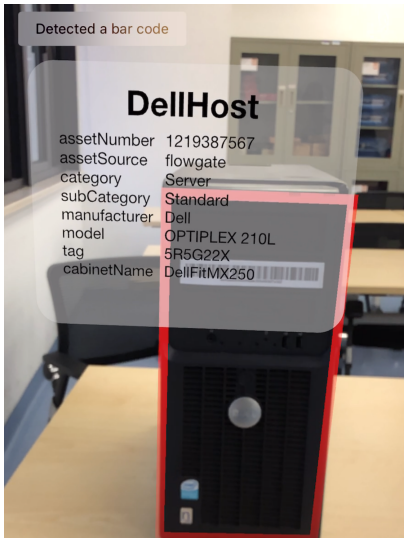
UM-SJTU Joint Institute
Major Design Experience
(Ve450 FA2020)

VMware#1 - AR in Data Centers
Design Review 3 Report

Sponsor: Gavin Lu, VMware
Mentor: Yixing Jia, VMware
Instructor: Mingjian Li, UMJI



An AR app for aiding data center maintenance

Group 21

Shuyi Zhou	Email: zhoushuyi_sue@sjtu.edu.cn
Chenyun Tao	Email: tcy1999@sjtu.edu.cn
Liying Han	Email: hanliying_17@sjtu.edu.cn
Yaxin Chen	Email: leepace666666@sjtu.edu.cn
Jinglei Xie	Email: xie_jinglei@sjtu.edu.cn

November 23, 2020

Contents

1 Problem Description and Introduction

We are Group 21 of Capstone Design Fall 2020 from University of Michigan - Shanghai Jiao Tong University Joint Institute. The title of our project is **AR in Data Center**, sponsored by VMware, Inc. As the name implies, the project is about applying augmented reality (AR) to work related with data centers. In this report, we will provide necessary background and introduction to the project, engineering specifications, concept generation, concept selection process, selected concept description, project plan and analysis of potential problems.

For this section, the background of our project will be presented, the problems and needs will be formally defined, the strategy will be generally discussed, and the expected outcome will be stated.

1.1 Background Overview

A Data Center (DC) is a major infrastructure and facility which contains a high quantity of servers and computers to provide internet services for many companies in the world (such as Google, Amazon, etc.) [1]. It usually requires high-level reliability and availability. In many DCs, various kinds of critical components keep running all time, and any interruption in its services may cause major problems. Thus, we need both regular and emergent DC maintenance to make sure that computers, servers, cooling systems, and many other critical devices are working properly. If any error happens, fast recovery should be conducted. In addition, regular auditing work is also necessary for DCs to ensure that every component or device is properly placed and used, which is labor consuming as well.

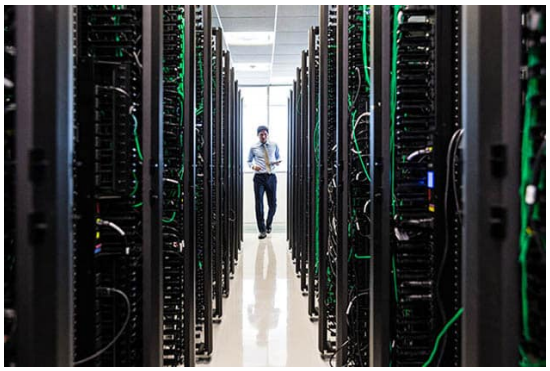


Figure 1: A typical data center (DC) (cisco.com).

To aid DC maintenance and audits, people in the industry have already developed some kinds of computer systems. For example, Data Center Infrastructure Management (DCIM) can monitor, check, control and manage the energy usage of IT devices in a DC [1]; Configuration Management Data Base (CMDB) contains all relevant information about the components of the information system and the relationships between those components [2]. Some more intelligent systems can even detect component failures and provide possible repairing strategies. Still, in many situations, we need human workers to fix the problem manually inside DCs, as well as to conduct on-site audits.

1.2 Problems and Needs

As mentioned in the previous section, we do have various computer systems to contain necessary information for on-site maintenance and auditing work. However, these systems are usually not integrated together, which means if we want to access different kinds of information (eg. data from environment sensors, power usage information, device models and parameters, etc.), we have to visit different systems to fetch the data, which could be a complex work.

Another problem is that on-site workers in DCs usually do not have user-friendly enough instructions and information access. Sometimes they have to face very tedious literal descriptions in manuals or documents, and sometimes they need to search information manually in different data systems. That could greatly reduce the efficiency of maintenance and audit work. Thus the two main problems for many current DC maintenance and audit work can be summarized as

- Lack of integrated information system;
- Lack of user-friendly instructions and information access.

To solve the above two problems, what we need are

- An integrated system that involves all the information together;
- A more user-friendly tool to aid and instruct on-site maintenance and audit work.

With the above improvements, the efficiency of on-site maintenance and audit work can be greatly improved.

1.3 General Strategy

To realize the needs, we can divide our project into two parts: the back end part and the front end part. For the back end part, we can establish an integrated data system to contain and provide all necessary information of the data center. For the front end part, we need to develop a user interface to display the information in a vivid way. More details are given in the following sections.

1.3.1 Back End Implementation

For the back end of the project, we already have different systems to contain different kinds of data, such as DCIM and CMDB mentioned previously. Also, VMware has developed a software system called *Flowgate* [3] to integrate the existing systems together and provide convenient access to all types of data (Figure 2). Thus we can use Flowgate as our back end database, and make use of API to fetch data from it.

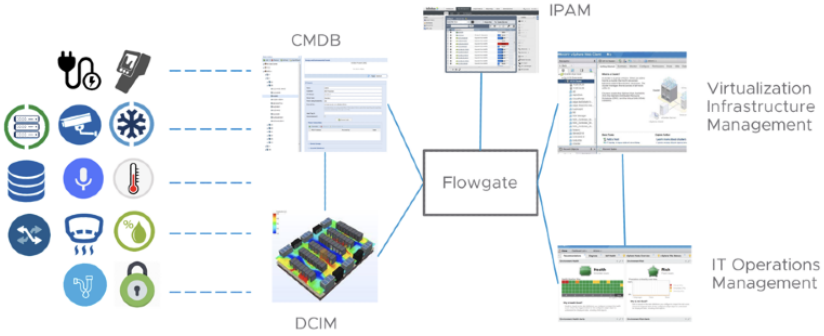


Figure 2: Flowgate: a system to integrate various kinds of data systems for data centers.

1.3.2 Front End Implementation

For the front-end application, we are going to use augmented reality, *AR*, which is a combination of information generated by computers and real-world scenarios. The three basic hardware components of AR are sensors, processors and displays [4], all of which could be contained in a smart phone. For software developers, there are some open source software development kits such as Google ARCore and ARKit. Almost all the AR toolkits support common operating systems like iOS and Android, so we can easily develop apps in these platforms.

An example of the usage of AR is shown in Figure 3, which is a function in Google Map. The App can label information onto real-world scenes to provide visual guide to its user.

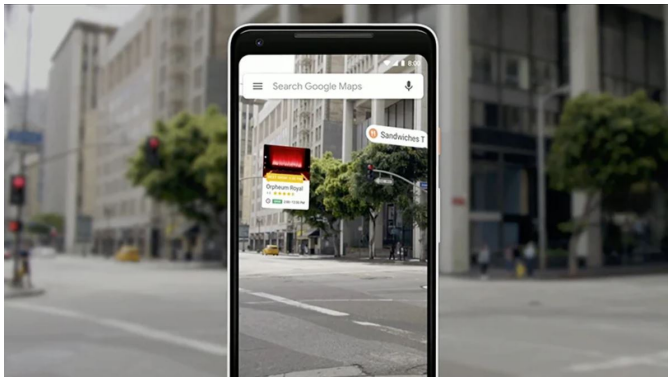


Figure 3: AR in Google Map.

If we use AR to aid DC maintenance and audit work, we can obtain information visually and easily, avoiding unnecessary visits to different data systems. Also, AR can provide instructions in a convenient and vivid way, which is more user-friendly for the on-site technicians.

1.4 Expected Outcome

The product of our project is expected to be an AR App which can obtain all kinds of information related to a specific data center from the back end database (Flowgate), and display necessary information and instructions in a real-time manner (using AR) to aid data center maintenance and audit work. More intuitively, the product could be similar to Figure 4.

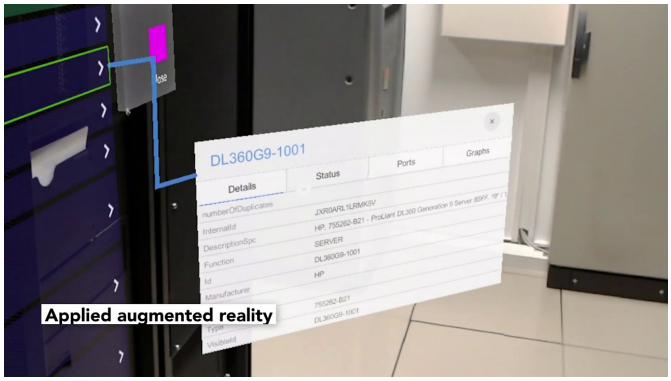


Figure 4: An example of AR software used in data center [1].

Also, the App is expected to have quick reaction time, relatively low power consumption and high compatibility to different types or versions of operating systems. More details about design specifications can be found in section 3.

2 Literature Reviews and Benchmarks

In order to have a better understanding of how to solve the problem, we have conducted several literature reviews on related technology. This can also help us meet customer requirements, and find out our novel solution on this project.

