

TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
ADVANCED COLLEGE OF ENGINEERING AND MANAGEMENT
DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
KALANKI, KATHMANDU



A Major Project Mid-Term Defense Report On
“ANTHROPOMETRIC MEASUREMENT SYSTEM USING
MACHINE LEARNING AND COMPUTER VISION”
[CT 707]

SUBMITTED BY:

Kshitiz Khanal ACE075BCT023
Milan Adhikari ACE075BCT028
Nirala Lamichhane ACE075BCT034
Rasik Nepal ACE075BCT044

SUPERVISED BY:

Er. Kritika Gulati

A Major Project Mid-Term report submitted to the department of Electronics and
Computer Engineering in the partial fulfillment of the requirements for degree of
Bachelor of Engineering in Computer Engineering
Kathmandu, Nepal
August 2022

ACKNOWLEDGEMENT

We take this opportunity to express our deepest and sincere gratitude to our Project Supervisor **Er. Kritika Gulati**, Lecturer, ACEM for her insightful advice, motivating suggestions, invaluable guidance, help and support in successful completion of this project and also for his/her constant encouragement and advice throughout our Bachelor's programme.

We express our deep gratitude to **Er. Ajaya Shrestha**, Head of Department of Electronics and Computer Engineering, **Er. Bikash Acharya**, Deputy Head, Department of Electronics and Computer Engineering for their regular support, co-operation, and coordination.

The in-time facilities provided by the department throughout the Bachelors program are also equally acknowledgeable.

We would like to convey our thanks to the teaching and non-teaching staff of the Department of Electronics & Communication and Computer Engineering, ACEM for their invaluable help and support throughout the period of Bachelor's Degree. We are also grateful to all our classmates for their help, encouragement and invaluable suggestions.

Finally, yet more importantly, I would like to express our deep appreciation to my grandparents, parents, siblings for their perpetual support and encouragement throughout the Bachelor's degree period.

Kshitiz Khanal	ACE075BCT023
Milan Adhikari	ACE075BCT028
Nirala Lamichhane	ACE075BCT034
Rasik Nepal	ACE075BCT044

ABSTRACT

With the evolution of technology, the pattern of shopping is also changing. From going shopping physically to buying them online, time has evolved. Meanwhile, some problems have arisen in this tech industry. Comfort and fit of garments affect everyone. For the garment to end up as an approved garment in the customer's wardrobe, it has to be improved and evaluated many times over. The final evaluation to pass is when the customer finds the garment, tries it on. In the world of fashion e-commerce platforms, maximum return the garment because it doesn't fit them well.

The purpose of our project is to build a body measurement system for the customers, so individualized digital body measurement of the customer can be collected in a hassle free-way. System can deliver personalized tailored size recommendations using Computer Vision and Machine Learning just by using a personal mobile phone. As a result of using body measuring applications for individualized tailored size, customers obtain the products they love in sizes that fit their bodies and are much less likely to return undesirable items.

Keywords: *online shopping, garments, e-commerce, body measurement system, Computer Vision, Machine Learning, customer*

TABLE OF CONTENTS

ACKNOWLEDGEMENT	i
ABSTRACT	ii
TABLE OF CONTENTS	iii
LIST OF FIGURES	v
LIST OF TABLES	vi
LIST OF ABBREVIATIONS/ACRONYMS	vii
CHAPTER 1	1
INTRODUCTION	1
1.1 Background	1
1.2 Motivation	2
1.3 Statement of the problem	2
1.4 Project objective	2
1.5 Significance of the study	3
CHAPTER 2	4
LITERATURE REVIEW	4
CHAPTER 3	6
REQUIREMENT ANALYSIS	6
3.1 Hardware Requirements	6
3.2 Software Requirements	6
3.3 Functional Requirements	6
3.4 Non-Functional Requirements	6
CHAPTER 4	8
SYSTEM DESIGN AND ARCHITECTURE	8
4.1 Algorithm:	8
4.2 Flowchart:	8
4.3 System Design	9
4.4 Use Case Diagram	11
4.5 DFD Level 0	12
4.6 DFD Level 1	12

CHAPTER 5	13
METHODOLOGY	13
5.1: MediaPipe Pose	13
5.2 Pose Detection	14
5.2.1 Front Pose	14
5.2.2 Side Pose:	15
5.3 Determining Ratio	16
5.4 Measurements	17
5.5 Basic Django	17
5.6 Working of webapp	19
CHAPTER 6	21
EXPECTED OUTPUT	21
CHAPTER 7	22
RESULT AND ANALYSIS	22
7.1 Work Progress	22
7.2 Problems Faced	26
7.3 Work Remaining	26
7.4 Time Schedule	27
CHAPTER 8	28
TOTAL COST	28
REFERENCES	29

LIST OF FIGURES

Figure 1: Flowchart.....	8
Figure 2: System Design.....	9
Figure 3: Use Case Diagram.....	11
Figure 4: DFD Level 0.....	12
Figure 5: DFD Level 1.....	12
Figure 6: Pose Landmarks	13
Figure 7: Front Pose.....	14
Figure 8: Side Pose	15
Figure 9: Basic Django	18
Figure 10: How the webapp works	19
Figure 11: Form to take user info	22
Figure 12: Page to take front photo.....	23
Figure 13:landmark detection from in browser javascript.....	23
Figure 14: Photo captured after correct pose	24
Figure 15:Calculation of measurement.....	25
Figure 16: Measurement result on the webpage	25

LIST OF TABLES

Table 1: Time Schedule of the project.....	27
--	----

LIST OF ABBREVIATIONS/ACRONYMS

API:	Application Program Interface
OpenCV:	Open Source Computer Vision Library
DOM:	Document Object Model

CHAPTER 1

INTRODUCTION

Shopping online is the process of buying goods or getting services without visiting the supplier or service provider physically. when it comes under clothing, the main requirement is proper fitting of the clothes in the body. Building trust is one of the major issue in online shopping. Customer satisfaction and the requirement fulfillment is the key to success for online business. Most of the customer prefer online mode rather than visiting the shops physically. Due to the dependency on internet and communication people prefer online means for easy and fast services. Physical goods like books, gadgets, furniture, and appliances are can be ordered online but goods related to fashion like clothes need proper measurement and fitting. [1]

Between 20-40% of the clothes bought online are returned among which 70% are returned due to fitting issues alone. [2].In fashion designing or clothing field we need precise measurements. Meanwhile, it is not possible for every customer to measure their body with measuring tape. Business related to product like books, gadgets, furniture, and appliances are easy to do online. But clothing sector moves towards a bit of complexity.

Also, the lack of standardized apparel size system has also created a lot of problems. It means that from brand to brand and from country to country- your size can fluctuate. A size ‘small’ in one store, could be ‘medium’ in another. Unfortunately, this can make online shopping confusing and stressful. It also makes shoppers crave the offline opportunity of trying clothes on to be sure of the right fit.

Our system tries to overcome this problem by measuring the customer body while creating the account and recommending the clothes for customers according to their measurements.

1.1 Background

Customers with online shopping can search for any product in many stores and purchase online with only a few steps instead of going to a retail store to find their desired products and standing in a line to check out. Clothes shoppers might prefer in-stores shopping rather than online shopping for many reasons that includes the inability to determine their proper clothing size accurately, try clothes on to see whether they are suitable or not, and assess the quality of the material of clothes.

Several studies have investigated the possibility of automatically estimating body measurements from customers' images. Most of these approaches utilize specialized devices (e.g., in-depth camera) to capture 3D images of the human body. Despite the measurement accuracy that a 3D based method provides, these methods are not applicable for all users who want to estimate their sizes on-the-go while purchasing items online. Only a few studies have used 2D images to estimate the body measurements. Studies that considered the 2D images estimate some basic measurements of human bodies or classify the human body into some predefined classes.

1.2 Motivation

Selecting the clothes which fit the best is pretty difficult for the customers using online stores. The sizes provided by different brands are not absolute. The expected size of the cloth and the actual size may vary. So, our proposed application provides a better alternative of selecting clothes which are best fitted for the customer. This project is motivated to create an application which would provide a user specific cloth section like a physical store with the convenience of online shopping.

