ZYOFISIK - YOUR PERSONAL AI FITNESS TRAINER

## A PROJECT REPORT

**Submitted by**

**ANGEL F [REGISTER NO:211418104020] BAIRAVI B [REGISTER NO:211418104033] JEY SNEHA A [REGISTER NO:211418104101]**

***in partial fulfillment for the award of the degree***

***of***

**BACHELOR OF ENGINEERING**

***in***

**COMPUTER SCIENCE AND ENGINEERING**



## PANIMALAR ENGINEERING COLLEGE

**(An Autonomous Institution, Affiliated to Anna University, Chennai)**

**MAY 2022**

# PANIMALAR ENGINEERING COLLEGE

###### (An AutonomousInstitution, Affiliated to Anna University, Chennai)

**BONAFIDE CERTIFICATE**

Certified that this project report **ZYOFISIK - YOUR PERSONAL AI FITNESS TRAINER** is the bonafide work of **ANGEL F [REGISTER NO:211418104020], BAIRAVI B [REGISTER NO:211418104033],**

**JEY SNEHA A [REGISTER NO:211418104101]** who carried out the project work under my supervision.

## SIGNATURE SIGNATURE

**Dr.S.MURUGAVALLI,M.E.,Ph.D DR.L.JABA SHEELA,M.E.,Ph.D HEAD OF THE DEPARTMENT SUPERVISOR**

**PROFESSOR**

DEPARTMENT OF CSE, DEPARTMENT OF CSE,

PANIMALAR ENGINEERING COLLEGE, PANIMALAR ENGINEERING COLLEGE, NASARATHPETTAI, NASARATHPETTAI,

POONAMALLEE, POONAMALLEE,

CHENNAI-600 123. CHENNAI-600 123.

Certified that the above mentioned students were examined in Anna University project viva-voice held on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

## INTERNAL EXAMINER EXTERNAL EXAMINER

**DECLARATION BY THE STUDENT**

We Angel F [REGISTER NO:211418104020], Bairavi B [REGISTER NO:211418104033], Jey Sneha A [REGISTER

NO:211418104101] hereby declare that this project report titled

**ZYOFISIK - YOUR PERSONAL AI FITNESS TRAINER**, under the

guidance of **DR.L.JABA SHEELA, M.E., Ph.D** is the original work done by us and we have not plagiarized or submitted to any other degree in any university by us.

# 1.

**2.**

**3.**

**ACKNOWLEDGEMENT**

We would like to express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We express our sincere thanks to our Directors **Tmt.C.VIJAYARAJESWARI**, **Dr.C.SAKTHI KUMAR,M.E.,Ph.D** and **Dr.SARANYASREE SAKTHI**

**KUMAR B.E.,M.B.A.,Ph.D.,** for providing us with the necessary facilities to undertake this project.

We also express our gratitude to our Principal **Dr.K.Mani, M.E., Ph.D.** who facilitated us in completing the project.

We thank the Head of the Computer Science and Engineering Department, **Dr.S.MURUGAVALLI,M.E.,Ph.D.,** for the support extended throughout the project.

We would like to thank my **Project Guide DR.L.JABA SHEELA** and all the faculty members of the Department of CSE for their advice and encouragement for the successful completion of the project.

We would like to thank our parents for their support rendered throughout the project.

## ABSTRACT

Yoga learning and Self-instruction systems can popularize and disseminate Yoga while ensuring proper practice. These computerized self-training systems for sports and fitness can improve participant performance and thwart injuries. Concocting an interactive web application that uses the webcam to cognize the user's yoga and exercise poses, estimates each pose to assist the user in practicing those postures, and tracks successful shots at various stances. Our approach seeks to cognize the yoga asanas based on the data obtained from an open-source Yoga-82 Dataset. The Mediapipe framework is employed for locating the landmarks of the images from the dataset and the frames obtained from the live stream webcam input. The detected critical points are passed to our model where the Convolutional Neural Network(CNN) finds patterns and the Keras-Sequential model analyzes their evolution over time. The Mediapipe is then used to retrieve the stick figure of the user for estimating their yoga poses and predicts its accuracy by passing it to the CNN model. On the contrary, in the exercise trainer module, the angles made by the lines of the stick figure are used to calculate the degree of the angle made between the arm and the forearm, which indeed helps in assisting the user as he or she can alter his or her posture accordingly. Finally the system contains a meditation coach, which is programmed to give commands at standard intervals, and subsequently the user is ushered by a voice that instructs them to maintain their breathing pace thereby creating a soothing environment.

