showed that with a small dataset of 500 images, YOLO performed faster and more accurately compared to Mask R-CNN (Sumit, et al., 2020). The study recommended a larger dataset to be used in order to increase Mask R-CNN accuracy. However, larger dataset requires more resources such as time and computer resources. SSD is not a viable solution due to its inability to segment images. Hence, YOLOv8 is chosen to be the algorithm that will be utilized for object detection.

2.3.3 Library and API for AR

For the AR training system development, a set of libraries will be utilized. A library refers to a collection of pre-written codes which developers can implement in order to optimize tasks. Examples of libraries for AR development includes:

2.3.3.1 AR.js

AR.js is one of the most popular libraries for AR on the Web that is available. One of the advantages of AR.js is that it is open-source and free to use. Other than that, it is lightweight and supports features such as location-based AR, marker tracking, and image tracking. AR.js believes that by having AR on the Web, it can reduce the expense of development of an app. Additionally, users are not required to download an AR app and waste their storage. Regarding the system requirements, AR.js will work on any devices (preferably phone) that supports webgl and webrtc.

2.3.3.2 A-Frame

A-Frame is a web framework that enables VR experience on the Web, but with updates, currently A-Frame has an out of the box support for AR for browsers with support for ARCore and ARKit. A-Frame was originally created within Mozilla and now it is maintained by the co-creators of A-Frame within Supermedium. HTTPS is needed in order for WebXR support. A-Frame also supports other libraries or frameworks such as Vue.js, Preact, D3.js, React, etc.

2.3.3.3 WebXR

The WebXR Device API provides the access to input and output capabilities which is commonly associated with VR and AR devices. With it, it enables development and the ability to host VR and AR experiences on the web, either on desktop or even on mobile. For desktop clients, the XR hardware will usually be a headset such as Microsoft Hololens, or for mobile clients it may be represented in the mobile phone itself along with its viewer harness for example Samsung Gear VR or Google Cardboard.

2.3.3.4 ARCore

ARCore is an AR SDK that is owned by Google. It offers cross-platform API to build AR experience on devices running Android, iOS, Unity, and the Web (need addition of WebXR). It offers features such as motion tracking, anchors that ensure an object's position over time, detects environment size and location, light estimation, and also depth measuring. This Google's AR tools and solution itself is already implemented by companies such as Mattel, NBA, Gap, Singapore Tourism Board, etc. Unfortunately, its native functions do not support AR on the Web as it is meant to be implemented in Android applications developed using Java or Kotlin.

2.3.3.5 ARKit

Apple with its ARKit 6 offers some features which cannot be found on the others including support for 4K video and support for front and back camera usage at the same time, so that users will be able to record their AR experience in an improved quality and experience. Just like Google's ARCore, it also supports Depth API which measures distances. Unfortunately, ARKit is only supported by Apple's devices which are iPhones and iPads. On top of that, natively, it does not support AR on the Web as it is meant to be implemented in iOS or iPadOS applications.

2.3.3.6 Review of Libraries, APIs, and SDKs for AR

	AR.js	A-Frame	WebXR	ARCore	ARKit
Support for AR on the Web	Yes, native feature	Yes, native feature	Yes, native feature	Not natively	Not natively
Main language	JavaScript	JavaScript	Primarily JavaScript	Kotlin/Java	Preferably Swift
Platform supported	Web-based	Web-based	Web-based	Android, Unity, and iOS	iOS and iPadOS devices
Pricing	Free and open-source	Free and open-source	Free	Free	Free
Distribution	Web	Web	Web	Google Play Store	Apple App Store
Marker-based support	Yes	Not natively	Not natively	Yes	Yes
3D Object recognition	Yes	Yes	No	No	Yes

Table 2.3.3.6.1

Through this literature review, comparison was made and it was decided that for the development of the AR-enabled assembly guidance system, the library AR.js and WebXR simultaneously, due to the flexibility and open-sourceness. On top of that, we would like to make our system available for everyone, regardless of the OS they are running, something that ARCore and ARKit does not really support. Furthermore, we would like our system to have location-based and marker-based AR, something that only AR.js could offer. On top of AR.js, we will implement WebXR to help us bring the AR system on to the Web.

2.3.4 Library for Visualization

2.3.4.1 Three.js

Three.js is an open-source JavaScript library and also API which allows for the creation and also the ability to display animated 3D computer graphics on the Web. Initially released in April 2010 in GitHub by Ricardo Cabello, Three.js is built on top of WebGL, a low-level API for rendering 3D graphics on the web. Several reasons contribute to the popularity of Three.js, one of them is because it is cross-browser compatible, so developers can be ensured that their graphics will be able to be displayed in popular browsers such as Chrome, Firefox, Safari, and even Edge (Johansson, 2021). As long as the browser is supported by WebGL 1.0, Three.js will work just fine on them.

2.3.4.2 Konva.js

Konva is also a JavaScript library. During the observation (July 26, 2023), Konva has gathered a respectable amount of almost 10-thousands of stars on its GitHub page. With Konva, developers can easily implement high performance animations, transitions, node nesting, layering, and others for both desktop and mobile clients. Unfortunately, Konva does not possess the ability to create 3D content, but it only supports 2D content.

2.3.4.3 Unity

Unity was first announced and released in 2005 as a Mac OS X game engine. It is more well known as a platform for 3D game development purposes. However, Unity can also be used to create and also operate real-time 3D content. Regrettably, unlike Three.js, Unity is not open-source and its Personal free tier lacks crucial features such as Unity Mars authoring tools for AR.

2.3.4.4 OpenCV

OpenCV is a versatile computer vision library. By importing the necessary libraries and loading the target image, OpenCV can also be utilized to create an overlapping mask to selectively reveal or hide specific regions based on defined criteria, or in this case, to highlight the components. Previously, OpenCV has been utilized in order to create AR systems by highlighting certain objects in the image (Amato, Venticinque, & Martino, 2013).

2.3.4.5 Review for library for 3D graphics

	Three.js	Konva.js	Unity	OpenCV
3D visualization	Yes	No	Yes	Yes
Pricing	Free	Free	Free, but limited, paid tier starting at US$399/year	Free
Compatibility	Most browser that supports WebGL 1.0	Most browser except IE11	Unity WebGL does not support mobile device	Compatible with desktop and mobile device

Table 2.3.4.5.1

Due to the needs of this system, which is to simply highlight the components required for the assembly process, we determined that OpenCV is the one that suits our needs the best. Konva was not suitable for our usage as it only supports 2D model development. Meanwhile Unity was not chosen due to its pricing and the incompatibility with mobile devices. Additionally, Three.js was not chosen as using 3D graphics to display information can be resource intensive, leading to higher operational cost.

2.4 Software Development Life Cycle methodologies

There exist a variety of SDLC models which could accommodate different needs or objectives. SDLC model itself can be defined as a model that describes the overall area of how software development is happening by defining each of the phases (Gurung, Shah, & Jaiswal, 2020). The following are a few examples of software development process models.

2.4.1 Agile

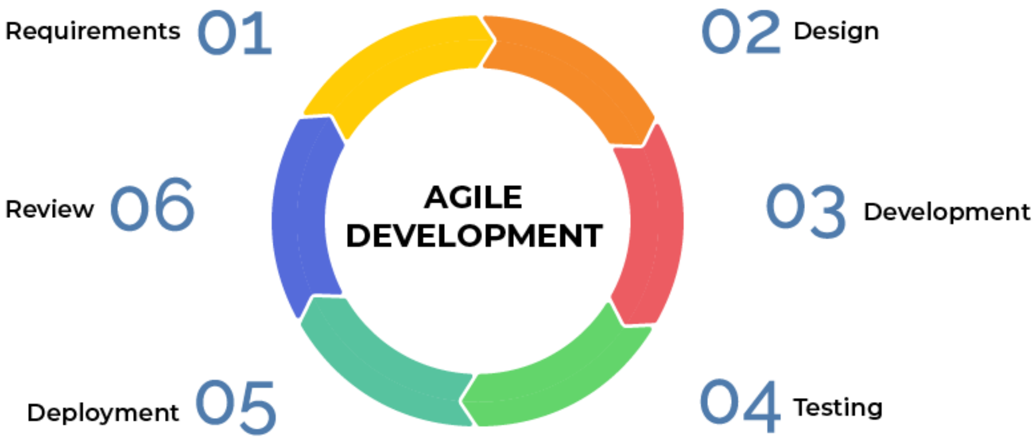


Figure 2.4.1.1

An agile or scrum SDLC methodology is based on collaborative decision making between the requirements and the solutions teams. It prioritizes individuals and interactions or communications over processes and tools. Agile values working software more over comprehensive documentation and customer collaboration over contract negotiation. It aims to deliver a functional software product in small increments, but still implementing feedback and making changes as required by the customer's needs with fast delivery time (Al-Saqqa, Sawalha, & Nabi, 2020). Agile itself consists of 6 steps which are requirements analysis, design, development, testing, deployment, and lastly reviewing. Following each iteration, a review process to ensure requirements compliance will take place. Whenever the client expresses dissatisfaction, the product will be updated again until it meets their expectation as this methodology places a strong emphasis on communication between clients and developers, making sure to comply with all of the software requirements.

2.4.2 Waterfall

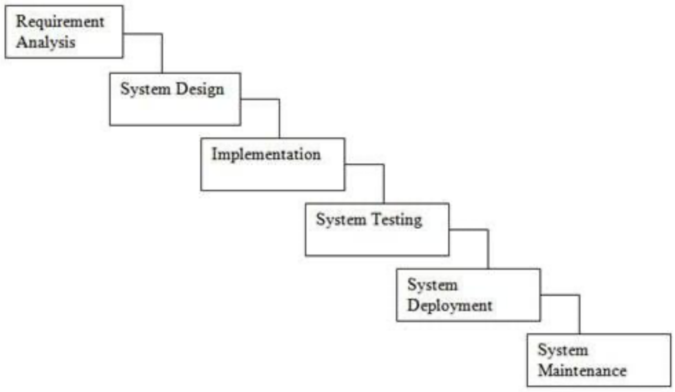


Figure 2.4.2.1

The waterfall model is one of the most basic and most used models of SDLC. It is running in a sequential order where the next phase will only be able to be started once the previous one ended which can lead to higher quality output (Wahid, 2020). The steps in Waterfall include Requirements, Design, Implementation, Verification, Maintenance. But the disadvantage of waterfall is that it is a rigid and linear approach. This lack of flexibility makes it hard to accommodate changes or requests, hence it might lead to cost overrun.

2.4.3 Spiral

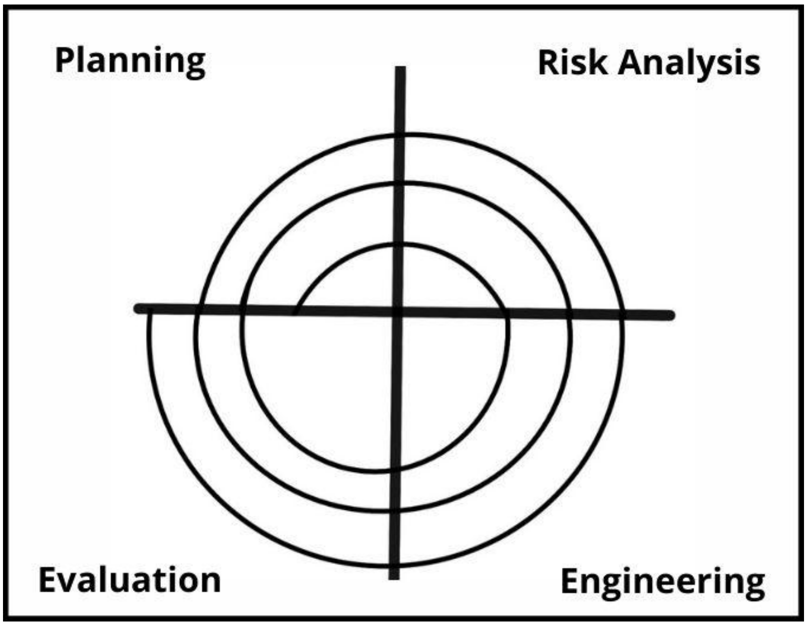


Figure 2.4.3.1

Spiral models can be described as a mix of waterfall, evolutionary, and prototyping models. The specialty of a spiral model is that it has the capability to handle risks, hence it is usually implemented on projects which are large, expensive, and complex. A spiral's loops are not defined or limited and are flexible depending on the project's requirements or needs.

2.4.4 Prototype

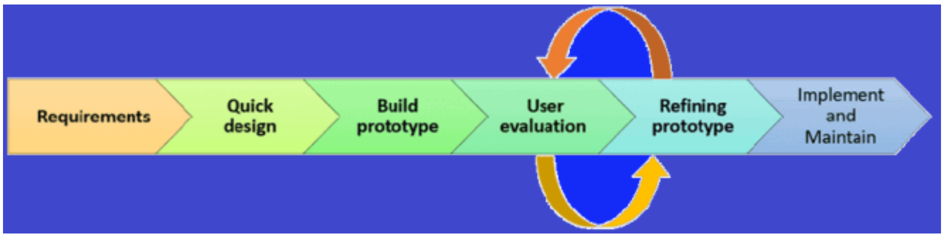


Figure 2.4.4.1

Prototype model is utilized when a prototype is produced, tested, and modified until an acceptable prototype is obtained. Once done, the accepted prototype may be turned into a full system. This model's downside is that it requires an intensive communication between developers and client to go through a trial-and-error process as the project itself may not get a complete requirement.

2.4.5 V-Shaped

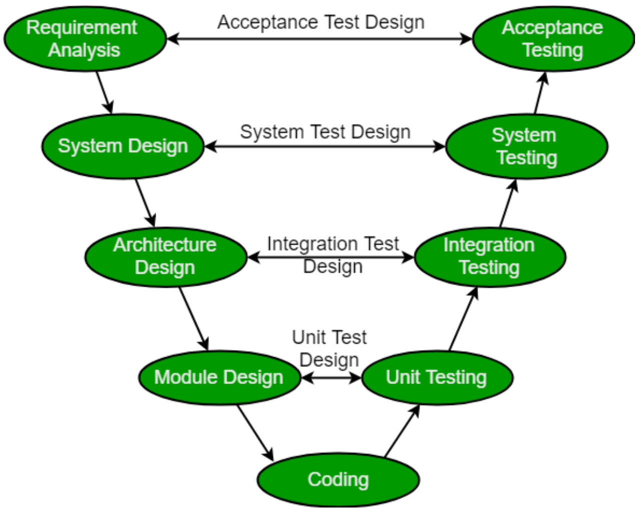


Figure 2.4.5.1

The V in V-shaped model represents the Verification and Validation model. In this model, every development stage will be associated with a testing phase and to continue to the next stage, a verification and validation for the previous state must be done and passed first (Dwivedi, Katiyar, & Goel, 2022). The upsides of this model are that problems will be detected or found at the early stage.

2.4.6 Review of Software Development Lifecycle Models

For this research, the software development life cycle that was chosen is the agile model. The reason why this was picked is because it values individual and interactions over processes and tools, working software over documentation, customer collaboration over contract negotiations. On top of it, agile also enables responding to change, instead of strictly following a plan.

2.5 Research Design

There are several research designs which can be picked for this research, this includes quantitative, qualitative, and mixed-methods. Each has their own upsides and downsides, and each is suitable for different projects, depending on the needs.

2.5.1 Quantitative Research

In quantitative research, it puts an emphasis on quantifying the collection and analysis of numerical data. The numerical values from observations will be used to explain the phenomena that was found during the observation (Taherdoost, 2022). Quantitative research itself aims to create and utilize mathematical models, theories, and hypotheses that relate to specific phenomena. After gathering the numerical values or data, the researchers will implement mathematical models in order to understand the information and also draw conclusions. To sum it up, quantitative research involves the utilization of numerical data to comprehend various phenomena present in the world.

2.5.2 Qualitative Research

Qualitative methods aim to gather genuine and analyze non-numerical data to understand concepts, opinions or experiences, basically to explore the problems which numbers cannot define. This allows for more in-depth explanation (Taherdoost, 2019) and the inclusion of a variety of problems including someone’s life experience and stories. As it possesses an exploratory nature, this method is mainly useful when researching a phenomenon with little to no information. Hence, the qualitative research method can be used to discover new perceptions, concepts, and theories.

2.5.3 Mixed-Method Research

Mixed-method research involves combining both qualitative and quantitative traits and approaches in order to gain a deeper understanding of a subject. Depending on the needs of the research, more focus may be put on either the quantitative side or qualitative side, or if required, the focus can also be split evenly across both methods. Through the analysis and interpretation of this method, researchers may construct, confirm and theorize at the same time (Dawadi, Shrestha, & Giri, 2021). Due to its potential, this approach has been used in sectors such as healthcare and even education.

2.5.4 Review of Research Design

Through this literature review, quantitative was determined to be the chosen research design due to its usage of numbers and data to understand and explain different phenomena in the world. The numerical data obtained, can be used in this research to compare the effectiveness of the AR-enabled assembly guidance system.

2.6 Research Instruments

Research instruments can be used to define tools which are used to collect, obtain, measure, and even analyze data, which are relevant to the research or study being conducted. These tools can include:

2.6.1 Questionnaire

A questionnaire refers to an instrument which consists of a set of questions or prompts that are designated to collect and obtain information from respondents. Usually, a questionnaire's questions are a mix of open-ended and closed-ended questions. The open-ended questions allow the respondents to elaborate their thoughts, but analyzing these answers can be costly (Krosnick, 2018). On the other side, the close-ended questions will offer some predefined responses which the respondent may pick. Close-ended questions consist of several types such as Likert scale, multiple choice question, checkboxes, etc. Questionnaires are commonly used as they are efficient in collecting data from a large group of respondents, but still being affordable at the same time.

2.6.2 Observation

The research instrument observation gathers data by observing the behavior of the target population in a controlled and uncontrolled environment. It can be done either directly or indirectly, depending whether it was done remotely or directly in real-life (Ciesielska, Boström, & Öhlander, 2017). Observation is a double-edged sword. Typically, an observation is easy to execute and also provides accurate data, but on the other hand it can be expensive and also it is subject to the observer's bias.

2.6.3 Experiment

Experiments can be used to describe tests that were done in order to learn or discover the reality of something or to find causal relationships between independent and dependent variables (Rogers &

Révész, 2019). An experiment's result aims to either support or reject a hypothesis, or it can also determine the likelihood of something that was untried before. Unfortunately, an experiment may be expensive, and on top of it, the controlled and non-realistic situation may put a burden on the respondents, leading to a corrupted result.

2.6.4 Interview

An interview is a structured conversation between a participant which asks questions and the other participants provides the answers to the questions. The participant which asks questions is the interviewer, meanwhile those being interviewed are called as an interviewee. There are 3 interview types: structured, semi-structured, and unstructured. In structured interviews, adherence to an interview protocol is essential. Semi-structured interviews maintain focus on the main topic while allowing exploration of relevant ideas. Meanwhile, in an unstructured interview, it resembles a normal conversation and lacks a predefined protocol or set of questions, which makes them relatively informal in style (Croucher & Cronn-Mills, 2021).

2.6.5 Focus Group

Focus Group allows a Focus Group Discussion (FGD) which involves interviewing a group of targeted respondents. Due to its group nature, usually participants are more comfortable in expressing their views compared to a one-on-one interview. Therefore, group dynamics may foster diverse perspectives and deeper insights on the topic. However, the downside of a FGD is that the discussion topic may accidentally expose confidential information of the participants as there is limited control over what participants may communicate outside the group (Sim & Waterfield, 2019).