2.1 Literature Search Methods

Search by Keywords Augmented Reality, Data center maintenance, Flowgate, Data Center Visualization, Collaborative AR, AR in Data Center

Search on Platforms Google Scholar, SJTU Digital Library, U-Mich Digital Library, IEEE Digital Library, Github

2.2 Competitive and Related Products

2.2.1 Augmented Assembly using a Mobile Phone [5]

This is a mobile phone based augmented reality (AR) assembly system. It is based on a client-server architecture, where complex model information is stored on a PC, and a mobile phone with the camera is used as a client device to access this information. With this system, users are able to see an AR view that provides step by step guidance for a real world assembly task.

[1] www.youtube.com/watch?v=1Pe028PjQhs

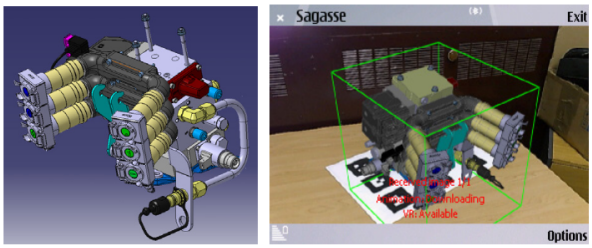


Figure 5: An example of CAD model and AR view of a hydraulic tractor.

This system is made up of two parts: a PC server software and a visualization/controller client on a smart phone. The phone client software takes images of the real assembly site, and sends them to the PC server. After the PC server receives the image, it will compute the AR model that matches the image, and send it back to the phone. The user then sees the augmented views of real world assembly tasks on the phone. The transmission of data between PC and mobile phone is realized by WLAN or Bluetooth.

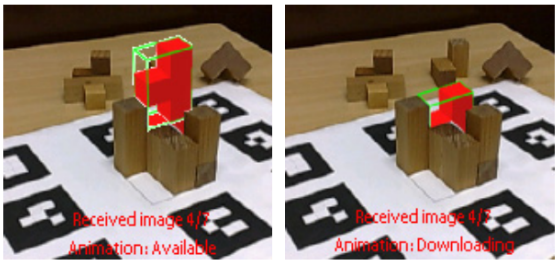


Figure 6: Augmented assembly of a 3D puzzle on a mobile phone.

To be more illustrative and intuitive, the augmented view is provided as an animated image, so that the step by step guide for assembly can be provided.

Advantages This system does not need to use many computing resources for AR on mobile phones, as the computation part is performed on the PC server.

Disadvantages The data transmission time between the phone and PC server is long. The average time to send and receive an image is 19.13 seconds over Bluetooth and 3.44 seconds over WLAN. Also, this system does not involve database of real-time information, so it is hard to be implemented on database.

2.2.2 Mobile augmented reality in the data center [6]

This is the product invented by IBM that can be used to help manage data base. It enables system administrators to easily identify various hardware assets in data center, and provides them with an additional tool to interact with those assets. Users scan QR code on the rack to fetch data from the Tivoli MEO server, which stores information of ID and location of each asset on the rack. AR on the phone marks the outline of each asset by showing a red frame of it.

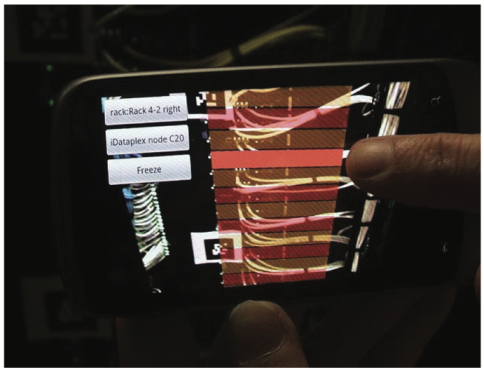


Figure 7: Visual overlay of data center assets on top of assets in an IBM iDataPlex rack.

Aside from information of ID and location, there is also a "freeze" button. Users are able to point the device at an area of interest and freeze the video capture of camera, so that they can check the device in a comfortable position. They do not need to hold the device up at an awkward angle while they are interacting with the application.

Advantages As users can freeze the camera capture, this product is user-friendly. Also, the AR visualizes the outline of each asset by red boxes.

Disadvantages There is limited information stored in Tivoli MEO server: only location, name of each machine and position where a new asset can be added. It does not check and use over physical information other than QR code, like signal lights.

3 Customer Requirements (CR) and Engineering Specifications (ES)

The benchmarks indicate that the basic requirement of our customers is to display real-time information of data center with AR, which could be divided into four detailed customer requirements. By conducting literature search and considering the constraints in reality, we further convert the requirements into engineering specifications.

3.1 Details of CR and ES

3.1.1 Short Reaction Time and the Corresponding ES

CR: Short Reaction Time

ES: Object Localization Time, Barcode Identification Time, Database Query Time, AR Image Generation Time

A research on user psychology shows that a user can get distracted in 1 second, so short reaction time is required in our project [7]. The execution steps of our application include the localization of objects, barcode identification, database query, and the AR image generation. Therefore, we need to control the time of these tasks to fulfill the requirement of short reaction time. Based on literature search, our APP should be able to localize the barcode in 0.5s [8], identify the barcode in less than 0.05s [9], achieve an O(log(n)) complexity in the time of

database query, which depends on the data size, and take less than 0.1s to generate the AR image [10].

3.1.2 Information Correctness and the Corresponding ES

CR: Information Correctness

ES: Object Localization Accuracy, Data Retrieval Accuracy

Another important feature of our product is information correctness. In our application, the tasks related to achieving information are barcode localization and data retrieval. According to state-of-art technologies, we need to realize a 90% accuracy on localizing the barcode [11] and more than 99% accuracy on retrieving data [12].

3.1.3 Comfortable Display and the Corresponding ES

CR: Comfortable Display

ES: Smooth Display, Appropriate Temperature

User experience is also essential for our project, and thus we would like to ensure a comfortable display, which is achieved by keeping a smooth display and an appropriate temperature of the devices in our application.

To realize a smooth display, we could set the frame rate above 15 frames per second [13]. The temperature of the device can keep increasing when AR is on, and thus we need to control the sensible temperature below 40 °C [14][15].

3.1.4 Portable Device and the Corresponding ES

CR: Portable Device

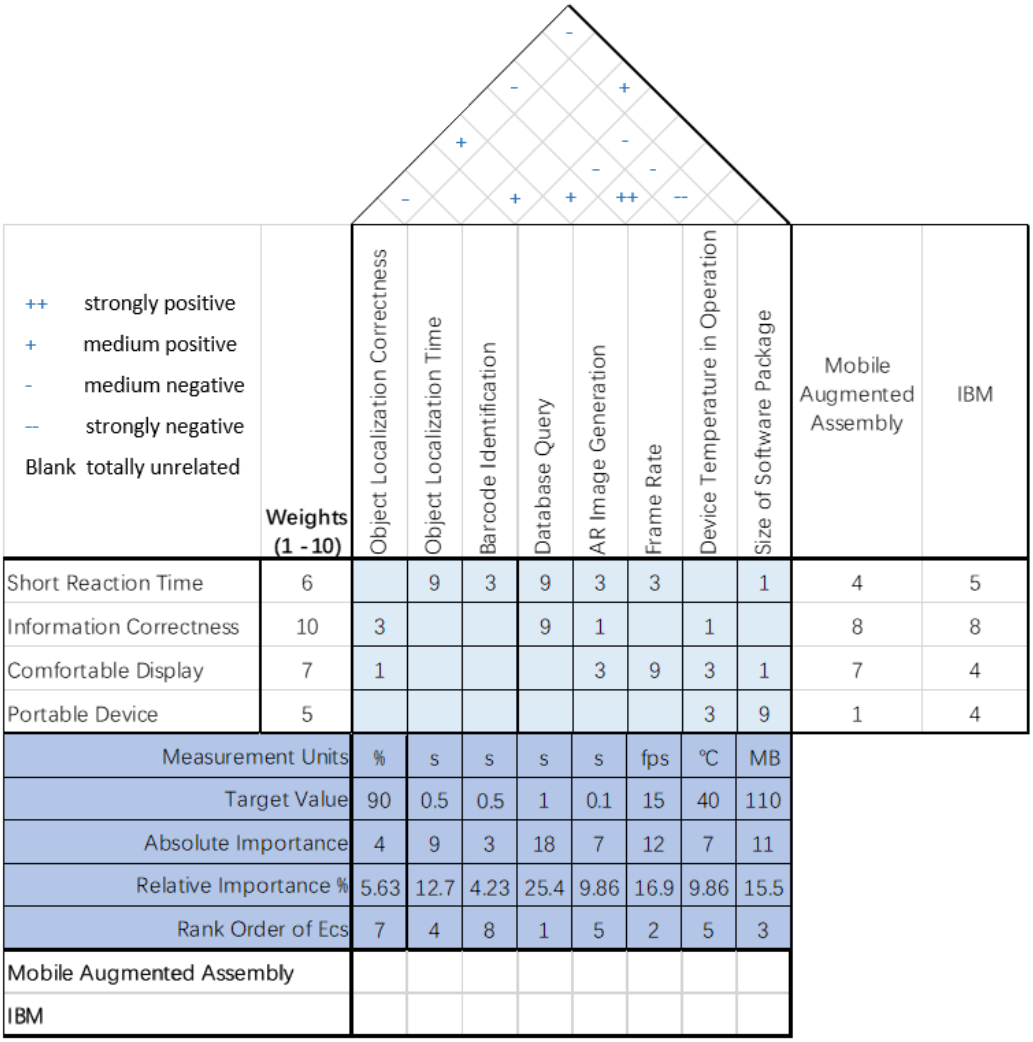
ES: Platform, Small Package Size

AR application can be loaded in portable devices like smart phones or pads. Nowadays, the operation system of most of the smart phones and pads on the market is either Andriod released by Google, or iOS released by Apple. Both Google and Apple launch AR toolkits. The toolkits are available on Android 7.0 or above and iOS 11.0 or above, which limits the platforms of the portable devices where our application could run [16][17].