# TABLE OF CONTENTS

CHAPTER NO. TITLE PAGE NO. ABSTRACT iii

[LIST OF TABLES vi](#_TOC_250014)

LIST OF FIGURES vii

[LIST OF SYMBOLS, ABBREVIATIONS viii](#_TOC_250013)

1. INTRODUCTION

1.1 [Overview](#_TOC_250012) 2

1.2 [Problem Definition](#_TOC_250011) 3

1. LITERATURE SURVEY 5
2. SYSTEM ANALYSIS
   1. Existing System 14
   2. Proposed system 16
   3. Feasibility Study 16
   4. Hardware Environment 20
   5. Software Environment 20
3. SYSTEM DESIGN

[4.1. ER diagram](#_TOC_250010) 22

* 1. [Data dictionary](#_TOC_250009) 23
  2. [Data Flow Diagram](#_TOC_250008) 24
  3. [UML Diagrams](#_TOC_250007) 26

CHAPTER NO. TITLE PAGE NO.

1. SYSTEM ARCHITECTURE
   1. [System overview](#_TOC_250006) 31
   2. [Module Design Specification](#_TOC_250005) 32
   3. [Algorithms 3](#_TOC_250004)5
2. SYSTEM IMPLEMENTATION
   1. Client-side coding 39
   2. Server-side coding 45
3. SYSTEM TESTING
   1. Unit Testing 50
   2. Integration Testing 51
   3. Test Cases & Reports 52
   4. Results and Discussions 54
4. CONCLUSION
   1. [Conclusion and Future Enhancements](#_TOC_250002) 59

[APPENDICES](#_TOC_250001)

* 1. [Sample Screens](#_TOC_250000) 60

**REFERENCES** 63

# LIST OF TABLES

**TABLE NO. TITLE PAGE NO.**

* 1. Survey Result- Health issue levels during lockdown by

different respondents 6

* 1. Survey Result - Health management technique used by

different respondents 6

* 1. Comparative study of Existing Android Applications 14
  2. Economic Feasibility 17

4.1 Data dictionary 23

* 1. Unit testing 50
  2. Integration testing 51
  3. Test Cases 52

|  |  |  |
| --- | --- | --- |
|  | **LIST OF FIGURES** |  |
| **FIGURE NO.** | **TITLE** | **PAGE NO.** |
| 4.1 | ER diagram of Zyofisik | 22 |
| 4.2 | Data flow diagram of Zyofisik - level 0 | 24 |
| 4.3 | Data flow diagram of Zyofisik - level 1 | 25 |
| 4.4 | Use case diagram of Zyofisik | 26 |
| 4.5 | Class diagram of Zyofisik | 27 |
| 4.6 | Activity diagram of Zyofisik | 28 |
| 4.7 | Sequence diagram of Zyofisik | 29 |
| 5.1 | System Architecture of Zyofisik | 31 |
| 5.2 | Yoga assistant architecture | 33 |
| 5.3 | Exercise trainer architecture | 34 |
| 5.4 | Meditation coach architecture | 35 |
| 7.1 | Exercise input | 52 |
| 7.2 | Exercise output | 52 |
| 7.3 | Yoga input | 53 |
| 7.4 | Yoga output | 53 |
| 7.5 | Dumbbell curl | 54 |
| 7.6 | Waiter curl | 55 |
| 7.7 | Hammer curl | 55 |
| 7.8 | Push-ups | 55 |
| 7.9 | CNN model accuracy graph | 56 |
| 7.10 | CNN model loss graph | 56 |
| 7.11 | Warrior - II pose | 57 |
| 7.12 | Tree pose | 57 |
| A1.1 | Meditation Coach Sample Screen | 60 |
| A1.2 | Exercise trainer Sample Screen - 1 | 60 |
| A1.3 | Exercise trainer Sample Screen - 2 | 61 |
| A1.4 | Exercise trainer Sample Screen - 3 | 61 |
| A1.5 | Yoga Assistant Sample Screen - 1 | 62 |
| A1.6 | Yoga Assistant Sample Screen - 2 | 62 |