2.6.6 Review of Research Instrument

After reviewing the literature on research instruments, interview was selected as the primary method, aligning with the qualitative research design. This method has been deliberately chosen due to its effectiveness and its reliability in producing the data to be used in the research. On top of it, with interviews, it allows for in-depth exploration of topics. Unlike with questionnaires, in interviews, the addition of follow-up questions can be included in order to explore interesting topics that may arise during the conversation.

Chapter 3: Preliminary Studies

3.1 Introduction

As mentioned in the literature review, the qualitative research method will be applied in this research. In addition, the interview conducted would allow for more in-depth discussion regarding the problem of training new assemblers.

3.2 Research Design

3.2.1 Research Design Table (RDT)

No	Research Questions	Research Objectives	Methods	Expected Outcome
1	What is the current method of training new assemblers?	To investigate the current method and techniques used to train new assemblers.	1. Literature Review 2. Studying existing system	List any existing methods and techniques which are being used to train new assemblers
2	How can the AR system help the process of assemblers training?	To design and develop AR applications for the manufacturing industry for assembler training.	1. Interview	Develop a prototype of the AR-enabled system that is suitable for assemblers training
3	Can the effectiveness and efficiency of the AR improve the training and onboarding process?	To validate the effectiveness and efficiency of AR application for manufacturing industry for assembler training.	1. Experimental 2. Conducting user acceptance testing	Provide and validate an accurate and effective training system

Table 3.2.1.1

3.2.2 Design Science Research Methodology (DSRM) Table

Phase	Details of Activities
Phase 1 Identification of problem and motivation	• According to relevant articles, it was stated that in general, manufacturing companies, especially the smaller ones, is conducting training using text-based guidelines which are not sufficient and clear enough, thus causing more errors to happen and wasting more resources. • To verify the existence of this issue with the training, literature review has been carried out to ensure the effectiveness of the problem solution proposed for the problem statement. • Based on the literature review, the implementation of AR may be able to minimize the errors made, which in turn would lower operational cost. The need of supervision by experienced assembler is also reduced, hence improving productivity.
Phase 2 Define scope and objectives of the solutions	• Literature review and interview helped in understanding the issues and problems that is occurring in the training processes. • A conceptual framework has been created in order to show how the product works.
Phase 3 Data Collection	• To create a dataset of product images, a collection of images representing various products needs to be gathered. Each image in the dataset should be accompanied by a data annotation, providing relevant information about the product depicted in the image. • Clean out all of the noisy data in order to improve the model's accuracy.
Phase 4 Model Training and Deployment	• The Object Detection model is train with different parameters to achieve the best accuracy rate. • Using Django Framework, develop a website • Integrate the model with the website and deploy it.
Phase 5 Evaluation	• Conduct experiment to test the system's effectiveness. • Evaluate the system and conduct testing to gather feedback. • Fix all of the errors spotted and implement feedbacks gathered.
Phase 6 Communication	• Compile all findings, such as from literature review, development process, evaluation, into a complete and detailed documentation.

Table 3.2.2.1

3.2.3 Overall Research Design Table

Phase	Input	Process	Output
Phase 1	• Motivation • Problem	• Validate the motivation • Analyze the problem	• Project scope • Proposed solutions
Phase 2	• Project space • Proposed solution	• Further research • Suggestion and refinement from supervisor	• Conceptual Framework
Phase 3	• Conceptual framework	• Artefact design and development	• Artefact Design • Hierarchical Task Analysis
Phase 4	• Artefact Design • Hierarchical task analysis	• Application and demonstration of artifact	• Validated working prototype
Phase 5	• Validated working prototype	• Performance measurement (Quality Assurance)	• Validated artifact
Phase 6	• Validated artifact	• Documentation and compilation	• FYP report

Table 3.2.3.1

3.2.4 Evaluation Table

AR to train new assemblers		
1	Participant	New assemblers which require training in order to fulfill the skills requirement
2	Task	1. Select product to assemble 2. Inform user of steps to do 3. Visualize the steps in collection of images 4. Run object segmentation tasks on frames received from the phone.
3	Technique	1. Image processing
4	Measurement	1. Accuracy of components detection 2. Effectiveness in training
5	Outline	1. Explain the goals and objectives of the interview to the interviewee

Table 3.2.4.1

3.2.5 Research Operational Framework

Item	Phases	Activity Description	Output	Expected Contribution
Idea	Phase 1: Problem Identification and motivation	• To conduct literature search and review • Interview	Conceptual model	Knowledge on AR for training system
Conceptual model	Phase 2: Suggestion of solutions	• Literature review • Task analysis • In-depth survey • In-site observations	Refined conceptual model	Artefact: Prototype of training system with AR Technique to identify components
Working prototype	Phase 3: Artefact design and development		Artefact design	
Artefact	Phase 4: Artefact evaluation	• Develop performance measures assessment • Users are the artifact and give performance measures	Identified performance measure	
	Phase 5: Model validation	• To conduct an experiment, with and without the AR system	Validated model	Final model of the AR system for training
	Phase 6: Documentation	• To combine all results from literature review, preliminary study, artifact design, artifact evaluation report, framework validation report, framework validation report into a complete and comprehensive report		

Table 3.2.5.1

3.2.6 Hardware and Software

Hardware	For Assemblers: Mobile devices that are equipped with a camera and ability to connect to the internet For Admin: Desktop that can connect to the internet	Justification: Camera will be utilized to see the components and internet will be required to access the server hosted in within the LAN.
Software	1. Django Framework 2. Python 3. Visual Studio Code 4. MacOS Ventura 13.3 or Windows 10	Justification: These are the software that will be used in order to develop this system

Table 3.2.6.1

3.3 Research Instruments

An interview will be implemented as the only and primary research instruments in order to obtain useful information for the research. The interview will be utilized in order to gather the information regarding assembler training.

3.4 Data Collection

Population and Sample		
1	Participant	Owner of a Selangor company that designs & supplies industrial automation systems
2	Representative tasks	1. Select process 2. Following the steps displayed
3	Technique	1. Interview
4	Measurement	1. Accuracy of components detection 2. Effectiveness in training
5	Outline	Explain the goals and objectives of the questionnaire to the participant

Table 3.4.1

3.5 Instruments

The interview will be conducted physically with an owner of a Selangor company that designs & supplies industrial automation systems. The questions in the interview will be regarding the issues regarding the training of new manufacturer assemblers. The main objective of this interview will be to gather the current method used to train new assemblers. Results from this interview will enable the analysis of the feasibility of an AR training system. Additionally, an experiment will be conducted in order to evaluate the accuracy and effectiveness of the system.

3.6 Software Development Life Cycle (SDLC)

The software industry employs the Software Development Life Cycle (SDLC) as a systematic approach to create, design, and deliver software products that meet high standards of quality, reliability, cost-effectiveness, and timely delivery (Dwivedi, 2022). The SDLC serves as a well-structured model for the software development process. Referring to another research paper (Okesola, 2020), there are 6 crucial phases of an SDLC, these include: planning and requirement analysis, defining requirements, designing, development, testing, and deployment and maintenance.

As reviewed in the literature review, the research design chosen will be the agile model due to its advantages over the other models reviewed.

3.6.1 Proposed Agile Model

In total, there are 6 steps in the proposed agile model, which consists of requirements, design, development, testing, deployment, and lastly, review.

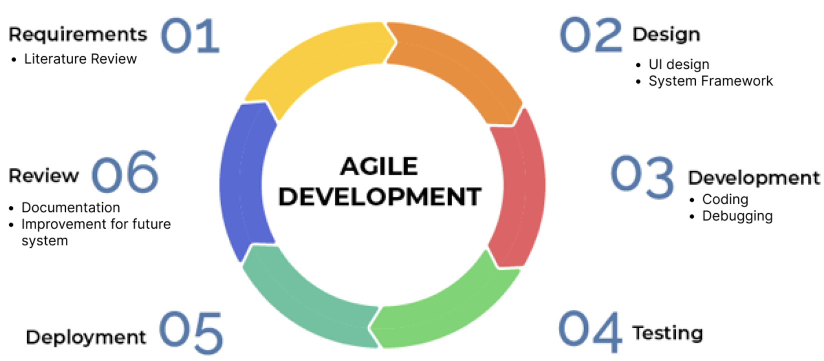


Figure 3.6.1.1

3.6.2 Agile Unified Project

	Phases	**Output**
Inception	Identify Problem - Literature review	List of problems identified
Elaboration	- Identify problem statement - Identify research objective - Identify Scope and Limitation	- Problem Statement identified - Research Objective identified - Scope and Limitation identified
Construction	Identify the tools for system development - Laptop - Smartphone	- Tools identified - Prototype development identified
Transition	- Conduct experiment - Collect feedback	Gain feedback

Table 3.6.2.1

Chapter 4: Analysis

4.1 Introduction

An interview was conducted with the owner of a company that designs and supplies industrial automation systems for motion control applications located in Selangor. The purpose of this interview is to get more understanding on the current method used to train new assemblers in the company. The type of interview is a semi structured interview where the interviewee is able to share their relevant ideas while still focusing on the main topic.

4.2 Objectives, Questions, and Results of Interview

Objective 1

In order to understand more regarding the role of assemblers in the factory.

Questions

1. How many assemblers are employed?
2. Can you describe the work of the assemblers?
3. Are the roles of these assemblers flexible?
4. How many assemblers are needed to assemble the FRL unit?

Answer

In total, the company employed 8 skilled assemblers, complemented by a constant flow of interns staying during their internships. Their role is to assemble or disassemble various pneumatics products. All have a flexible role, considering the products in question are generally not too challenging to assemble for the experienced assembler. For instance, the average time for an experienced assembler to assemble a FRL is around 10 minutes, while a new assembler may need 20 minutes. An FRL unit itself only needs a single assembler.

Objective 2

In order to have a more clear or complete idea of the training process for the assemblers.

Questions

1. What is the current method of training new assemblers?
2. Are the guidelines up-to-date?

3. Is it possible for the admin to make changes to the guidelines?
4. How long does it take to train a new assembler to assemble the FRL unit?
5. What are the drawbacks of that method?

Answers

Currently, a PDF document containing the steps are shared to the assembler for them to use as guidance during the assembly process, hence training themselves. The guidelines itselves are up to date as it is given by the manufacturer, but if required the admin can make some changes to it. Regarding the duration of training, it depends on each skill set. But a new unskilled assembler may need up to 20 minutes for the first unit and a few days to fully master the process, meanwhile interns generally need more time. On top of that, the absence of illustrations in the text-based guidelines presents a challenge. This is a problem as interns will generally stay for a short period of time, hence it is crucial to shorten the training time.

4.3 Use Case Diagram

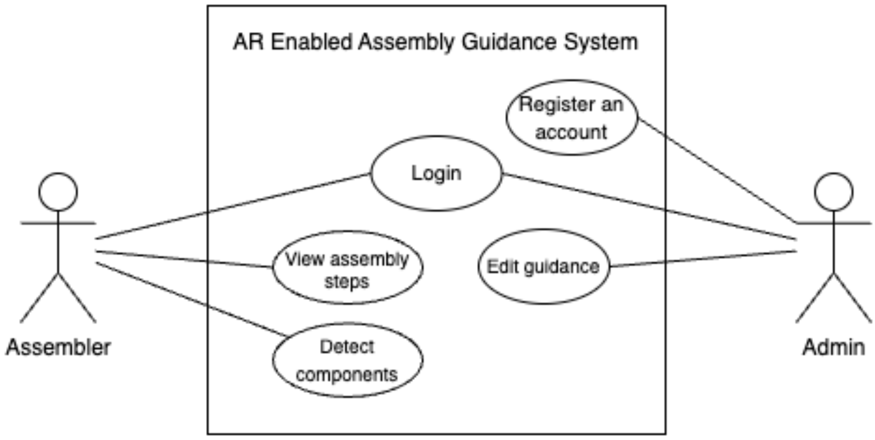


Figure 4.3.1

4.4 Use Cases Description

4.4.1 Register an account

Use case ID	UC001	Version	1.0
Feature	F001 Register account		
Purpose	Allowing admin to register a new account to the system		
Actor	Admin		
Trigger	Admin launched the system.		
Precondition	● The admin is logged in ● The admin is on the home page		
Main flow	1. Admin register an Assembler account with a new and unique employee ID 2. System verifies that the ID has not been used before 3. System redirects admin to the homepage		
Alternate flow	1. Admin register an Assembler account with an old, used ID 2. System verifies that the ID has been used before 3. System displays an error message		
Rules	Username and password entered must follow the requirement set		
Author	Nicholas Dylan		

Table 4.4.1.1

4.4.2 Login

Use case ID	UC002	Version	1.0
Feature	F002 Login		
Purpose	Allowing users to login to the system		
Actor	Assembler and Admin		
Trigger	User launched the system.		
Precondition	● The user is on the login page. ● The user is not logged in		
Main flow	1. Users enters their correct email and password 2. Users click the "Login" button 3. System ensures the credentials provided are valid 4. System redirects user to the homepage		
Alternate flow	1. Credentials entered by users are invalid 2. Users click the "Login" button 3. System found the credentials provided are invalid 4. System shows an error message warning the users		
Rules	Username and password entered must match to the one registered in the database		
Author	Nicholas Dylan		

Table 4.4.2.1

4.4.3 Edit guidance

Use case ID	UC003	Version	1.0
Feature	F003 Edit guidance		
Purpose	Allowing admin to edit the assembly guidance		
Actor	Admin		
Trigger	Admin clicked the “Edit” button		
Precondition	● The admin is logged in ● The admin is on the homepage.		
Main flow	1. Admin click on the “Edit guidance” button 2. Admin make changes to the guidance 3. Admin click on the “Confirm” button to confirm changes 4. System saved the changes made		
Alternate flow	-		
Rules	-		
Author	Nicholas Dylan		

Table 4.4.3.1

4.4.4 View assembly steps

Use case ID	UC004	Version	1.0
Feature	F004 View assembly steps		
Purpose	Allowing assembler to read the step guidance		
Actor	Assembler		
Trigger	Assembler clicked the “Start” button		
Precondition	• The assembler is logged in • The assembler is on the homepage. • The assembler picked the type of products to assemble		
Main flow	1. Assembler click on the “Start” button 2. System shows the instructions for the assembler 3. Assembler read the instructions		
Alternate flow	1. Assembler does not click on the “Start” button 2. System does not show the instructions for the assembler		
Rules	-		
Author	Nicholas Dylan		

Table 4.4.4.1

4.4.5 Detect components

Use case ID	UC006	Version	1.0
Feature	F006 Detect components		
Purpose	Allowing assemblers to show the components to the camera for the server to detect components		
Actor	Assemblers		
Trigger	Assemblers clicked the “Check” button		
Precondition	● The assembler is logged in ● The assembler confirmed the product to assemble		
Main flow	1. The assembler picked the type of product to assemble 2. Assembler placed the component to the inside of the camera view 1. Assembler clicked “Check” button 2. System detected the component		
Alternate flow	1. Assembler placed the wrong component to the camera view 2. System unable to detect the components 3. System shows a warning sign and prompt user to make another attempt		
Rules	-		
Author	Nicholas Dylan		

Table 4.4.5.1

4.5 Functional Requirements

4.5.1 FR-01 Register account

FR-01 Login	
FR-01-001	System needs to be able to let Admin register a new account
FR-01-002	System should be able to verify that the new credentials is following the requirement set (e.g. password need to contain numbers)
FR-01-003	System should warn Admin if the new credentials entered was not following the requirement set
FR-01-004	System should prompt Admin to re-enter the new credential if the credential entered before is not following the requirement

Table 4.5.1.1

4.5.2 FR-02 Login

FR-02 Login	
FR-02-001	System needs to be able to validate the entered user's credential
FR-02-002	System should be able to allow user to log in into the system
FR-02-003	System should warn user if the credential entered was incorrect
FR-02-004	System should prompt user to re-enter their credential if the credential entered before is incorrect

Table 4.5.2.1

4.5.3 FR-03 Edit guidance

FR-03 Edit guidance	
FR-03-001	The system shall be able to allow the admin to make changes to the guidance
FR-03-002	System should be able to save the new edited guidance

Table 4.5.3.1

4.5.4 FR-04 View assembly steps

FR-04 Read instructions	
FR-04-001	The system should be able to display the step required to be completed

Table 4.5.4.1

4.5.5 FR-05 Detect components

FR-05 Detect components	
FR-05-001	System should be able to detect the completeness of the components
FR-05-002	System should be able to prompt the assembler to make another attempt if any components were to be found missing
FR-05-003	System should be able to show something that indicates that the components required are detected

Table 4.5.5.1

4.6 Non-Functional Requirements

4.6.1 Reliability

NFR-01 Reliability	
NFR-01-001	Fault tolerance
NFR-01-001-1	In the absence of an internet connection, the system should be able to reconnect once connection is recovered and continue from the same step

Table 4.6.1.1

4.6.2 Usability

NFR-02 Usability	
NFR-02-001	Learnability
NFR-02-001-1	Ensure the system is easy for a first-time user to understand and use effectively
NFR-02-002	Efficiency
NFR-02-002-1	The system should be able to allow user to finish their task in an efficient manner
NFR-02-003	Effectiveness
NFR-02-003-1	The system should be able to allow user to finish their task in an accurate and effective manner
NFR-02-004	User satisfaction
NFR-02-004-1	User should be able to use the system in a satisfactory manner

Table 4.6.2.1

4.6.3 Compatibility

NFR-03 Compatibility	
NFR-03-001	Interoperability
NFR-03-001-1	Ensure the system can be run on different systems such as iOS and Android

Table 4.6.3.1

Chapter 5: Design

5.1 Introduction

This chapter contains technical drawings, system framework, and design principles which underpin this project.

