

A PROJECT REPORT
ON
AI-Based Yoga Guru App

Submitted to the Savitribai Phule Pune University in the partial fulfillment
of the requirements for the award of degree
of

BACHELOR OF COMPUTER ENGINEERING

BY

Rachana Deodhar	71706870G
Mugdha Chindhe	71706840E
Abhishek Ranjane	71706616K
Palash Gangamwar	71706965G

PROJECT GROUP NO: 20

DEPARTMENT OF COMPUTER ENGINEERING

Accredited by NAAC

STES
SINHGAD COLLEGE OF ENGINEERING

S.N. 44/1, Vadgaon Bk, Off Sinhgad Road
Pune - 411041

YEAR 2019-20



Sinhgad Institutes

SINHGAD COLLEGE OF ENGINEERING

S.N. 44/1, Vadgaon Bk, Off Sinhgad Road

Pune - 411041

CERTIFICATE

This is to certify that the preliminary project report entitled

”AI-BASED YOGA GURU APP”

Submitted by

Rachana Deodhar 71706870G

Mugdha Chindhe 71706840E

Abhishek Ranjane 71706616K

Palash Gangamwar 71706965G

is a bonafide work carried out and is approved for the partial fulfillment of the requirement of Savitribai Phule Pune University, Pune for the award of the Degree of Bachelor of Computer Engineering.

This project work has not been earlier submitted to any other Institute or University for the award of any degree or diploma.

Place: Pune

Date: 13 April 2020

Prof. G.G. Chiddarwar

Internal Guide

Department of Computer Engineering

Prof. M.P. Wankhade

Head,

Department of Computer Engineering

Dr. S. D. Lokhande

Principal

SCOE, Pune

Acknowledgement

The success and final outcome of this project required a lot of guidance and assistance from many people and we are extremely privileged to have got this all along the completion of our project report. All that we have done is only due to such supervision and assistance and we would not forget to thank them.

We respect and thank Prof. G.G. Chiddarwar, for providing us an opportunity and giving us all support and guidance which made us complete the project report duly. We are extremely thankful to her for providing such a nice support and guidance.

We owe our deep gratitude to Dr. S.D. Lokhande, Principal, Sinhgad College of Engineering and Prof. M.P. Wankhade, HOD Department of Computer Engineering, who took keen interest on our project work and for providing all facilities and every help for smooth progress of project.

We are thankful and fortunate enough to get constant encouragement, support and guidance from all Teaching staffs of Department of Computer Engineering which helped us in successfully completing our project. At last we would like to thank all the unseen authors of various articles on the Internet, helping us become aware of the research currently ongoing in this field and all my colleagues for providing help and support in our work.

Abstract

The importance of yoga is renowned worldwide and its health benefits, which were preached by ancient sages, have stood the test of time. Even though Yoga is becoming preeminent, there are important challenges faced while doing yoga such as performing it with incorrect form, the classes being expensive and the shortage of time in our busy lives. Computer vision techniques exhibit promising solutions for human pose estimation. However, these techniques are seldom applied in the domain of health or exercise, with no literature or projects cited specifically for yoga. In this project, we present a unique solution to the problems faced while doing yoga at home by building an android application that estimates the human pose and provides corrective instructions to the person doing yoga. The app uses PoseNet, an open-sourced Google project, to predict the coordinates and then draw a skeleton based on 17 points on the body such as elbows, knees, etc. It compares these with the correct values, calculated on actual poses using OpenCV and displays the incorrect angles, the accuracy score and dictates the corrections that need to be made. This process completely works on real-time video, on the client-side, which yields faster performance and ensures user privacy. The application also has the added functionality of being operated by voice commands, enabling the user to perform yoga without interruption.

List of Figures

2.1	App Working Overview	9
2.2	Time line chart	13
3.1	Sequence Diagrams	16
3.2	Use-Case Diagram	17
3.3	State Machine Diagram	18
3.4	UML Deployment Diagram	19
3.5	Module Diagram	20

Acronyms

SRS System Requirements Specification

UML Unified Modelling Language

GUI Graphical User Interface

AI Artificial Intelligence

ML Machine Learning

CNN Convolutional Neural Network

Contents

Certificate	i
Acknowledgement	ii
Abstract	iii
List Of Figures	iv
Acronyms	v
1 INTRODUCTION	1
1.1 Background and Basics	1
1.2 Literature Survey	2
1.3 Project Undertaken	6
1.3.1 Problem Definition	6
1.3.2 Scope Statement	6
1.4 Organization of Project Report	6
2 PROJECT PLANNING AND MANAGEMENT	7
2.1 Introduction	7
2.2 Detail System Requirement Specification (SRS)	7
2.2.1 System Overview	7
2.2.2 Functional Requirements	10
2.2.3 Non-Functional Requirements	10
2.2.4 External Interface Requirements	11
2.2.5 Other Requirements	12
2.3 Project Process Modeling	12

2.4	Project Scheduling	13
3	ANALYSIS AND DESIGN	14
3.1	IDEA Matrix	14
3.2	Mathematical Model	15
3.3	Feasibility Analysis	15
3.4	Sequence Diagrams	16
3.5	Use-Case Diagram	17
3.6	State Machine Diagram	18
3.7	UML Deployment Diagram	19
3.8	Module Diagram	20
4	Implementation and Coding	21
4.1	Introduction	21
4.2	21
4.2.1	Operational Details	21
4.2.2	Major Classes	23
4.2.3	Code Listing	24
4.2.4	Screen shots	28
5	Testing	29
5.1	Unit Testing	29
5.2	Acceptance Testing	30
6	Results and Discussion	31
6.1	Main GUI snapshots	31
6.2	Discussions	36
7	Conclusion	37
8	Future Work	38
	References	39

INTRODUCTION

1.1 Background and Basics

Over the last few decades, there has been an upsurge in the prevalence of yoga. Yoga is not a religion, it's a way of living that aims towards a healthy mind in a healthy body. Medical professionals and celebrities are also adopting and recommending the regular practice of yoga due to its various benefits. Looking from the outside in, yoga may seem like just another form of physical activity. While it does offer many of the benefits of exercise, it does come with a few subtleties that make it hard to learn on your own. Improper training postures can cause serious harm to the muscles and ligaments of the body. Yoga is a path towards total harmony of body, mind, and spirit. Yoga is not merely a form of exercise for the body. It is an ancient wisdom - for a healthier, happier, and more peaceful way of living - which ultimately leads to union with the Self. The growing popularity of yoga is logical and not accidental it's all about the physical and psychological benefits that yoga brings. First of all, it's beneficial to physical health. But yoga classes bring another benefit psychological. At the moment, people's main priority in practicing yoga is shifting toward self-development, self-knowledge, and spiritual practices. It's worth taking this into account in yoga mobile app.

People are constantly trying to achieve the challenging and impressive asanas that they see on social media, and it's easy to forget that the ancient practice of yoga is about so much more than contorting your body into unique physical shapes. Even in the simplest of supervised beginner classes, all yogis are still at risk of developing some of the most common yoga injuries. Yoga is now one of the most popular forms of exercise. Not only can it help you get fit, improve flexibility, and build strength, but there are also hidden health benefits, like reviving your immune system and fighting migraines. As yoga has become more popular, though, the rate of yoga injuries has also risen. Injuries of the neck, wrist, shoulder, elbow, lower back are common injuries can happen due to the wrong position of yoga. So, our yoga guru app is the best solution to perform the correct position of yoga, avoiding injuries and stay fit.

1.2 Literature Survey

Table 1.1: Literature Survey.

Sr.No	Title	Description	Author
1	Machine learning gesture analysis of yoga for exergame development.	Machine learning algorithm in a basic computer video exergame can assess yoga skill acquisition in targeted select populations as a means to promote healthy physical activity.	Paula Pullen, William Seffens.
2	3D Human Pose Estimation with Relational Networks	Each pair of different body parts generates features, and the average of the features from all the pairs are used for 3D pose estimation.	Sungheon Park, Nojun Kwak.
3	Human Pose Co-Estimation and Applications	PCE leads to better pose estimation in such images, and it learns meaningful prototypes which can be used as priors for pose estimation in novel images.	Marcin Eichner, Vittorio Ferrari.
4	DeepPose: Human Pose Estimation via Deep Neural Networks	Method for human pose estimation based on Deep Neural Networks (DNNs). The pose estimation is formulated as a DNN-based regression problem towards body joints.	Alexander Toshev, Christian Szegedy.

Table 1.2: Literature Survey.