**SYMBOL /**

# LIST OF SYMBOLS & ABBREVIATIONS

**DESCRIPTION**

**ABBREVIATION**

θ Angle made by three points (x1,y1),(x2,y2), (x3,y3)

σ Softmax activation function

β Correction Constant added in adam optimizer COCOMO Constructive cost model

KLOC Kilo lines per code

ROI Region of interest

CNN Convolution neural network

**INTRODUCTION**

**CHAPTER 1**

**INTRODUCTION**

**1.1 Overview**

The main goal of our project is to develop an interactive web application that has three main modules, namely The Yoga Assistant module, The Exercise Trainer module, and The Meditation Module.

The Yoga Assistant module uses a webcam to recognize the user's yoga and exercise poses and estimates each pose to assist the user practicing those postures to infer whether or not practiced effectively. In the Yoga Assistant module, the CNN algorithm is deployed by using the Keras - Sequential Model that uses categorical cross entropy to calculate the loss function and adam optimizer to reduce the loss. The user can select the asana, or allow the system to return a random yoga pose that is to be performed by the user and the system will predict the accuracy to what extent the pose is correctly performed by the user.

The Exercise Trainer module takes the frame from a real-time webcam stream at specific intervals and gets all the keypoints using pose estimation algorithm from the BlazePose Model in MediaPipe. Then the spatial location between the keypoints are found to draw a line, thereby creating a stick figure. According to the selected exercise the angle and distance made by the joints are calculated. The distance between the points of interest is calculated using Euclidean distance. From the set of two lines obtained angle is calculated in radians and then it is converted

to degrees. The calculated angle is checked with the coded limits and if it passes the conditions the count is increased and stage is changed.

In the Meditation Module the user specifies the duration of the meditation. The module provides meditation commands for the time set by the user. Consequently, when the client seeks the server every half a second and checks for a change in the audio file name to switch its instruction, the server provides the audio file depending on the command at standard intervals such as breathe in, breathe out. After the timer ends the thread stops and clears the audio file.

## Problem Definition

Yoga and Exercise have become ineluctable for the health and relaxation of the multitude as a result of the rise in stress in modern living. Asanas can be learned in a yoga studio or at home with home instructions or self-taught with the help of books and videos, where self-learning is the most desired option, especially in this pandemic crisis, but certainly is the most daunting as well. The possibility of practicing it incorrectly is high, which ultimately is futile. Therefore, our system helps the practitioners to perceive whether or not it is an asset to their fitness routine instead of stalling their time and effort.

**LITERATURE SURVEY**

## CHAPTER 2 LITERATURE SURVEY

1. **Chhaihuoy Long, Eunhye Jo, and Yunyoung Nam, (2022), “Development of a yoga posture coaching system using an interactive display based on transfer learning” - J Supercomput 78, 5269–5284 [3]**

This paper uses TL-MobileNet-DA through which they have achieved 98.43% accuracy. The Model is trained on 14 poses with a total of 120 images. By Performing data augmentation including sheer range of 0.2, zoom at 0.2, width shift and height shift in the range of 0.12, brightness at range (1, 1), and horizontal flip. Transfer learning (TL) is performed on six different pre-trained models, including VGG16, VGG19, MobileNet, MobileNetV2, InceptionV3, and DenseNet201, using the ImageNet dataset. This model uses the categorical cross-entropy loss function and Adam optimizer, with 100 epochs, in the training process and min delta value of 0.001 at patience 10 on EarlyStopping.