5.2 Technical Drawings

5.2.1 Class Diagram

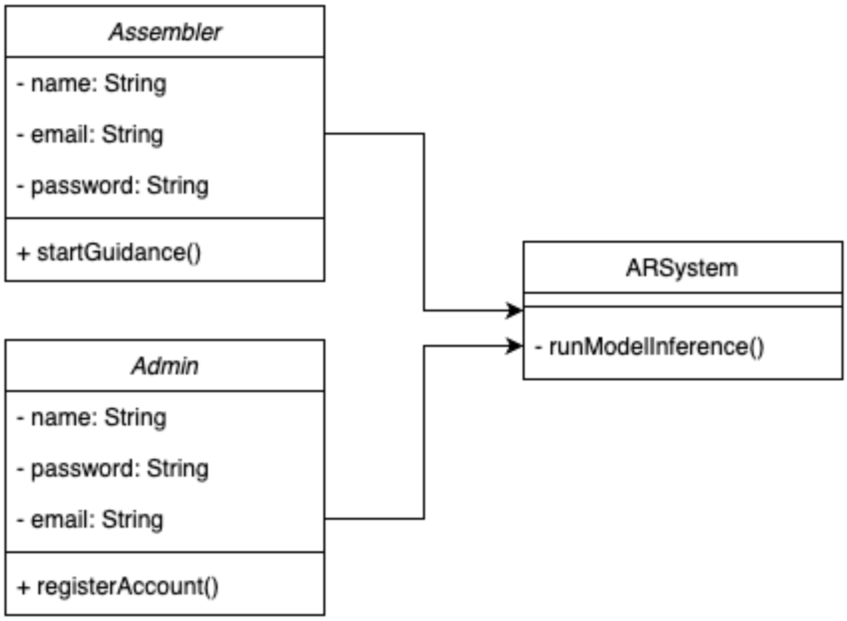


Figure 5.2.1.1

5.2.2 Activity Diagram

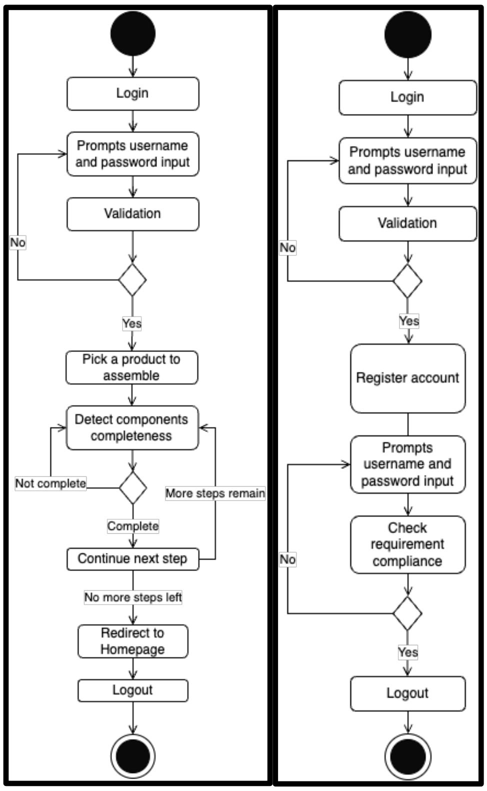


Figure 5.2.2.1

5.3 System Architecture

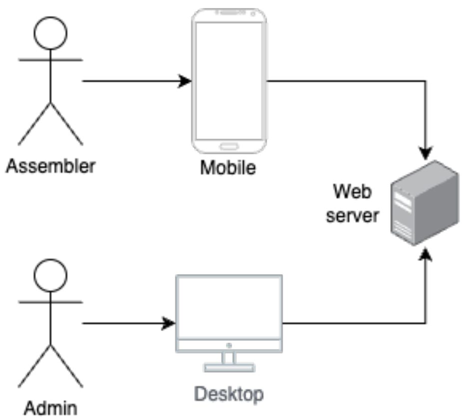


Figure 5.3.1

As depicted by the figure above, the assembler will be required to access the website-based system through a mobile phone. Once logged in, the assembler will be able to access the AR system that will assist them in the assembly process. On the other hand, the admin will exclusively access the website-based system through a desktop interface. Upon logging in, the admin may register a new assembler account or and modify the description of guidance steps as needed.

5.4 Design Principles

The design principles considered in this project were based on the UX principles by Hartson and Pyla (2020).

5.4.1 Visibility

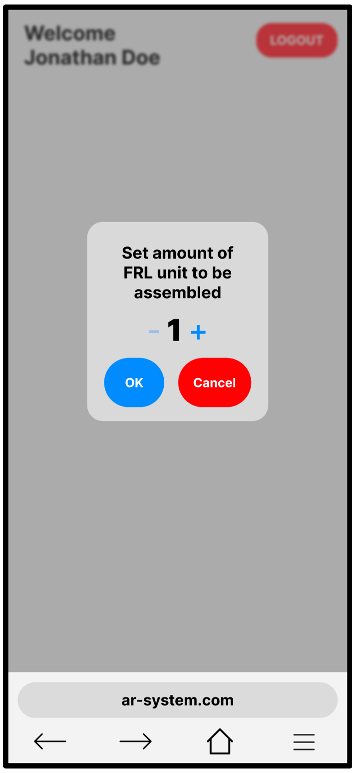


Figure 5.4.1.1

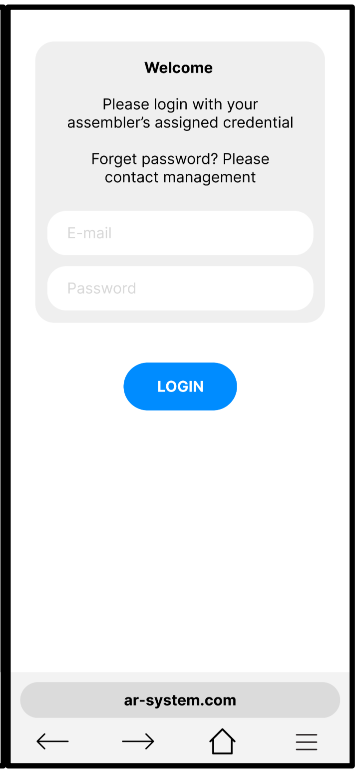


Figure 5.4.1.2

Visibility is one of the design principles that the more visible an element is, the more users will notice it and also its usage. Conversely, when an element is less visible, users are less likely to acknowledge it. As can be seen here, all of the objects that the user might consider crucial are displayed on the screen without any unnecessary clicks to reveal it. For instance, the login functions are the first thing the user will see once the website is launched and the login button was also designed to be clearly labeled.

5.4.2 Noticeability

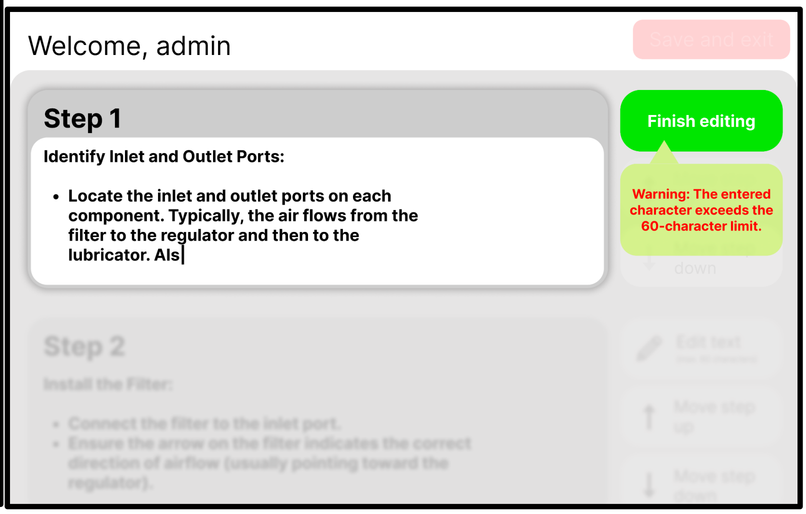


Figure 5.4.2.1

Figure 5.4.2.2

If an object is important, being put in a random position inside the screen is not helping the user to notice it. In order for it to be useful, the user needs to be able to notice it. On occasions, a user might not notice the object at times when they are not expecting or looking for it. Hence, this design is designed in a way to foster the user's awareness of the necessary objects. For instance, in the mobile interface, a pop up is used for important messages that are crucial for the task. The color yellow and red was used for the pop up due to the common color convention where red is usually associated with something urgent or emergency. On top of that, in the mobile desktop a similar yellow pop up was used to attract the user's attention.

5.4.3 Text Legibility

Figure 5.4.3.1

Legibility here is not regarding whether the user can easily understand the words easily. Rather, it is about discernibility of the text and how the text can be read or sensed. In order to achieve this, the font color, type, and size were designed in a way to make it easy for the users on both desktop and mobile to understand. The utilization of a distinct contrast between the background and font color was employed to augment legibility. For instance, black text on a light gray background and white text on a red background.

5.4.4 Use of Color

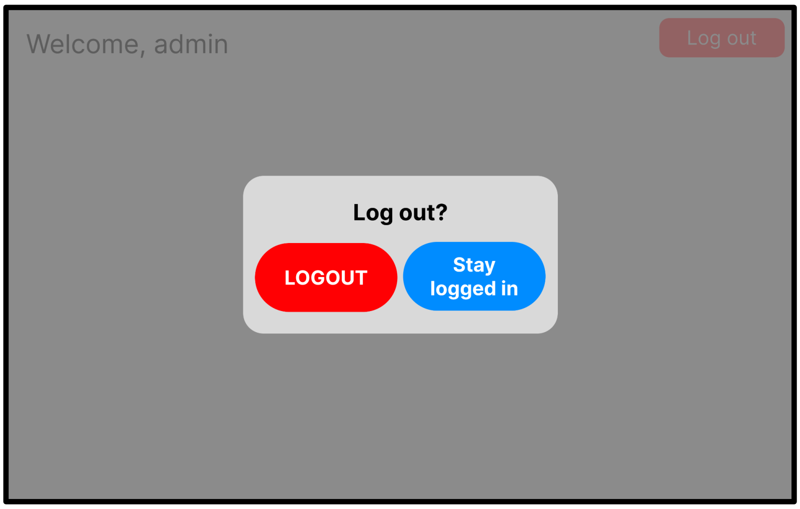


Figure 5.4.4.1

Color is one of the most powerful instruments in a designer toolkit and can be used to make an impact on the user's emotion and attention. As seen above, the distinct red and blue hues of the buttons effectively serve the purpose of clearly signaling to individuals that they indeed function as buttons with the blue symbolizing calmness and the red serving as an indicator for alert situations, which in this situation is logging out from the system. On top of it, the font color for the button labels were changed to white in order to increase the contrast, hence increasing text legibility at the same time.

5.4.5 Consistency of Cognitive Affordances

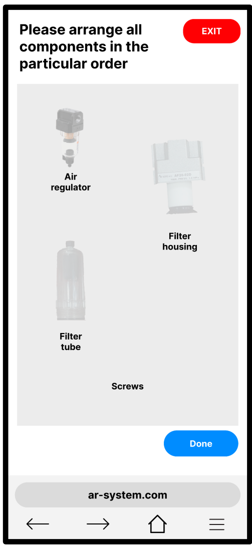


Figure 5.4.5.1

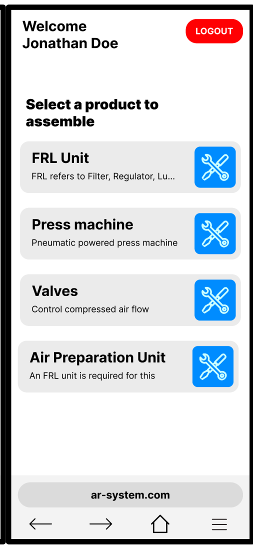


Figure 5.4.5.2

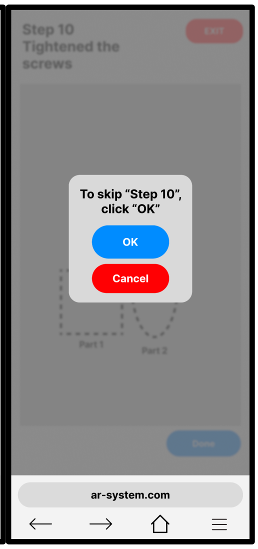


Figure 5.4.5.3

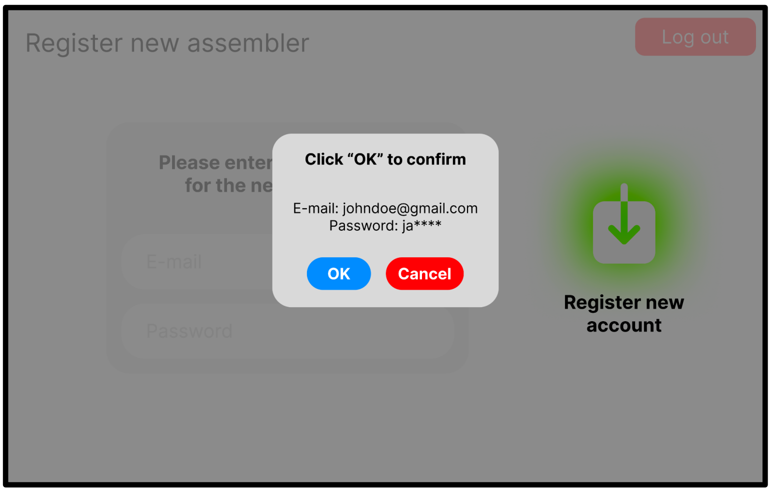
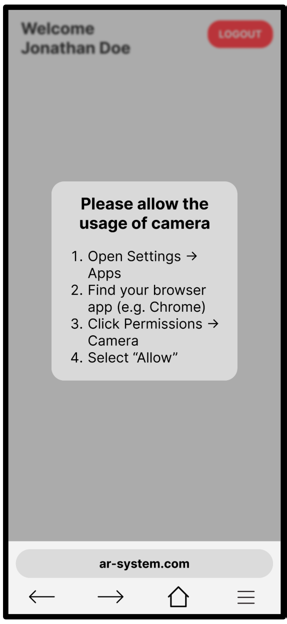
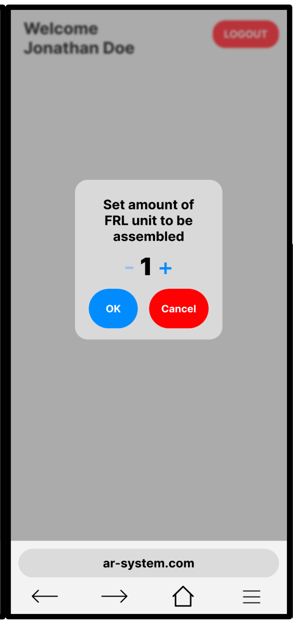


Figure 5.4.5.4

Consistency of cognitive affordances is consistency regarding the use of wording labels for menus, buttons, icons, fields in order to make it easier for users to understand the system. Being consistent here does not only translate to using the same terms for the same things, it also means using different terms for different things. The latter is crucial, especially when the difference is subtle. For instance, 2 different terms: "logout" and "exit" were used in order to allow the user to differentiate between exiting from the assembly process and exiting from the system or logging out. On top of that, as can be seen from the rightmost image, the term "OK" is consistently utilized for both buttons and text information labels. Additionally, the interface on both mobile and desktop were made to be similar, for instance, the usage of blue and red for the colors.

5.4.6 Constraints as requirements





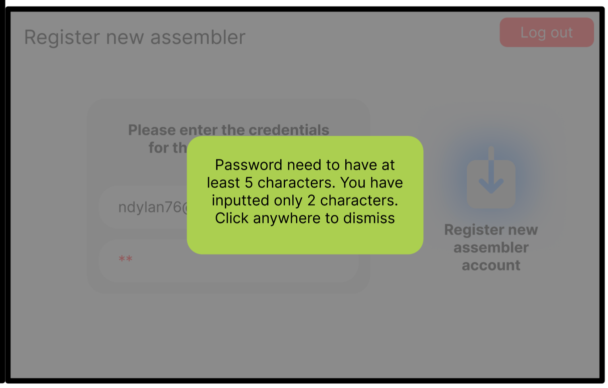
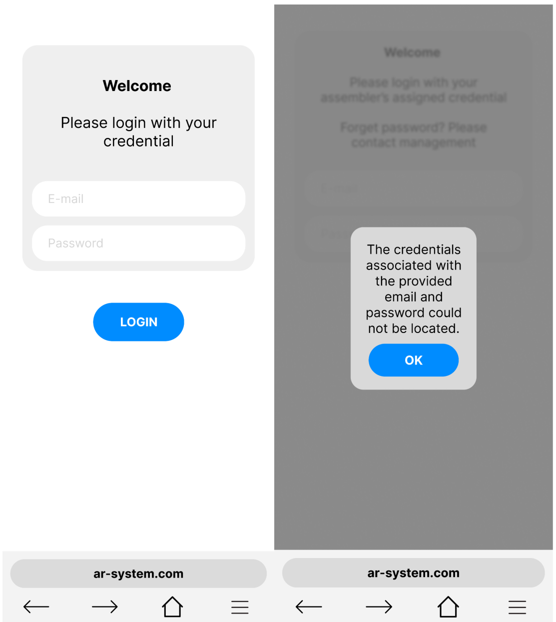


Figure 5.4.6.1 *Figure 5.4.6.2* *Figure 5.4.6.3*

Within the system, constraints will be implemented to ensure users adhere to established norms and meet specified standards. As an example, mobile devices accessing the system must be equipped with a camera, and users will be prompted to grant permission for camera access as part of the requisite criteria. Constraint was also applied on the user input for the amount of FRL units to be constructed as it is impossible to build 0 units. In the desktop interface, the admin will also face a constraint in the form of the minimum number of characters for the password.

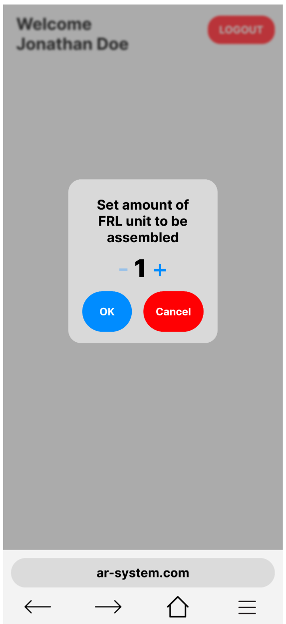
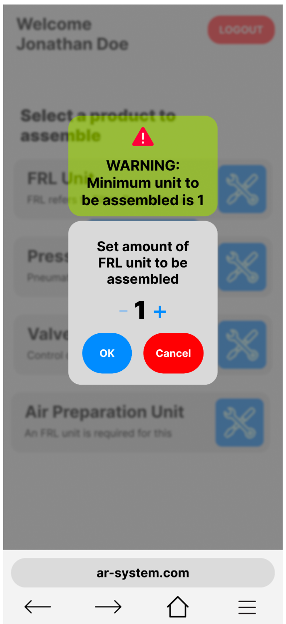
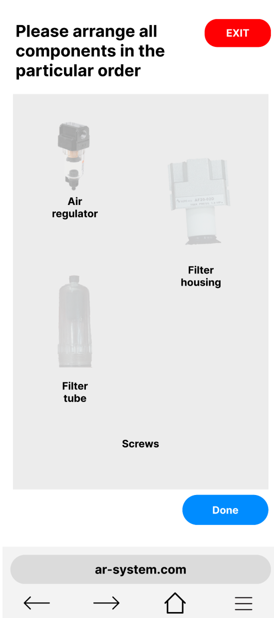
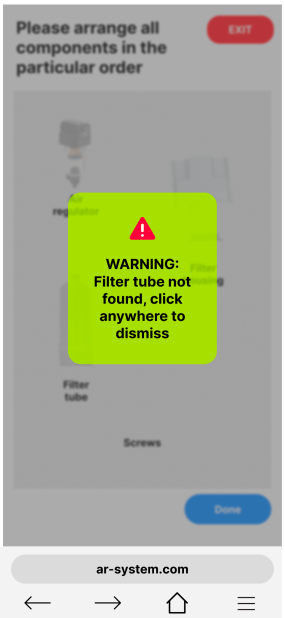
5.5 UI Flow of Control

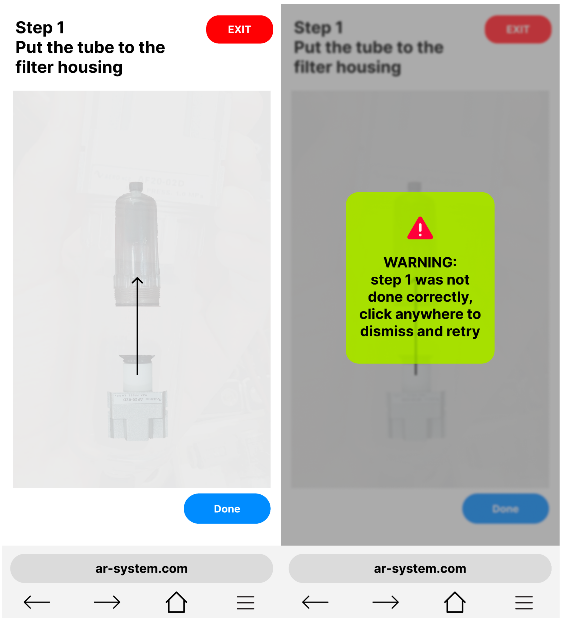
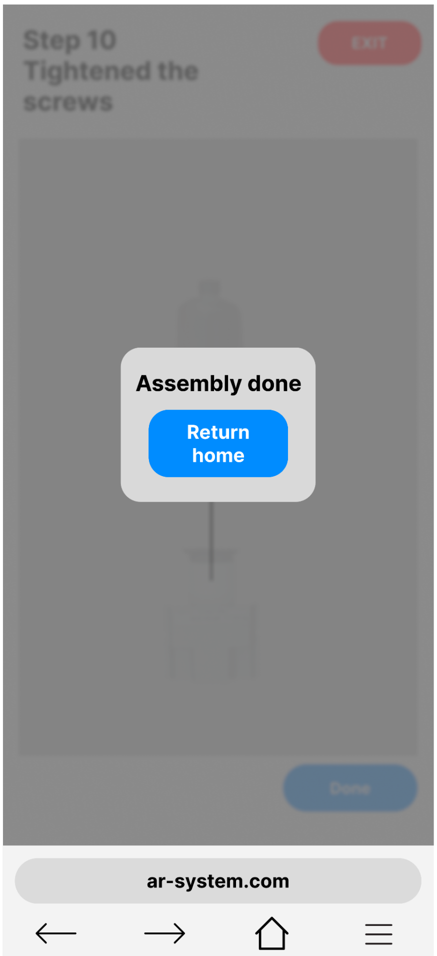
5.5.1 Flow as assembler

<table>
<tr>
<td>


</td>
<td>Once the website was launched on a mobile device, the assembler will be prompted with their registered account. If the account with the entered credential was not found, then an error message depicted in the rightmost illustration will be shown.</td>
</tr>
<tr>
<td>


</td>
<td>Subsequent to logging in, the assembler will be greeted with the home screen with a list of products which are available that they may pick to assemble. In this case, the assembler will click on the blue button next to the FRL unit.</td>
</tr>
</table>

 	Post picking the FRL unit, a pop up will appear prompting the assembler to fill how many FRL units that they want to assemble. After clicking the "+" and "-" button to increase or decrease the number of units to be assembled, the assembler may click on "OK" to confirm it. In the case the amount of 1 and the assembler clicked "-", then the system will show a pop-up error warning the user that the minimum is 1.
 	Upon confirming the amount, the assembler will need to arrange all the components following the placement presented on the screen. After that, the assembler may click "Done" and the components need to be detected through the phone's camera in order for the assembler to continue to the assembly part. In the case of any components not found, the system will show a pop-up message.