Sr.No	Title	Description	Author
5	Efficient Object Localization Using Convolutional Networks	Recent state-of-the-art performance on human-body pose estimation has been achieved with Deep Convolutional Networks (ConvNets). This model is trained in cascade with a state-of-the-art ConvNet model to achieve improved accuracy in human joint location estimation.	Jonathan Tompson, Ross Goroshin, Arjun Jain.
6	Convolutional Pose Machines	Convolutional pose machines provide an end-to-end architecture for tackling structured prediction problems in computer vision without the need for graphical-model style inference.	Shih-En Wei, Varun Ramakrishna, Takeo Kanade
7	Deep High-Resolution Representation Learning for Human Pose Estimation	the human pose estimation problem with a focus on learning reliable high resolution representations. Most existing methods recover high-resolution representations from low-resolution representations produced by a high-to-low resolution network	Ke Sun, Bin Xiao, Dong Liu

Table 1.3: Literature Survey.

Sr.No	Title	Description	Author
8.	Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation	In this work, they propose to combine the advantages from both methods. Specifically, our proposed model, DeepLabv3+, extends DeepLabv3 by adding a simple yet effective decoder module to refine the segmentation results.	Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, Hartwig Adam
9.	MonoCap: Monocular Human Motion Capture using a CNN Coupled with a Geometric Prior	This paper is concerned with the challenge of recovering the 3D full-body human pose from a markerless monocular RGB image sequence. A 3D human pose estimation framework from a monocular image or video has been presented.	Xiaowei Zhou, Menglong Zhu, Georgios Pavlakos, Spyridon Leonardos
10.	Yoga Posture Recognition By Detecting Human Joint Points In Real Time Using Microsoft Kinect	In this paper, we have proposed a system to recognize three major yoga pose by detecting of human joint points using Microsoft Kinect. We have detected the yoga poses considering above 97% accuracy in every angle between different body parts.	Muhammad Usama Islam, Hasan Mahmud, Faisal Bin Ashraf, Iqbal Hossain

Table 1.4: Literature Survey.

Sr.No	Title	Description	Author
11.	Recognition of Yoga Poses through an Interactive System with Kinect device	This paper has presented an interactive system that uses Kinect v2 for 6 Yoga poses recognition with command voices to visualize the instructions and pictures about the poses to be performance.	Edwin W. Trejo, Peijiang Yuan
12.	Realtime Multi-person 2D Pose Estimation Using Part Affinity Fields	It presents an approach to efficiently detect the 2D pose of multiple people in an image. The approach uses a nonparametric representation, which we refer to as Part Affinity Fields (PAFs), to learn to associate body parts with individuals in the image.	Zhe Cao, Tomas Simon, Shih-En Wei, Yaser Sheikh
13.	PersonLab: Person Pose Estimation and Instance Segmentation	It present a box-free bottom-up approach for the tasks of pose estimation and instance segmentation of people in multi-person images using an efficient single-shot model.	George Papandreou, Tyler Zhu, Liang-Chieh Chen, Jonathan Tompson, Kevin Murphy

1.3 Project Undertaken

1.3.1 Problem Definition

To develop an AI based yoga instructor app, that estimates and corrects poses in real time.

1.3.2 Scope Statement

To help people practice Yoga more effectively. Also, Make people understand and replicate Yoga poses more effectively. This project will increase awareness of a healthy lifestyle through practicing Yoga.

1.4 Organization of Project Report

There are a total of 6 chapters. Chapter 2 presents project planning and management which contain system required specifications. It has requirements at different levels like functional, non-functional, deployment, external interface. It also describes project process mapping, project scheduling. Chapter 3 provides a mathematical model, architecture diagram, UML diagram and analysis, design of the product. Chapter 4 summarizes the implementation, operational part, and coding part. Chapter 5 presents testing and it's test cases. Chapter 6 shows results and discussion. Chapters 7 and 8 presents the conclusion and future work respectively.

PROJECT PLANNING AND MANAGEMENT

This chapter covers the project planning and management details. It also covers System Requirement specifications. SRS is considered as the base for the effort estimations and project scheduling.

2.1 Introduction

This chapter covers the project planning and management details. It also covers System Requirement specifications. SRS is considered as the base for the effort estimations and project scheduling

2.2 Detail System Requirement Specification (SRS)

2.2.1 System Overview

- The system is an android app. The main function of this app is to guide users to perform yoga poses more effectively on your own without getting any physical injury. Because there are so many users who want to do yoga but they don't have that much time to go to yoga classes and pay money for it. Also, there is a risk of some physical injury that can happen due to a wrongly performed yoga pose. [16]
- The system works with the help of some technologies or frameworks like PoseNet, OpenCV, android studio.
- PoseNet is a vision model that can be used to estimate the pose of a person in an image or video by estimating where key body joints are. [15]
- PoseNet is used for processing user videos. It helps for pose estimation of real-time user video. Using PoseNet, we can find the coordinates of 17 key points of the body which are then used to find angles between those respective points. [15]
- OpenCV is a library of programming functions mainly aimed at real-time computer vision. The images of the 4 yoga poses that our app offers for practice are processed using OpenCV and the set of 17 key points for each of them are found out. A skeleton formed by joining the 17 key points is displayed. [8]
- OpenCV is used for processing target image i.e. to find coordinates of the target image,

with the help of this we can find target angles to which we will compare angles of user's yoga pose and total accuracy will be found. Only if the accuracy is less than 80%, the angles will be displayed on screen for the user to correct them. [8]

- Android studio is used to make a user-friendly android app. Our app also uses voice commands. Google Text-to-Speech has been used to develop them in the app. The user can choose the pose to practice by just saying first, second, third, or fourth. Also once the user's pose is accurate the app tells the user that your pose is correct through voice command.

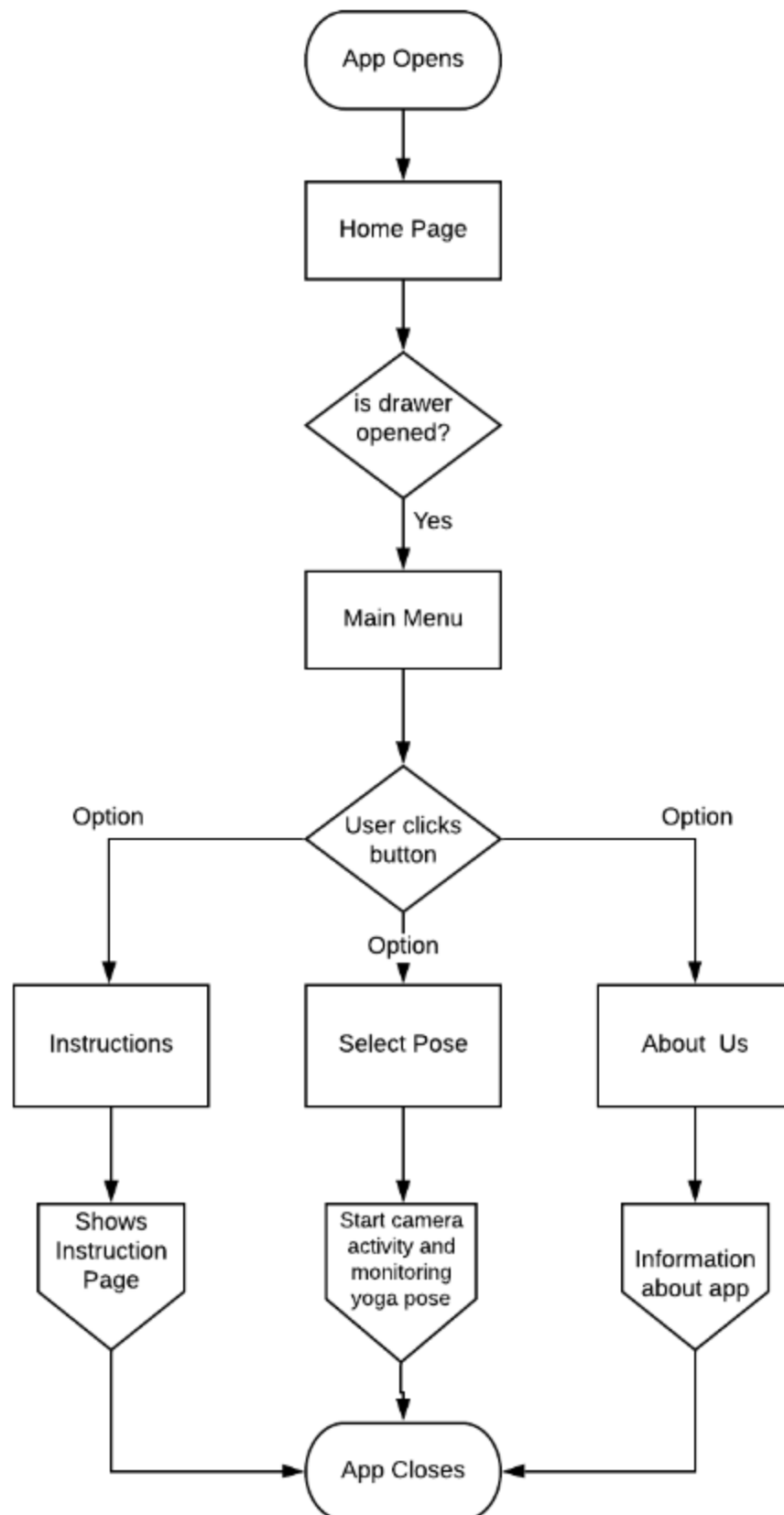


Figure 2.1: App Working Overview

2.2.2 Functional Requirements

- User will select yoga asana image which he/she wishes to imitate.
- Permission will be asked to start the mobile camera and if given the user video will start.
- The real-time poses of the user are then processed using PoseNet.
- The selected image from database is also processed using the same algorithm.
- The system compares both these poses and prompts the user whether he/she has successfully imitated the pose.