1.3 Statement of the problem

According to GlobalWebIndex, in 2017 alone consumers returned \$351 billion of all online and offline purchases. Online shopping returns are more prevalent than in-store returns. Among which 52% of the customers say the main reason for returning was that the size did not fit and they were unable to try before buying it. The National Retail Federation reported that fraudulent returns cost retailers as much as \$15 billion in the U.S. in 2017. Almost 80% of people in the US and UK check the returns policy on a retailer's website before making a purchase. [3]

According to 48% of the customers the top solution would be detailed/accurate description of sizing. We can provide this with the help of a model which can be customized for each customer from which they can select the product size that actually fits them. [3]

1.4 Project objective

Our main objective is:

- To develop a system that can take the measurements of customers through their own device camera and recommend garments for custom fit.

1.5 Significance of the study

An accurate 3D model of a customer can help the customer to select the clothes which are best fitted for them. This will decrease the number of online shopping returns. Returning purchased goods are not only just an inconvenience for customers and retailers but there is also massive loss in capital for the retailers in transportation fees. Our proposed application can help eliminate the returning cost for the retailers and decrease the inconvenience for both involved parties.

CHAPTER 2

LITERATURE REVIEW

Estimating body measurements from 2D images is a very challenging task and only a small number of studies investigated this particular problem. While the rest of the studies used more complex and in-depth cameras to obtain 3D images that then are used to estimate the body measurements. In this section, we present the research effort that focus on estimating clothing size or body parts measurements using 3D and 2D images. Chang, et al. [4] proposed a dynamic fitting room that utilizes Microsoft Kinect and augmented reality technologies to allow individuals to visualize a real-time image of themselves while trying on different digital clothes. The system evaluates the user's body height according to head/foot joints and the depth information using two Kinect cameras one for taking a front image and the other for side image. The result of evaluating the proposed model shows that the estimated size is quite close to users' claimed sizes. This study however requires sophisticated hardware (i.e., Kinect's cameras) to estimate the measurements.

Xiaohua, et al. [5]proposed method aims to provide an automatic way to extract feature points and measure the sizes on 3D human bodies. The feature extraction and measurement estimation stage are usually serving as a pre-processing step for garment designer or virtual fitting application. The proposed method is automatic in a data-driven way, and it is insensitive to postures and different shapes of 3D human body. The approach also requires depth camera to estimate body measurements which is not appropriate and easy to use for online shoppers who want to estimate their body measurements. Another research Mojarrad & Kargar [6] proposed an approach that examine various images of people and estimate the size, body measurements, and other properties of the body (e.g., tall fat, short fat, tall thin, and short thin). The model starts by using many remote sensing applications such as image registration, image segmentation, and object detection and recognition as a pre-processing stage for feature extraction. Then, the model extracts feature and detect the edges and points of the body using canny edge detector approach. The findings of the study revealed that the proposed method was able to work on all of the human sizes. The limitation of this approach is that they didn't focus on specifying all the measurements of body parts.

One of the most related research works to our proposed approach is developed by Chandra, et al. [7] who presented an approach that estimate human body measurements by utilizing phone cameras. The proposed method tracks the photographed bodies using full body detection methodology to determine if there is a human body in the photo or not. The system then uses upper body detection to identify the face, neck, and shoulders then the system activates the upper body marker. Then, the body part will be estimated by calculating the difference between the most left-side and most right-side of any body part (e.g., shoulders). They also applied the golden ratio of human body to estimate the body measurements. Despite of the automated functionalities of their proposed system, the application recommends user to use some specific environmental settings (i.e., background and light). The method also only detects the upper body of customers while the lower body is still very important to estimate the proper size. Another limitation of the system is that the detection and measurement have to be repeated five times, and the system will find the mode of the results to give the user the most accurate size.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 Hardware Requirements

- A Working PC
- A Mobile Phone

3.2 Software Requirements

- **Python:** Python is a high-level, interpreted, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. [8]
- **Django:** Django is a free and open-source, Python-based web framework that follows the model–template–views architectural pattern. [9]
- **JavaScript:** is a programming language that is one of the core technologies of the World Wide Web, alongside HTML and CSS. As of 2022, 98% of websites use JavaScript on the client side for webpage behavior, often incorporating third-party libraries. All major web browsers have a dedicated JavaScript engine to execute the code on users' devices.
JavaScript is a high-level, often just-in-time compiled language that conforms to the ECMAScript standard. It has dynamic typing, prototype-based object-orientation, and first-class functions. It is multi-paradigm, supporting event-driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM). [10]

3.3 Functional Requirements

- Take the measurements accurately
- Make the model as detailed as possible
- Recommend clothes as according to the measurements taken

3.4 Non-Functional Requirements

- **Response Time:** This system should be able to take measurements in less than a minute.

- Portability: The system should be able to perform on any device.
- Reliability: Every functionality on the code must be able to work smoothly without failure under given normal conditions.
- Extensibility: Ability to extend the system is limitless. New functionalities can be added to the system anytime.

CHAPTER 4

SYSTEM DESIGN AND ARCHITECTURE

4.1 Algorithm:

- Step 1: Start
- Step 2: Ask height information from the user
- Step 3: Access Webcam feed from the client device
- Step 4: Transfer the obtained frames to backend
- Step 5: Determine Correct Pose of the User and capture Image
- Step 6: Determine the Ratio
- Step 7: Calculate necessary Measurements
- Step 8: Stop