	During the assembly part, the assembler will be required to show their product and click “Done” to start the system checking process. If the end product passes the system checking then the assembler may continue to the next step. If not, an error message depicted on the second illustration will appear.
	Once the assembly process is finished, the assembler will click on the “Return home” button which will redirect the user to the home screen.

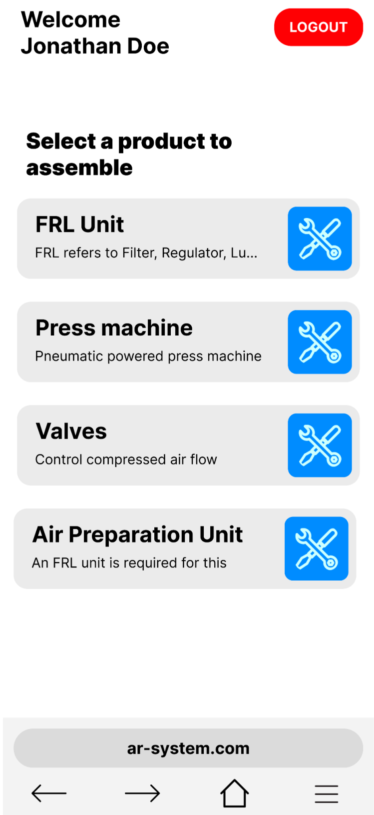
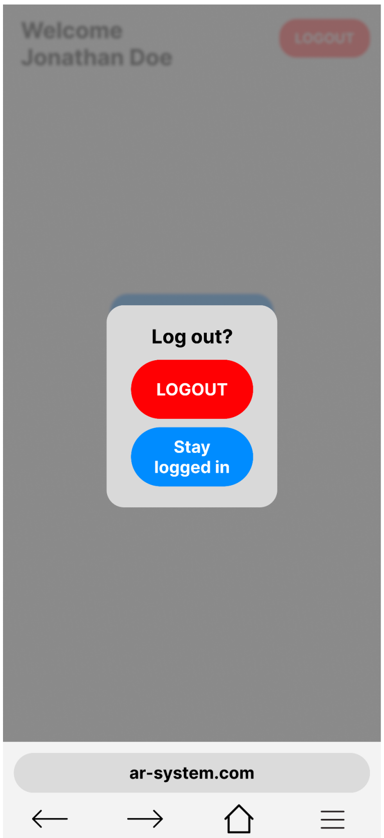
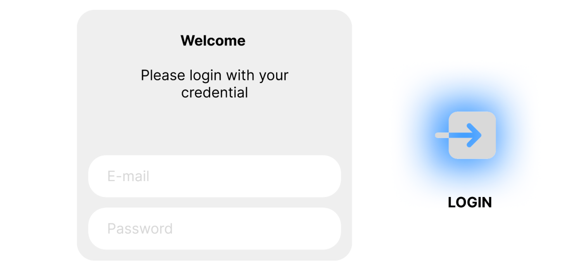
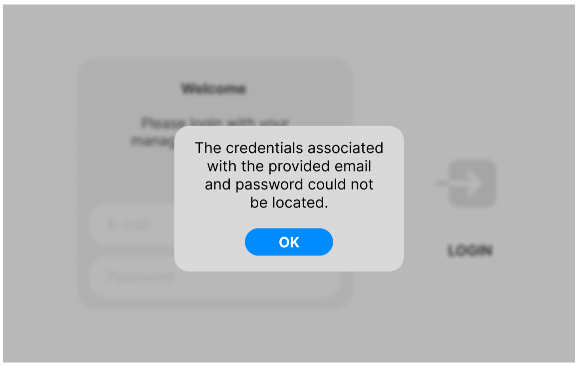
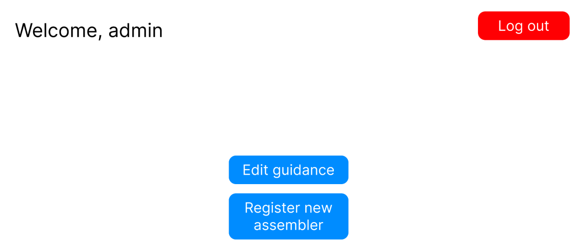
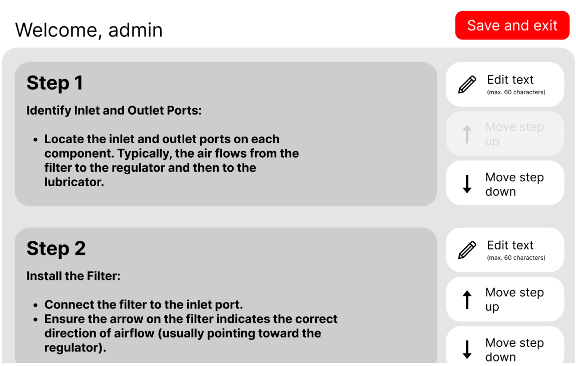
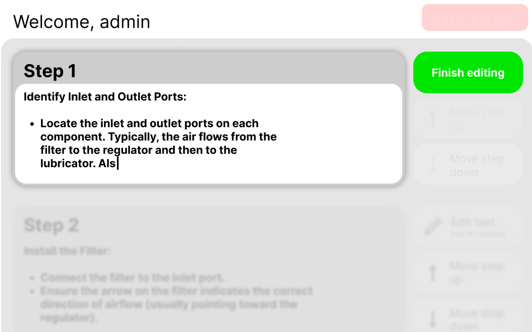
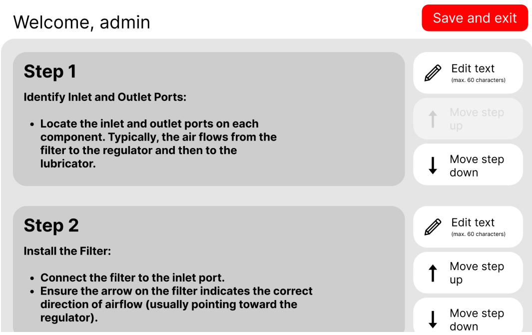
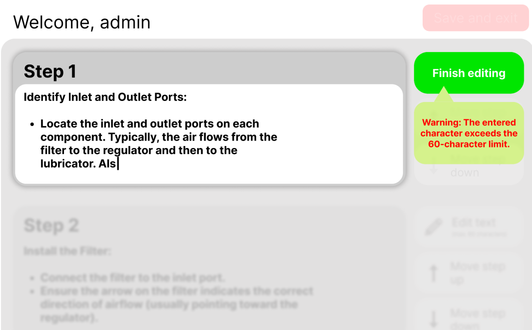
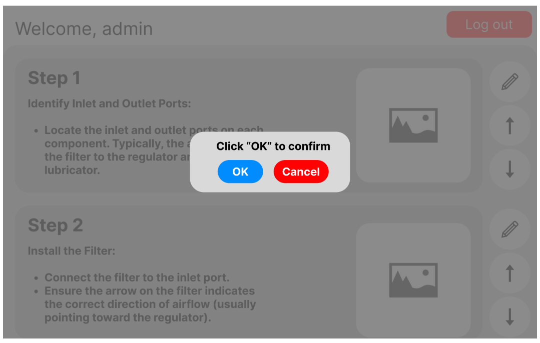
	Once the assembler finishes with their tasks, they may click on the logout button which is visible on the top right corner.
	Clicking on the "LOGOUT" button will log the assembler out of the system.

Table 5.5.1.1

5.5.2 Flow as admin editing the text of the guidance

 	When the website is launched on a desktop, it will prompt the user with a screen for them to log in on the desktop interface. In the case of a wrong password or wrong email then the system will show a pop-up message warning the user.
	Once logged in, the admin may edit the text on the guidance or register a new assembler account.
	Subsequently, the admin may click on the "Edit text" button to edit the text, or the "Move step up" or "Move step down" in order to move the step up or down. In this case the admin will click on the edit button.

<table>
<tr>
<td>
</td>
<td>Once done typing, admin may click on the finish editing button to save it.</td>
</tr>
<tr>
<td>


</td>
<td>Upon editing as needed, admin may click the “Save and exit” button on the top right corner. If the characters entered are more than the limit, then an error message will pop up to inform the admin.</td>
</tr>
<tr>
<td>
</td>
<td>Subsequently, a pop up confirming the changes made to the guidance will pop up.</td>
</tr>
</table>

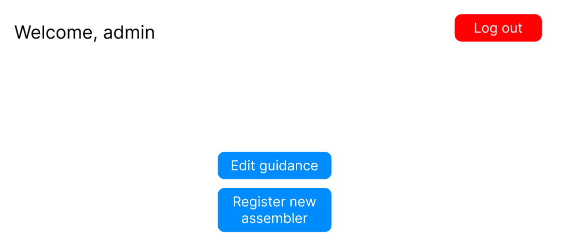
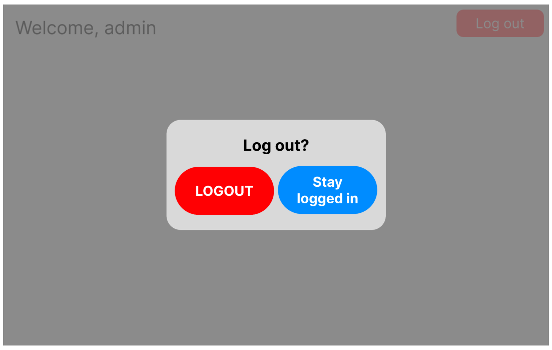
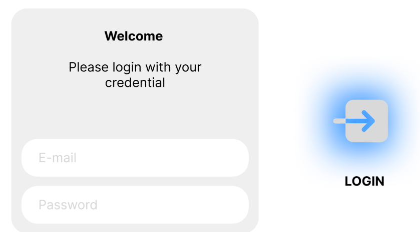
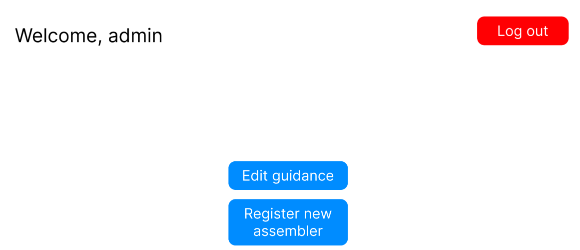
	After confirming, the user will be automatically redirected to the homescreen where the user may log out if needed.
	Confirming logging out from the system

Table 5.5.2.1

5.5.3 Flow as admin registering a new assembler account

	Similar to the previous flow, here the admin may input their email and password to login.
	In this scenario, the user will click on the “Register new assembler” to make a new assembler account

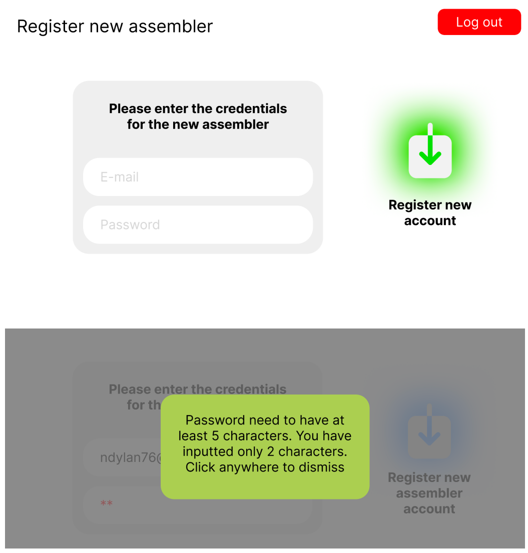
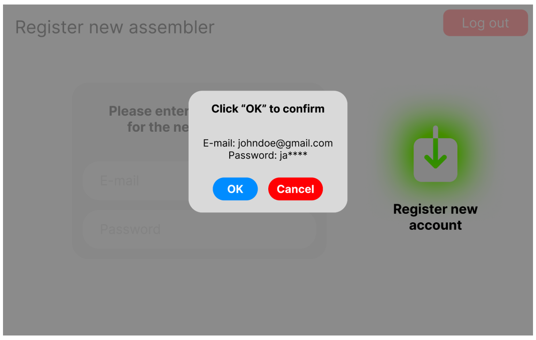
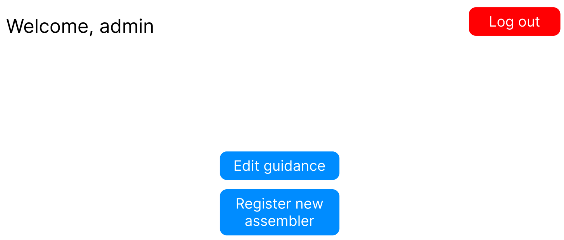
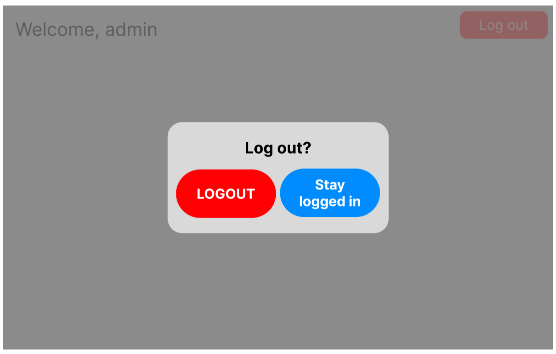
	Here, the admin may input the assembler's email and assign the account with a password. Once done, the admin may click on the green button labeled "Register new assembler account". If the password entered has less than 5 characters, then the system will pop up an error message.
	After clicking the button, a pop up confirming the new account's credential will appear and prompt the user for a response confirming.
	After confirming, the user will be automatically redirected to the home screen.
	The admin will then confirm to log out from the system.

Table 5.5.3.1

Chapter 6: Implementation

6.1 Introduction

In this chapter, a thorough recapitulation of the considerations throughout the development of this system. This chapter is a comprehensive overview that introduces the technical aspects, methodologies, and also the tools employed during the implementation phase. In addition, all the libraries, modules, and perhaps, framework which are necessary will be addressed.

6.2 Creating Model

6.2.1 Libraries and Modules Required

```
# Data creation
!pip install -q supervision==0.9.0

import supervision as sv

# Data annotation
!pip install -q autodistill \
autodistill-grounded-sam \
autodistill-yolov8 \
roboflow

from autodistill.detection import CaptionOntology
from autodistill_grounded_sam import GroundedSAM

# Model training
!pip install -q ultralytics

from ultralytics import YOLOv8

# PyTorch with GPU support for data annotation and model training
!pip install --pre torch torchvision torchaudio -f https://download.pytorch.org/whl/nightly/cu121/torch_nightly.html

# Miscellaneous (Data cleaning, visualization, etc.)
from tqdm.notebook import tqdm
import cv2
import re
import os
```

Figure 6.2.1.1

The code snippet above illustrates the installation of several essential libraries required for the implementation of this project. These libraries include supervision which will assist in the data creation and also to create visualization of the data created.

Additionally, *autodistill* is installed in order to aid the annotation process. It enables researchers to distill large models like, in this case GroundedSAM, into smaller and more lightweight models. Without distillation, base models like GroundedSAM and GPT-4 can be very time consuming and resource intensive. Together, the combined use of GroundedSAM and *autodistill* facilitates streamlined data annotation processes.

To train the model, the library of Ultralytics is installed together with PyTorch. Utralytics provides the implementation for training deep learning models including YOLOv8.

Meanwhile, PyTorch with GPU support is installed to leverage GPU acceleration in the training of the model. The PyTorch nightly build for CUDA 12.1 was chosen as it was the only version that compatible with the GPUs that were used, a Tesla T4 and NVIDIA A100 Tensor Core GPU, both were equipped with the latest CUDA version, 12.2, which, as of March 2024, is not yet supported by stable build of PyTorch.

Finally, miscellaneous libraries such as *tqdm* for progress bar, cv2 to read images, re and os for data processing.

6.2.2 Data Creation

```
import os
HOME = "/content/drive/MyDrive/dataset"
print(HOME)
```

```
!mkdir {HOME}/images
!mkdir {HOME}/videos
```

```
VIDEO_DIR_PATH = f"{HOME}/videos"
IMAGE_DIR_PATH = f"{HOME}/images"
FRAME_STRIDE = 6
```

```
import supervision as sv
from tqdm.notebook import tqdm

video_paths = sv.list_files_with_extensions(
    directory=VIDEO_DIR_PATH,
    extensions=["mov", "mp4"])

print(video_paths)

for video_path in tqdm(video_paths):
    video_name = video_path.stem
    image_name_pattern = video_name + "-{:05d}.png"
    with sv.ImageSink(target_dir_path=IMAGE_DIR_PATH, image_name_pattern=image_name_pattern) as sink:
        for image in sv.get_video_frames_generator(source_path=str(video_path), stride=FRAME_STRIDE):
            sink.save_image(image=image)
```

Figure 6.2.2.1

Instead of manually capturing a large number of pictures one by one, a time and manpower-intensive task, the provided script is designed to automate the extraction of frames from video files. The process begins by creating two folders inside Google Drive, named "videos" and "images" respectively.