2.2.3 Non-Functional Requirements

2.2.3.1 Performance Requirements

- The project will require a good quality device to run because of the heavy client side inference processes.
- There should not be any lag in the application during the time user is practicing the pose and the angles are being displayed.

2.2.3.2 Safety Requirements

- User should hold the yoga pose for 5 seconds and then check with the app before continuing to avoid injuries.
- The results should be displayed in real-time, with minimal lag.

2.2.3.3 Security Requirements

- The inferences for the user's real-time video will be enclosed on the client-side, ensuring user privacy.

2.2.3.4 System Security

- User's real time yoga video will only be seen by user.

2.2.4 External Interface Requirements

2.2.4.1 User Interfaces

- The product will run as an android application. The product will have good usability and intuitive controls and navigation. The application will have a dashboard and an interactive guide.

2.2.4.2 Software Interface

- Front-end, back-end interface – The back-end framework, such as OpenCV, PoseNet, will host the web application running on the client side. The requests sent from the front-end will be used by the back-end framework to handle user authentication, navigation and various functions.
- Application interfaces- The android application makes API calls to PoseNet functions for extracting coordinates of essential points.

2.2.5 Other Requirements

Software Requirements

1. PoseNet
2. Tensorflow Lite
3. OpenCV
4. Android Studio

Important Packages:

- a) Kotlin

Hardware Requirements

1. Node component (PC/Laptop)
2. Internet connection
3. Smartphone

2.3 Project Process Modeling

In this, the agile process model is followed. The agile process model refers to a software development approach based on iterative development. Agile methods break tasks into smaller iterations, or parts do not directly involve long term planning. The project scope and requirements are laid down at the beginning of the development process. Plans regarding the number of iterations, the duration and the scope of each iteration are clearly defined in advance. The division of the entire project into smaller parts helps to minimize the project risk and to reduce the overall project delivery time requirements. Each iteration involves a team working through a full software development life cycle including planning, requirements analysis, design, coding, and testing.

2.4 Project Scheduling

A timeline is a type of chart which visually shows a series of events in chronological order over a linear timescale. The power of a timeline is that it is graphical, which makes it easy to understand critical milestones, such as the progress of a project schedule.

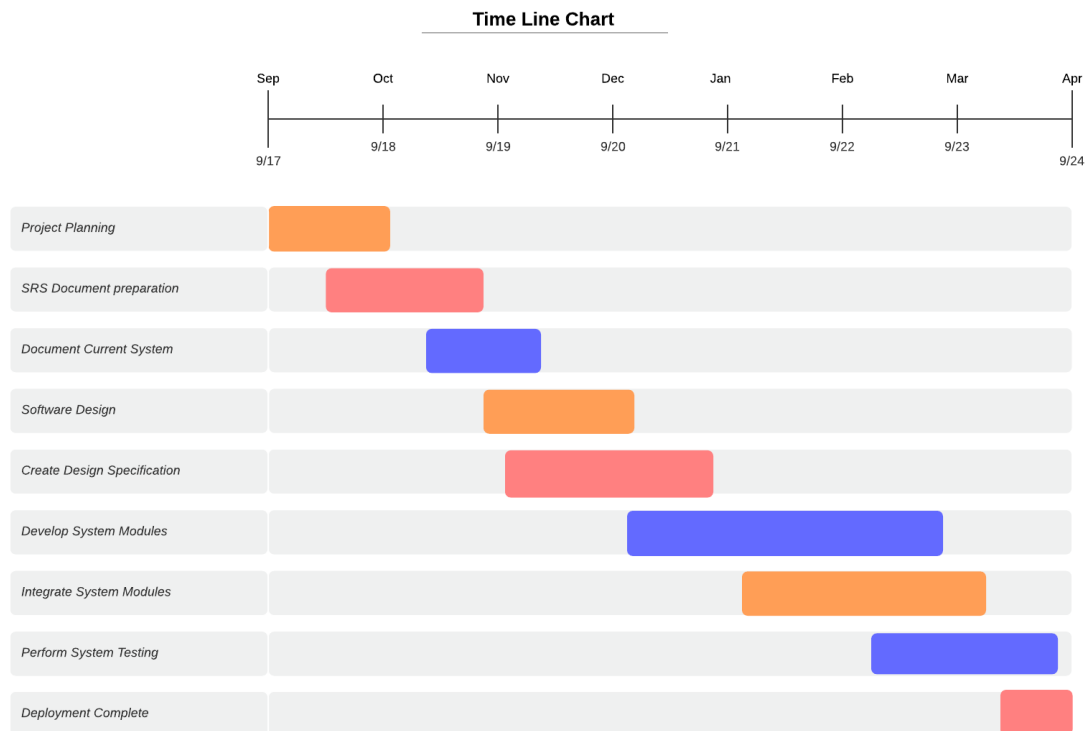


Figure 2.2: Time line chart

ANALYSIS AND DESIGN


3.1 IDEA Matrix

Table 3.1: IDEA Matrix

IDEA	DELIVERABLE	PARAMETER AFFECTED
Increase	Increase number of people practicing yoga everyday	People
Improve	Improve people postures and forms correctly	Position of body
Innovate	Innovate the way people perform yoga	People
Discover	Help you discover the health benefits of yoga	Health of person
Deliver	Deliver more efficient and a better way of performing yoga	People
Decrease	1. Decrease the problems and issues faced due to wrong practicing of yoga	Person
Educate	1. Educate project members 2. Educate people to perform yoga in accurate manner	1. Project member 2. People
Evaluate	1. Continuous monitoring and showing position angle. 2. Real time analysis	Analysis
Eliminate	1. Eliminate physical injuries usually occurring when yoga poses go wrong.	People
Advance	There is no existing system focusing on yoga	System
Actualize	Realization of accurate pose of yoga	Person
Advertise	Avoid problems due to wrong postures	Person

3.2 Mathematical Model

The main purpose of our android app is to show the actual and target angle. To find angles between coordinates a mathematical formula is used which shown below. [16] These two



$$\text{Angle_B} = \text{Math.acos} \left(\frac{(\text{Math.pow}(\text{len}(b, c), 2) + \text{Math.pow}(\text{len}(a, b), 2) - \text{Math.pow}(\text{len}(a, c), 2))}{2 * \text{len}(b, c) * \text{len}(a, b)} \right) * 180 / \text{Math.PI}$$

lines AB and BC are like human body parts, and we are finding the angle between those two lines. This formula is the same as the formula used for finding the angle between two straight lines. PoseNet, a famous machine learning model, acts as a mathematical model for real-time human pose estimation. It uses poseNet() function which takes some parameters like video, type and callback for estimation. [16]

3.3 Feasibility Analysis

The importance of a feasibility study is based on organizational desire to “get it right” before committing resources, time, or budget. A feasibility study might uncover new ideas that could completely change a project’s scope. It’s best to make these determinations in advance, rather than to jump in and to learn that the project won’t work. Many people wish to go to yoga classes or gym but may not have time or money to do so, for these people our ”AI-based yoga guru app” is the best solution. So it is technically feasible as well as economically justifiable. Also, it is legally feasible i.e. it does not conflict with zoning laws, data protection acts or social media laws.

3.4 Sequence Diagrams

A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place.