4.2 Flowchart:

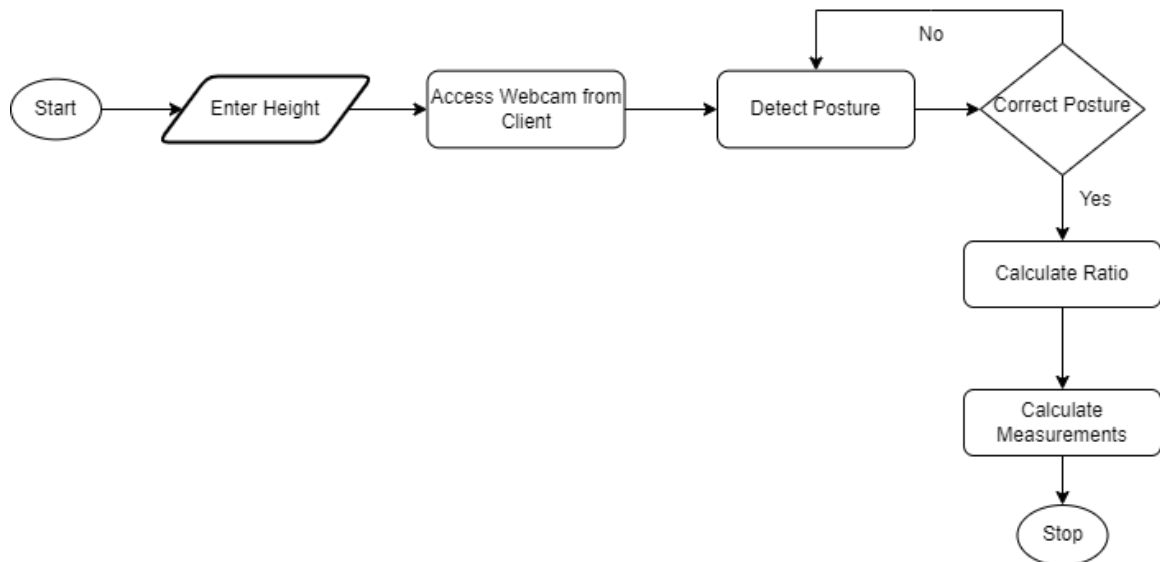


Figure 1: Flowchart

4.3 System Design

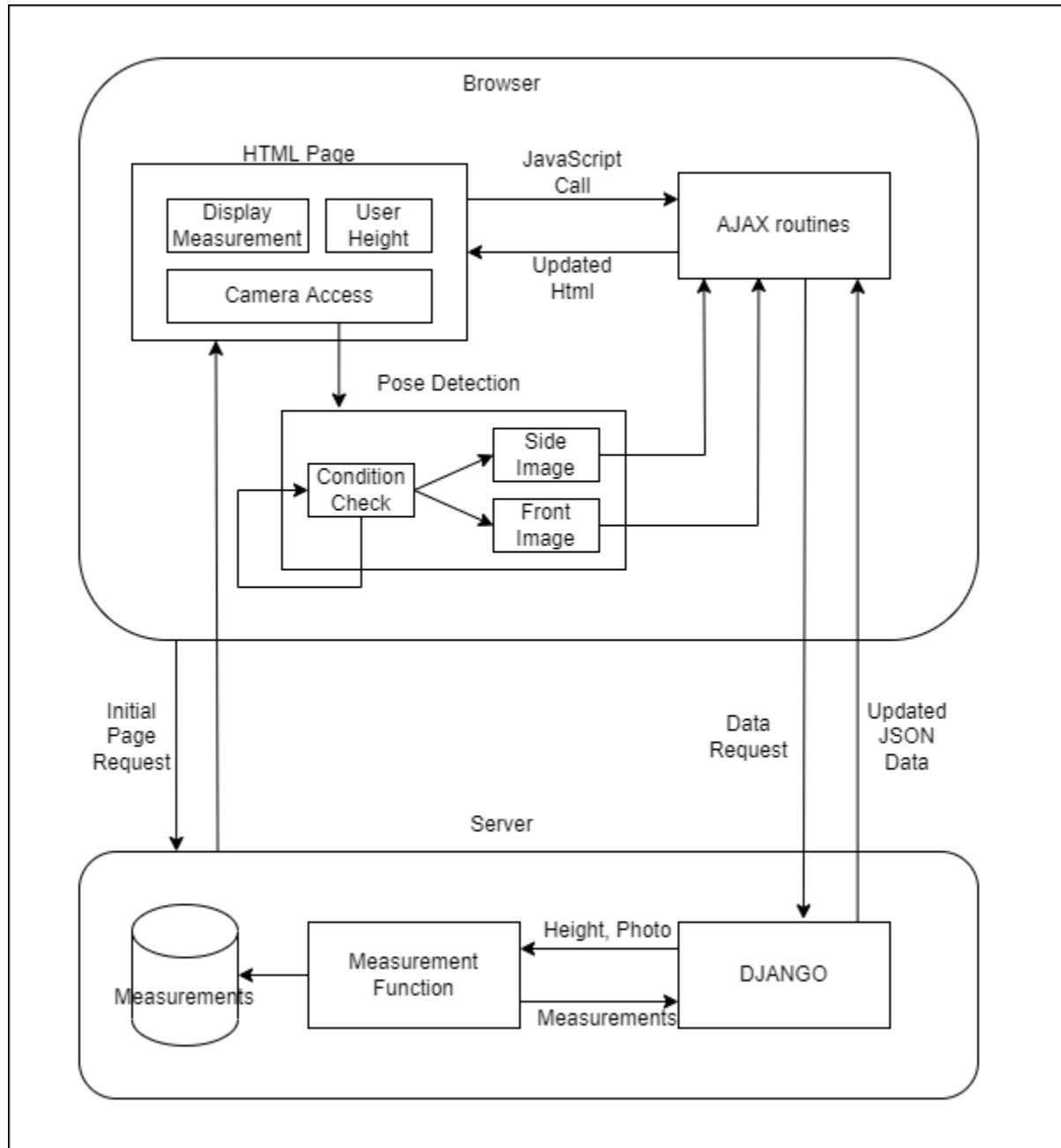


Figure 2: System Design

First, a user requests for resources, namely in this case the initial form for inputting the basic user information. Django considers the request as a URL and matches it with the existing URL paths in the `urls.py` file. Once the URL is mapped, Django jumps to the `views.py` folder and calls a view. The triggered view looks for models and templates to interact with, and then it returns the response

back to the user. Here, the model deals with the data associated with the user request. On the other hand, the template deals with the HTML and the static files, such as CSS files, JS files, images, etc., that are required to complete the view.

After the template is rendered on the browser, the user then inputs their basic information. This information is passed to the database of the server through AJAX calls. After submitting the basic information, the user is redirected to the next page where the front-side photo of the user is taken. When the browser requests for the front-photo page, the same steps are repeated and Django serves a template to the browser. Here the template also includes a JavaScript file which contains a script to check the posture of the user to capture their photo. To check the posture, we have used the mediapipe library. The posture is checked and once the correct posture is detected, a photo of the user is captured. The photo is converted in base64 format and passed to the server using AJAX. Side- photo is also taken in the same way.

Once the photo and height are passed to the views.py of the server, it calls the OpenCV program and passes these values to it. First the base64 image is converted to OpenCV compatible form and the measurements are calculated.

After the calculation is complete, the value is then sent back to the views.py file. The calculated values along with user information are stored in the database. The calculated values are then responded back to the browser through AJAX routine in JSON format. After the response is received the browser updates the user interface with the values.