Considering the storage capacity of portable devices, we also need to keep a small package size, with less than 110MB for Android and 940MB for iOS [18][19].

3.2 Quality Function Deployment

Based on the customer requirements and engineering specifications described above, we can evaluate our benchmarks and develop our house of quality (HOQ), which is shown in Figure 8.



++ strongly positive + medium positive - medium negative -- strongly negative Blank totally unrelated	Weights (1 - 10)	Object Localization Correctness	Object Localization Time	Barcode Identification	Database Query	AR Image Generation	Frame Rate	Device Temperature in Operation	Size of Software Package	Mobile Augmented Assembly	IBM
Short Reaction Time	6		9	3	9	3	3		1	4	5
Information Correctness	10	3			9	1		1		8	8
Comfortable Display	7	1				3	9	3	1	7	4
Portable Device	5							3	9	1	4
Measurement Units		%	s	s	s	s	fps	℃	MB		
Target Value		90	0.5	0.5	1	0.1	15	40	110		
Absolute Importance		4	9	3	18	7	12	7	11		
Relative Importance %		5.63	12.7	4.23	25.4	9.86	16.9	9.86	15.5		
Rank Order of Ecs		7	4	8	1	5	2	5	3		
Mobile Augmented Assembly											
IBM											

Figure 8: House of Quality.

3.2.1 Customer Requirements and Weights

Our customers are companies that own data centers. By doing customer survey and studying our benchmarks, we conclude the customer requirements and their corresponding weights, and list them in Figure 9, which is part of our HOQ. The weights range from 0 to 10, and a larger weight means a bigger importance. The most crucial requirement of our product is information correctness, since our product will be of no use if the information it provides is incorrect. The other three requirements are related to user experience. Since portable device is the easiest to be satisfied, it has the lowest weight.

Short Reaction Time	6
Information Correctness	10
Comfortable Display	7
Portable Device	5

Figure 9: Customer Requirements and Weights.

3.2.2 Benchmark Competitions against CR

The information correctness of both Mobile Augmented Assembly product and the IBM product is high, as the correctness of data is the most important factor to solve database maintenance problem. Mobile Augmented Assembly product is not portable, as it needs a nearby computer server. The animated way of AR display is good, but the reaction time is not short. IBM product only has one phone client, so it is portable. The AR display and reaction time is fair. See Figure: 10

Mobile Augmented Assembly	IBM
4	5
8	8
7	4
1	4

Figure 10: Benchmark Competitions against CR.

3.2.3 Generate ES and Cross Correlate ES

The details of generating ES from CR are described in Section 4.1. Their cross correlations are shown in Figure 11. A "++" symbol shows there is a strong positive correlation between the two engineering specifications, while a "–" symbol shows there is a strong negative correlation; a "+" or "-" symbol means there is a weak correlation.

The object localization correctness has weak correlations with some specifications. One is object localization time. If we want to achieve a larger localization accuracy, then the time needed to localize it will be increased. Besides, a larger localization accuracy means we need a better localization model, which can result to a larger software package size. The device temperature also has weak correlations with some several specifications. If frame rate is increased or AR image generation time is reduced, then the device temperature will probably increase. The only strong correlation exists between AR image generation time and the frame rate; if AR image generation time is reduced, then the frame rate could be increased to give user a smoother display.

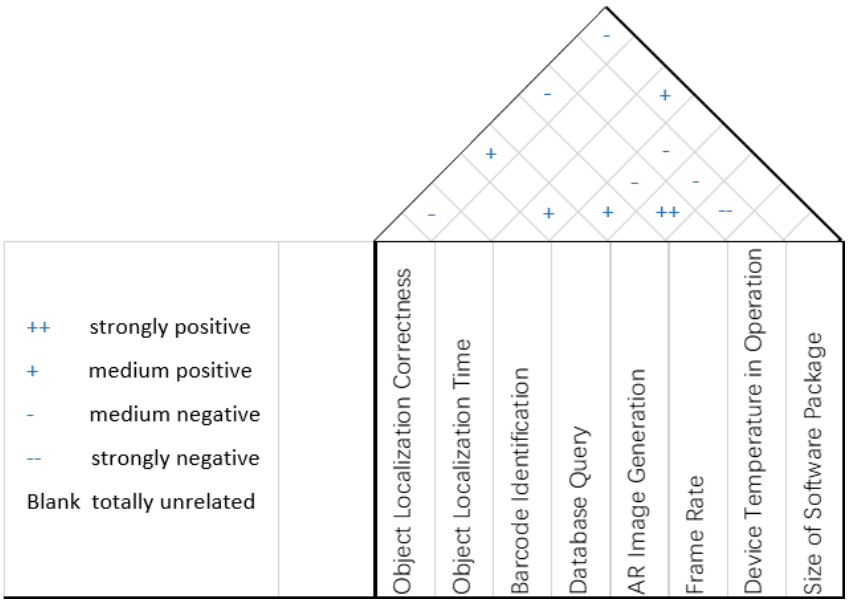


Figure 11: Cross Correlation of ES.

3.2.4 Correlate CR to ES

The correlations between CR and ES are shown in Figure 12. Correlation values include ”9”, ”3” and ”1”, where ”9” means strongly related, ”3” means somewhat related, ”1” means weakly related and empty means unrelated. The requirement of short reaction time is related to the time of a series of operations in our application as well as the frame rate. Among these operations, object localization and database cost more time than the others, which have a larger effect on short reaction time. As for information correctness, the most important step is to fetch the correct information from database. Comfortable display requires a high frame rate. The time for AR image generation and the temperature of the device can also influence the display. Finally, a small software package size is needed for the portable device requirement.

	Object Localization Correctness	Object Localization Time	Barcode Identification	Database Query	AR Image Generation	Frame Rate	Device Temperature in Operation	Size of Software Package
Short Reaction Time		9	3	9	3	3		1
Information Correctness	3			9	1		1	
Comfortable Display	1				3	9	3	1
Portable Device							3	9

Figure 12: Correlation between CR and ES.

3.2.5 Importance Rating

According to the correlations between CR and ES, we calculate the absolute importance as well as the relative importance of our engineering specifications, and then rank them. As shown in Figure 13, the critical-to-quality ES of our product are database query, frame rate and size of software package.

	Object Localization Correctness	Object Localization Time	Barcode Identification	Database Query	AR Image Generation	Frame Rate	Device Temperature in Operation	Size of Software Package
Absolute Importance	4	9	3	18	7	12	7	11
Relative Importance %	5.63	12.7	4.23	25.4	9.86	16.9	9.86	15.5
Rank Order of Ecs	7	4	8	1	5	2	5	3

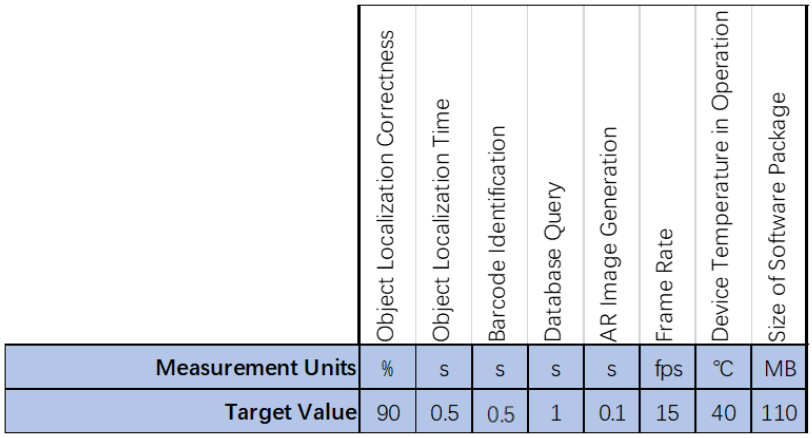
Figure 13: Importance Rating.

3.2.6 Benchmark Competitions Against ES

For both Mobile Augmented Assembly and IBM products, the methods they use are different from our ES. In this way, it is hard to do a direct comparison on object location correctness and time, bar code identification and data query. AR image generation, frame rate, device temperature and the size of software packages are not provided in the literature.

3.2.7 Set Targets for ES

Figure 14 shows the target values and the units for ES. The target values are set mainly according to state-of-the-art technologies and the details of setting the target values for ES are presented in Section 4.1.



	Object Localization Correctness	Object Localization Time	Barcode Identification	Database Query	AR Image Generation	Frame Rate	Device Temperature in Operation	Size of Software Package
Measurement Units	%	s	s	s	s	fps	℃	MB
Target Value	90	0.5	0.5	1	0.1	15	40	110

Figure 14: Targets for ES.

3.3 Summary of Engineering Specifications

Engineering Specifications	Measurement Units	Target Value
Object Localization Correctness	%	90
Data Retrieval Accuracy	%	99
Object Localization Time	s	0.5
Barcode Identification Time	s	0.5
Database Query Time	s	1
AR Image Generation Time	s	0.1
Frame Rate	frames/s	15
Device Temperature in Operation	°C	40
Size of Software Package	MB	110

Table 1: Summary of Engineering Specifications.

4 Concept Generation

Based on customer requirements and engineering specifications, we think about how to implement our project accordingly. We first come up with the concept flowchart of our project. Our application should be able to work as shown in Fig: 15.