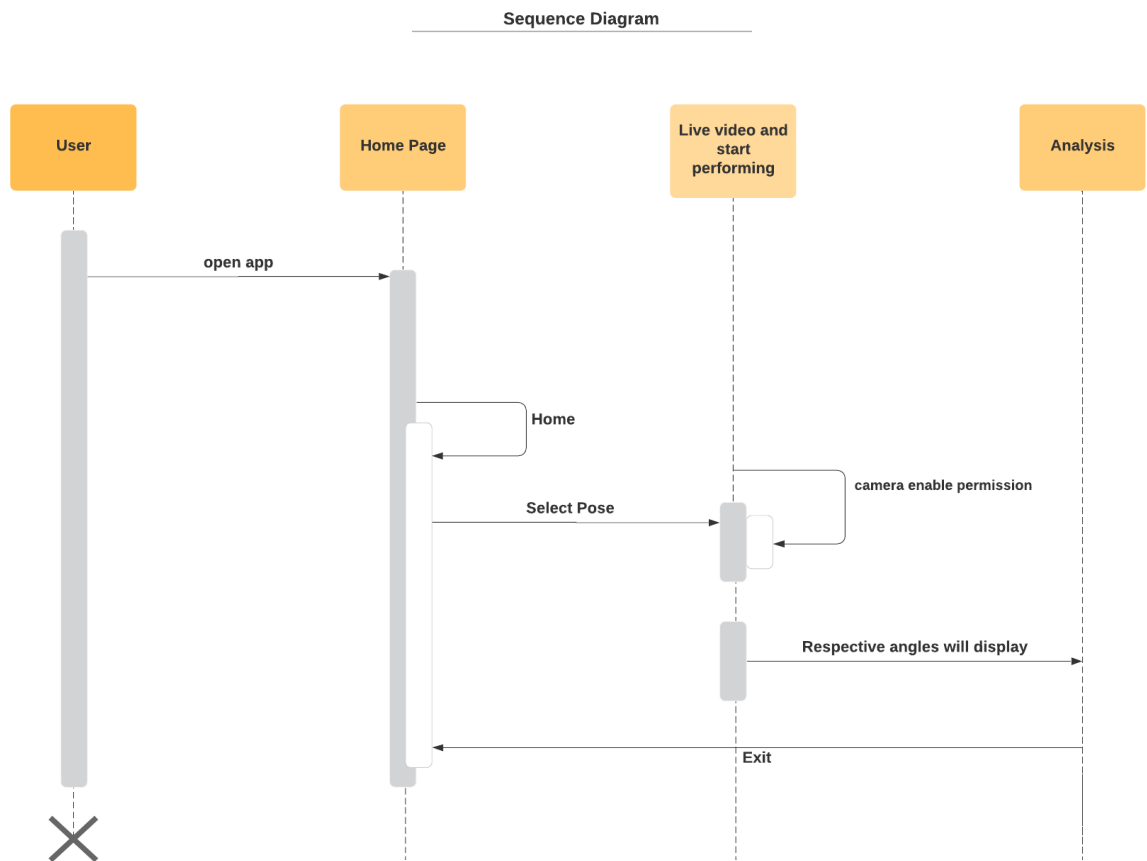


Figure 3.1: Sequence Diagrams

3.5 Use-Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved.

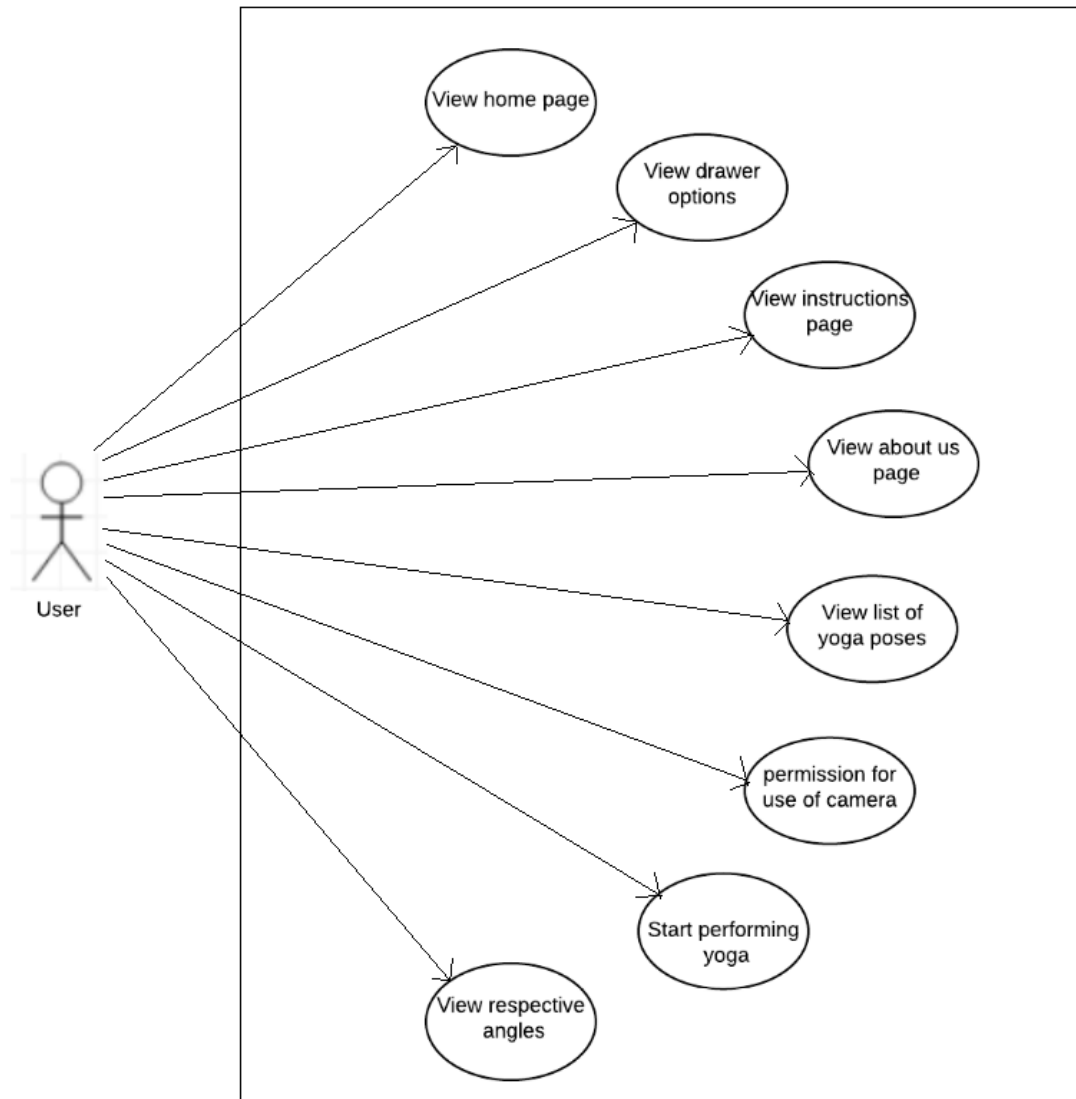


Figure 3.2: Use-Case Diagram

3.6 State Machine Diagram

State-transition diagrams describe all of the states that an object can have, the events under which an object changes state (transitions), the conditions that must be fulfilled before the transition will occur (guards), and the activities undertaken during the life of an object (actions).

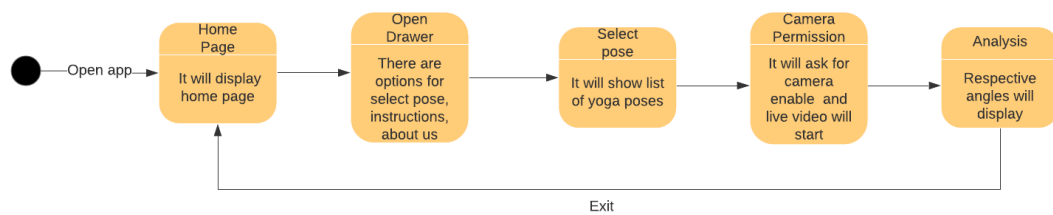


Figure 3.3: State Machine Diagram

3.7 UML Deployment Diagram

A deployment diagram is a UML diagram type that shows the execution architecture of a system, including nodes such as hardware or software execution environments, and the middle ware connecting them.