4.4 Use Case Diagram

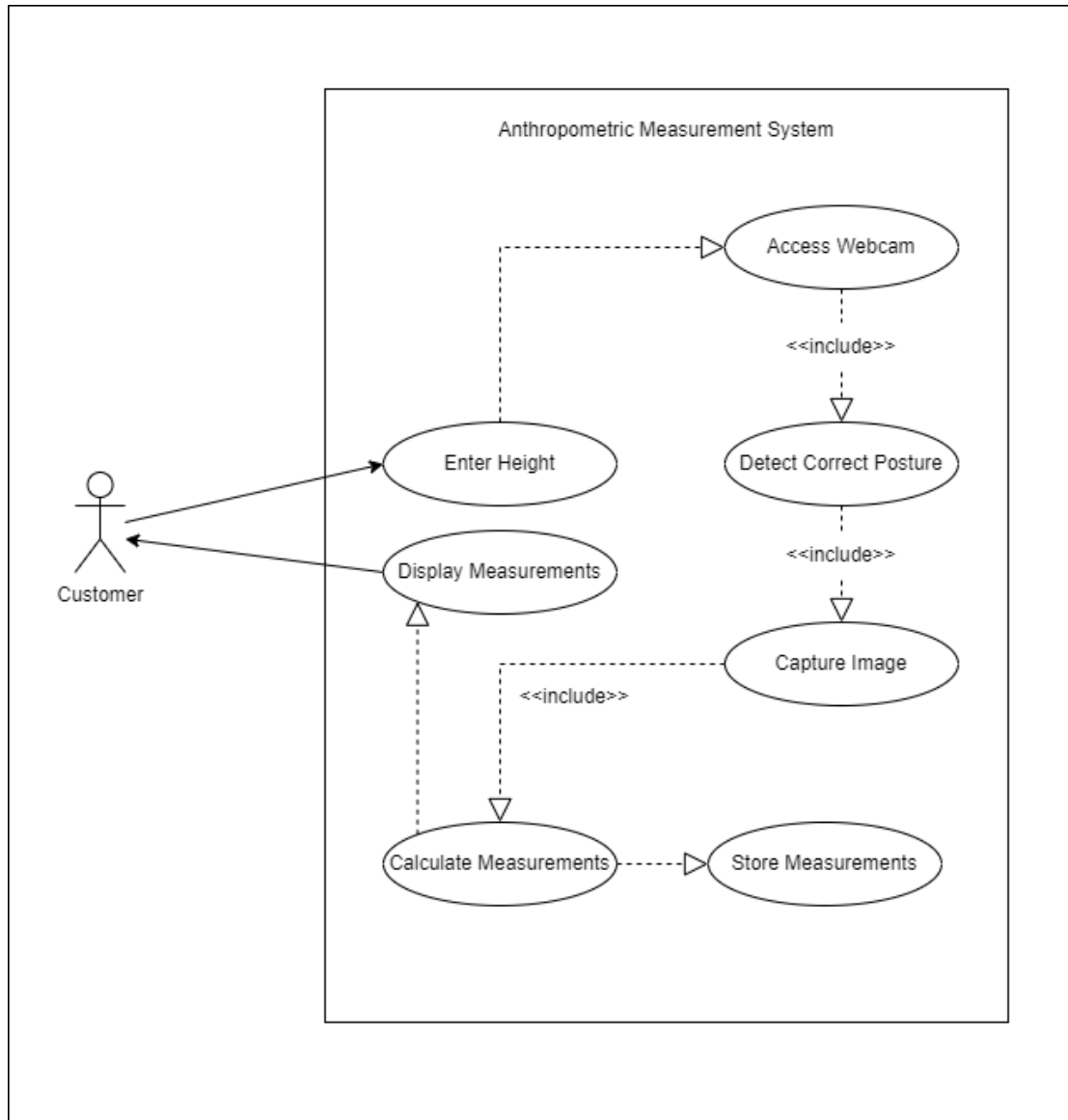


Figure 3: Use Case Diagram

Use Case Diagram shows how the customer interacts with the system. When customers enter the system initially, they are asked to enter their height. Then a webcam is accessed where they have to maintain the pose until the system detects the correct posture. After which image is captured and sent to the server where measurements are calculated. Then finally these measurements are displayed to the customer.

4.5 DFD Level 0

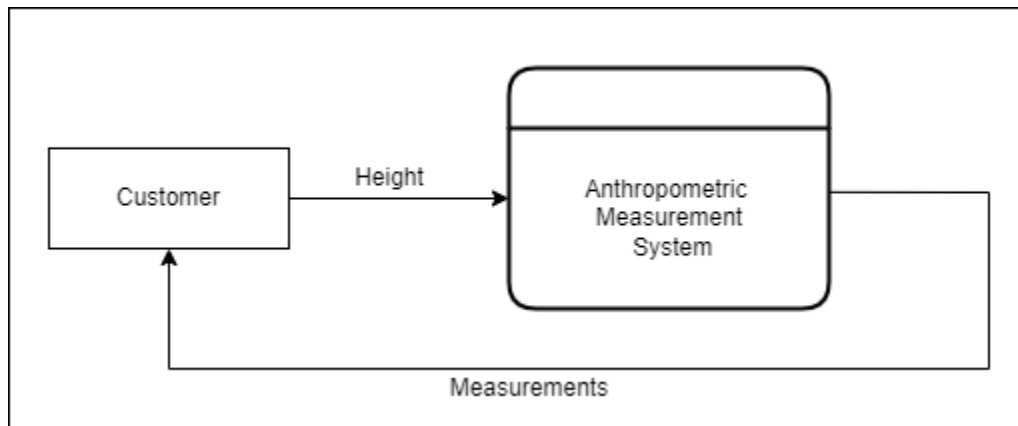


Figure 4: DFD Level 0

4.6 DFD Level 1

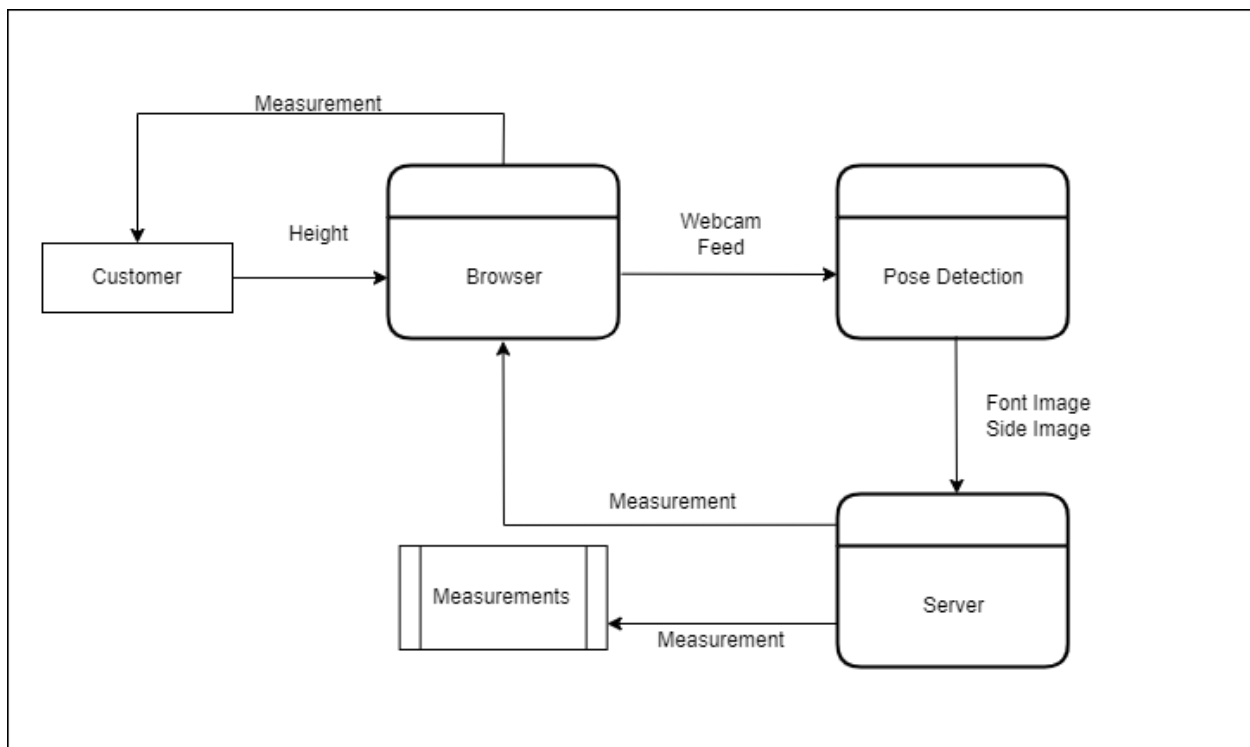


Figure 5: DFD Level 1

CHAPTER 5

METHODOLOGY

5.1: MediaPipe Pose

Obtaining Anthropometric measurements requires detection of the human body. For detecting human bodies in the feed captured from the client webcam, MediaPipe Pose is used.

MediaPipe Pose is a ML solution for high-fidelity body pose tracking, inferring 33 3D landmarks.

The landmark model in MediaPipe Pose predicts the location of 33 pose landmarks.

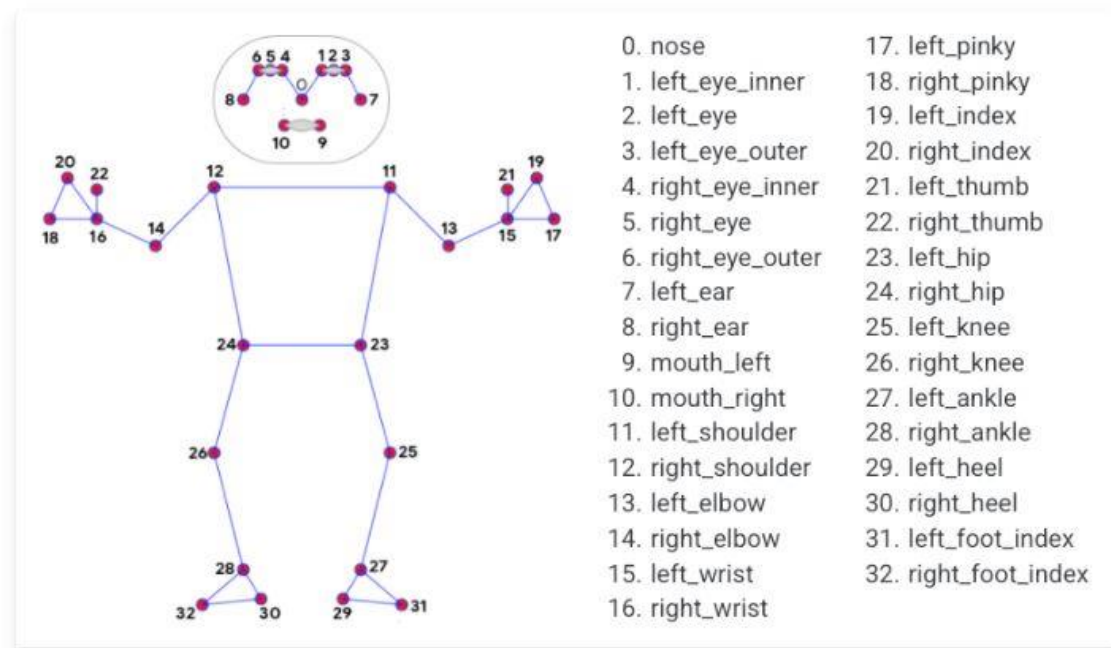


Figure 6: Pose Landmarks

Source: https://google.github.io/mediapipe/images/mobile/pose_tracking_full_body_landmarks.png [Accessed 22 08 2022]

MediaPipe Pose returns a list of pose landmarks. Each landmark consists of the following:

- x and y: Landmark coordinates normalized to [0.0, 1.0] by the image width and height respectively.