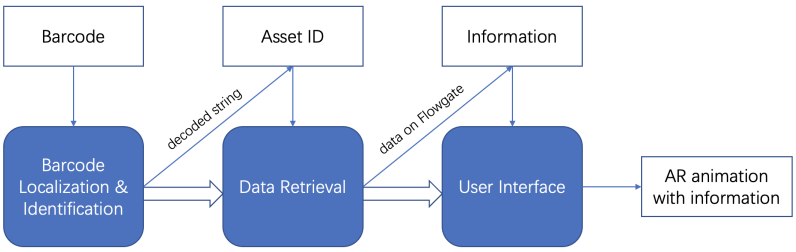


Figure 15: Concept Generation Flowchart.

First, at the beginning of a session, the app need to detect a barcode in **Barcode Localization & Identification** part, which is attached on the device a user wants to check. The app will try to detect barcode from one picture caught by the mobile camera. This picture with barcode is then the input for barcode localization and identification part. In this part, we will use some APIs to detect the location of barcode, and decode the information of the barcode. The output will be a string containing the asset ID information, which can be later used as a token to fetch detailed data of the device from Flowgate server.

Then the decoded string will be used as the input of **Data Retrieval** part. For dthis part, we use the decoded asset ID as the token to fetch data from Flowgate. On Flowgate, we have saved various data for different devices. For example, for each asset token we can get the name of the device, which rack this device belongs to, and the location of this device on that rack, etc. Those data will then be send to the UI interface part, as we need to present such detailed data of the device of interest to the user.

The last step to complete this session is to show the information users want to see by good **User Interface**. We would like to use AR to draw a red box on the device which the user is checking, and display detailed information in text view. We need to consider how to show such text view. After we have completed this part, users can start to maintain the device with the help from our app, which shows some detailed information of the device with AR animation.

In order to generate concepts on application implementation, we break the whole structure of our project into three sub-functions, and use the **morphology analysis** method to help generate various concepts for each sub-function. As is shown in Fig: 16, each part has several concepts generated by brainstorming. For Barcode Detect part, we have considered different APIs, ZXing, ML Kit and Scandit. For Data Retrieval part, we considered to directly retrieve data from Flowgate each time or use a cache to store information. For the User Interface part, we come up with two ways of showing text view, one by 3D AR display, and another by 2D text page display. The detailed brainstorming process will be shown in the next part. After that, we will pick up final concept by concept selection.

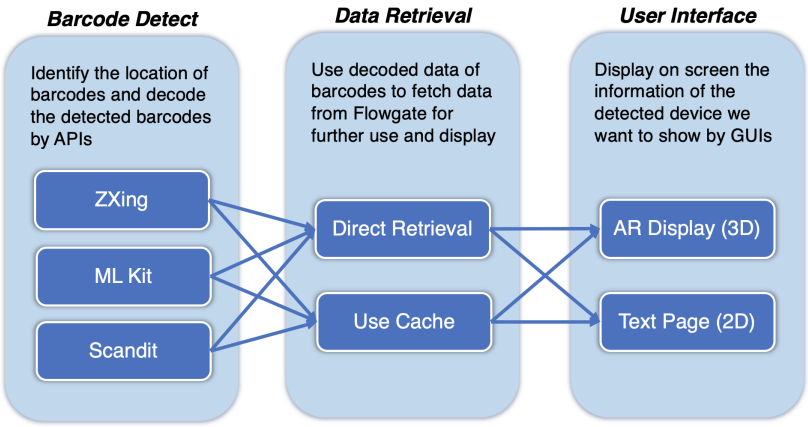


Figure 16: Morphological analysis diagram.

4.1 Barcode Localization & Identification

Barcode Localization & Identification is the fundamental step for us to realize the function of our app. Due to time limit, we decided not to write barcode detection by ourselves, but to use some Software Development Kit (SDK) as an API. Each time we need to detect and decode barcode, we call this API to return the decoded information from the barcode we detected. After research on the internet, we focus on three types of SDK that realize barcode detection and decoding function.

The first one is **ZXing**. This is an open source SDK developed in Java language, found on Github. It can support multiple types of barcode decoding, and can detect and decode one barcode at one time, basicly. This is commonly used by android developers to realize the function of barcode identification in apps.

The second SDK we found applicable is **ML Kit**. This is an open source SDK developed by Google. Barcode detection is one of the function ML Kit can provide. Google has a sound and thorough documentation of guidance for this SDK. It can support various types of barcode decoding by specifications, and can detect and decode more than barcodes in one picture.

The third SDK for bacode identification is **Scandit**. This is a closed source SDK, which aims at high accuracy of barcode detection under various environment. It can also detect and decode more than barcodes at the same time. To use this SDK, you need to register on its company's website and contact them.

4.2 Data Retrieval

Data retrieval is an important part in our app. The two concepts related data retrieval is shown in Fig 17 and Fig 18. One concept is to retrieve data directly every time a bar code is scanned. In this case, we directly send its ID to the remote database and obtain its information through network transmission. Another concept is generated from the concern of short reaction time. Usually, it takes a considerable amount of time to transmit data through network, since it should establish the connection first and then transmit the data one packet by one packet. Thus, we came up with an idea, which utilizes cache and applies least recently used strategy. Cache is commonly used in today's computer, for example, memory management in operating

system and data accessed in cpu. If we use cache to store the recently scanned devices, next time when the same device is scanned, we can directly retrieve the information from cache. In this case, when a query comes, it first checks whether the cache contains the corresponding information and only reaches remote database when miss.

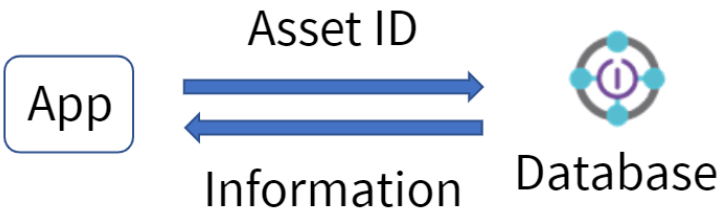


Figure 17: Direct data retrieval.

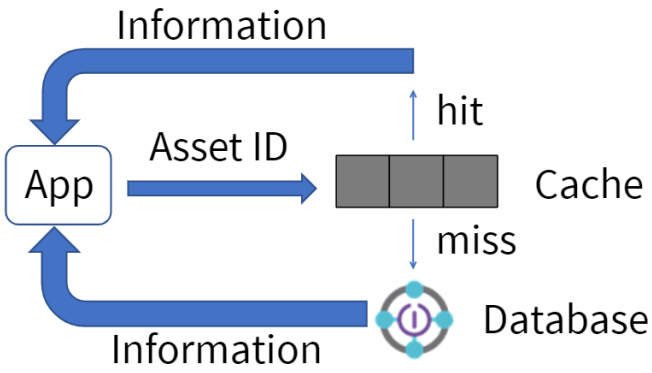


Figure 18: Retrieval with cache.

4.3 User Interface

About User interface, our concern is how to display data. The concepts are generated from the customer requirements of good user experience and the existing products we saw during the literature survey.

The first concept is to place information based on 3-D coordinate in reality world(Fig 20). This concept is mainly generated from the AR products already existing on the markets, namely, AR products will normally use this way. However, we also thought of a display method which may be easier to read: to always place information on the screen based on view controller 2-D coordinate as a normal app(Fig 19).

In either of the two ways, we will mark the cabinet and servers with red box based on the coordinates in reality world.

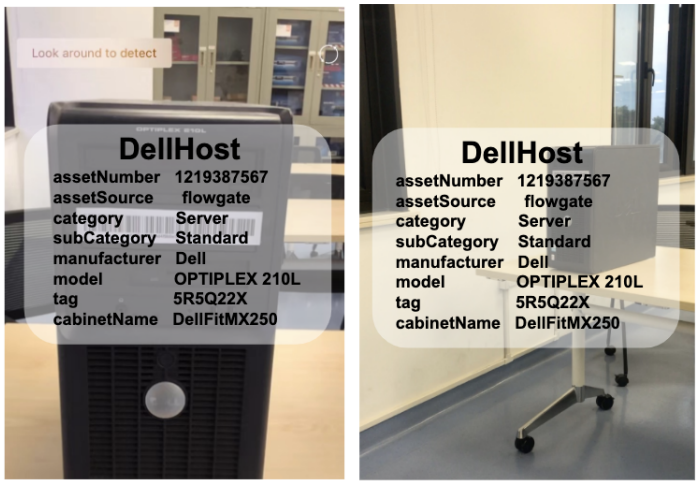


Figure 19: Put information on the screen based on view controller 2-D coordinate

Figure 20: Place information based on 3-D coordinate in reality world

5 Concept Selection Process

5.1 Barcode Localization & Identification

To evaluate the three concepts related to barcode detection and decoding, we formulate the design criteria used to evaluate them and draw a pair-wise comparison matrix in Fig: 21 to decide the weight factors of each criteria. According to the comparison matrix, information correctness turns out to be the most important criteria, which is reasonable because the correctness of data we fetch from Flowgate must be ensured, so we first need to decode the correct barcode in correct way. This will meet our ES of high accuracy on barcode identification.

Criteria	Reaction time	Information correctness	Implement. difficulty	Cost	Row total	Weighting factor
Reaction time	-	0	0	1	1	0.17
Information correctness	1	-	1	1	3	0.5
Implement. difficulty	1	0	-	0	1	0.17
Cost	0	0	1	-	1	0.17
					6	1.00

Figure 21: Pair-wise comparison matrix for barcode localization & identificatio.

All three concepts have advantages and disadvantages. For ZXing SDK, it is good as it is for free and open-source. It mainly has two disadvantages, first is the lack of documentation. Since this not is developed by a company, the developers team do not provide enough guidance for it. We need more time to learn how to integrate ZXing into our app. Also, ZXing SDK is not well encapsulated, so it may cause inconvenience when we integrate it into our app. The second disadvantage is that the time for ZXing to detect and decode a barcode is slow, and the accuracy to decode a barcode is also low.