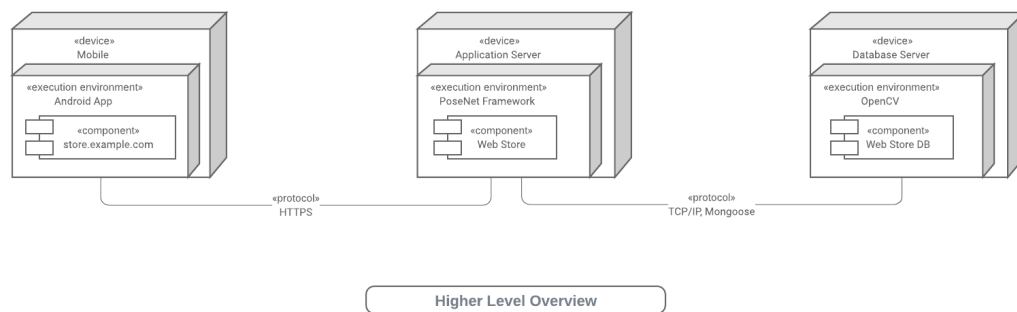


Figure 3.4: UML Deployment Diagram

3.8 Module Diagram

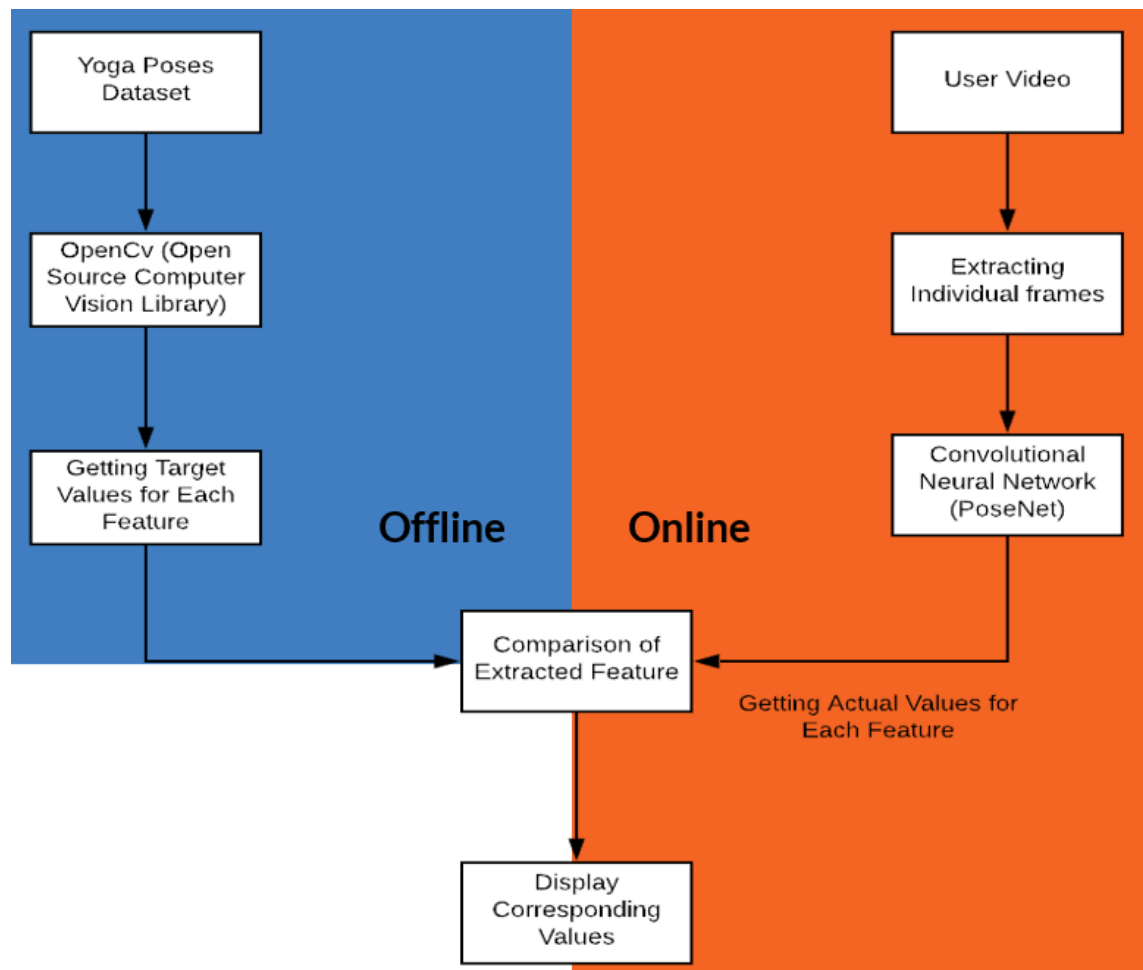


Figure 3.5: Module Diagram

Implementation and Coding

4.1 Introduction

This chapter covers the role of various subsystems/modules/classes along with implementation details listing of the code for the major functionalities

4.2

4.2.1 Operational Details

The system is an android app. The system's module divided into two parts offline and online. The online part uses PoseNet for processing user video while the offline uses OpenCV for processing images. Offline part works for pose estimation of yoga pose dataset i.e. images. The system uses OpenCV for the estimation of the yoga pose image. Google Text-to-Speech is used in android studio for voice recommendation part.

PoseNet Module

- PoseNet is a vision model that can be used to estimate the pose of a person in an image or video by estimating where key body joints are. PoseNet can be used to estimate either a single pose or multiple poses, meaning there is a version of the algorithm that can detect only one person in an image/video and one version that can detect multiple persons in an image/video. [15]
- The single-pose estimation algorithm is the simpler and faster of the two. Its ideal use case is for when there is only one person centered in an input image or video. The single-pose estimation algorithm takes input like image element, scale factor, flip horizontal, output stride. [15]
- Input image element is an html element that contains an image to predict poses for, such as a video or image tag.
- Image scale factor is a number between 0.2 and 1. Defaults to 0.50. Set this number lower to scale down the image and increase the speed when feeding through the network at the cost of accuracy. [16]

- Flip horizontal is defaults to false. This should be set to true for videos where the video is by default flipped horizontally (i.e. a webcam), and you want the poses to be returned in the proper orientation.
- Output stride — Must be 32, 16, or 8. Defaults to 16. Internally, this parameter affects the height and width of the layers in the neural network. [15]
- The outputs for the single-pose estimation algorithm contain two things :
 1. A pose, containing both a pose confidence score and an array of 17 keypoints.
 2. Each keypoint contains a keypoint position and a keypoint confidence score. Again, all the keypoint positions have x and y coordinates in the input image space, and can be mapped directly onto the image. [15]

OpenCV Module

- OpenCV is the huge open-source library for the computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today's systems. [10]
- By using it, one can process images and videos to identify objects, faces, or even handwriting of a human.
- When it integrated with various libraries, such as Numpy, python is capable of processing the OpenCV array structure for analysis.
- OpenCV is a library of programming functions mainly aimed at real-time computer vision. When the user selects a desired image that he wants to imitate, the selected image is processed using Opencv and a set of 17 key points are stored in the system. A skeleton formed by joining the 17 key points is displayed. [10]
- Yoga Poses Dataset is a set of different Yoga pose images. It is useful to find a target angle.
- The target values for each feature gets calculated.

Google Text-to-Speech

- Google Text-to-Speech (GTTS) is a screen reader application developed by Google for its Android operating system. It powers applications to read aloud (speak) the text on the screen with support for many languages. [16]
- It is a great piece of technology that was developed to help individuals with visual im-

pairments. However, device manufacturers these days enable text-to-speech Android that allows books to be read out loud and new languages to be learned. [16]

- Our application also uses voice commands. Google Text-to-Speech has been used to develop them in the app. The user can choose the pose to practice by just saying first, second, third or fourth. Also once the user's pose is accurate the app tells the user that your pose is correct through voice command. [14]
- Speech recognition is designed to listen commands for a brief moment and deactivate on its own. We wanted to mimic the "OK Google" recognition pattern, hence this library is added.
- The recognizer is managed on its own via the `KontinuousRecognitionManager` and exposes a few utility methods such as `startRecognition`, `stopRecognition`, `cancelRecognition` and `destroyRecognizer`.
- The recognizer will be listening and be respawned all the time once you call `startRecognition` and until you call `stopRecognition`.

4.2.2 Major Classes

PosenetActivity : Responsible for processing the users video and extracting coordinates for the 17 points.

CameraActivity : Responsible for taking in the users video and giving it to the posenet for further processing. [11]

4.2.3 Code Listing

To calculate angle using PoseNet of user's video and using OpenCV of target image :

```

rhx=person.keyPoints[8].position.x
rhy=person.keyPoints[8].position.y
rkx=person.keyPoints[6].position.x
rky=person.keyPoints[6].position.y
rax=person.keyPoints[12].position.x
ray=person.keyPoints[12].position.y

x1 = Math.sqrt(Math.pow((((this.rhx - this.rkx).toDouble()), 2.0) +
    Math.pow((((this.rhy - this.rky).toDouble()),2.0)).toInt()
y1 = Math.sqrt(Math.pow((((this.rhx - this.rax).toDouble()), 2.0) +
    Math.pow((((this.rhy - this.ray).toDouble()),2.0)).toInt()
z1 = Math.sqrt(Math.pow((((this.rkx - this.rax).toDouble()), 2.0) +
    Math.pow((((this.rky - this.ray).toDouble()),2.0)).toInt()

resultangle = Math.round((Math.acos((Math.pow(this.z1.toDouble(), 2.0)+
    Math.pow(this.x1.toDouble(),2.0)-Math.pow(this.y1.toDouble(),
    2.0))/(2*this.z2*this.x2)))*180/Math.PI).toInt()

```

```

lsx=points[6][0]
lsy=points[6][1]
lex=points[5][0]
ley=points[5][1]
lwx=points[11][0]
lwy=points[11][1]

p12 = math.sqrt(math.pow((lsx - lex),2) + math.pow((lsy - ley),2));
p13 = math.sqrt(math.pow((lsx - lwx),2) + math.pow((lsy - lwy),2));
p23 = math.sqrt(math.pow((lex - lwx),2) + math.pow((ley - lwy),2));

resultAngle = round(math.degrees(math.acos(((math.pow(p23, 2)) +
    (math.pow(p12, 2)) - (math.pow(p13, 2))) / (2 * p23 * p12)))) ;

```


Three main functions performed by poseNet [17]