- z: Represents the landmark depth with the depth at the midpoint of hips being the origin, and the smaller the value the closer the landmark is to the camera. The magnitude of z uses roughly the same scale as x.
- visibility: A value in [0.0, 1.0] indicating the likelihood of the landmark being visible (present and not occluded) in the image.

5.2 Pose Detection

The video captured from the client webcam is passed to MediaPipe Pose. It detects different landmark information in the images. Using the information provided by MediaPipe Pose, the x and y coordinates of the landmarks are retrieved along with their visibility and z-index.

Slopes of individual body parts are calculated from the coordinates obtained from MediaPipe.

$$m = \frac{y2 - y1}{x2 - x1} = \frac{\Delta y}{\Delta x}$$

where,
 m= slope of the line
 x1,x2= coordinates of the x axis
 y1,y2 = coordinates of the y axis

In order to capture the images for the measurement, the customer has to stand in a specific predefined pose.

There are two poses that the customer has to stand in:

5.2.1 Front Pose

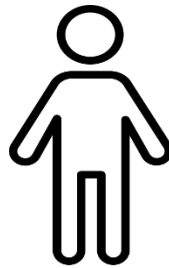


Figure 7: Front Pose

In order to detect if the Customer is standing in the correct Pose, the slopes of the body parts are compared with the pre-defined slope limits. Twelve Body parts and their landmark information were used for front pose detection.

```
front_slope_limit = {  
    'left_forearm': (-1.5, -0.2),  
    'left_arm': (-1.5, -0.2),  
    'shoulder': (-0.06, 0.06),  
    'right_arm': (0.2, 1.5),  
    'right_forearm': (0.2, 1.5),  
    'left_shoulder_hip': (3, 7),  
    'right_shoulder_hip': (-20, -7),  
    'waist': (-0.08, 0.08),  
    'left_thigh': (-8, -4),  
    'right_thigh': (4, 8),  
    'left_leg': (-8, -4),  
    'right_leg': (4, 8)  
}
```

These slope limits were obtained by several trials.

In each frame, the slopes of each body part are calculated and compared with the slope limits. The parts whose slope limits lie within the desired range are highlighted to indicate the correct position. After all the parts' slopes fall within the desired range, an image is captured and the landmark information is stored.

5.2.2 Side Pose:



Figure 8: Side Pose

Five body parts and their landmark information were used for Side Pose Detection.

```
side_slope_limit = {  
    'left_forearm': (-0.8, -0.2),  
    'left_arm': (-0.8, -0.2),
```

```

'left_shoulder_hip': (8, 50),
'left_thigh': (-50, -8),
'left_leg': (-50, -8)
}

```

These slope limits were obtained by several trials.

In each frame, the slopes of each body part are calculated and compared with the slope limits. The parts whose slope limits lie within the desired range are highlighted to indicate the correct position. After all the parts' slopes fall within the desired range, an image is captured and the landmark information is stored.

5.3 Determining Ratio

The user is asked to enter his height while using the application. The images captured are 2D images. In order to map the obtained measurements from the image into real world measurements, a ratio is necessary i.e we needed to find the relation between the persons height in the image and his/her's real height.

According to research, it is proven that a normal human being's height is Four times the length of the Femur. The landmarks information stored after detecting Front Pose is used to get the coordinates of Left Hip, Left Ankle, Right Hip, Right Ankle. Then the distance between them is calculated to obtain the length of Femur.

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

where
 x_1, x_2 = x coordinates of landmarks
 y_1, y_2 = y coordinates of landmarks

Then, the average length of femur is taken to obtain the calculated height:

$$\text{calculated height} = \text{avg_length_of_femur} * 4$$

And

$$\text{Ratio} = \frac{\text{Actual Height}}{\text{Calculated Height}}$$

5.4 Measurements

The landmarks information stored after capturing Front Pose and Side Pose are used to calculate the measurements. Using the coordinates, distances are calculated and made available to the customers.

```
front_pairs = [('left_wrist', 'left_elbow'), ('left_elbow', 'left_shoulder'),
               ('left_shoulder', 'right_shoulder'), ('right_shoulder', 'right_elbow'),
               ('right_elbow', 'right_wrist'), ('left_shoulder', 'left_hip'),
               ('right_shoulder', 'right_hip'), ('left_hip', 'right_hip'),
               ('left_hip', 'left_knee'), ('right_hip', 'right_knee'),
               ('left_knee', 'left_ankle'), ('right_knee', 'right_ankle')]
side_pairs = [('left_wrist', 'left_elbow'), ('left_elbow', 'left_shoulder'),
               ('left_shoulder', 'left_hip'), ('left_hip', 'left_knee'),
               ('left_knee', 'left_ankle')]
```

Above mentioned pairs are used to calculate the necessary measurements. Coordinates of above listed pairs are extracted from stored landmark information and distance between them is calculated.

5.5 Basic Django

In a traditional data-driven website, a web application waits for HTTP requests from the web browser (or other client). When a request is received the application works out what is needed based on the URL and possibly information in POST data or GET data. Depending on what is required it may then read or write information from a database or perform other tasks required to satisfy the request. The application will then return a response to the web browser, oftened dynamically creating an HTML page for the browser to display by inserting the retrieved data into placeholders in an HTML template.

Django web applications typically group the code that handles each of these steps into separate files:

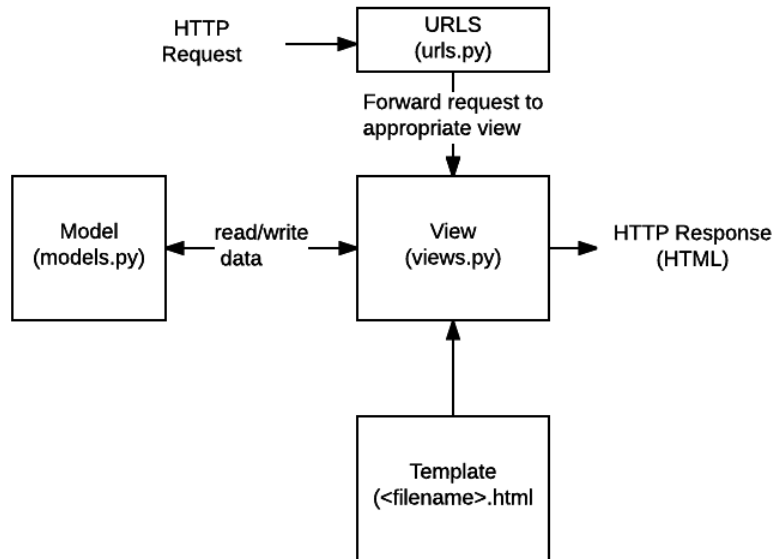


Figure 9: Basic Django

[Source: <https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Introduction>]

- **URLs:** While it is possible to process requests from every single URL via a single function, it is much more maintainable to write a separate view function to handle each resource. A URL mapper is used to redirect HTTP requests to the appropriate view based on the request URL. The URL mapper can also match particular patterns of strings or digits that appear in a URL and pass these to a view function as data.
- **View:** A view is a request handler function, which receives HTTP requests and returns HTTP responses. Views access the data needed to satisfy requests via *models*, and delegate the formatting of the response to *templates*.
- **Models:** Models are Python objects that define the structure of an application's data, and provide mechanisms to manage (add, modify, delete) and query records in the database.
- **Templates:** A template is a text file defining the structure or layout of a file (such as an HTML page), with placeholders used to represent actual content. A *view* can dynamically create an HTML page using an HTML template, populating it with data from a *model*. A template can be used to define the structure of any type of file; it doesn't have to be HTML.