For ML Kit, this is an open-source SDK developed by Google. It has a thorough and detailed documentation for developers, and is already well-functioning for barcode detection and decoding. This makes the implementation much easier. Although ML Kit may be a little slower than Scandit on the time of barcode identification, it is still acceptable as it has a moderate time that is in accordance with our ES on reaction time.

Last, Scandit SDK performs the best in both time and accuracy for barcode identification. However, this is a close-source SDK, so we need to spend money to buy it. We also need to keep in contact with the company each time we need some modifications of barcode identification function, since we cannot modify the codes by ourselves.

Fig: 22 and the final scoring result reflects that using a ML Kit SDK is more suitable.

Design criterion	Weight factor	Unit	ZXing			ML Kit			Scandit		
			Value	Score	Rating	Value	Score	Rating	Value	Score	Rating
Reaction time	0.17	Exp	Long	5	0.85	Fair	6	1.02	Short	8	1.36
Information correctness	0.5	Exp	Low	5	2.5	Fair	6	3	High	7	3.5
Implement. difficulty	0.17	Exp	High	4	0.68	Low	8	1.36	Low	8	1.36
Cost	0.17	$	0	10	1.7	0	10	1.7	>100	2	0.34
					5.73			7.08			6.56

Figure 22: Weighted decision matrix for barcode Localization & Identification.

5.2 Data Retrieval

To evaluate the two concepts related to data retrieval, we first formulate the criteria used to evaluate them and draw a pair-wise comparison matrix in Fig 23 to decide the weights of criteria. According to the comparison matrix, time requirement and space requirement turn out to be the most important criteria, which is reasonable since the reaction time will influence the CR of user experience and the space usage will influence the ES of our software package size.

Criteria	Time requirement	Space requirement	Implementation difficulty	History record	Row total	Weighting factor
Time requirement	-	0	1	1	2	0.33
Space requirement	1	-	1	0	2	0.33
Implementation difficulty	0	0	-	1	1	0.17
History record	0	1	0	-	1	0.17
					6	1.00

Figure 23: Pair-wise comparison matrix for data retrieval.

Although using cache may help to save the time on network transmission, it has several drawbacks. First, it needs an cache to store the visited information, which costs memory space. Second, it requires extra implementation of maintaining and updating the cache based on the direct retrieval, so it is harder to implement. Third, although it can reduce the retrieval time theoretically, the actual time depends on the hit rate of the cache, which means when the requested device information is not in the cache, it needs extra time to traverse the whole cache. However, maintaining a cache can support the function of history query, and thus the user can check which devices he/she has maintained. The decision matrix is shown in Fig 24 and the final scoring result reflects that using a cache is more suitable.

Criteria	Read Data	Select Data	Go to Data	Implementation	Row total	Weighting factor
Read Data	-	0	1	0	1	0.17
Select Data	1	-	1	1	3	0.5
Go to Data	0	0	-	1	1	0.16
Implementation	1	0	0	-	1	0.17
					6	1.00

Figure 25: Pair-wise comparison matrix for user interface.

Design criterion	Weight factor	Unit	Direct Retrieval			Cache [1]		
			Value	Score	Rating	Value	Score	Rating
Time requirement	0.33	Exp	Fair	5	1.65	Low	6	2.31
Space requirement	0.33	Exp	Low	7	2.31	Fair	5	1.65
Implementation difficulty	0.17	Exp	Fair	5	0.85	High	3	0.51
History record	0.17	Exp	Unsupported	3	0.51	Supported	7	1.19
					5.32			5.66

Figure 24: Weighted decision matrix for data retrieval.

5.3 User Interface

We compare our two solutions based on four criteria: whether the shown data is easy to read, whether it is easy to select to read the data of a certain bar code, whether it is easy to go to and get access to the data of the bar code and whether it is easy to be implemented. As shown in fig 25, we think whether it is easy to select certain data is most important. And others take almost the same weight.

We then compare our solutions based on the criteria. The first solution – putting information based on 3-D coordinate – has two shortcomings. First, imagine the computer screen is our server and the contents on the screen are our data. If I want to read the data clearly, I have to be in front of the data and keep an appropriate angle. This is the same situation as in reality. That's why we think reading data and going to data is less convenient.

However, putting information based on 2-D coordinate also has two shortcomings. Imagine

Design criterion	Weight factor	Unit	3-D display			2-D display		
			Value	Score	Rating	Value	Score	Rating
Read data	0.17	Exp	Fair	6	1.02	Clear	10	1.7
Select data	0.5	Exp	Easy	10	5	Difficult	7	3.5
Go to data	0.16	Exp	Fair	8	1.28	Easy	10	1.6
Implementation	0.17	Exp	Easy	8	1.36	Difficult	6	1.02
					8.66			7.82

Figure 26: Weighted decision matrix for user interface.

we have several bar codes in the same scene. Which information should we display? This is difficult to be selected by the program and inconvenient to be selected by the user. Thinking of the above, it makes the program difficult to be implemented.

The decision matrix is shown in Fig 26. After comparison, we decide to use the first solution, that is, putting information based on 3-D coordinate [2]. Our challenge is still how to make the data easy to read.

6 Selected Concept Description

6.1 Overview

As is stated above, the chosen concept in our project is to use ML Kit for barcode localization and identification, cache for data retrieval, and 3-D display for user interface, which is shown in Figure 27. In our design, the user would first scan a barcode, and using ML Kit for barcode localization and identification, our app would be able to decode the barcode. Then the app will use the decoded information to fetch data from Flowgate, where cache is used to improve the reaction time. Finally, the app will place the obtained information based on 3-D coordinate in reality world.

[2]This is the response to the Judge Panel's comment 1 in section 9.

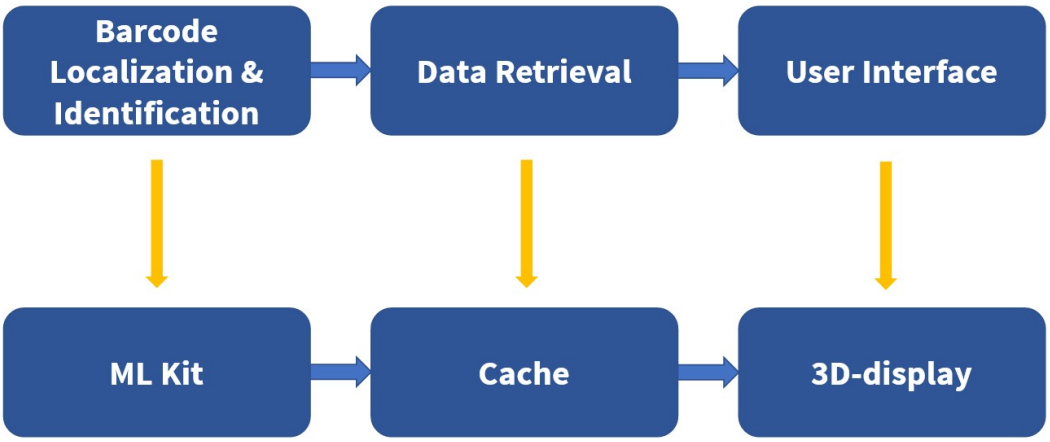


Figure 27: An overview of the chosen concept.

6.2 Engineering Design Analysis

Our major design lies on how to optimize user experience so that the staff in data center can utilize our app easily for maintenance. One important perspective is the steps needed to be completed by the user in order to get device information. User experience is also very important, so we need to set specific parameters to achieve a comfortable display of information.

6.2.1 Steps

A general overview of the steps is that the user needs to scan a barcode and then read the information of the corresponding device from AR interface. When considering the final design, a lot of details need to be specified. For example, can the user scan multiple bar codes with all the information displayed on the same screen? Is it better if the app can support rack identification?

After consulting our sponsor, the final steps that needs user to complete are:

1. Scan one bar code
2. Step back to catch the whole view of the rack
3. Step closer to read the information displayed by the corresponding device

After step 1, our app will retrieve information from the back-end server, which contains the location information of the device. After step 2, we get the location of the whole rack and then we can mark the specific device in this rack according to its location information. Next, we can display all its information by this device so that the user can check and maintain the device. After the user finishing maintaining this device, he/she can start a new session to scan the next bar code.

As shown above, our app can scan only one bar code. The idea of scanning multiple bar codes seems fascinating, but it has a big drawback. If multiple bar codes are required, then our app needs to keep scanning, which could lead to a high temperature. If we follow the steps above, the app is able to stop scanning when user starts to check device information and maintain the device, which can greatly reduce the cpu usage. Also, displaying information of several device at a time will let it looks messy and hard to check which information corresponds

to which device. Besides, scanning the rack can help the localization of the device, and thus the user can have a clear view of both the device and its information.

6.2.2 User Experience

As indicated in the book [4], if the frame rate is too low, the scene in AR can not be displayed smoothly, which will lead to a bad user experience. As suggested, the frame rate needs to be above 15 frames per second. One obstacle in setting a high frame rate is that it needs to keep refreshing the scene displayed on your phone and this may lead to a counter reaction of high temperature. Our final design sets the frame rate as 60 frames per second, which is a common frame rate in today's AR application. This frame rate guarantees both smooth display and acceptable phone temperature.

6.3 Design Description

The detailed flowchart of the design is shown in Figure 28. In our design, when a user wants to maintain a device in Figure 28, he scans the barcode, and our device starts to detect and decode the barcode. Then the app will use the decoded information to fetch data from Flowgate. Next, the user steps back to catch the whole view of the rack A. Our app then recognizes the rack and devices, and marks them by those red blocks in AR interface. The location information is provided in data from Flowgate. Then the user steps closer. Our app can show the detailed information of the device, and then the user can start maintaining the device with those information. After the user finishes, he can choose to maintain another device by starting a new session. Our app will be refreshed, and the user can scan another barcode now.