- 1.To draw a line on a canvas
- 2.Draws a pose skeleton by looking up all adjacent keypoints/joints
- 3.Draw pose keypoints onto a canvas [17]

```
function drawSegment([ay, ax], [by, bx], color, scale, ctx) {
  ctx.beginPath();
  ctx.moveTo(ax * scale, ay * scale);
  ctx.lineTo(bx * scale, by * scale);
  ctx.lineWidth = lineWidth;
  ctx.strokeStyle = color;
  ctx.stroke();
}

function drawSkeleton(keypoints, minConfidence, ctx, scale = 1) {
  const adjacentKeyPoints = posenet.getAdjacentKeyPoints(keypoints, minConfidence);
  adjacentKeyPoints.forEach((keypoints) => {
    drawSegment(toTuple(keypoints[0].position),
      toTuple(keypoints[1].position), color, scale, ctx);
  });
}

function drawKeypoints(keypoints, minConfidence, ctx, scale = 1) {
  for (let i = 0; i < keypoints.length; i++) {
    const keypoint = keypoints[i];

    if (keypoint.score < minConfidence) {
      continue;
    }

    const { y, x } = keypoint.position;
    ctx.beginPath();
    ctx.arc(x * scale, y * scale, 3, 0, 2 * Math.PI);
    ctx.fillStyle = color;
    ctx.fill();
  }
}
```

Opencv applied on image

```
MODE = "COCO"

if MODE is "COCO":
  protoFile = "pose/coco/pose_deploy_linevec.prototxt"
  weightsFile = "pose/coco/pose_iter_440000.caffemodel"
  nPoints = 18
  POSE_PAIRS = [ [1,0],[1,2],[1,5],[2,3],[3,4],[5,6],[6,7],[1,8],[8,9],
    [9,10],[1,11],[11,12],[12,13],[0,14],[0,15],[14,16],[15,17]]

elif MODE is "MPI" :
  protoFile = "pose/mpi/pose_deploy_linevec_faster_4_stages.prototxt"
  weightsFile = "pose/mpi/pose_iter_160000.caffemodel"
  nPoints = 15
  POSE_PAIRS = [[0,1], [1,2], [2,3], [3,4], [1,5], [5,6], [6,7], [1,14],
    [14,8], [8,9], [9,10], [14,11], [11,12], [12,13] ]
```

```
frame = cv2.imread("D:\BE_project\OpenPose\tadasana.jpg")
frameCopy = np.copy(frame)
frameWidth = frame.shape[1]
frameHeight = frame.shape[0]
threshold = 0.1
net = cv2.dnn.readNetFromCaffe(protoFile, weightsFile)

t = time.time()
# input image dimensions for the network
inWidth = 368
inHeight = 368
inpBlob = cv2.dnn.blobFromImage(frame, 1.0 / 255, (inWidth, inHeight),
                                (0, 0, 0), swapRB=False, crop=False)

net.setInput(inpBlob)

output = net.forward()
print("time taken by network : {:.3f}".format(time.time() - t))

H = output.shape[2]
W = output.shape[3]

# Empty list to store the detected keypoints
points = []

for i in range(nPoints):
    # confidence map of corresponding body's part.
    probMap = output[0, i, :, :]

    # Find global maxima of the probMap.
    minVal, prob, minLoc, point = cv2.minMaxLoc(probMap)
```

```
# Scale the point to fit on the original image
x = (frameWidth * point[0]) / W
y = (frameHeight * point[1]) / H

if prob > threshold :
    cv2.circle(frameCopy, (int(x), int(y)), 8, (0, 255, 255),
                thickness=-1, lineType=cv2.FILLED)
    cv2.putText(frameCopy, "{}".format(i), (int(x), int(y)),
                cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2, lineType=cv2.LINE_AA)

    # Add the point to the list if the probability is greater than the threshold
    points.append((int(x), int(y)))
else :
    points.append(None)

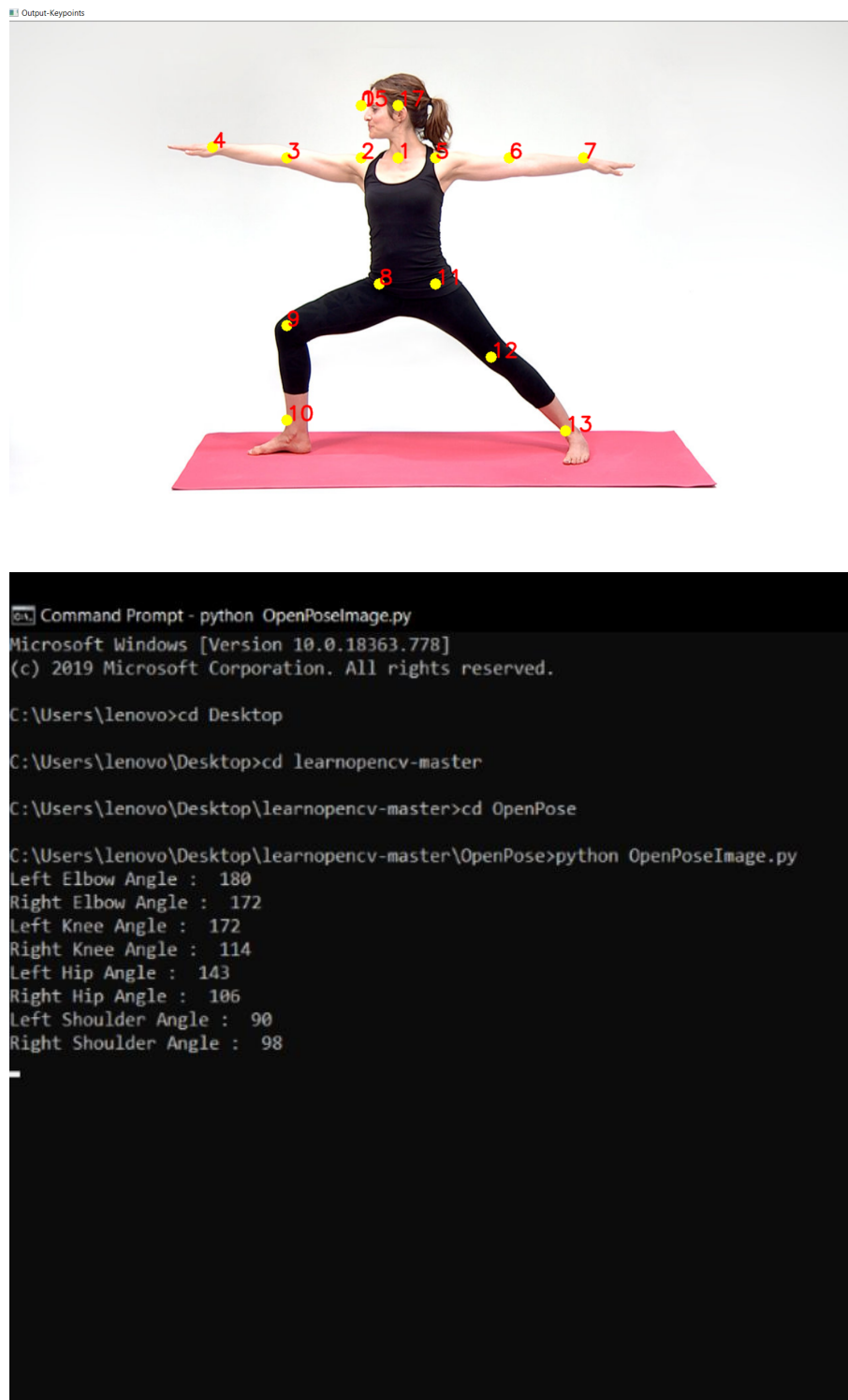
# Draw Skeleton
for pair in POSE_PAIRS:
    partA = pair[0]
    partB = pair[1]

    if points[partA] and points[partB]:
        cv2.line(frame, points[partA], points[partB], (0, 255, 255), 2)
        cv2.circle(frame, points[partA], 8, (0, 0, 255), thickness=-1,
                    lineType=cv2.FILLED)

cv2.imshow('Output-Keypoints', frameCopy)
print('Left shoulder x :', points[5][0])
print('Left shoulder y :', points[5][1])
print('Left elbow x :', points[6][0])
print('Left elbow y :', points[6][1])
print('Left wrist x :', points[7][0])
print('Left wrist y :', points[7][1])
```

4.2.4 Screen shots

Screenshot of result after running OpenCV algorithm



Testing

5.1 Unit Testing

It is technique using which individual modules are tested to determine if there are any issues by the developer himself

Test Scenario	Input	Expected Result
Pose selection module	Text/Image	Successful selection of pose
PoseNet module	Real time video	Body pose detection with skeleton
Body pose angles detection	Real time video	Successful angle calculation for 17 key points.
Pose matching	Real time video/images	Successful correction of pose
Estimation of correct pose	Body pose angles	Correctness of pose in percentage
Test working of camera	Camera input	Motion picture recording