[11]

5.6 Working of webapp

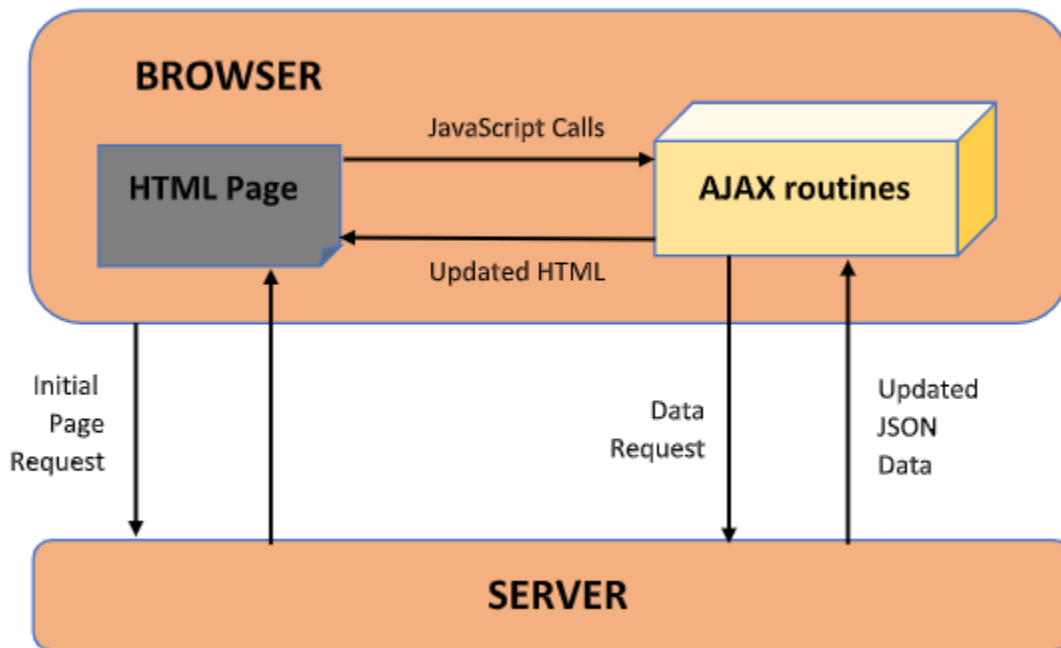


Figure 10: How the webapp works

- **JavaScript Code on the client** – Side/ browser makes a request when an event occurs on the web page. The JavaScript code will generate an XHR object and is sent as a request object to the server. The XHR object contains some JavaScript data/ object.
- The XHR object also contains the URL or name of the call-back function on server.
- **The request is handled by the server with a callback function** – The appropriate view function or callback function handles the request. It will send a success or failure response. Since the request is asynchronous, the rest of the code executes without interruptions.
- At that time, the server processes the request.
- **The response is received as success and failure** – The success response can contain some server data in multiple formats like:
 1. Text Format:
Html Format (This contains HTML elements).
 2. JSON Format:
JSONP Format (It is JSON but only when the response comes from a different domain).
Script (This contains some JavaScript to be changed in page).

3. XML Format

Similarly a failure response can be formatted.

- **The JavaScript will execute according to the response received** – The AJAX routine will now execute according to data from the server. We can change the HTML elements or run some JavaScript. Many things are possible using it. The best example is Facebook's like button.

AJAX has multiple implementations but this is the common flow.

CHAPTER 6

EXPECTED OUTPUT

- Scan within a minute
- Display the accurate measurements
- Display the apparel according to the user measurements

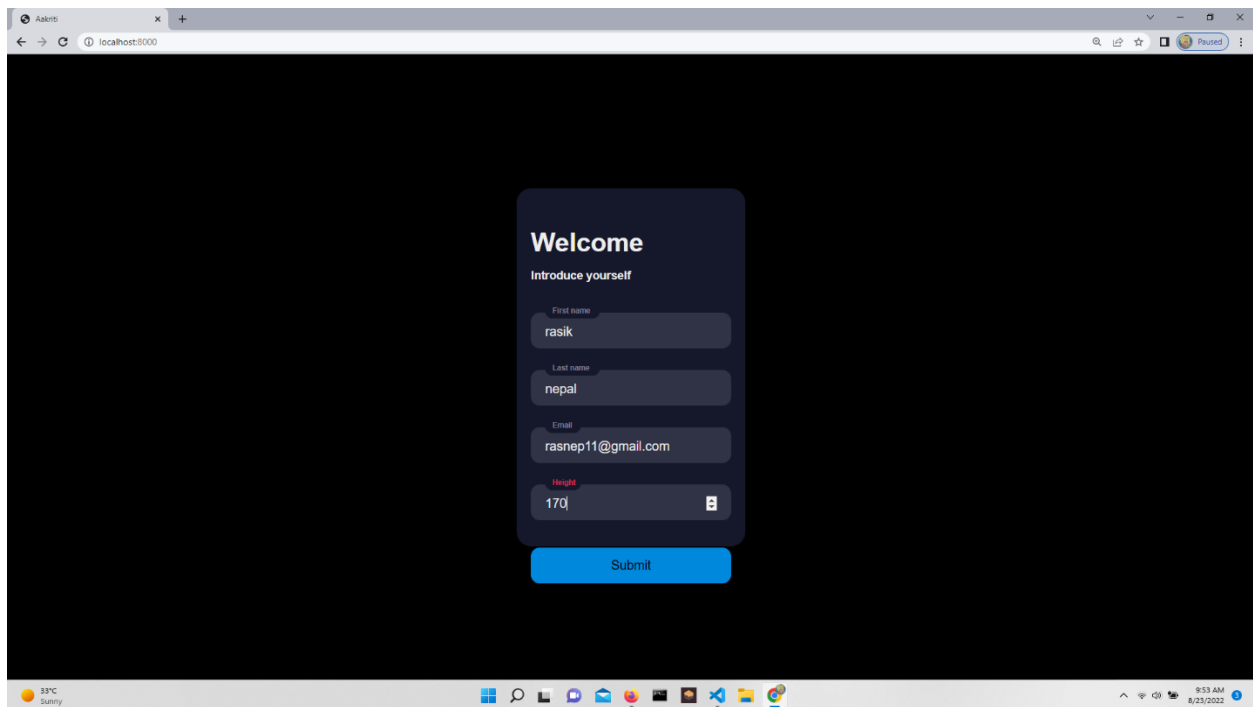
CHAPTER 7

RESULT AND ANALYSIS

7.1 Work Progress

- Design of UI of webapp
- System can now take image from the client webcam, authenticate it and pass the image to the server where the measurements will be taken and passed to the frontend.

Output:



The screenshot displays a web browser window with a dark theme. The main content is a 'Welcome' form with the subtitle 'Introduce yourself'. The form includes four input fields: 'First name' with the value 'rasik', 'Last name' with 'nepal', 'Email' with 'rasnep11@gmail.com', and 'Height' with '170'. A blue 'Submit' button is positioned below the 'Height' field. The browser's address bar shows 'localhost:8000'. The Windows taskbar at the bottom indicates the system time as 9:53 AM on 8/23/2022.