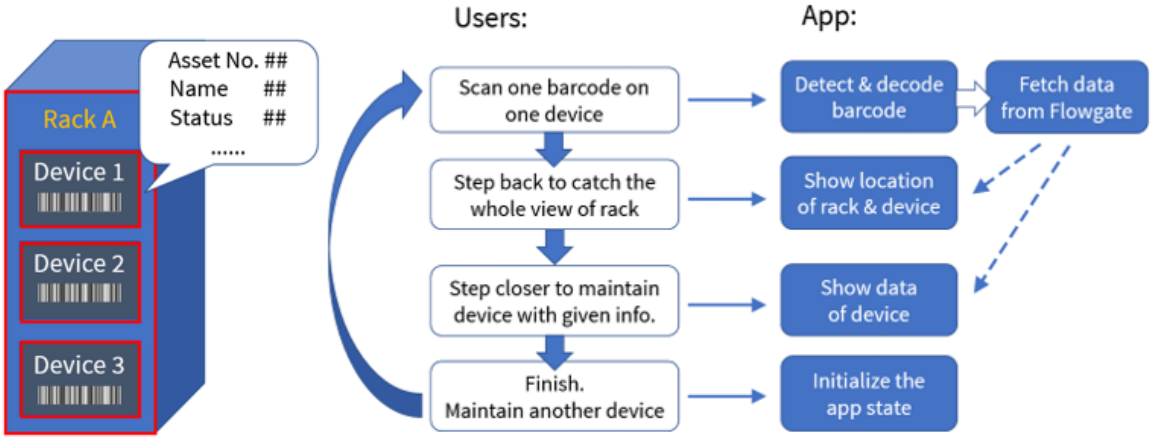


Figure 28: Detailed flowchart of the system.

To make the design work, we divide it into several tasks based on subfunctions, and implement these tasks through collaborative programming. We first install the Flowgate sever. For Andriod, we use ML Kit for barcode scanning, and ARCore and Sceneform packages to enable the AR environment and image recognition. For iOS, only ARKit is needed for all these tasks. In this process, we take the non-ideal cases into consideration. For instance, for the case that

the barcode information is missing or the barcode is damaged, the program will display nothing on the card and show an error message [3].

After implementing the subfunctions, we integrate them to build a complete AR app. The app will work as long as the user install it on their mobile phones, and give the required camera permission to it. As the app is an integration of the desired subfunctions, users would be able to conveniently obtain the necessary information that could help them in maintenance and audits in data centers. All the parts needed in the process are just our laptops, mobile phones, server machines and data center devices borrowed from our institute, which have no cost. For details on implementation and required materials, please refer to section 6.4.

6.4 Implementation Plan

6.4.1 Materials

For the back-end part of our project, we use Flowgate as our back-end system to contain all data center information. The Flowgate system needs to be installed on an EXSi system, and requires at least 4 vCPUs, 8GB of memory, and 60GB of disk space. In addition, we need Python to manage asset data in Flowgate via API.

For the front-end part of our project, since we are implementing our AR app in both Android and iOS platforms, we need to use both Java and Swift for app development. The coding can be done in Android Studio and Xcode respectively. To enable AR, bar code scanning, and image recognition, we can use only ARKit for iOS, since it contains all necessary packages and libraries. For Android, we use the ARCore and Sceneform packages for AR development and image recognition, and use the ML Kit package for bar code scanning.

Regarding hardware devices and equipment, we need an Android smart phone and an iOS smart phone for app development; a server machine to install Flowgate system; and a rack and several physical devices for testing and demonstration purposes.

To summarize, the required materials are listed as follows:

1. Programming languages:
    a. Java
    b. Swift
    c. Python
2. Software and systems:
    a. Flowgate
    b. EXSi
    c. Android Studio
    d. Xcode
3. Packages and libraries:

[3] This is the response to the Judge Panel's comment 3 in section 9.

a. ARKit

b. ARCore

c. Sceneform

d. MLKkit

4. Hardware devices and equipment:

   a. A sever machine

   b. An Android smart phone

   c. An iOS smart phone

   d. A rack

   e. 2 - 3 physical data center devices

6.4.2 Implementation Process

The implementation of our project is mainly divided into 7 steps, as shown in the flow chart in figure 29. Details about the process are listed as follows:

- Step 1: The first step is to install and configure the backend server. We have installed Flowgate in the ESXi system of our server, and have created information sets for data center assets. Sensors and servers are also mapped together to ease information retrieval.

- Step 2: The second step is to realize subfunctions of front-end App separately, including data retrieval, bar code scanning, and AR. Data retrieval is realized by Http requests. For Android, we use ARCore and Sceneform to implement the AR test programs, and use ML Kit to realize bar code recognition. For iOS, all necessary functions for AR and bar code scanning are included in ARKit, so we use it to implement our test programs. In a test program for AR, we managed to find a plane in the environment and place a text box on it, as shown in figure 30.

Figure 30: Test program for AR

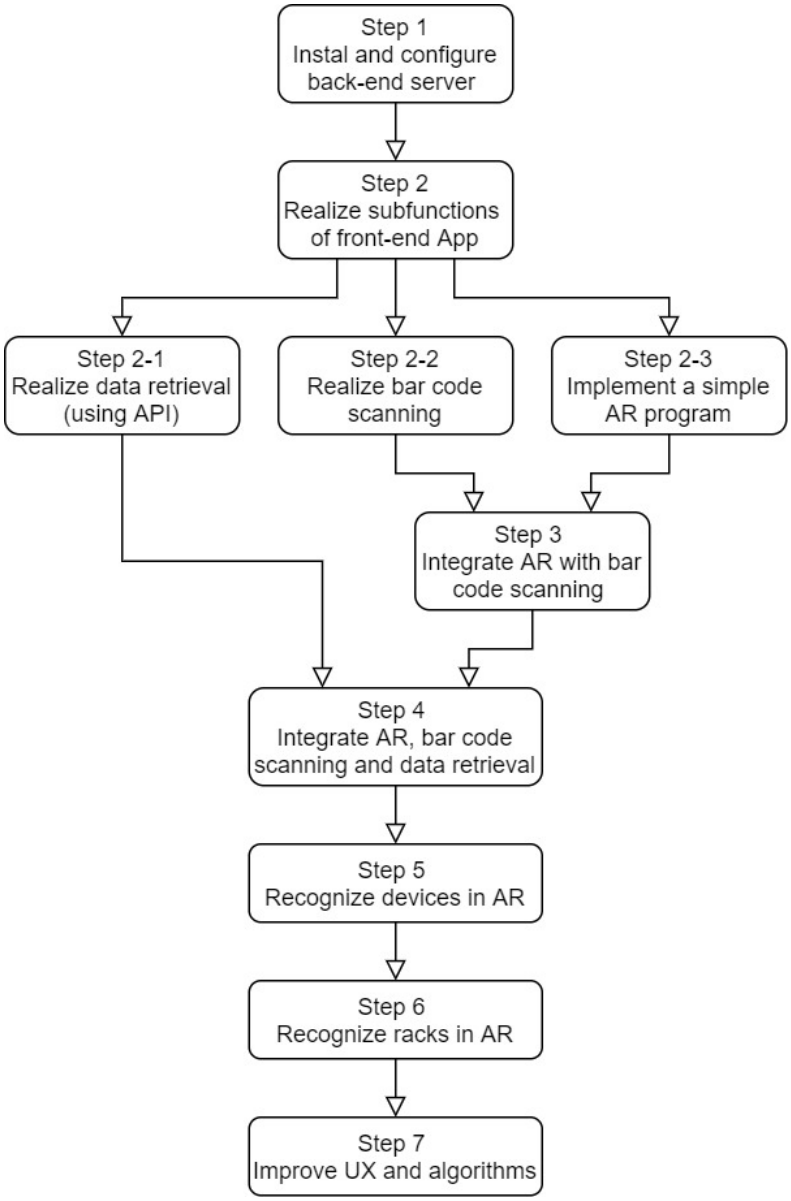


Figure 29: Implementation process

- Step 3: The third step is to integrate AR with bar code scanning. In this step, we have implemented an app to scan any bar code and then place the readings onto a 3D window in AR (figure 31). In Android, since Sceneform and ML Kit cannot use camera simultaneously, we adapt the algorithm of keep capturing image from AR interface and then sending it to ML Kit modules to process the image in the background (for bar code recognition).

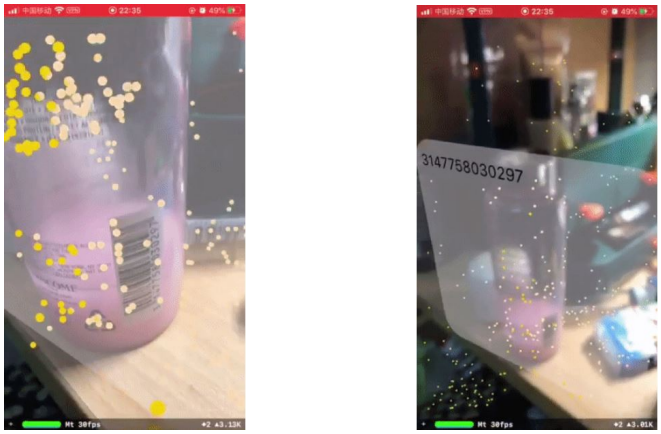


Figure 31: Integrate AR with bar code scanning

- Step 4: Step 4 is to combine step 3 with data retrieval. The backend data obtained using the bar code is displayed onto a 3D window (figure 32). Asynchronous Http request mechanism is applied to ensure that network problem will not influence the operation of the main AR activity.