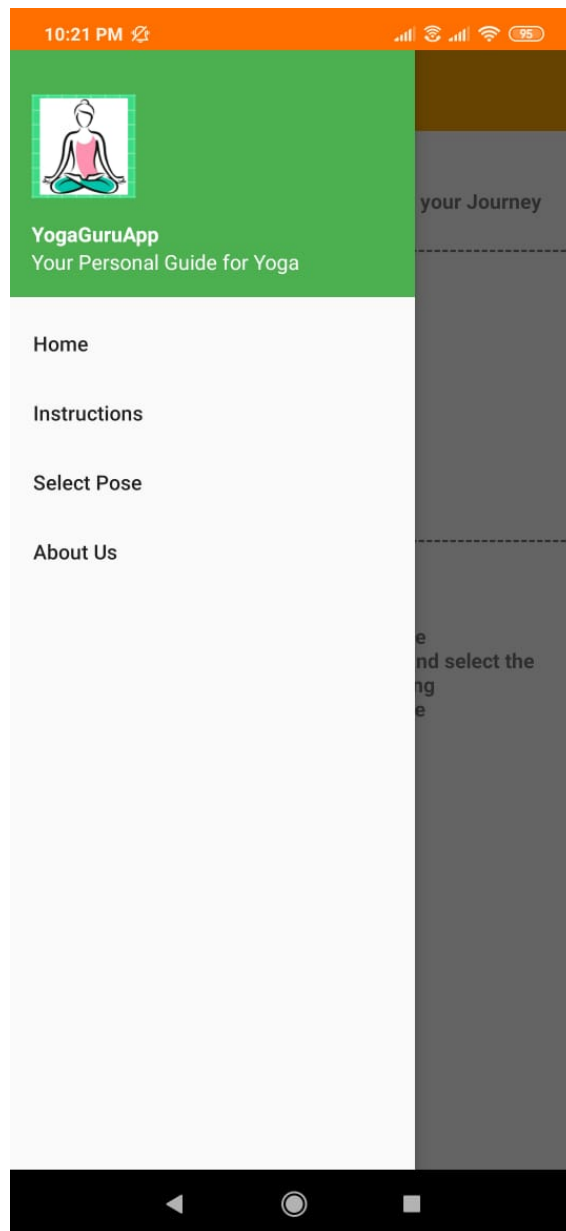
5.2 Acceptance Testing

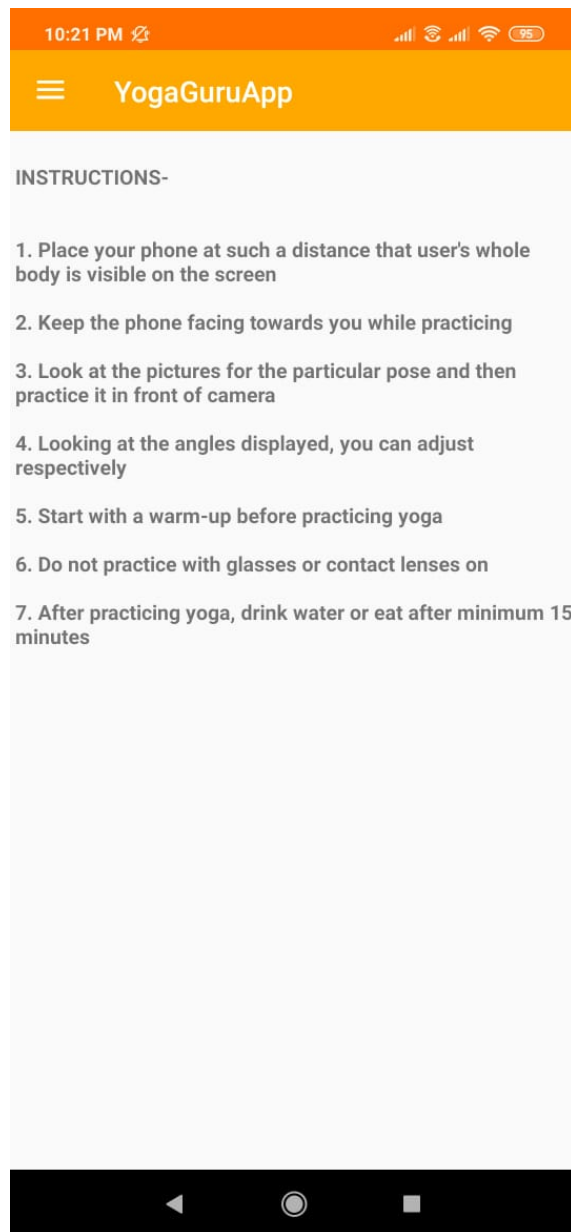
It is the formal testing according the user needs, requirements and business processes conducted to check whether the system satisfies the acceptance criteria or not and to enable the users, customers or authorized entities to determine whether to accept the system or not.

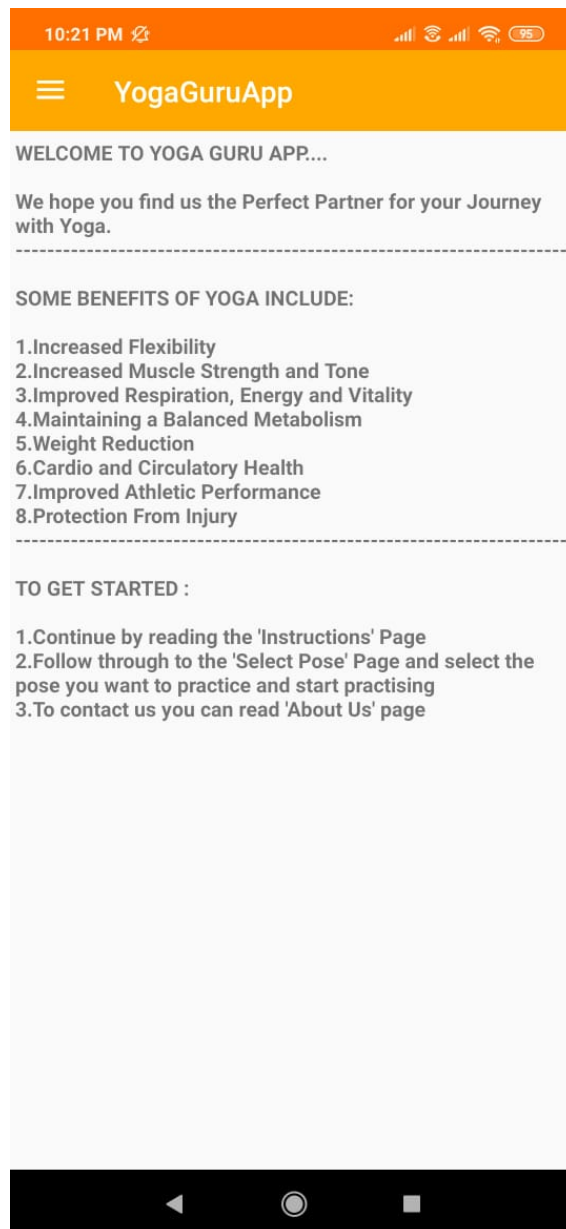
Test Case	Description	Input/Event	Expected Output	Actual Output
1.	Click Events	All the click events on the UI	Should lead to the desired page	Leads to the desired page
2.	Speed	Click event	Fast	Fast
3.	Camera access	Real-time user performance video	Camera should be linked as input channel	Camera is linked as input channel
4.	Voice command	User's voice to select index number of yoga pose	Whichever index number is chosen by the user through the voice it's respective activity should start	Respective activity is working properly
5.	Analysis of user's performance	Real-time user performance video	Angles of body part whose accuracy is below 80% should be display	Angles are displaying properly