Videos of components of the FRL unit then may be uploaded into the “videos” folder and the parameter FRAME_STRIDE may be set to control the interval between consecutive frames to be extracted.

Subsequently, the code lists down all the video paths inside the folder. Next, it iterates through each video path while showing a progress bar from the *tqdm* library. For each video, it extracts frames using *supervision.get_video_frames_generator(),* with a specified interval defined by FRAME_STRIDE.

With supervision’s ImageSink, each of the frames are saved into the “images” folder. This method specifies the target directory and establishes the image naming pattern based on the video name, with additional numbers at the end to differentiate between frames.

```python
import supervision as sv

IMAGE_DIR_PATH = f"/content/images"
image_paths = sv.list_files_with_extensions(
    directory=IMAGE_DIR_PATH,
    extensions=["png", "jpg", "webp"])
print('image count:', len(image_paths))
```

```python
SAMPLE_SIZE = 16
SAMPLE_GRID_SIZE = (4, 4)
SAMPLE_PLOT_SIZE = (16, 16)
```

```python
import cv2
import supervision as sv

titles = [
    image_path.stem
    for image_path
    in image_paths[:SAMPLE_SIZE]]
images = [
    cv2.imread(str(image_path))
    for image_path
    in image_paths[:SAMPLE_SIZE]]

images, titles, grid_size, size = images, titles, SAMPLE_GRID_SIZE, SAMPLE_PLOT_SIZE
sv.plot_images_grid(images=images, titles=titles, grid_size=grid_size, size=size)
```

Figure 6.2.2.2

Afterward, the code snippet above will display the frames in a grid, just like shown on the right, to provide a preview of the images created.

6.2.3 Data Annotation

As mentioned previously, creating a dataset with high-quality annotations that requires manual labor can be financially and manpower-intensive. Rasmussen's, Kirk's, & Moeslund's (2021) study found that during annotation, it is not unusual to have detailed processes with multiple stages and users with different roles and that it can be difficult to implement in smaller projects where resources are limited. Hence, as the dataset reached 3,330 images, it was decided that utilizing *autodistill* would be the best choice here, despite its inability to match the quality of manual annotation.

```python
ontology=CaptionOntology({
    # key : value
    "black gear": "black gear"
})
```

Figure 6.2.3.1

The code starts by importing the *CaptionOntology* from *autodistill.detection*. An ontology defines how the base model is prompted and what the target model will predict. The ontology accepts a key-value pair, with the key being the input fed to the model and the value to act as the human readable name. The key input should be simple but explanative enough for the model to understand.

When dealing with niche objects, the extensive use of ontology is highly probable to overwhelm the model, leading to wrongfully labeled components. As a result, a decision was made to segregate the dataset based on components and execute the annotation process individually for each component.

In the code snippet above, the ontology was initially designed for black gears. An attempt to automate the process further by implementing a loop to handle all components was made. However, due to limitations in computer resources, namely GPU, the attempt to automate the process for multiple components proved unsuccessful.

```
DATASET_DIR_PATH = f"{HOME}/dataset"
IMAGE_DIR_PATH = f"{HOME}/images"

base_model = GroundedSAM(ontology=ontology)
dataset = base_model.label(
    input_folder=IMAGE_DIR_PATH,
    extension=".png",
    output_folder=DATASET_DIR_PATH)
```

Figure 6.2.3.2

Subsequently, a variable, DATASET_DIR_PATH, is defined in order to specify the destination where the results of the annotation should be saved. GroundedSAM was then imported, as acknowledged before, it will be used as the base model that will be distilled. GroundedSAM itself is based on Meta AI's Segment Anything Mode (SAM). With GroundedSAM, segmentation masks for individual picture segments are created using SAM, and the contents of a mask are labeled using Grounding DINO, which is an object detection model.

With the base model, ontology, and unlabeled images from IMAGE_DIR_PATH, the annotation process can be done in order to create a labeled dataset. GPU acceleration is not obligatory, but it is highly recommended. In general, compared to CPU (Intel Xeon CPU with 2 vCPUs (virtual CPUs), GPU acceleration decreases processing time by 98.5% or 66.73x faster with a Tesla T4 GPU and up to 99.4% or 165.46x faster with an NVIDIA A100 GPU. This improvement is particularly very notable due to the size of the dataset.

6.2.4 Data Cleaning

To enhance annotation quality, a cleaning process is needed due to two main issues: the model annotating an object multiple times or correctly annotating the object but incorrectly labeling the background with it. Both scenarios resulted in one image of an object having multiple annotations. Given the dataset size (2619 training and 711 validation images) and minimal occurrence of misannotated data, it was decided to remove the data that has multiple annotations. The function below was utilized to do the aforementioned task.

```
def delete_images_and_labels_for_multiple_objects(label_dir, image_dir):
    for label_file in os.listdir(label_dir):
        if label_file.endswith('.txt'):
            label_path = os.path.join(label_dir, label_file)
            with open(label_path, 'r') as file:
                lines = file.readlines()
            if len(lines) > 1:
                base_filename = os.path.splitext(label_file)[0]
                for ext in ['.png', '.jpg', '.jpeg', '.webp']:
                    image_path = os.path.join(image_dir, base_filename + ext)
                    if os.path.exists(image_path):
                        os.remove(image_path)
                        print(f"Deleted image file: {image_path}")
                os.remove(label_path)
                print(f"Deleted label file: {label_path}")
```

Figure 6.2.4.1

A total of 173 training and 15 validation images from each component were annotated wrongly, resulting in a combined total of 188 inaccurately annotated data points.

```
SAMPLE_SIZE = 500
SAMPLE_GRID_SIZE = (50, 40)
SAMPLE_PLOT_SIZE = (100, 30)

image_names = list(dataset.images.keys())[:SAMPLE_SIZE]
mask_annotator = sv.MaskAnnotator()
box_annotator = sv.BoxAnnotator()

images = []
for image_name in image_names:
    image = dataset.images[image_name]
    annotations = dataset.annotations[image_name]
    labels = [
        dataset.classes[class_id]
        for class_id
        in annotations.class_id]
    annotates_image = mask_annotator.annotate(
        scene=image.copy(),
        detections=annotations)
    annotates_image = box_annotator.annotate(
        scene=annotates_image,
        detections=annotations,
        labels=labels)
    images.append(annotates_image)

sv.plot_images_grid(
    images=images,
    titles=image_names,
    grid_size=SAMPLE_GRID_SIZE,
    size=SAMPLE_PLOT_SIZE)
```

Figure 6.2.4.2

With the code snippet above, the remaining 3,142 images were reviewed manually to make sure there are no multiple annotations left in a single image. The code was designed to display the first 500 images in a grid. The amount of 500 was deemed enough as this process was done separately for each component.

6.2.5 Data Manipulation

As the data annotation was done to each component's data separately, in each component's folder, there exist a 'valid' and a 'train' subfolder containing text annotation files. Within these files, the class label is uniformly assigned as 0, translating to a total of 1 class, even though there are a total of 10 of different components.

```
folder_path = "/content/dataset_tube/train/labels"
new_number = 1

# Iterate over each .txt inside the directory
for filename in os.listdir(folder_path, new_number):
    file_path = os.path.join(folder_path, filename)
    if filename.endswith(".txt"):
        with open(file_path, 'r') as file:
            content = file.read()

            # if .txt is not empty and first character is 0
            if len(content) > 1 and content.split()[0] == "0":
                new_content = re.sub(r'^0\b', new_number, content, count=1, flags=re.MULTILINE)

                # Write the new number back to the file
                with open(file_path, 'w') as modified_file:
                    modified_file.write(new_content)
                    print(f"Modified {filename}")
```

Figure 6.2.5.1

Implementing the script above entails the transformation of each component's class label data, replacing the default value of 0 with a unique identifier represented by the *new_number* variable. This manipulation results in each component having a different and unique class number. Afterwards, these 10 separated datasets can be combined into a single dataset with 10 different classes, totaling in 2446 training and 696 validation data.

6.2.6 Model Training

At the time of the development of this system in December 2023, YOLOv8 stood as the latest iteration of the YOLO model. However, by the end of February 2024, a new version of YOLO model, YOLOv9, was released, boasting claims of superior accuracy and computational efficiency.

Addressing the accuracy rate and computational efficiency rate discrepancy, a study (Wang, Yeh, & Liao, 2024), revealed that on the Microsoft Common Objects in Context Dataset (MS COCO), YOLOv9 surpassed YOLOv8's accuracy by up to 0.6%, despite YOLOv9 requiring 43% fewer calculations compared to YOLOv8.

In spite of these advantages, due to the unavailability of resources, namely GPU, and also the marginal discrepancy between YOLOv8's and YOLOv9's accuracy rate, it was determined that YOLOv8 is sufficient for this project. This decision was further influenced by the researcher's laptop, a MacBook Air M1, which will serve as the server for model inference and is still capable of performing this task with acceptable latency.

```
model = YOLOv8("yolov8n-seg.pt")
model.train(DATA_YAML_PATH, epochs=80, device=0)
```

Figure 6.2.6.1

With the code snippet above, we set the base model to be *yolov8n-seg.pt*. This signifies that we are using YOLOv8 Nano. In total, YOLOv8 offers five scaled versions: YOLOv8n (nano), YOLOv8s (small), YOLOv8m (medium), YOLOv8l (large), and YOLOv8x (xtra-large).

The YOLOv8 Nano was chosen due to its lightweight nature. A study from the Journal of Advances in Information Technology revealed that, in a multiclass classification problem, YOLOv8 Nano achieved a respectable accuracy of 69.1%, ranking second only to the xtra-large model (Pham & Le, 2024).

6.3 Model Deployment

In total, there are 2 main Django apps, one for the model inference, and the other is the rest of the system. This section will discuss *arweb*, the app that was built for the model inference.

6.3.1 Libraries, Modules, and Framework Required

As mentioned in literature review, the framework chosen will be the Django Framework. To be specific, Django 4.2.9. This version of Django was chosen as Django 4.2 is designated as a long-term support (LTS) release and will be supported until at least Q2 2026, which is longer compared to the newer Django 5.0 which is only supported until Q2 2025.

```
from django.shortcuts import redirect, render
from functools import wraps
import json
import cv2
import numpy as np
import base64
import asyncio

from channels.generic.websocket import AsyncWebsocketConsumer
from channels.exceptions import StopConsumer

from ultralytics import YOLO
from ultralytics.utils.plotting import Annotator, colors

from arweb.models import Product

import torch
```

Figure 6.3.1.1

The code snippet above contains all the necessary import statements to develop this particular Django App.

Asyncio is a Python library that provides facilities for writing asynchronous code. It allows the handling of many input/output bound tasks such as fetching data from multiple sources over the network without it halting the execution of other tasks. This is crucial as the server may need to handle multiple client connections at the same time.

The *AsyncWebsocketConsumer* class, imported from *channels.generic.websocket*, is a fundamental component for handling WebSocket connections in Django Channels-based applications. It enables bidirectional and real-time communication between a client and a server over a single, long-lived connection. This functionality empowers clients, or in this case, assemblers, to utilize their phone's camera, while allowing the server to perform model inference on the frames received from the client.

Still a part of Django Channels library, the *StopConsumer* is also imported. It is an exception which will be raised to signal the termination of a WebSocket consumer once the client is finished with their task. It will invoke the consumer to stop processing messages and disconnect the WebSocket connection.

Additionally, Ultralytics' YOLO is imported as it is required for the utilization of the YOLO-based .pt model created previously. Meanwhile, *Annotator* is imported for the purpose of annotating images in segmentation format.

```
INSTALLED_APPS = [
    "daphne",
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'arweb',
    'main',
    'captcha',
    'django_recaptcha',
    'allauth',
    'allauth.account',
    'bootstrap5',
    'channels',
]
```

```
# asgi.py
from channels.auth import AuthMiddlewareStack
from channels.routing import ProtocolTypeRouter, URLRouter
from channels.security.websocket import AllowedHostsOriginValidator
from django.core.asgi import get_asgi_application
from django.urls import path

import os
from arweb.views import ClassConsumer

os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'arproject.settings')

django_asgi_app = get_asgi_application()

application = ProtocolTypeRouter({
    # Django's ASGI application to handle traditional HTTP requests
    "http": django_asgi_app,
    "https": django_asgi_app,

    # WebSocket chat handler
    "websocket": AllowedHostsOriginValidator(
        AuthMiddlewareStack(
            URLRouter([
                path("", ClassConsumer.as_asgi()),
            ])
        )
    ),
})
```

Figure 6.3.1.2

By default, Django implements WSGI (Web Server Gateway Interface), which is synchronous and is suitable for handling synchronous request-response cycles. However, with the addition of WebSocket and asynchronous functions, the limitations of WSGI become apparent.

Asynchronous functionality necessitates a more flexible and scalable approach. Therefore, the use of ASGI (Asynchronous Server Gateway Interface) becomes imperative in this scenario as it enables concurrent processing of multiple connections and asynchronous operations, ensuring optimal performance and responsiveness. Hence, *daphne* is implemented into this system in order to enable the Django project to implement ASGI.

6.3.2 Django Models (model.py)

```python
class Product(models.Model):
        name            = models.CharField(max_length=255)
        product_id      = models.CharField(max_length=20, unique=True, blank=False)
        desc            = models.TextField(blank=True)
        created_at      = models.DateTimeField(auto_now_add=True)
        updated_at      = models.DateTimeField(auto_now=True)

        def __str__(self):
                return self.name
        def preview_name(self):
                return self.name[:40] + '...' if len(self.name) > 40 else self.name
        def preview_desc(self):
                return self.desc[:40] + '...' if len(self.desc) > 40 else self.desc
        def get_steps(self):
                steps = list(self.step_set.all())
                return steps
        def get_components(self):
                all_components = set()
                for step in self.get_steps():
                        all_components.update(step.components.all())
                return all_components

        steps                   = property(get_steps)
        components_property     = property(get_components)
```

Figure 6.3.2.1

```python
class Step(models.Model):
        product             = models.ForeignKey(Product, on_delete=models.CASCADE)
        title               = models.CharField(max_length=250)
        order               = models.IntegerField(null=False, validators=[MinValueValidator(1)])
        instruction         = models.TextField(blank=True)
        components          = models.ManyToManyField('Component', blank=True, related_name='step_components')

        def __str__(self):
                return self.title

class Component(models.Model):
        name                    = models.CharField(max_length=250)
        component_id            = models.CharField(max_length=15, unique=True)
        created_at              = models.DateTimeField(auto_now_add=True)
        updated_at              = models.DateTimeField(auto_now=True)
        is_active               = models.BooleanField(default=True)
        picture                 = models.ImageField(upload_to='component_pictures/', null=True, blank=True)

        def __str__(self):
                return self.name
```

Figure 6.3.2.2

In total, there are 3 models created inside *arweb* to represent the components, products, and steps.

First, the *Product* model consists of a name, ID, description column, creation timestamp, and an update timestamp. The ID column is mandatory to be filled and needs to be unique for each product. Meanwhile, the timestamps are filled automatically by the system. Additionally, there are functions to retrieve the product's related steps, and all the components associated with the product.

The *Step* model represents individual steps within the assembly process of each product. Each step is associated with a certain product and includes attributes such as title, order, text-based instruction, and also the components related to it.

Lastly, the *Component* model depicts the individual components utilized in the assembly process. It consists of fields such as name, component ID, creation and update timestamps, and an image of the component. The model also includes a boolean field *is_active* to indicate whether the component is currently active or not.

6.3.3 Model Inference

```python
def segment(frame):
    # Checking if CUDA-enabled GPU is available
    if torch.cuda.is_available():
        print("GPU detected")
    else:
        print("No GPU, expect slower inference")
    # Initiating the model first
    model = YOLO("/Users/nichdylan/Documents/AR_Assembly/Remote model/yolov8n.pt")
    # Replacing it with the trained model
    model_path = '/Users/nichdylan/Documents/AR_Assembly/arproject/arweb/static/trained_model.pt'
    model = YOLO(model_path)
    # Overwrite the names from the model, giving it more user-friendly names
    names = ["filter gear", "filter tube", "filter disc", "gear combined with tube",
             "gear, tubed, and disc combined", "filter body", "filter body screwed",
             "tube", "spring", "final"]
    # Set a threshold or confidence rate of the classification
    threshold = 0.1

    H, W, _ = frame.shape
    results = model.predict(frame)
    annotator = Annotator(frame, line_width=4)

    detected_object_names = []

    if results[0].masks is not None:
        clss = results[0].boxes.cls.cpu().tolist()
        confidences = results[0].boxes.conf.cpu().tolist()
        masks = results[0].masks.xy
        for mask, cls, confidence in zip(masks, clss, confidences):
            if confidence > threshold:
                mask_fill_color = colors(int(cls), True)
                filled_mask = frame.copy()
                cv2.fillPoly(filled_mask, [mask.astype(int)], mask_fill_color)
                object_name = names[int(cls)]
                detected_object_names.append(object_name)
                # Transparency level
                alpha = 0.3
                cv2.addWeighted(filled_mask, alpha, frame, 1 - alpha, 0, frame)
                #Give annotation on top of the image
                annotator.seg_bbox(mask=mask,
                                   mask_color=colors(int(cls), True),
                                   det_label=f"{names[int(cls)]} {confidence:.2f}")
        ret, buffer = cv2.imencode('.jpg', frame)
        segmented_frame = buffer.tobytes()
        return segmented_frame, detected_object_names
    else:
        ret, buffer = cv2.imencode('.jpg', frame)
        frame = buffer.tobytes()
        return frame, detected_object_names
```

Figure 6.3.3.1

The model inference utilizes the function *segment* and it takes an argument, which is the frame. With torch, it then checks whether a CUDA-enabled GPU is available. GPU usage is not a must but recommended. During testing, employing a dedicated GPU, NVIDIA GeForce RTX 3050, led to approximately a tenfold increase in inference speed compared to the Apple M1 chip's integrated GPU.

Subsequently, the model will proceed to run inference on the frame with the trained YOLOv8 model. If the results are not 0, meaning at least 1 component is detected, it will check whether the confidence rate met the threshold level.

The confidence rate refers to the measure of certainty or confidence that a detected component belongs to a certain class and it is calculated by multiplying the probability of an existing class object with the intersection over the union (IoU) score.

The function will then compare the confidence rate to the threshold value which currently is sitting at 0.1. A study found that using the confidence rate of 0.1 rather than higher rate such as 0.5, has the possibility of improving the model's mAP values and robustness (Azevedo et al., 2020). Mean Average Precision or mAP provides an overall measure of the model's performance across different classes. While the study also acknowledged that using a threshold value of 0.1 may negatively impact label and spatial quality metrics, the emphasis here is on maximizing mAP and robustness. This decision aligns with the primary objective of achieving accurate and reliable object detection.

If the confidence rate of a detection exceeds the value of threshold, using OpenCV, it will then proceed to draw an overlapping-colored mask on top of the components, segmenting it from the background. On top of it, Ultralytics' *Annotator* will annotate the segmentation mask with the class label and also the confidence rate.

The confidence rate is displayed, acknowledging the prototype nature of this system. Furthermore, the percentage can also serve to clarify instances of misdetection.