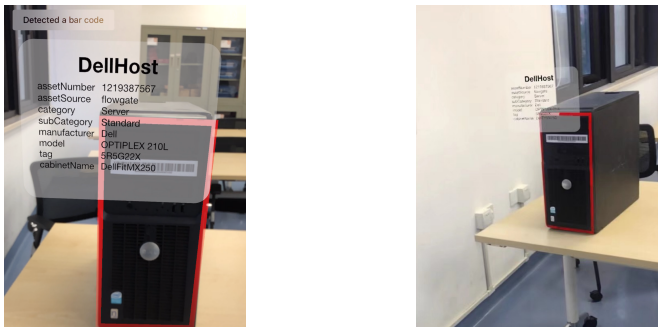


Figure 32: Display the device information onto a 3D window, and mark the device using a red rectangle

- Step 5: In Step 5, we have implemented a new function of recognizing devices in the AR interface. In Android, we use the `AugmentedImage` library in Sceneform to realize image recognition. An image of the device is added to the database, so that the real device can be detected by the app using camera. The iOS version is done similarly, using tool kits from ARKit. To mark the device vividly and clearly, we program our app to draw a red rectangle onto the device in the AR interface, as shown in figure 32.

- Step 6: In step 6, we will realize recognition of racks in data center, and display their corresponding information onto the AR interface. The method is similar to step 5. Until this step, the first complete prototype will be finished.

- Step 7: In step 7, we will improve the user experience (UX) by easing the work flow and making better user interface. Also, we need to improve the algorithms by reducing sampling rate (for bar code scanning) to make the operation smoother, and enabling scene freezing and session pausing to reduce power consumption. In addition, we will investigate

on how to handle multiple bar codes in one scene and display the information in a proper way.

Until now we have finished step 1 to 5. Step 6 and 7 are still under progress, and will be completed soon.

6.4.3 Budget

To implement our project, we use our own smart phones, and have borrowed a rack, several host machines, and a server with ESXi installed from our institute. All other software is either free or open-source. Thus currently we do not need to spend any budget for our project.

6.5 Validation Plan

For the validation of our project, we want to conduct experiments in real data center, for instance, the data center in Longbin building which contains enough racks and servers. The detailed validation methods for all the Engineering Specifications can be divided into 4 types:

1. Repeated trials to calculate success rate:

   For "Object Localization Correctness" and "Data Retrieval Accuracy", we will conduct 50 repeated trials to localize a device and retrieve information for a device respectively. We will count the number of failures in the 100 trials to calculate the success rate for object localization and data retrieval. As shown in table 1, we expect the success rate for object localization to be higher than 90% and the success rate for data retrieval to be higher than 99%.

2. Repeated trials to calculate average time:

   For "Object Localization Time", "Barcode Identification Time", "Database Query Time", and "AR Image Generation Time", we will conduct 50 repeated trials of object localization, bar code identification, database query and AR image generation respectively. The total time for each of the operation will be measured and written to the log of our program. Then the average time for each operation can be found simply by

   $$\text{average time} = \frac{\text{total time}}{50}. \tag{1}$$

   The average time will be considered as the testing result. As shown in table 1, we expect the average time for object localization to be smaller than 0.5 s, the average time for bar code identification to be smaller than 0.5 s, the average time for database query to be smaller than 1 s, and the average time for AR image generation to be smaller than 0.1 s.

3. Sampling from continuous measurements to find the mean value:

   For "Frame Rate" and "Device Temperature in Operation", our IDE can provide us with continuous measurements of those two parameters. We will let our app run for 5 minutes, and record values of frame rate and device temperature every 10 seconds. Then, we will calculate the mean frame rate and temperature as our testing results. As shown in table

1, we expect the mean frame rate to be larger than 15 fps, and the device temperature to be lower than $40°C$.

4. Direct reading:

   For "Size of Software Package" can be directly read from the app management panel in the "Settings" of smart phones. As shown in table 1, we expect the size of our app package to be smaller than 110 MB.

A summary of the validation methods for all the Engineering Specifications is shown in table 2.

Engineering Specifications	**Method for testing**
Object Localization Correctness	Repeated trials to calculate success rate
Data Retrieval Accuracy	Repeated trials to calculate success rate
Object Localization Time	Repeated trials to calculate average time
Barcode Identification Time	Repeated trials to calculate average time
Database Query Time	Repeated trials to calculate average time
AR Image Generation Time	Repeated trials to calculate average time
Frame Rate	Sampling from continuous measurements to find the mean value
Device Temperature in Operation	Sampling from continuous measurements to find the mean value
Size of Software Package	Direct reading

Table 2: Summary of validation methods for Engineering Specifications.

A rough current testing result for part of the Engineering Specifications is shown in figure 33. Since we have not finished the app development completely, we have not measured the size of our software package. From the table, we can see that most of parameters of our app have already met the specifications. For bar code identification time and device temperature, we are planning to solve the problems by changing for clearer bar codes (avoiding light reflections and using high resolution images) and reducing image sampling rate (to reduce CPU usage) respectively. We believe that our final product can meet all the specifications after the improvements.

	Current performance	**Requirement**
Barcode identification time	≈ 0.85s	< 0.5s
Object localization time	≈ 0.23s	< 0.5s
Object localization correctness	≈ 98%	> 90%
AR image generation	< 0.1s	< 0.1s
Data retrieval accuracy	100%	> 99%
Database query time	≈ 0.89s	< 1s
Frame rate	≈ 60 fps	> 15 fps
Device temperature	≈ 47 ºC	< 40 ºC

Figure 33: A rough current testing result

7 Project Timeline and Plan

7.1 Project Timeline

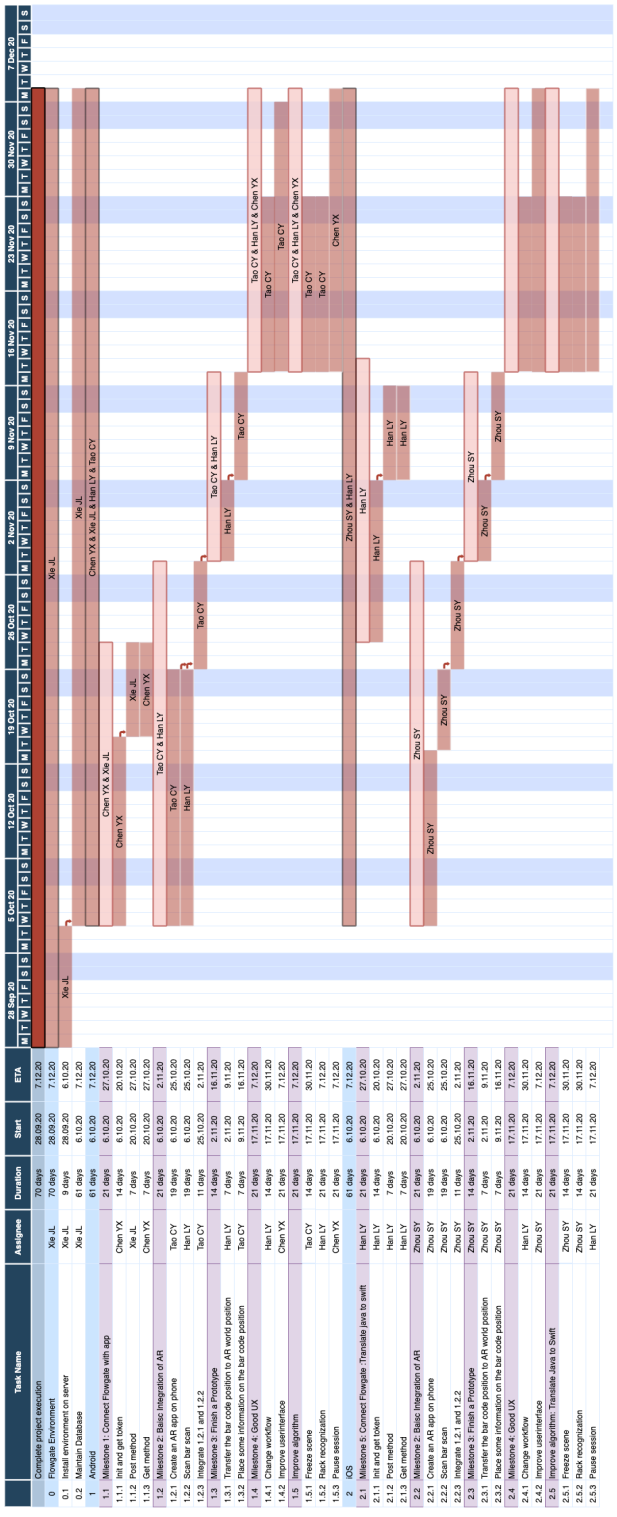


Figure 34: Gantt chart of our whole plan.

Here is our gantt chart (Fig 34). And to take a closer look at the gantt chart, let's look at Fig 35.

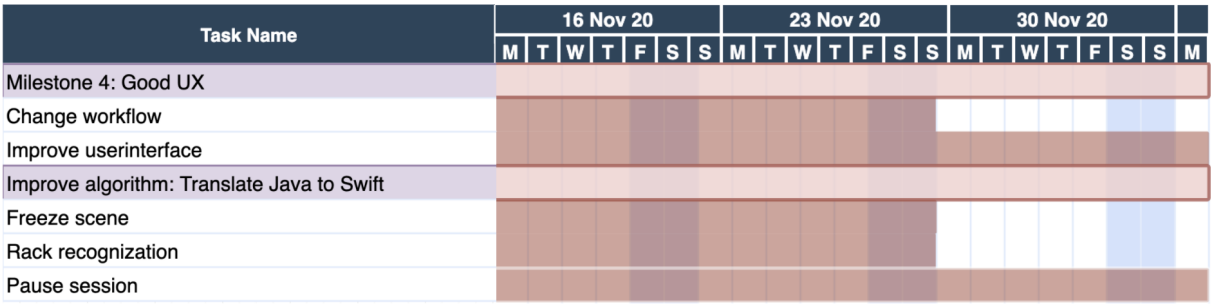


Figure 35: Gantt chart focusing on what to do from now on.