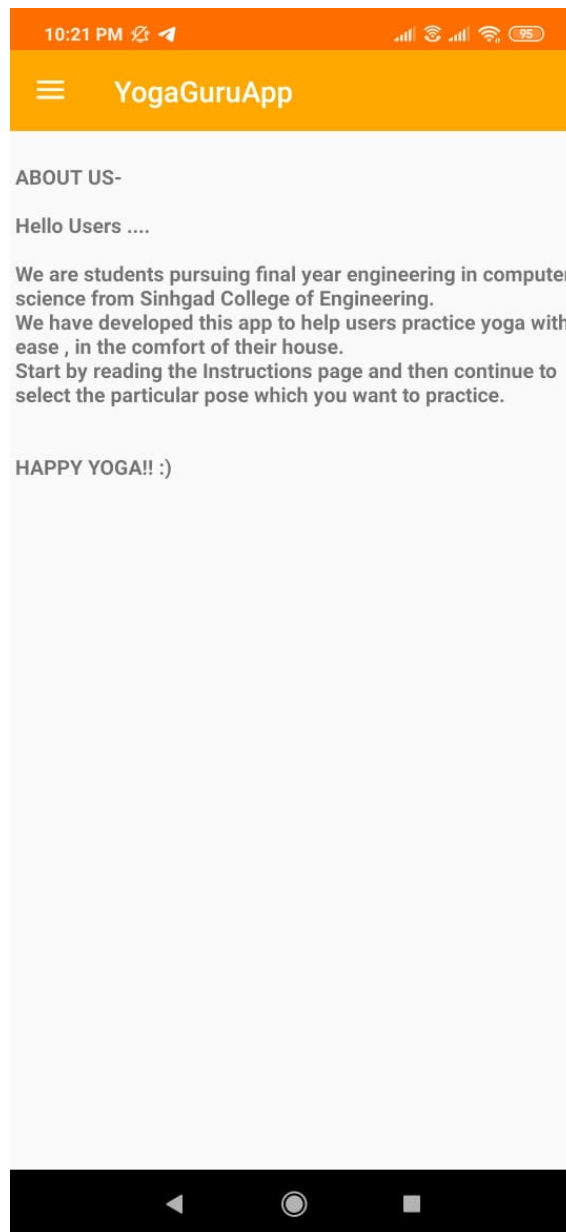
Results and Discussion

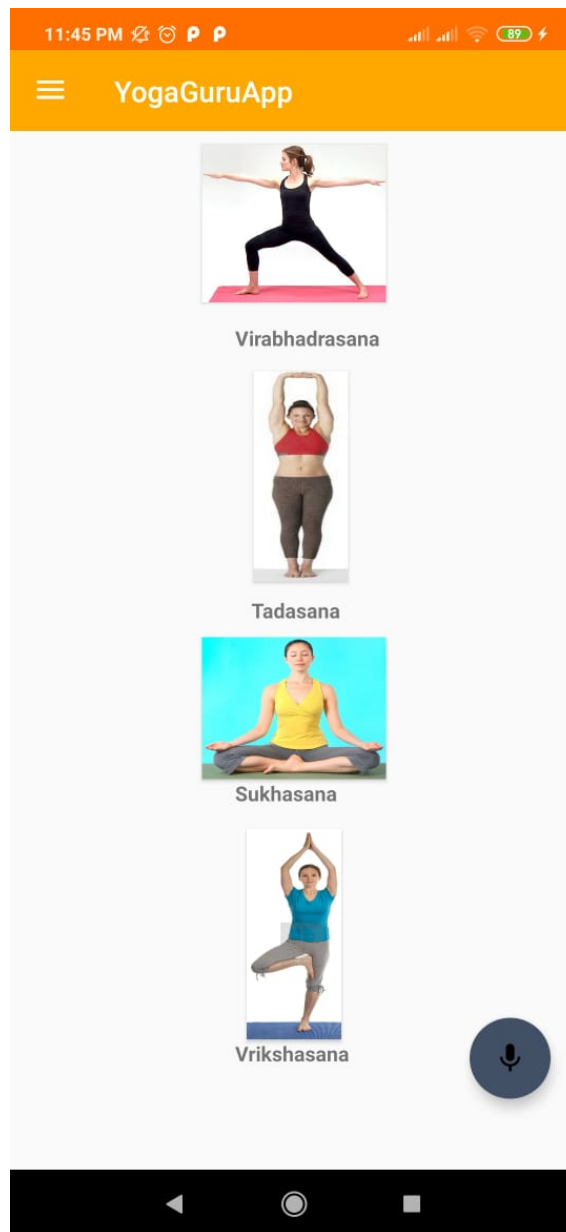
6.1 Main GUI snapshots











6.2 Discussions

User of the app will need a smartphone with good camera quality along with a strong internet connection. Once the user installs the app he can then start by reading the instructions page to know how to go about the app and how to use the app for performing yoga . The user can then go to the Select Pose page to select the particular pose he/she wants to practice. Once the user has selected the pose, the user can then start practicing the pose in front of the camera and the app will show results accordingly. Only if the accuracy of users pose is less than 80% the particular angle which needs correction will be displayed on screen so that the user can correct it. Once users accuracy is more than 80% the angles will no longer be displayed. Because the human body is not a robot and the flexibility can vary from person to person , the threshold is kept 80% and not 100%. User can choose from 4 yoga poses that the application currently supports mainly Virabhadrasana, Tadasana, Sukhasana, Vrikshasana.

The app enables the user to perform yoga according to his/her convenience in the comfort of their own house.

Conclusion

In this report, how Deep learning models are very useful for pose estimation and can be used for yoga pose estimation and correction is discussed. PoseNet is a model built and deployed on TensorFlow [17]. It can use the TensorFlow lite version for deploying it on an android app. Users need a smartphone with good quality hardware and the app. Once the user installs the app he can then start by reading the instructions page to know how to go about the app and how to use the app for performing yoga. The user can then go to the Select Pose page to select the particular pose he/she wants to practice. Once the user has selected the pose, the user can then start practicing the pose in front of the camera and the app will show results accordingly. The app captures the live video from the user's camera and estimates the 17 essential points and draws an overlapping skeleton for the same. It can use this skeleton to get the angles between various body parts and calculate the percentage of the correctness of the selected yoga pose. Only if the accuracy of users pose is less than 80% the particular angle which needs correction will be displayed on the screen so that the user can correct it. Once the user's accuracy is more than 80% the angles will no longer be displayed. Because the human body is not a robot and the flexibility can vary from person to person, the threshold is kept 80% and not 100%. Users can choose from 4 yoga poses that the application currently supports mainly Virabhadrasana, Tadasana, Sukhasana, Vrikshasana. [11]

The app also uses text to speech and speech to text features so that the user can control and navigate the app by using their voice. The app enables the user to perform yoga according to his/her convenience in the comfort of their own house without interruption. Thus, yoga pose estimation and correction by using a Deep Learning model for pose estimation can be successfully deployed on an android app.

Future Work

Many different adaptations, tests and experiments have been left for the future due to lack of time. Future work concerns deeper analysis of particular mechanisms, new proposals to try different methods, or simply curiosity.

Future work may include-

1. We could incorporate more poses and train on a larger dataset for more options of yoga poses for the user to practice from.
2. We would like to extend the functionality to providing the user feedback in our application
3. We could provide user login and dashboard so that the user could track his/her performance over specified time.
4. We could make the whole app voice-controlled for better accessibility and convenience for the user.
5. We could also extend the functionality by connecting the app to various other fitness equipment like the Fitbit so that the user can track his/her overall fitness performance.
6. Gesture recognition: This feature has not been implemented yet. However, since PoseNet already gives the coordinates of various body parts, it will be fairly to implement a gesture recognition system to navigate the app.
7. Progress tracking: This feature will store the measures of the correctness of the user's yoga poses. Giving users the ability to monitor their progress will not only be informative to them of their improvement but it will also keep the user motivated to keep on doing yoga.

REFERENCES

- [1] “Efficient Object Localization Using Convolutional Networks” The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 648-656
- [2] “Convolutional Pose Machines” The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 4724-4732
- [3] “Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields” The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 7291-7299
- [4] “Deep High-Resolution Representation Learning for Human Pose Estimation” The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 5693-5703
- [5] “PersonLab: Person Pose Estimation and Instance Segmentation with a Bottom-Up, Part-Based, Geometric Embedding Model” The European Conference on Computer Vision (ECCV), 2018, pp. 269-286
- [6] “Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation” The European Conference on Computer Vision (ECCV), 2018, pp. 801-818
- [7] “DeepPose: Human Pose Estimation via Deep Neural Networks” The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 1653-1660
- [8] Dan Oved, Irene Alvarado, and Alexis Gallo. “Real-time Human Pose Estimation in the Browser with TensorFlow.js” Internet: <https://medium.com/tensorflow/real-time-human-pose-estimation-in-the-browser-with-tensorflow-js-7dd0bc881cd5>, May 7, 2018 [Sept. 13, 2019].

- [9] "Recognition of Yoga Poses through an Interactive System with Kinect device" 2nd International Conference on Robotics and Automation Sciences, Edwin W. Trejo, Peijiang Yuan, 2018
- [10] "Human Pose Co-Estimation and Applications", IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 34, NO. 11, NOVEMBER 2012
- [11] "Yoga Posture Recognition By Detecting Human Joint Points In Real Time Using Microsoft Kinect" by Department of Computer Science and Engineering, Asian University of Bangladesh, Muhammad Usama Islam
- [12] "3D Human Pose Estimation with Relational Networks" S. PARK AND N. KWAK: 3D HPE WITH RELATIONAL NETWORKS
- [13] "MonoCap: Monocular Human Motion Capture using a CNN Coupled with a Geometric Prior" TO APPEAR IN IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, 2018
- [14] <https://www.fritz.ai/pose-estimation/>
- [15] <https://medium.com/beyondminds/an-overview-of-human-pose-estimation-with-deep-learning-d49eb656739b>
- [16] <https://github.com/pradnyaborkar90/Match-Pose-A-system-for-learning-Yoga-and-Dance-poses-using-Pose-Estimation-Posenet-Tensorflow>
- [17] <https://blog.tensorflow.org/2018/05/real-time-human-pose-estimation-in.html>