6.3.4 URL (urls.py)

Inside a Django project, the *urls.py* file serves as the routing mechanism, where it maps incoming requests to the appropriate views.

```
from django.urls import path
from .views import yolo_view

app_name = 'arweb'

urlpatterns = [
    path('yolo-view/<product_id>', yolo_view, name='yolo'),
]
```

Figure 6.3.4.1

For instance, the arweb's *url.py* above contains an entry which defines the url pattern for *yolo-view* together with a *product_id*, mapped into the *yolo_view* inside the *arweb* application. The *yolo_view* itself renders the *detect.html* template which contains product details, steps, and components.

6.3.5 HTML (detect.html)

```html
<audio id="correctAudio" src="{% static 'correct.mp3' %}" type="audio/mpeg"></audio>
<audio id="errorAudio" src="{% static 'error.mp3' %}" type="audio/mpeg"></audio>

<div class="p-4" style="border-radius: 0px;">
    {% for step in steps %}
        <div class="step plus-sans twohundred {% if forloop.first %}active-step{% endif %}" id="step{{ forloop.counter }}">
            <h4 class="sevenhundred" id="title{{ forloop.counter }}">Step {{ forloop.counter }}: {{ step.title }}</h4>

            <p id="whatyouneed{{ forloop.counter }}">What you need: {% for component in step.components.all %}{{ component }}, {% endfor %}</p>
        </div>

        <div class="modal" id="popup-box{{ forloop.counter }}" tabindex="-1" role="dialog" style="padding-top: 15%; max-width: 100%;">
            <div class="modal-dialog" role="document">
                <div class="modal-content">
                    <div class="modal-header">
                        <h5 class="modal-title plus-sans sevenhundred">Oops! Here's a hint for step {{ forloop.counter }}</h5>
                        <button type="button" class="close" onclick="closePopUp({{ forloop.counter }})">
                            <span aria-hidden="true">&times;</span>
                        </button>
                    </div>
                    <div class="modal-body">
                        {% if step.picture %}
                            <img src="{{ step.picture.url }}">
                        {% else %}
                            <em><p>Image not available</p></em>
                        {% endif %}

                        {% if step.instruction %}
                            <p>{{ step.instruction }}</p>
                            {% else %}
                            <em><p>Instructions are not available</p></em>
                        {% endif %}

                        <em><p>Click the X button on the top right to dismiss this message</p></em>
                    </div>
                </div>
            </div>
        </div>
    {% endfor %}

    <div class="row" style="max-width: 100%;">
        <div class="col">
            <button class="button-link" id="checkButton">Check</button>
        </div>
        <div class="col my-auto">
            <p class="my-auto" id="countdownTimer"></p>
        </div>
        <div class="col text-right">
            <button class="button-link" id="nextButton" style="display: none;">Next</button>
        </div>
    </div>
</div>

<video style="width: 100%; border-radius: 25px;" id='camera_src' autoplay playsinline></video>
<canvas id="canvas" style='display:none; max-width: 100%;'></canvas>

<div style="border-radius: 10px; width: 100%;" class="align-center-scale">
    <img style="display: none; width: 100%; border-radius: 25px; padding: 0;" id='client' src="https://live.staticflickr.com/65535/53529488187_bff285b40b_b.jpg" class="video" alt="screen">
</div>
```

Figure 6.3.5.1

Referring to the previous statement, the *detect.html* contains the instructions for each step, the components needed, and also the buttons to either check, show hints, and proceed to the next step. Additionally, it also contains 2 audios that are to be played, depending on whether the model detected the required components or not. The "Check" button will invoke the *sendViaWebSocket()* function inside the JavaScript.

6.3.6 JavaScript to Facilitate Media Stream and WebSocket

The JavaScript file serves multiple purposes such as displaying the current steps, timer, pop up box, and others. However, its main purpose is to facilitate the WebSocket connection by sending captured frames from the client's camera to the server to be processed and receiving the processed frames back as well.

```
function startCamera() {
    $("#client").hide();
    var samsung_camera = false
    var deviceId

    if (navigator.mediaDevices.getUserMedia) {
        var constraints = {
            video: {
                facingMode: 'environment'
            }
        };
        navigator.mediaDevices.enumerateDevices()
            .then(function(devices) {
                devices.forEach(function(device) {
                    if (device.kind === 'videoinput' && device.label.toLowerCase().includes('camera2 0, facing back')) {
                        deviceId = device.deviceId;
                        samsung_camera = true
                    }
                });

                if (samsung_camera) {
                    constraints.video.deviceId = { exact: deviceId };
                    console.log('Found main camera, device id ', deviceId);
                } else {
                    console.log('Camera "camera2 0, facing back" not found.');
                }
                console.log(constraints)
                navigator.mediaDevices.getUserMedia(constraints)
                    .then(function(stream) {
                        video.srcObject = stream;
                        video.setAttribute('autoplay', true);
                        video.play();
                    })
                    .catch(function(error) {
                        console.error('Error accessing camera:', error);
                    });
            })
            .catch(function(error) {
                console.error('Error enumerating devices:', error);
            });
    }
}
```

Figure 6.3.6.1

The startCamera() function relies on navigator.mediaDevices.getUserMedia to initiate MediaStream from the user's phone camera. In *navigator.mediaDevices.getUserMedia*, the *facingMode* property is set to *environment* to make sure that the system will utilize the phone's back camera. However, due to the wide variety of Android phone models, selecting the correct camera is challenging. Modern Android phones often have multiple cameras, such as the Samsung Galaxy A52's quad-camera setup. Identifying the main camera is difficult, especially as cameras like the Ultra-Wide distort images and the Telephoto zoom unnecessarily, impacting model inference.

```
function enumerateDevices() {
    navigator.mediaDevices.enumerateDevices()
        .then(function(devices) {
            console.log("Devices:", devices);
        })
        .catch(function(err) {
            console.error("Error enumerating devices:", err);
        });
}
```

Figure 6.3.6.2

However, when tested with a Samsung Galaxy A52, a Samsung Galaxy Tab S7 FE, and a Samsung Galaxy S10, all devices consistently labeled their main camera as "camera2 0, facing back". Therefore, the startCamera() function was updated to utilize enumerateDevices() to identify the camera with this specific label. This led to the adjustment of the startCamera() function to use enumerateDevices() to locate the camera with that certain label. However, when a non-Samsung device, like a MacBook Air, attempted to access the system, it encountered an error because its camera was labeled differently. To resolve this, the function was updated to use any camera when the main camera label is unknown.

```
function sendViaWebSocket(){
    if (checkCount >= 5) {
        $("#nextButton").show();
    }
    if (mode == true){
        mode = false;
        checkCount++;
        var ws = new WebSocket(
            ws_scheme + '://' + window.location.host + '/'
        );
        ws.onopen = (event) => {
            console.log('WebSocket Connected!');
        };
        ws.onmessage = (event) => {
            var data = JSON.parse(event.data);
            console.log("HERES DATA", data)

            if ('detected_object_names' in data) {
                console.log("detected object names", data.detected_object_names)

                currentComponents = stepComponents[currentStep];
                console.log("components CURRENT", currentComponents)

                var detectedObjects = data.detected_object_names;
                foundMatch = currentComponents.every(component => detectedObjects.includes(component));
                frameUpdate = data.image;
                if (foundMatch){
                    // Display the processed frame into the HTML
                    img.src = "data:image/jpeg;base64," + frameUpdate;
                    $("#client").show();
                    // Close the WebSocket connection
                    ws.close();
                    // Play the correct audio
                    var audioElement = document.getElementById('correctAudio');
                    audioElement.play();

                    if (currentStep === totalSteps) {
                        $("#nextButton").text("Done");
                    }
                    var title = document.getElementById("title"+currentStep);
                    if (title) {
                        title.style.color = 'green';
                        title.textContent += " ✅";
                    }
                    var whatyouneed = document.getElementById("whatyouneed"+currentStep);
                    if (whatyouneed) {
                        whatyouneed.textContent = "Great job! 👏";
                    }
                    $("#checkButton").hide();
                    $("#nextButton").show();
                    stopMode()
                }
            }
        };
        ws.onclose = (event) => {
            console.log('WebSocket connection terminated');
        };
        if (navigator.mediaDevices.getUserMedia) {
            var constraints = {
                video: {
                    facingMode: 'environment'
                }
            };
            var samsung_camera = false
            navigator.mediaDevices.enumerateDevices()
                .then(function(devices) {
                    devices.forEach(function(device) {
                        if (device.kind === 'videoinput' && device.label.toLowerCase().includes('camera2 0, facing back')) {
                            deviceId = device.deviceId;
                            samsung_camera = true
                        }
                    });

                    if (samsung_camera) {
                        constraints.video.deviceId = { exact: deviceId };
                        console.log('device id ', deviceId);
                    } else {
                        console.error('Camera "camera2 0" not found.');
                    }

                    navigator.mediaDevices.getUserMedia(constraints)
                        .then(function(stream) {
                            video.srcObject = stream;
                            video.setAttribute('autoplay', true);
                            video.play();
                            video.addEventListener('loadedmetadata', setCanvasSize);

                            var delay = 500; // adjust the delay to fit your needs (ms)
                            var jpegQuality = 0.6; // adjust the quality to fit your needs (0.0 -> 1.0)

                            setInterval(function() {
                                context.drawImage(video, 0, 0, video.videoWidth, video.videoHeight);
                                canvas.toBlob(function(blob) {
                                    if (ws.readyState == WebSocket.OPEN) {
                                        if (mode == false){
                                            // Converting blob to base64
                                            var reader = new FileReader();
                                            reader.readAsDataURL(blob);
                                            reader.onloadend = function() {
                                                // Construct the message object with base64-based blob & currentComponents
                                                var base64data = reader.result;
                                                var message = {
                                                    image: base64data,
                                                    currentComponents: stepComponents[currentStep]
                                                };
                                                // Convert the message to JSON before sending
                                                ws.send(JSON.stringify(message));
                                            };
                                        }
                                    }
                                }, 'image/jpeg', jpegQuality);
                            }, delay);
                        });
                })
                .catch(function(error) {
                    console.error('Error enumerating devices:', error);
                });
        }
    }
    else if (mode == false){
        mode = true;
        console.log("Mode stopped, status: ", mode)
        stopMode()
    }
};
```

Figure 6.3.6.3

Another crucial function that will be invoked when the user clicks the button "Check" is the *sendViaWebSocket()*, which is responsible for starting capturing frames and sending it over to the server via WebSocket. There is a 500ms delay to prevent any backlog of images awaiting processing, as the average latency sits at 291ms per image, occasionally spiking to as high as 387ms.

Its execution ceases either upon the expiration of a 10-second timer or upon the detection of the necessary components, ensuring the server's resources are used efficiently. When the necessary components are detected, the system will then show a message and play a sound indicating the completion of the step. It will also display a "Next" button for the user to continue to the next step.

```
timeoutId = setTimeout(function() {
    if (!mode){
        sendViaWebSocket();
        $("#checkButton").text("Check");
        $("#countdownTimer").text("Times out").addClass("plus-sans fivehundred");;
        clearInterval(countdownInterval);
        openPopUp(currentStep) // To show the hints (if available)
        var audioElement = document.getElementById('errorAudio');
        audioElement.play();
    }
},10000); // Time out in milliseconds, 10000ms = 10s
```

Figure 6.3.6.4

Upon completion of the sendViaWebSocket() function without detecting the necessary components, it will trigger an audible signal, signifying incompleteness. Additionally, openPopUp() will be triggered, displaying a popup that provides hints on object positioning or procedural guidance to complete the step.

Chapter 7: Testing

7.1 Introduction

The current chapter will primarily discuss the testing phase of the developed system. The testing phase is crucial in order to evaluate the effectiveness, and efficiency of the system. This chapter provides an evaluation and examination of the testing procedure, addressing the creation of test cases, the implementation of the aforementioned test cases, and the outcome.

It is essential to highlight that the system under consideration is a prototype. As such, it is essential to let the testers acknowledge its limitations, notably its ability to detect objects only from specific angles due to its preliminary development stage. Additionally, testers must be aware that, owing to its prototype status, the system might misdetect other objects as the intended target due to the current limitation of resources, such as GPU, manpower, and dataset. This acknowledgment is particularly pertinent given the challenging nature of managing such constraints within the framework of a solo project. This recognition holds particular significance, given the formidable challenge of managing these constraints within the scope of an independent project.

In addition to that, it is also crucial to recognize that this system is a guidance system that aims to guide assemblers on assembling the products. Similar to text-based guidelines, it is not a quality assurance system which is able to verify the quality of the assembled product.

7.2 Test Cases

<table>
<tr><td colspan="3">Test Case ID: 01</td><td colspan="2">Test Priority: Medium</td></tr>
<tr><td colspan="5">Test Title: Device compatibility</td></tr>
<tr><td colspan="5">Description:

This test case is to make sure that the system can be accessed successfully from the client's phone, no matter the operating system.</td></tr>
<tr><td colspan="5">Pre-Conditions:

1. The Django Project must be running a local area network (LAN).</td></tr>
<tr><td colspan="5">Valid Test</td></tr>
<tr><td>Test Step</td><td>Test Description</td><td>Expected Result</td><td>Actual Result</td><td>Status</td></tr>
<tr><td>The user uses a Samsung Galaxy A52 to access the system.</td><td>The user uses an Android-based, Samsung Galaxy A52 to access the web-based system.</td><td>The web browser displays the web application and all functions work as expected.</td><td>The user can access the web application and utilizes all the functions.</td><td>Success</td></tr>
<tr><td colspan="5">Post Condition:

The web application loads successfully, and the user can enter the website.</td></tr>
<tr><td colspan="5">Invalid Test</td></tr>
<tr><td>Test Step</td><td>Test Description</td><td>Expected Result</td><td>Actual Result</td><td>Status</td></tr>
<tr><td>The user uses an iPhone XR to access the system.</td><td>The user accesses the system with an iOS device, iPhone XR.</td><td>The web browser displays the web application and all functions work as expected.</td><td>The user’s phone is unable to let the system access the phone’s camera due to security measures done in iOS and iPadOS.</td><td>Fail</td></tr>
<tr><td colspan="5">Due to Apple’s strict control over iOS and iPadOS, this is something that is outside the control of the developer and would need Apple itself to intervene.</td></tr>
</table>

Table 7.2.1

Test Case ID: 02			Test Priority: Medium	
Test Title: Registering Assembly Account				
Description: This test case is to make sure that the admin can only register an account using a new and unique Assembler's ID.				
Pre-Conditions: 1. The admin must be already logged in to the system. 2. The admin possesses the Assembler's credentials.				
Valid Test				
Test Step	Test Description	Expected Result	Actual Result	Status
The admin registers an account using a new and unique ID.	The admin registers a new Assembler's account with a new Assembler's ID.	The system registers the new account.	The system successfully registered the account.	Success
Post Condition: The system allows the admin to register a new account.				
Invalid Test				
Test Step	Test Description	Expected Result	Actual Result	Status
The admin uses a used Assembler's ID.	The admin uses an old and used Assembler's ID. Each assembler is supposed to only have one account.	The system displays an error message.	The system warns that the ID has been used before, therefore the account is not registered.	Success

Table 7.2.2

Test Case ID: 03			Test Priority: High	
Test Title: Edit Guidance Test				
Description: This test case is to make sure that the admin can edit the guidance text or steps descriptions.				
Pre-Conditions: 1. The admin is logged in into the system. 2. The user's account type is admin, not assembler.				
Valid Test				
Test Step	Test Description	Expected Result	Actual Result	Status
The admin changes the step's descriptions, title, and also the components needed.	The admin made some modification to the step's descriptions, title, and also the components needed.	The changed steps can be displayed and will not affect the system.	The modification resulted in error in the model inference process.	Fail
Post Condition: The editing step function will be disabled as the model is already trained and tailored for only FRL Unit assembly process.				

Table 7.2.3

Test Case ID: 04			Test Priority: Medium	
Test Title: Authentication/Login Test				
Description: This test case is to make sure that the client (assembler and admin) can authenticate themselves to the system.				
Pre-Conditions: 1. The web application can be launched without any trouble. 2. The user has knowledge of their credentials.				
Valid Test				
Test Step	Test Description	Expected Result	Actual Result	Status
The user types in the correct credentials.	The user enters their correct email and password to login to the web application.	The web browser authenticates the user.	The user managed to login to the web application.	Success
Post Condition: User is authenticated and allows the user to use the system.				
Invalid Test				
Test Step	Test Description	Expected Result	Actual Result	Status
The user types in the wrong credentials.	The user enters the password to login to the system.	The system will not allow the user to login.	The user is unable to login to the system.	Success

Table 7.2.4

Test Case ID: 05			Test Priority: High	
Test Title: View Assembly Steps Test				
Description: This test case is to make sure that the assembler can view the assembly steps preview before starting the AR System.				
Pre-Conditions: 1. The assembler is logged in into the system. 2. The assembler is on the homepage.				
Valid Test				
Test Step	Test Description	Expected Result	Actual Result	Status
The assembler clicks on the “Details” or “**i**” button, next to the product title.	The assembler clicks on the button to display more details about the product.	The system displays a pop-up box containing information on the steps.	The system displayed a pop-up box containing correct information on the steps.	Success
Post Condition: The system is able to show a pop-up containing details of the assembly steps.				
Invalid Test				
Test Step	Test Description	Expected Result	Actual Result	Status
The assembler does not click on the “Details” or “**i**” button.	The assembler does not click on the button to display more details about the product.	The system does not display a pop-up box containing information on the steps.	The system did not display a pop-up box containing correct information on the steps.	Success

Table 7.2.5

Test Case ID: 06			Test Priority: High	
Test Title: Object Detection Test				
Description: This test case is to make sure that the model inference and segmentation process is able to run successfully.				
Pre-Conditions: 1. The user is logged in into the system. 2. The user's account type is assembler, not admin.				
Valid Test				
Test Step	Test Description	Expected Result	Actual Result	Status
The user starts the assembly and shows the correct components inside the camera view.	The user opens the system and starts the assembly process of an FRL Unit. Then the user proceeds to put the required components inside the camera view.	Model detected the required components.	The model managed to detect the components correctly.	Success
Post Condition: The server is able to receive the images, run model inference on it, and send it back to the phone.				
Invalid Test				
Test Step	Test Description	Expected Result	Actual Result	Status
The user moves the camera away from the components.	The user moves the camera view away from the components, showing an unrelated background.	The model wrongly detected the background as a component.	The model wrongly detected the background as a component.	Success

Table 7.2.6

7.3 User Acceptance Testing (UAT)

7.3.1 UAT-1

<table>
<tr><td colspan="6">AR Enabled Assembly Guidance System

A system that guides and verifies the assembly process with the aid of computer vision, object detection and segmentation, and image masking. All using just a mobile phone, something that almost everyone nowadays has. This allows companies to train their assembler, while still keeping costs low. The object detection and segmentation model is trained on a custom dataset.</td></tr>
<tr><td>Tester Name: Ling Yang En</td><td colspan="3">Position: Student</td><td colspan="2">Date: March 19th, 2024</td></tr>
<tr><td rowspan="2">Name of Category</td><td colspan="5">Score (1 is the lowest, 5 is the highest)</td></tr>
<tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr>
<tr><td>User Friendliness</td><td></td><td></td><td>✓</td><td></td><td></td></tr>
<tr><td>Interface Design</td><td></td><td></td><td></td><td>✓</td><td></td></tr>
<tr><td>Model Accuracy</td><td></td><td></td><td></td><td>✓</td><td></td></tr>
<tr><td>System Performance</td><td></td><td></td><td></td><td></td><td>✓</td></tr>
</table>

Comment From Tester	The system is working as expected. For a prototype this fulfills all my expectations. However, I think it is better for the hint button to be placed at the top. For first-time users, the hint button is hidden at the bottom, potentially causing them to overlook it.
Action Taken by Developer	I have considered your recommendation and will proceed with implementing the change as you suggested. The hint button will be moved in order to improve visibility.