The due date of video is on November 30th, before then, we want to finish realizing rack recognition and adjusting the workflow of our final project. Also, we will add a function to allow user to freeze the scene so that they can read message clearly. Before design expo, we will keep improving our user interface. To improve the algorithm, we want to pause AR session to save CPU when we do not need to scan.

8 Analysis of Potential Problems

As we still have several tasks to finish, it is possible that some problems would appear. Based on our plan, we analyze the potential problems, and raise the corresponding solutions.

The first potential problem is that as we want to adjust the workflow, and add a new function to allow user to freeze the scene, we have to change the code, which may lead to some unexpected bugs. Fortunately, we use Github for collaborative programming, and we have timely backups of the source codes. Our solution is to continue utilizing the advantages of collaborative programming, and finish the tasks, as well as fix the bugs as soon as possible. In the worst case, at least we have backups of the source codes, so that we would still be able to produce the video ahead of the incoming deadlines.

The second potential problem is that we have not tested our app on real racks, and thus the app might not work as expected in real data centers. We will contact IT office as soon as possible to test our app in real data centers to ensure that the app would work well on design expo, based on the validation plan discussed in section 6.5.

The third potential problem is that we need to demonstrate our project in design expo, but we do not have appropriate devices. We will contact IT office as soon as possible to borrow racks and devices that could be used for our demonstration [4].

9 Q&A

In the oral presentation of Design Review 2 and 3, the Judge Panel raised three questions about our project, and the questions and answers are stated in this section.

[4]This is the response to the Judge Panel's comment 2 in section 9.

1. *(DR2)* How do you display the information with AR?
   Response: We place information based on 3-D coordinate in reality world.

2. *(DR3)* How will you demonstrate your project in expo?
   Response: We will borrow racks and devices from Longbin building, and we need to contact IT office as soon as possible.

3. *(DR3)* How do you deal with the case that the barcode information is missing or the barcode is damaged?
   Response: In this case, since the barcode ID is not inside the Flowgate database, the program will display nothing on the card and show an error message.

10 Conclusion

Our project, AR in Data Center, is about applying augmented reality in data centers. In this report, we provide necessary background and introduction to the project, engineering specifications, concept generation, concept selection process, selected concept description, project plan and analysis of potential problems. To sum up, our project should produce an app that could obtain all kinds of information related to a specific data center from the back end database Flowgate, and display necessary information using AR to facilitate data center maintenance and audit work. The customer requirements include short reaction time, information correctness, comfortable display, and portable device. By conducting literature search and considering the constraints in reality, we further set up the engineering specifications based on the requirements. We generate quite a few concepts through morphological analysis, and select a concept after comparison. This chosen concept is to use ML Kit for barcode localization and identification, cache for data retrieval, and 3-D display for user interface.

We still have several tasks to finish before design expo. The next one is to finish adjusting the workflow of our final project before the due date of group video on November 30th. Based on our plan, we also analyze the potential problems, and raise the corresponding solutions.

References

[1] M. F. Abadi, F. Haghighat, and F. Nasiri, "Data center maintenance: applications and future research directions," Facilities, vol. 38, no. 9/10, pp. 691–714, 2020.

[2] M. Ayat, M. Sharifi, S. Ibrahim, and S. Sahibudin, "CMDB Implementation Approaches and Considerations in SME/SITU's Companies," 2009 Third Asia International Conference on Modelling & Simulation, 2009.

[3] Vmware. "Flowgate." GitHub. [Online]. Available: https://github.com/vmware/flowgate. Accessed: Sept. 27, 2020.

[4] A. B. Craig, Ed., "Understanding Augmented Reality", San Francisco, Morgan Kaufmann, 2013.

[5] M. Hakkarainen, C. Woodward and M. Billinghurst, "Augmented assembly using a mobile phone," 2008 7th IEEE/ACM International Symposium on Mixed and Augmented Reality, Cambridge, 2008, pp. 167-168, doi: 10.1109/ISMAR.2008.4637349.

[6] S. Deffeyes, "Mobile augmented reality in the data center," in IBM Journal of Research and Development, vol. 55, no. 5, pp. 51-55, Sept.-Oct. 2011, doi: 10.1147/JRD.2011.2163278.

[7] N. Fiona, "A Study on Tolerable Waiting Time: How Long Are Web Users Willing to Wait?", Behaviour & Information Technology - Behaviour & IT, 2003, pp. 285, doi: 10.1080/01449290410001669914.

[8] Q. Peng and Y. Song, "Object recognition and localization based on Mask R-CNN," Journal of Tsinghua University (Science and Technology), 2019, vol. 59, no. 2, pp. 135-141.

[9] E. Ohbuchi, H. Hanaizumi and L. A. Hock, "Barcode readers using the camera device in mobile phones," 2004 International Conference on Cyberworlds, Tokyo, Japan, 2004, pp. 260-265, doi: 10.1109/CW.2004.23.

[10] A. Baek, K. Lee, and H. Choi, "CPU and GPU parallel processing for mobile Augmented Reality," vol. 1, pp.133-137, 2013, doi: 10.1109/CISP.2013.6743972.

[11] O. Oktay et al., "Stratified Decision Forests for Accurate Anatomical Landmark Localization in Cardiac Images," in IEEE Transactions on Medical Imaging, vol. 36, no. 1, pp. 332-342, Jan. 2017, doi: 10.1109/TMI.2016.2597270.

[12] LabCE, "Data center operations supported by augmented reality," MediaLab. [Online]. Available: https://www.labce.com/spg650115_barcode_reading_and_accuracy.aspx. Accessed: Oct. 10, 2020.

[13] A. Craig. Augmented Reality Hardware, pp. 69-124, 2013. doi: 10.1016/B978-0-240-82408-6.00003-5.

[14] Apple. “Keeping iPhone, iPad, and iPod touch within acceptable operating temperatures.” Apple. [Online]. Available: https://support.apple.com/en-us/HT201678. Accessed: Oct. 10, 2020.

[15] Google. “Safety & regulatory manual (Pixel 3 & Pixel 3 XL 2018).” Google. [Online]. Available: https://support.google.com/pixelphone/answer/9134668?hl=en. Accessed: Oct. 10, 2020.

[16] Apple. “ARKit.” Apple. [Online]. Available: https://developer.apple.com/documentation/arkit. Accessed: Oct. 10, 2020.

[17] Google. “Supported Devices.” Google. [Online]. Available: https://developers.google.com/ar/discover/supported-devices. Accessed: Oct. 10, 2020.

[18] Apple. “App Store.” Apple. [Online]. Available: https://www.apple.com/app-store. Accessed: Oct. 10, 2020.

[19] Google. “Google Play.” Google. [Online]. Available: https://play.google.com/store/apps. Accessed: Oct. 10, 2020.

11 Appendix

11.1 Bios

11.1.1 Shuyi Zhou

I am Shuyi Zhou, a senior student major in Electrical and Computer Engineering (ECE) exchanged to Germany last year, and the attached picture of myself is taken in Germany. I have learnt courses like data structures and algorithms, computer organization, and computer networks, and become familiar with programming languages like C/C++ and Python. I am currently working as a teaching assistant for *Introduction to Computer and Programming* course. In the future, I will pursue a master degree in Tokyo.

Figure 36: Shuyi Zhou

11.1.2 Chenyun Tao

I am Chenyun Tao, a senior student major in Electrical and Computer Engineering (ECE) at University of Michigan - Shanghai Jiao Tong University Joint Institute (UM-SJTU JI). I like travelling, and the attached picture of myself is taken during my trip to Japan. I have learnt courses like data structures and algorithms, computer organization, and computer networks, and become familiar with programming languages like C/C++ and Python. In the future, I will pursue a master degree, and I have participated in JI's GDP program with University of Michigan, School of Information.

Figure 37: Chenyun Tao

11.1.3 Liying Han

I am Liying Han, a senior student major in Electrical and Computer Engineering (ECE) at University of Michigan - Shanghai Jiao Tong University Joint Institute (UM-SJTU JI). I have learnt courses like data structures and algorithms, computer organization, and machine learning. I am familiar with C/C++ and Python programming languages. In the future, I want to go abroad and study in the field of machine learning algorithms.

Figure 38: Liying Han

11.1.4 Yaxin Chen

I am Yaxin Chen, a senior student major in Electrical and Computer Engineering (ECE) at University of Michigan - Shanghai Jiao Tong University Joint Institute (UM-SJTU JI). I have learnt courses like data structures and algorithms, computer organization and methods and tools for big data, and currently I work as a teaching assistant for operating systems course. I am familiar with C/C++ and Python programming languages. In the future, I want to study in the field of distributed systems.

Figure 39: Yaxin Chen

11.1.5 Jinglei Xie

I am Jinglei Xie, a senior student major in Electrical and Computer Engineering (ECE) at University of Michigan - Shanghai Jiao Tong University Joint Institute (UM-SJTU JI). I am really interested in the field of computer science and information systems, and have taken various related courses like operating systems, big data, computer network, artificial intelligence, etc. I am familiar with several kinds of programming languages, and has been a teaching assistant for the course *Programming and Elementary Data Structures*. Besides that, I also have robotics related experience, and has attended global competitions as a member of SJTU VEX robotics team. After graduation, I plan to go abroad for further studies, and pursue a master's degree in computer science. Maybe I would like to focus more on systems and networks, since they attract me the most.

Figure 40: Jinglei Xie