Figure 11: Form to take user info

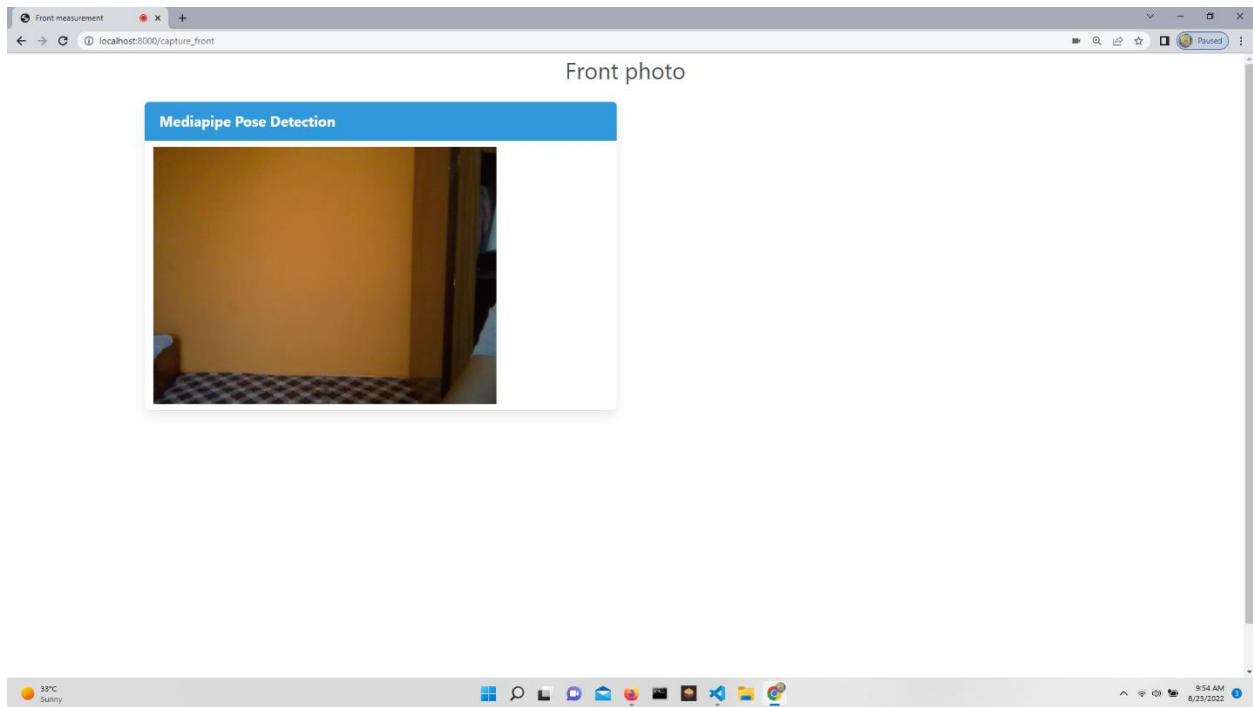


Figure 12: Page to take front photo

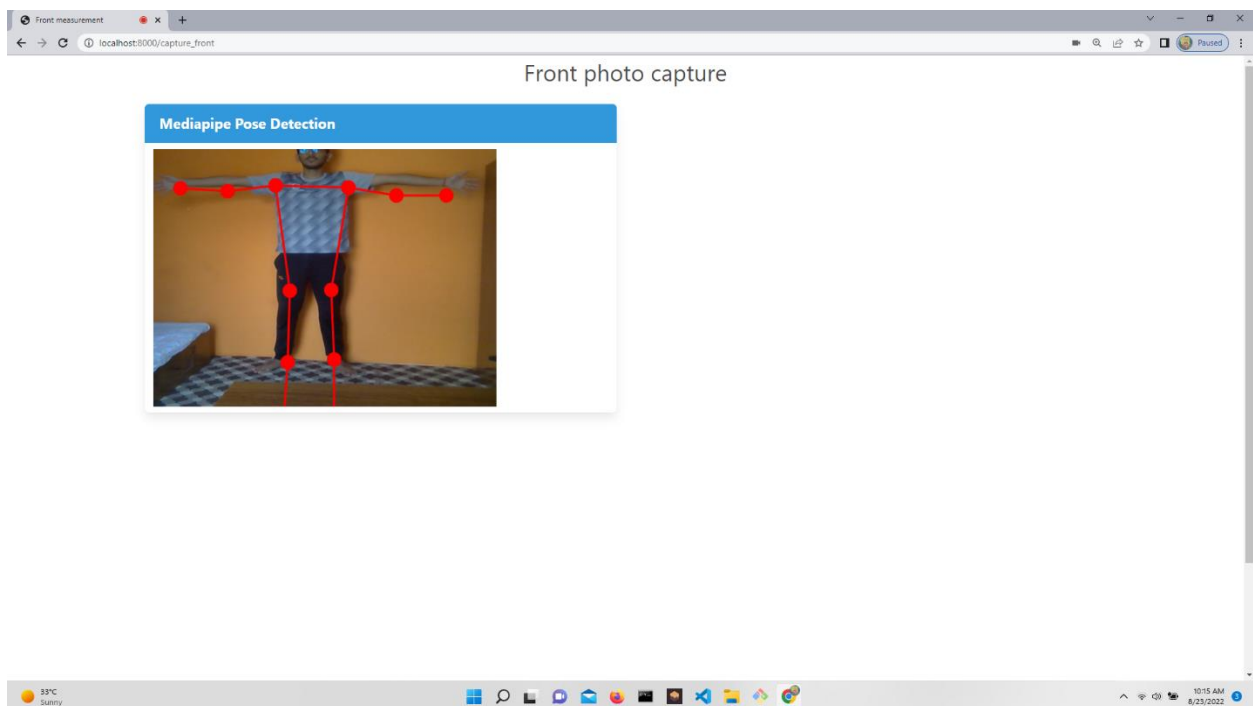


Figure 13: landmark detection from in browser javascript

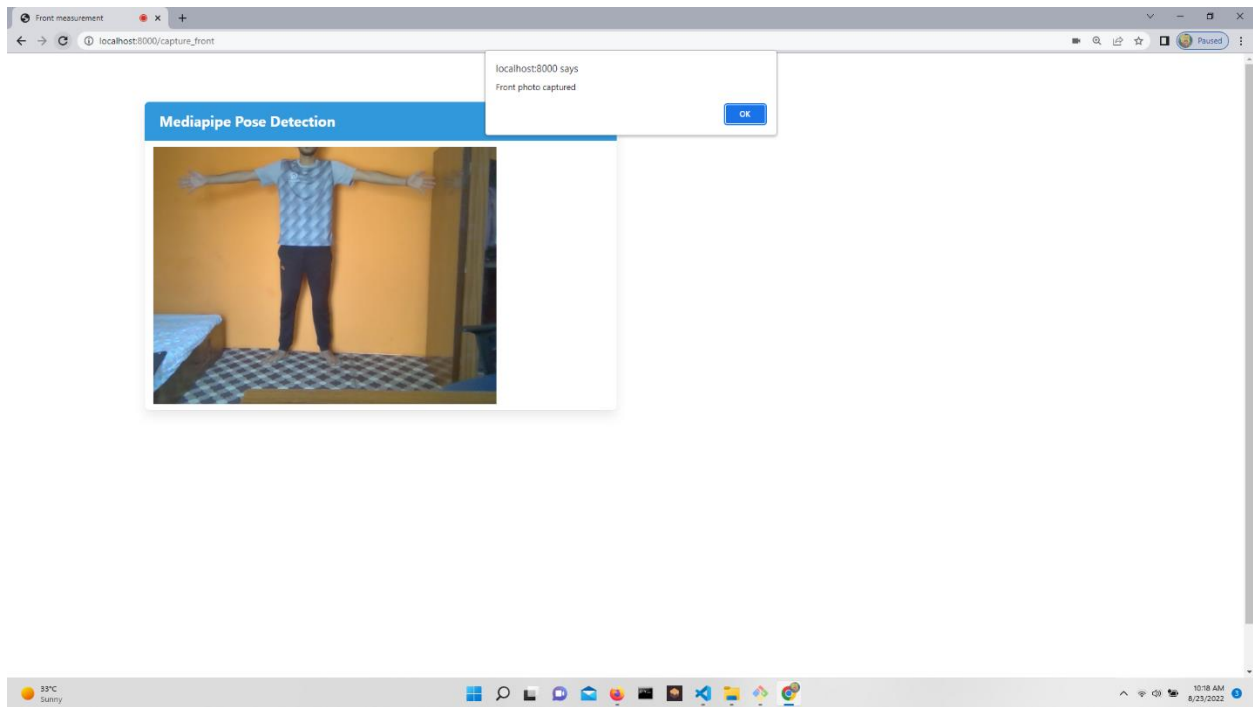


Figure 14: Photo captured after correct pose


```
MINGW64/c/Users/LENOVO/Documents/GitHub/Major-project
LENOVODESKTOP-T733RU4 MINGW64 ~/Documents/GitHub/Major-project (main)
$ python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
August 23, 2022 - 10:19:03
Django version 4.0.6, using settings 'webcamimage.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
[23/Aug/2022 10:19:20] "GET /capture_front HTTP/1.1" 200 2748
[23/Aug/2022 10:19:20] "GET /static/js/FrontPose.js HTTP/1.1" 304 0
INFO: Created TensorFlow Lite XNNPACK delegate for CPU.
{'left_forearm': 28.929659926386627, 'left_upper_arm': 26.497406550187968, 'shoulder_length': 36.84101223067031, 'right_upper_arm': 26.352985367278677, 'right_forearm': 28.59497567545202, 'hip_length': 21.235081054980228, 'left_thigh': 42.46362808937213, 'right_thigh': 42.53637191062787, 'left_lower_leg': 35.784442219630066, 'right_lower_leg': 35.38615396377103}
[23/Aug/2022 10:19:39] "POST /post/ajax/friend HTTP/1.1" 200 400
```

Figure 15: Calculation of measurement

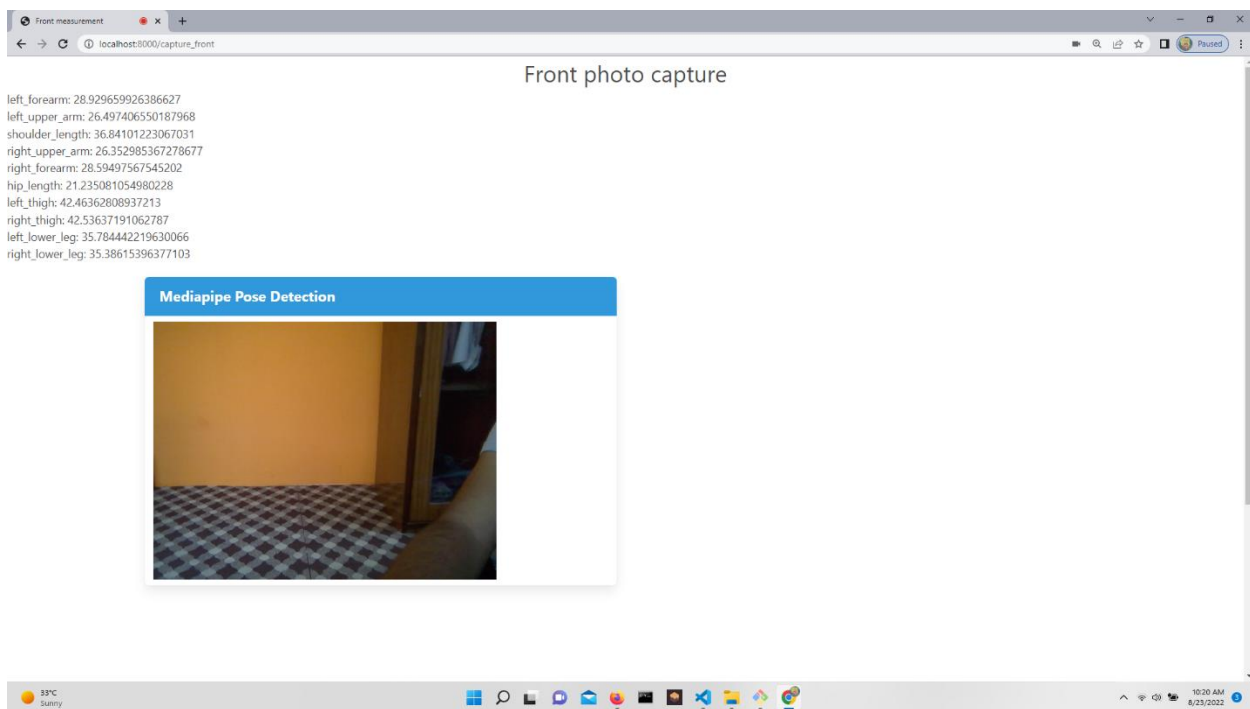


Figure 16: Measurement result on the webpage

7.2 Problems Faced

- Initially, we tried using BodyPix model in tensor.js but it had more percentage error and was difficult to optimize and integrate in the system. So, we had to shift to other models.
- The system is currently not user friendly because of the problem in synchronization among all dependencies.
- It was difficult to calculate slopes for the proper body pose to take measurement.

7.3 Work Remaining

- The improvement in UI is to be done according to designs.
- Algorithm for calculation of circumference for various parts of a body is to be develop.
- API is to be built.

7.4 Time Schedule

S.N.	Tasks	Month 1	Month 2	Month 3	Month 4-7	Month 8
1	Research					
2	Designing					
3	Coding					
4	Testing					
5	Implementation					
6	Documentation					

Table 1: Time Schedule of the project

CHAPTER 8

TOTAL COST