Table 7.3.1.1

7.3.2 UAT-2

<table>
<tr><td colspan="6">AR Enabled Assembly Guidance System

A system that guides and verifies the assembly process with the aid of computer vision, object detection and segmentation, and image masking. All using just a mobile phone, something that almost everyone nowadays has. This allows companies to train their assembler, while still keeping costs low. The object detection and segmentation model is trained on a custom dataset.</td></tr>
<tr><td>Tester Name: Tetsuro Horie</td><td colspan="2">Position: Student</td><td colspan="3">Date: March 13th, 2024</td></tr>
<tr><td rowspan="2">Name of Category</td><td colspan="5">Score (1 is the lowest, 5 is the highest)</td></tr>
<tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr>
<tr><td>User Friendliness</td><td></td><td></td><td>✓</td><td></td><td></td></tr>
<tr><td>Interface Design</td><td></td><td></td><td>✓</td><td></td><td></td></tr>
<tr><td>Model Accuracy</td><td></td><td></td><td></td><td>✓</td><td></td></tr>
<tr><td>System Performance</td><td></td><td></td><td></td><td></td><td>✓</td></tr>
</table>

Comment From Tester	Overall, the system is acceptable. However, in my opinion, the user experience can be improved by allowing the user to return to the previous step.
Action Taken by Developer	Changes were made on the system in order to allow the assemblers to return to their previous step.

Table 7.3.2.1

7.4 FRL Unit Construction Time Comparison

To demonstrate the advantages of this system, an experiment will be conducted comparing the time required to build an FRL unit using text-based guidelines alone versus using the AR-enabled system.

Two separate tests were conducted on two different subjects in order to validate the effectiveness and efficiency of the system. One subject was asked to assemble the FRL unit with just the guidance of a text-based guidance. Meanwhile, the other one had the opportunity to assemble the product with the aid of the AR Enabled Assembly Guidance System.

In the first test, Ling Yang En, a student with no prior knowledge of FRL Unit assembly practice, was tasked with assembling the FRL unit using the AR-enabled system and Ling completed the task in just 3 minutes and 45 seconds.

Meanwhile, in the second test, Foo Fang Khai, another student with no prior experience with an FRL Unit, was also tasked to assemble the FRL Unit, but only with text-based guidance. Foo needed 6 minutes and 10 seconds to assemble the unit. Foo also stated that he was having problems as there was a lack of visual aid.

Subject	Details	Time recorded
Ling Yang En	Using AR-Enabled Assembly Guidance System	3 minutes 45 seconds
Foo Fang Khai	Using a text-based guidance	6 minutes 10 seconds

Table 7.4.1

Chapter 8: Critical Evaluation and Conclusion

8.1 Introduction

In this chapter, a comprehensive evaluation of the project and also self-assessment will be included. Challenges encountered during the development of this project and potential improvement to be implemented in the future will also be addressed.

8.2 Process Evaluation

The initial concept of the AR-Enabled Assembly Guidance System emerged from the observation that traditional text-based instructions have remained static, relying solely on text-based manuals with limited or even no visual aids. Leveraging the widespread adoption of smartphones, this system utilizes technology to revolutionize assembly guidance. By utilizing the camera and display capabilities of smartphones, users can receive dynamic instructions. Through the overlay masks on the camera display, users can be guided step by step in almost real-time, increasing confidence and reducing errors.

Initial phase involved conducting a comprehensive review of current literature and conducting research, particularly on the manufacturing and assembly business. The literature review covered diverse technical domains such as the IDE, algorithms to be used for object detection, the framework for Augmented Reality, and lastly, which library to be used for showing extra information on top of the camera view. However, during the implementation process of this project, it was realized that the use of an AR Framework is redundant, leaving only YOLOv8, as the object detection model, and OpenCV, to visualize the overlaying information, to be implemented and allowing the implementation of AR. On top of that, several libraries that were not covered during the literature review, were found to be useful and were implemented during the implementation process. Most notably, these libraries include *autodistill*, *supervisor*, *django-channels*, and *daphne*.

Subsequently, a face-to-face interview with the owner of a Selangor company that designs & supplies industrial automation systems was conducted. The owner highlighted the challenge of training new interns and the need for a system to facilitate self-directed training, reducing the need for constant supervision. This system would enable interns to learn autonomously, freeing up senior staff to focus on more critical tasks and enhancing overall efficiency within the company.

The following phase entailed the development process of the system. The process involved the collection of pictures, which are required for the dataset. A total of 3,330 pictures were collected and then annotated with the help of *autodistill* and GroundedSAM. Among these, 188 images were discarded due to inaccuracies or erroneous data. With the obtained and cleaned dataset, a YOLOv8 model was then trained and implemented inside the Django-based web-based system, where it can run model inference on frames that are inputted into it. The system was designed in a way in order to enable almost real-time data processing. In addition to the resources discussed in the literature review, other supplementary libraries and resources have also been incorporated into the implementation process. As mentioned before, these include *autodistill*, *supervisor*, *django-channels*, and *daphne*.

Documentation, especially Chapter 6, was continuously updated throughout the entirety of the development process. Subsequently, the system underwent comprehensive testing, with each stage meticulously documented. As feedback and insights from supervisors, testers, and subject tests were received, the system underwent iterative changes, incorporating valuable input to enhance its functionality and usability.

8.3 Self-Evaluation

In the dataset creation, model training, and website development and deployment, I have gained a lot of valuable insights and learning experiences. Initially, I extensively researched and created a Python script that assists me in the dataset creation process. After gathering the much-needed data, I then train the model, ensuring its accuracy and reliability. Throughout the development process, I've learned about the many methods of image segmentation, image processing, asynchronous functions, object detection algorithms, WebSocket, and the tiring process of dataset creation.

In more detail, I experimented with various ways to create a dataset to train the object segmentation model and even developed a Python script that helps with the data labeling process. Upon deployment through the website, I conducted thorough testing and validation to ensure the model's usability.

In the implementation of the model inference process inside the Web-based system, I encountered issues on connecting the phone's camera with the server and also the slow inference process due to the unavailability of a dedicated GPU. However, these issues were resolved through the implementation of asynchronous functions and some modifications to the process of sending frames from the phone's camera. Moreover, feedback from testers refined the website based on user preferences, which enhanced my understanding of the user's preferences. Overall, the deployment process resulted in a seamless and accessible platform, providing assemblers with a user-friendly web-based system that can guide them in the assembly process.

Finally, I am proud of the AR-Enabled Assembly Guidance System that I developed. It showcases my ability to plan and execute a solution for a problem using technology and skill sets which

I gained during my study. This project has taught me valuable skills and knowledge that will help build a solid foundation for my future career, whether it be a Web Developer or Data Analyst.

8.4 Challenges

Numerous challenges were encountered during the development process of this system. There is a lack of information on a system that is similar. Furthermore, it's worth noting that the FRL Unit, specifically chosen for integration within this system, does not currently have an accessible dataset available for use. Thus, a Python script was developed, it automatically turns videos into images. After that, using *autodistill*, it distills down Meta AI’s *Segment Anything Model* (SAM) in order to make it usable inside the developer’s computer. Utilizing the distilled model, it proceeds to provide the labeled and segmented object data for each image.

Second, due to modern phone’s multiple camera setup, the system was facing a challenge in detecting the correct main camera. Using the telephoto camera or Ultrawide camera instead of the main one can lead to failure in detecting the component due to distortion and zooming. It is possible to pick a camera based on its ID, however, due to the variety of phones available in the market, it is almost impossible for the JavaScript API to always find the correct camera. Testing on 3 Samsung devices, shows that Samsung is consistently labeling their main camera with the same label. Hence, the JavaScript was then modified to be able to find the main camera using the label if it’s a Samsung device, or any camera if it is not a Samsung device.

Third, another issue was encountered when the website was hosted within the Local Area Network (LAN). Initially, instead of using JavaScript to access the phone’s camera, OpenCV was used instead. However, this resulted in the client, or in this case the phone, trying to access the server’s camera, instead of its own camera. Therefore, JavaScript's getUserMedia was then implemented, allowing all devices, except iOS and iPadOS devices, to use their own camera. WebSocket would then be used to enable communication between the server and client.

Fourth, due to the implementation of WebSocket, another issue was encountered where the utilization of an asynchronous function was required. This would also mean that Asynchronous Server Gateway Interface (ASGI) servers like Daphne are crucial. However, Django's development server is using WSGI servers like Gunicorn. Therefore, *daphne* for ASGI implementation and *django-channels* to assist WebSocket on server side were then installed. Afterward, the ASGI configuration and Django Channels functions were modified to fulfill the requirements of the system.

Fifth, the web-based system is hosted with a MacBook Air M1, a laptop that does not possess a dedicated GPU. Due to this, the model inference runs slowly and this may cause a backlog of images to be processed. When compared to another device with a dedicated GPU, NVIDIA GeForce RTX

3050, it led to around tenfold increase in inference speed compared to the Apple M1 chip. Hence, the JavaScript code was then modified in a way that the client will only send an image twice in a second. Additionally, the image will be resized into 40% of the original size, this was done in order to speed up the inference and also the latency.

The final challenge encountered pertained to interoperability among iOS and iPadOS devices, which was primarily attributed to Apple's stringent security and privacy protocols within its operating systems. This prevented the utilization of the phone's media devices on websites lacking security measures. This unsolved nature of this challenge stemmed from factors beyond the developer's control and would require Apple itself to intervene. It should be noted that an experimental app called Bowser, which was reported in a 2014 study can enable the usage of getUserMedia() on websites it deemed as insecure (Holmgren, Johansson, & Andersson, 2014). However, as of the current year, 2024, it has since been delisted from the App Store.

8.5 Future Enhancement

One of the biggest challenges of this system was the unavailability of GPUs. This made the process of data labeling much slower. Compared to a CPU (Intel Xeon CPU with 2 vCPUs), GPU acceleration reduces processing time significantly: by 98.5% (or 66.73x faster) with a Tesla T4 GPU and up to 99.4% (or 165.46x faster) with an NVIDIA A100 GPU. This improvement is particularly noteworthy given the dataset's size. This means, a 100-minute data labeling task with the CPU can be done in just 0.6 minutes or 36 seconds with the A100 GPU. Faster labeling task allows for more images to be labeled, hence increasing the size of the dataset.

In the future development, GPUs may be involved in the implementation in order to increase the size of dataset and number of images which can be used for training. On top of that, GPU acceleration together with more RAM memory can also be implemented in the model training process in order to speed up training duration and also facilitate the use of larger batches. More epochs and larger batch sizes can help in increasing the accuracy of the YOLOv8 model.

Lastly, further research into hosting a website with HTTPS can be done in order to enable camera usage on Apple iOS and iPadOS devices. Such efforts will not only enhance the user experience but also contribute to increasing the security of the web-based system. This would also allow more people to be able to access the system.

In conclusion, potential future improvements cover computational resources and enhance the security of the web-based system, while allowing iOS and iPadOS devices to access the system at the same time. Increasing the dataset size significantly and experimenting with more epochs during model training would lead to substantial improvements. These enhancements would make the system more inclusive, especially for iOS and iPadOS devices, faster by leveraging additional computational resources, and more accurate by allowing the model to learn from a larger and more diverse set of examples.

9. References

Amato, A., Venticinque, S., & Beniamino Di Martino. (2013). Image Recognition and Augmented Reality in Cultural Heritage Using OpenCV. https://doi.org/10.1145/2536853.2536878

Anantharaman, R., Velazquez, M., & Lee, Y. (2018). *Utilizing Mask R-CNN for Detection and Segmentation of Oral Diseases. IEEE Xplore*. https://doi.org/10.1109/BIBM.2018.8621112

Al-Saqqa, S., Sawalha, S., & Nabi, H. A. (2020). Agile Software Development: Methodologies and Trends. *International Journal of Interactive Mobile Technologies (iJIM)*, 246–270.

Azevedo, T., Jong, R., Mattina, M., & Maji, P. (2020). *Stochastic-YOLO: Efficient Probabilistic Object Detection under Dataset Shifts*. https://arxiv.org/pdf/2009.02967.pdf

Bottani, E., & Vignali, G. (2019). Augmented reality technology in the manufacturing industry: A review of the last decade. *IISE Transactions*, 284-310.

Botto, C., Cannavò, A., Cappuccio, D., Morat, G., Sarvestani, A. N., Ricci, P., Saturnino, A. (March 2020). Augmented Reality for the Manufacturing Industry: The Case of an Assembly Assistant. *2020 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, 299-304.

Ciesielska, M., Boström, K. W., & Öhlander, M. (2017). Chapter 2: Observation Methods. In Qualitative Methodologies in Organization Studies: Volume II: Methods and Possibilities (pp. 33–52). Palgrave Macmillan.

Croucher, S. M., & Cronn-Mills, D. (2021). Interviewing. In S. M. Croucher, & D. Cronn-Mills, Understanding Communication Research Methods (pp. 113-128). New York: Routledge.

Dawadi, S., Shrestha, S., & Giri, R. A. (2021). Mixed-Methods Research: A Discussion on its Types, Challenges, and Criticisms. *Journal of Practical Studies in Education* (S. 25–36).

Douglas, A., Kimbaris, G., & Stein, L. (2021). A Video is Worth a Thousand Words: Perceived Learning Value in Video versus Text-Based Cases (No. 46). *Neurology*.

Dwivedi, N., Katiyar, D., & Goel, G. (March 2022). A Comparative Study of Various Software Development Life Cycle (SDLC) Models. *International Journal of Research in Engineering, Advanced Technology and Management*, 5(3), 141-144.

Fariyanto, F., Suaidah, & Ulum, F. (June 2021). Perancangan Aplikasi Pemilihan Kepala Desa Dengan Metode Ux Design Thinking (Studi Kasus: Kampung Kuripan). *Jurnal Teknologi dan Sistem Informasi (JTSI)*, 52-60.

Gurung, G., Shah, R., & Jaiswal, D. P. (August 2020). Software Development Life Cycle Models-A Comparative Study. *International Journal of Scientific Research in Computer Science Engineering and Information Technology,* 6(4), 30-37.

Hartson, R., & Pyla, P. S. (2019). The UX book: Agile UX design for a quality user experience (2nd ed.). Morgan Kaufmann.

Holmgren, M., Johansson, D., & Andersson, K. (n.d.). A Web-based Time Machine with Augmented Reality. Retrieved March 23, 2024, from https://www.diva-portal.org/smash/get/diva2:1005220/FULLTEXT01.pdf

Johansson, J. (2021). *Performance and Ease of Use in 3D on the Web: Comparing Babylon.js with Three.js*. https://www.diva-portal.org/smash/get/diva2:1523176/FULLTEXT01.pdf

Kraiger, K., & Ford, J. (2021). The Science of Workplace Instruction: Learning and Development Applied to Work. *Annual Review of Organizational Psychology and Organizational Behavior*, 8(1), 45-72.

Krosnick, J. A. (2018). Chapter 9: Questionnaire Design. In D. L. Vannette, & J. A. Krosnick, *The Palgrave Handbook of Survey Research* (pp. 264-313). Palgrave Macmillan.

Kühn-Kauffeldt, M., & Böttcher, J. (2020). Open Source Augmented Reality Applications for Small Manufacturing Businesses. *Augmented Reality and Virtual Reality* (pp. 243-251).

Lai, Z.-H., Tao, W., Leu, M. C., & Yin, Z. (April 2020). Smart augmented reality instructional system for mechanical assembly towards worker-centered intelligent manufacturing. *Journal of Manufacturing Systems*, 69-81.

Okesola, O. J., Adebiyi, A. A., Owoade, A. A., Adeaga, O., Adeyemi, O., & Odun-Ayo, I. (2020). Software Requirement in Iterative SDLC Model. *Intelligent Algorithms in Software Engineering*, 26-34.

Pham, T.-L., & Le, V.-H. (2024). Ovarian Tumors Detection and Classification from Ultrasound Images Based on YOLOv8. *Journal of Advances in Information Technology*, 15(2), 264–275. https://doi.org/10.12720/jait.15.2.264-275

Plainer, M. (2021). *Practical Study of Visual Studio Code Practical Course-Contributing to an Open-Source Project*. https://www21.in.tum.de/teaching/osp/WS20/assets/fr-plainer-vscode.pdf

Rasmussen, C., Kirk, K., & Moeslund, T. B. (2022). *The Challenge of Data Annotation in Deep Learning—A Case Study on Whole Plant Corn Silage*. 22(4). https://doi.org/10.3390/s22041596

Rogers, J., & Révész, A. (2019). Experimental and quasi-experimental designs. In J. McKinley, & H. Rose, *The Routledge Handbook of Research Methods in Applied Linguistics* (S. 133-143). London: Routledge.

Sim, J., & Waterfield, J. (2019). Focus group methodology: some ethical challenges. *Quality & Quantity*, 3003–3022.

Sjöberg, A. & Hyberg, J. (2023). *Investigation regarding the Performance of YOLOv8 in Pedestrian Detection*. https://www.diva-portal.org/smash/get/diva2:1778368/FULLTEXT01.pdf

Taherdoost, H. (2019). What Is the Best Response Scale for Survey and Questionnaire Design; Review of Different Lengths of Rating Scale / Attitude Scale / Likert Scale. *International Journal of Academic Research in Management (IJARM)*, 1-12.

Taherdoost, H. (2022). What are Different Research Approaches? Comprehensive Review of Qualitative, Quantitative, and Mixed Method Research, Their Applications, Types, and Limitations. *Journal of Management Science & Engineering Research*, 53-63.

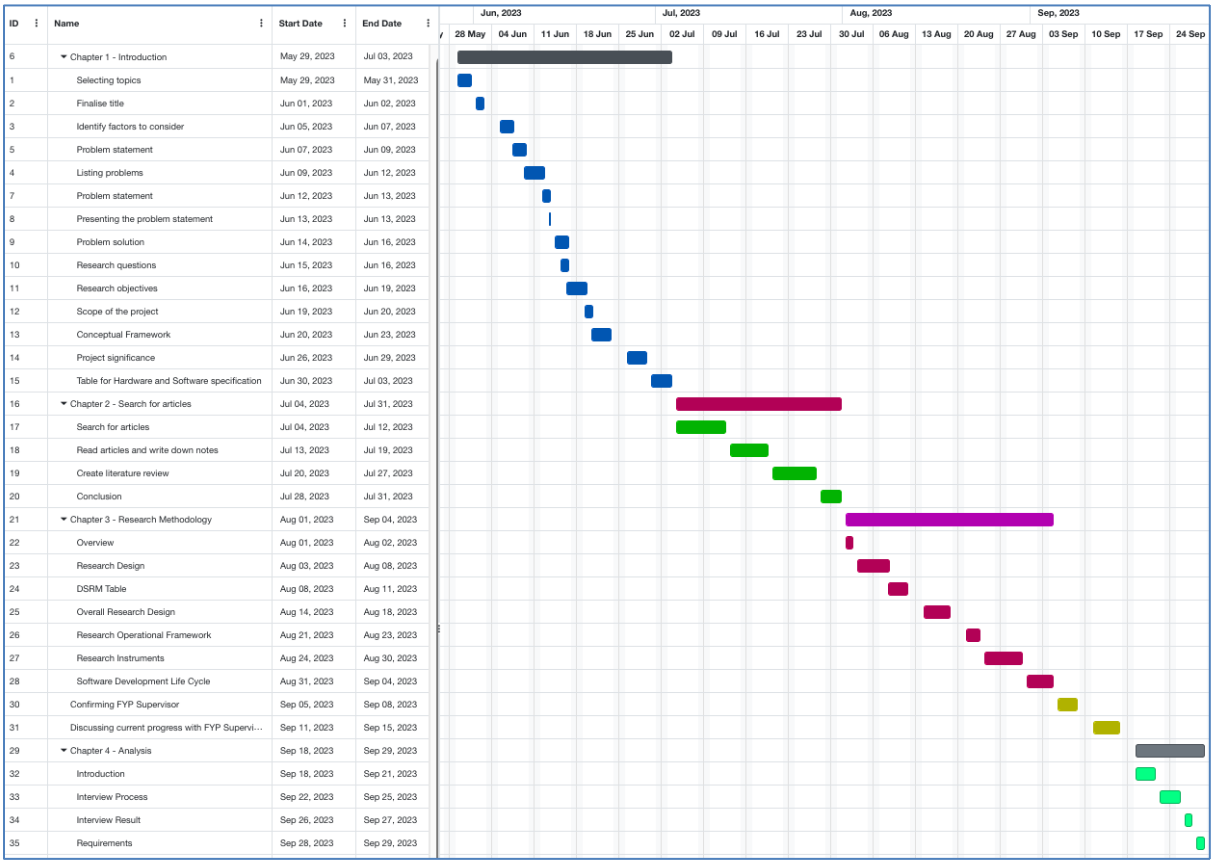
Wahid, A. A. (2020). Analisis Metode Waterfall Untuk Pengembangan Sistem Informasi. *Jurnal Ilmu-ilmu Informatika dan Manajemen STMIK*, 1-5.