Since our project is totally software dependent and the hardware needed comes already built in on our devices, total economic cost for project is almost equal to zero.

REFERENCES

- [1] D. I. Dewann, B. Chapain, M. P. Jaiswal and D. S. Kumar, "Estimating Human Body Measurement from 2D Images Using Computer Vision," 2014.
- [2] "Main reasons for online shoppers worldwide to return clothes purchased online as of 2021," 2021. [Online]. Available: <https://www.statista.com/statistics/1300981/main-reasons-return-clothes-bought-online/>. [Accessed 21 06 2022].
- [3] K. Gilsenan, "Online Shopping Returns: Everything Retailers Need to Know," 18 December 2018. [Online]. Available: <https://blog.gwi.com/chart-of-the-week/online-shopping-returns/>. [Accessed 22 08 2022].
- [4] Y.-W. L. H.-T. C. S.-Y. F. T.-T. C. H.-T. Chang, "A Dynamic Fitting Room Based on Microsoft Kinect and Augmented Reality Technologies," *International Conference on Human-Computer*, pp. 177-185, 2013.
- [5] P. X. L. L. X. Q. Xiaohui, "Automatic human body feature extraction and personal size," *Proceedings of Journal of Visual Languages and Computing*, vol. 47, pp. 9-18, 2018.
- [6] M. & K. S. Mojarad, "Measuring the main parameters of the human body in images by canny edge detector," 2013.
- [7] R. N. F. F. & R. T. H. Chandra, "R. N. F. F. & R. T. H. Chandra," *10th International Conference on Computer and Automation Engineering*, 2018.
- [8] "Python(Programming Language)," Wikipedia, [Online]. Available: [https://en.wikipedia.org/wiki/Python_\(programming_language\)](https://en.wikipedia.org/wiki/Python_(programming_language)). [Accessed 22 08 2022].
- [9] "Django(web framework)," Wikipedia, [Online]. Available: [https://en.wikipedia.org/wiki/Django_\(web_framework\)](https://en.wikipedia.org/wiki/Django_(web_framework)). [Accessed 22 08 2022].
- [10] "JavaScript," Wikipedia, [Online]. Available: <https://en.wikipedia.org/wiki/JavaScript>. [Accessed 22 08 2022].
- [11] "Django Introduction," mdn web docs_, [Online]. Available: <https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Introduction>. [Accessed 23 08 2022].

- [12] "Pose Tracking Full Body Landmarks," [Online]. Available: https://google.github.io/mediapipe/images/mobile/pose_tracking_full_body_landmarks.png. [Accessed 22 08 2022].