Wang, C.-Y., Yeh, I-Hau., & Liao, H.-Y. (2024). *YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information*. https://arxiv.org/pdf/2402.13616.pdf

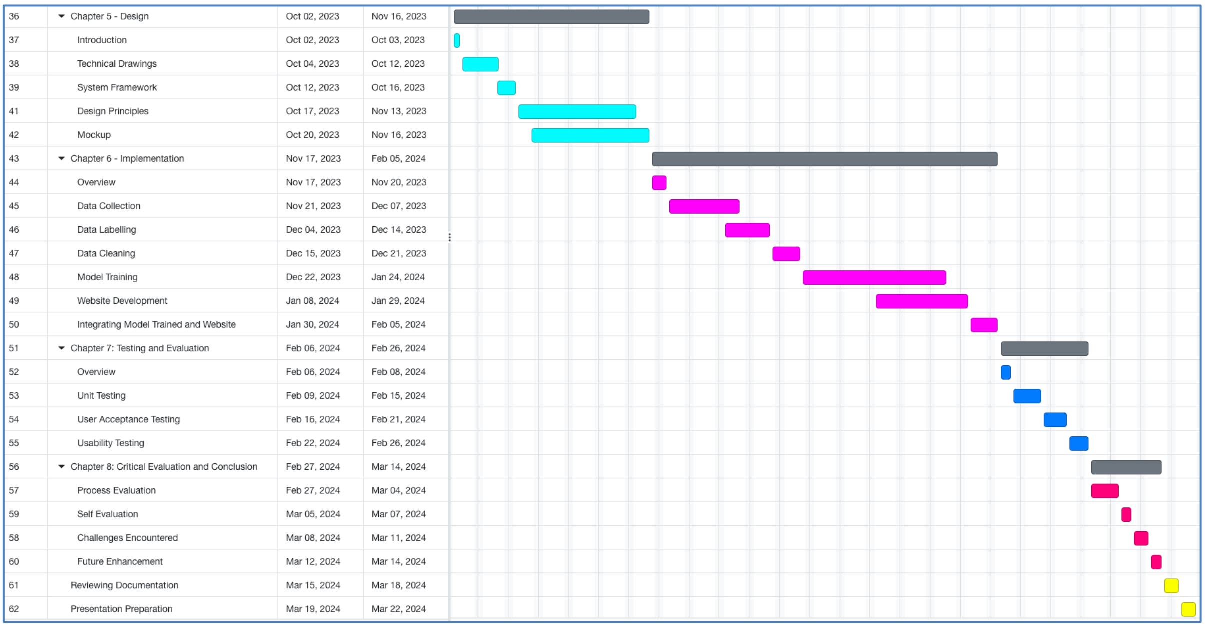
Xie, X., Han, X., Liao, Q., & Shi, G. (2017). Visualization and Pruning of SSD with the base network VGG16. *Proceedings of the 2017 International Conference on Deep Learning Technologies*. https://doi.org/10.1145/3094243.3094262

10. Appendix

10.1 Appendix A: Gantt Chart



ID	Name	Start Date	End Date
6	Chapter 1 - Introduction	May 29, 2023	Jul 03, 2023
1	Selecting topics	May 29, 2023	May 31, 2023
2	Finalise title	Jun 01, 2023	Jun 02, 2023
3	Identify factors to consider	Jun 05, 2023	Jun 07, 2023
5	Problem statement	Jun 07, 2023	Jun 09, 2023
4	Listing problems	Jun 09, 2023	Jun 12, 2023
7	Problem statement	Jun 12, 2023	Jun 13, 2023
8	Presenting the problem statement	Jun 13, 2023	Jun 13, 2023
9	Problem solution	Jun 14, 2023	Jun 16, 2023
10	Research questions	Jun 15, 2023	Jun 16, 2023
11	Research objectives	Jun 16, 2023	Jun 19, 2023
12	Scope of the project	Jun 19, 2023	Jun 20, 2023
13	Conceptual Framework	Jun 20, 2023	Jun 23, 2023
14	Project significance	Jun 26, 2023	Jun 29, 2023
15	Table for Hardware and Software specification	Jun 30, 2023	Jul 03, 2023
16	Chapter 2 - Search for articles	Jul 04, 2023	Jul 31, 2023
17	Search for articles	Jul 04, 2023	Jul 12, 2023
18	Read articles and write down notes	Jul 13, 2023	Jul 19, 2023
19	Create literature review	Jul 20, 2023	Jul 27, 2023
20	Conclusion	Jul 28, 2023	Jul 31, 2023
21	Chapter 3 - Research Methodology	Aug 01, 2023	Sep 04, 2023
22	Overview	Aug 01, 2023	Aug 02, 2023
23	Research Design	Aug 03, 2023	Aug 08, 2023
24	DSRM Table	Aug 08, 2023	Aug 11, 2023
25	Overall Research Design	Aug 14, 2023	Aug 18, 2023
26	Research Operational Framework	Aug 21, 2023	Aug 23, 2023
27	Research Instruments	Aug 24, 2023	Aug 30, 2023
28	Software Development Life Cycle	Aug 31, 2023	Sep 04, 2023
30	Confirming FYP Supervisor	Sep 05, 2023	Sep 08, 2023
31	Discussing current progress with FYP Supervi...	Sep 11, 2023	Sep 15, 2023
29	Chapter 4 - Analysis	Sep 18, 2023	Sep 29, 2023
32	Introduction	Sep 18, 2023	Sep 21, 2023
33	Interview Process	Sep 22, 2023	Sep 25, 2023
34	Interview Result	Sep 26, 2023	Sep 27, 2023
35	Requirements	Sep 28, 2023	Sep 29, 2023



36	Chapter 5 - Design	Oct 02, 2023	Nov 16, 2023
37	Introduction	Oct 02, 2023	Oct 03, 2023
38	Technical Drawings	Oct 04, 2023	Oct 12, 2023
39	System Framework	Oct 12, 2023	Oct 16, 2023
41	Design Principles	Oct 17, 2023	Nov 13, 2023
42	Mockup	Oct 20, 2023	Nov 16, 2023
43	Chapter 6 - Implementation	Nov 17, 2023	Feb 05, 2024
44	Overview	Nov 17, 2023	Nov 20, 2023
45	Data Collection	Nov 21, 2023	Dec 07, 2023
46	Data Labelling	Dec 04, 2023	Dec 14, 2023
47	Data Cleaning	Dec 15, 2023	Dec 21, 2023
48	Model Training	Dec 22, 2023	Jan 24, 2024
49	Website Development	Jan 08, 2024	Jan 29, 2024
50	Integrating Model Trained and Website	Jan 30, 2024	Feb 05, 2024
51	Chapter 7: Testing and Evaluation	Feb 06, 2024	Feb 26, 2024
52	Overview	Feb 06, 2024	Feb 08, 2024
53	Unit Testing	Feb 09, 2024	Feb 15, 2024
54	User Acceptance Testing	Feb 16, 2024	Feb 21, 2024
55	Usability Testing	Feb 22, 2024	Feb 26, 2024
56	Chapter 8: Critical Evaluation and Conclusion	Feb 27, 2024	Mar 14, 2024
57	Process Evaluation	Feb 27, 2024	Mar 04, 2024
59	Self Evaluation	Mar 05, 2024	Mar 07, 2024
58	Challenges Encountered	Mar 08, 2024	Mar 11, 2024
60	Future Enhancement	Mar 12, 2024	Mar 14, 2024
61	Reviewing Documentation	Mar 15, 2024	Mar 18, 2024
62	Presentation Preparation	Mar 19, 2024	Mar 22, 2024

10.2 Appendix B: Logbook

PART OF THE UNIVERSITY
OF WOLLONGONG AUSTRALIA
GLOBAL NETWORK

Department of Computing

Notes on use of the project logbook

1. The purpose of the Project Logbook to document these meetings and therefore build up a record of the student's progress throughout the project.
2. The student should prepare for the meetings by deciding which questions he or she needs to ask the lecturer and what progress has been made since the last meeting (if applicable) and noting these in the relevant sections of the sheet, effectively forming an agenda for the meeting.
3. Log sheets are compulsory assessment criteria for **Final Year Project**. Students who fail to meet the requirements of log sheets will not be allowed to submit **Final Year Project Report**.

Project Title	:	AR Enabled Assembly Guidance System
Name	:	Nicholas Dylan
ID	:	0133646
Email	:	ndylan76@gmail.com
Mobile Number	:	+601111385378
Programme/ Specialization	:	Bachelor of Computer Science
Main Supervisor	:	Bridget Merliza
Email	:	bridget.p@uow.edu.my
Co-Supervisor (If applicable)	:	-

Week / Date	Task: Review of progress made during FYP 1	
26 January 2024		
	Student Meeting Minute/Achievements/Activities)	The meeting commenced with a review of the progress made thus far. I then engaged in a brief discussion regarding the initial development phase.
	Supervisor (Suggestion & Comments)	The supervisor provided valuable feedback, emphasizing the need to rectify errors in the documentation. Additionally, a suggestion was made to conduct further research in order to improve the development process.
	Next Meeting Plan	Review back on what have been researched to Ms. Bridget Merliza
	Supervisor's Signature/Date	

Week / Date	Task: Discussing the founding of my initial research	
2 February 2024		
	Student Meeting Minute/Achievements/Activities)	The meeting commenced with a review on what my research found. On top of that, I also showcased an object detection model which was trained to detect a speaker. A speaker was chosen due to the unavailability of the FRL Unit.
	Supervisor (Suggestion & Comments)	Ms. Bridget Merliza suggested to research more on how to integrate the object detection model with the Web-based system, and whether it is possible to use Django Framework.
	Next Meeting Plan	Showcase the Web-based system to Ms. Bridget Merliza. Due to Chinese New Year, it is hoped that by the next meeting a lot of progress can be made.
	Supervisor's Signature/Date	

Week / Date	Task: Showcasing the improved Web-based System	
28 February 2024	**Student Meeting Minute/Achievements/Activities)**	After some brief discussion regarding the whole development process, I showcased the improved Web-based System. It has key features such as login and password reset. The website was using Django Framework. I used OpenCV to access the laptop's camera and used the trained model from the previous meeting in order to run model inference.
	Supervisor (Suggestion & Comments)	Ms. Bridget Merliza urged me to get access to the FRL Unit as soon as possible as data labelling may consume a lot of time. Also, she suggested me to start the documentation process.
	Next Meeting Plan	To showcase a model that can recognize the FRL Unit.
	Supervisor's Signature/Date	

Week / Date	Task: Showcasing the object detection and segmentation model	
13 March 2024	**Student Meeting Minute/Achievements/Activities)**	Rather than only object detection, the model was also trained to be able to segment components. Today, I showcased the new and improved model which can detect all components that are required for the 10 steps of assembling an FRL Unit. I also asked for feedback on my Chapter 6 documentation.
	Supervisor (Suggestion & Comments)	I should fix some bugs on the website that Ms. Bridget Merliza found. On top of it, I should fix some mistakes I made in the Chapter 6 Documentation.
	Next Meeting Plan	Show a finished website and also finished the testing.
	Supervisor's Signature/Date	

Week / Date	Task: Reviewing the final product and documentation	
18 March 2024		
	Student Meeting Minute/Achievements/Activities)	Ms. Bridget Merliza tested my final system. Afterwards, she checked my documentation and proceeded to ask me to fix some of the mistakes which I made in Chapter 7.
	Supervisor (Suggestion & Comments)	Improve the testing documentation and redo the testing that needs to be repeated.
	Next Meeting Plan	Finalizing the documentation
	Supervisor's Signature/Date	

Week / Date	Task: Final Meeting	
20 March 2024		
	Student Meeting Minute/Achievements/Activities)	Reviewing back all of the documentation and finalizing Chapter 8.
	Supervisor (Suggestion & Comments)	Fix some of the grammar mistakes and also to caption all images and tables.
	Next Meeting Plan	Not applicable
	Supervisor's Signature/Date	

10.3 Appendix C: Turnitin Report

1

ORIGINALITY REPORT

6%	5%	2%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	eprints.utar.edu.my Internet Source	1%
2	www.coursehero.com Internet Source	<1%
3	www.researchgate.net Internet Source	<1%
4	rufus31415.github.io Internet Source	<1%
5	Hartson, Rex, and Partha S. Pyla. "UX Design Guidelines", The UX Book, 2012. Publication	<1%
6	www.trentonsocial.com Internet Source	<1%
7	dspace.daffodilvarsity.edu.bd:8080 Internet Source	<1%
8	www.cse.unr.edu Internet Source	<1%
9	www.mdpi.com Internet Source	<1%

10	cursoslibres.url.edu.gt Internet Source	<1%
11	5dok.org Internet Source	<1%
12	Christoffer Bøgelund Rasmussen, Kristian Kirk, Thomas B. Moeslund. "The Challenge of Data Annotation in Deep Learning—A Case Study on Whole Plant Corn Silage", Sensors, 2022 Publication	<1%
13	himachalservices.nic.in Internet Source	<1%
14	www.radarmasters.uct.ac.za Internet Source	<1%
15	completeload.mystrikingly.com Internet Source	<1%
16	mafiadoc.com Internet Source	<1%
17	scholar.lib.vt.edu Internet Source	<1%
18	Rex Hartson, Pardha Pyla. "UX Design Guidelines", Elsevier BV, 2019 Publication	<1%
19	dspace.lboro.ac.uk Internet Source	<1%

20	12jri.service-finder.eu Internet Source	<1 %
21	en.wikipedia.org Internet Source	<1 %
22	hdl.handle.net Internet Source	<1 %
23	m.moam.info Internet Source	<1 %
24	deci.ai Internet Source	<1 %
25	eprints.utem.edu.my Internet Source	<1 %
26	www.classcentral.com Internet Source	<1 %
27	www.elevatus.io Internet Source	<1 %
28	"Augmented Reality, Virtual Reality, and Computer Graphics", Springer Science and Business Media LLC, 2019 Publication	<1 %
29	www.riskreductionafrica.org Internet Source	<1 %
30	www.thamesvalleyvision.co.uk Internet Source	<1 %

31	Marko Letic, Kosa Nenadic, Lazar Nikolic. "Real-time map projection in virtual reality using WebVR", 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2018 Publication	<1 %
32	Zhaoqian Wu, Yongjie Yao, Jinglun Liang, Fei Jiang, Shaotao Chen, Shaohui Zhang, Xiaohui Yan. "Digital Twin-Driven 3-D Position Information Mutuality and Positioning Error Compensation for Robotic Arm", IEEE Sensors Journal, 2023 Publication	<1 %
33	spectrum.library.concordia.ca Internet Source	<1 %
34	Mahendra Awale, Daniel Probst, Jean-Louis Reymond. "WebMolCS: A Web-Based Interface for Visualizing Molecules in Three-Dimensional Chemical Spaces", Journal of Chemical Information and Modeling, 2017 Publication	<1 %
35	online-journals.org Internet Source	<1 %
36	repo.itera.ac.id Internet Source	<1 %

wlv.openrepository.com

37	Internet Source	<1%
38	Carl Lawrence, Tuure Tuunanen, Michael D. Myers. "Chapter 7 Extending Design Science Research Methodology for a Multicultural World", Springer Science and Business Media LLC, 2010 Publication	<1%
39	eprints.dkit.ie Internet Source	<1%
40	eprints.lincoln.ac.uk Internet Source	<1%
41	www.thetechedvocate.org Internet Source	<1%
42	www.toshiba-storage.com Internet Source	<1%
43	repository.atmaluhur.ac.id Internet Source	<1%
44	testsigma.com Internet Source	<1%
45	bikehike.org Internet Source	<1%
46	cdma.ap.gov.in Internet Source	<1%
	immersive-web.github.io	

47	Internet Source	<1%
48	opus.bath.ac.uk Internet Source	<1%
49	www.gsmarena.com Internet Source	<1%
50	Sonia Budz. "Research Regarding the Influence of Social Media on the Quality of Services in the Sharing Economy", Springer Science and Business Media LLC, 2023 Publication	<1%
51	digital.library.adelaide.edu.au Internet Source	<1%
52	eprints-training.geomar.de Internet Source	<1%
53	eprints.mdx.ac.uk Internet Source	<1%
54	repository.ub.ac.id Internet Source	<1%
55	studentsrepo.um.edu.my Internet Source	<1%
56	"ICOA 2023 Conference Proceedings", 2023 9th International Conference on Optimization and Applications (ICOA), 2023 Publication	<1%

57	"Microgrid Technologies", Wiley, 2021 Publication	<1 %
58	Aymeric Koenig, Benjamin Allaert, Emmanuel Renaux. "NEURAL-UML: Intelligent Recognition System of Structural Elements in UML Class Diagram", 2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C), 2023 Publication	<1 %
59	Blumenfeld, Hal. "Neuroanatomy through Clinical Cases", Oxford University Press Publication	<1 %
60	Laubacher, Matthew; Navarre-Jackson, Layana C.; Williams, Amanda N.; Dillard, Dana; Pate, Jennifer;Choi, Kunsoo Paul. "An Introduction to Social Science: Individuals, Society, and Culture", UAGC, 2022 Publication	<1 %
61	eprints.qut.edu.au Internet Source	<1 %
62	escholarship.org Internet Source	<1 %
63	pdfcoffee.com Internet Source	<1 %
64	research.brighton.ac.uk Internet Source	

		<1%
65	rosap.ntl.bts.gov Internet Source	<1%
66	vdocuments.mx Internet Source	<1%
67	www.vtpi.org Internet Source	<1%
68	Ton Duc Thang University Publication	<1%
69	repository.dinamika.ac.id Internet Source	<1%
70	José Andery Carneiro. "Enhanced tooth segmentation algorithm for panoramic radiographs", Universidade de São Paulo. Agência de Bibliotecas e Coleções Digitais, 2023 Publication	<1%

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		