## Nagalakshmi Vallabhaneni and Dr. P. Prabhavathy, (2021), “The Analysis of the Impact of Yoga on Healthcare and Conventional Strategies for Human Pose Recognition”, Turkish Journal of Computer and Mathematics Education. (TURCOMAT).,vol. 12 no. 6, pp. 1772–1783 [7]

Yoga is perhaps the most straightforward physical, mental, and Spiritual work during this isolation period. Yoga Practice deals with these psychological issues, and along with breathing, meditation is the best general practice that will be careful with our body, brain, and soul. Yoga's impact on the people in the lockdown period is analyzed in this paper. The data was collected from 109 people, including 64 people who are male and 45 are female. Five parameters are considered: stress level, peace of mind, consciousness, physical health, and the respondent's mental health. The result of the survey is given in Table 1 and Table 2

Table 2.1. Survey Result- Health issue levels during lockdown by different respondents [7]

|  |  |  |  |
| --- | --- | --- | --- |
| **Health Issue** | **Stage during lockdown (%)** | | |
| ***Good*** | ***Average*** | ***Bad*** |
| Stress | 50 | 40 | 10 |
| Peace Of mind | 60 | 30 | 12 |
| Consciousness | 60 | 33 | 5 |
| Physical health | 75 | 20 | 2 |
| Mental health | 60 | 30 | 10 |

Table 2.2. Survey Result - Health management technique used by different respondents [7]

|  |  |
| --- | --- |
| **Percentage** | **Technique**  **(Health Management technique used by different respondents)** |
| 68% | Yoga |
| 14% | Walking |
| 7% | Relaxation |
| 6% | Exercise |
| 2% | Meditation |
| 3% | Others |

The survey shows that mental and physical health has had a noticeable change during the lockdown period. Yoga has been the top most technique used in managing health during this period which is also helpful in normal times.[14]

## Wu, Y., Lin, et al.(2021). “A Computer Vision-Based Yoga Pose Grading Approach Using Contrastive Skeleton Feature Representations.” Healthcare (Basel), 10(1):36 [10]

They proposed a grading approach to input two yoga pose images from the learner and the coach, respectively, and then extract the human skeleton key points and calculate the feature similarity between them. This system used both the coarse triplet example and the fine triplet example. Mediapipe framework was used to extract the key points. Yoga pose classification image dataset adopted from Kaggle with 45 categories and 1931 images selected gave accuracy of 83.21%. Yoga pose grading image dataset constructed by the authors containing 3000 triplet examples with each triplet example consisting of three pose images that belong to the same yoga pose category produced 63.58% accuracy.

## Yash Agrawal, Yash Shah, and Abhishek Sharma, (2020), “Implementation of Machine Learning Technique for Identification of Yoga Poses”, 9th IEEE International Conference on Communication Systems and Network Technologies, pp. 40-43 [11]

This paper tested six classification models of machine learning namely Logistic Regression, Random Forest, SVM, Decision Tree, Naive Bayes and KNN .But by using CNN and LSTM an accuracy of 99.04% was attained on a single frame when the Key Points were Open Pose was used to acquire the Keypoints. MediaPipe is faster because of its use of GPU. Dataset- 10 yoga poses, each class containing around 400 to 900 images, and the compiled Color Image dataset consists of 5459 images. Overall 94.28% accuracy was attained of all machine learning models, with Random Forest producing the most accuracy of approx 99%.

## Deepak Kumar, and Anurag Sinha, (2020), “Yoga Pose Detection and Classification Using Deep Learning” International Journal of Scientific Research in Computer Science, Engineering and Information Technology, Volume 6 Issue 6 Page: 160-184 [4]

Here keypoint Extraction of the human joint areas is done utilizing OpenPose which gives 18 keypoints. The system works at a rate of 3 FPS(frames per second). Dataset used comprises recordings of 6 yoga asanas performed by 15 distinct people (5 females and 10 guys) containing a total of 88 recordings (rate=30 FPS). This system uses the Support Vector Machine(SVM) algorithm which is best suited for classification giving a Train precision of 0.9953 and Validation exactness of 0.9762 and Test precision of 0.9319. But the CNN outperforms the SVM in terms of testing accuracy with Train exactness of 0.9878 Validation precision of 0.9921 and Test precision of 0.9858.

## Girija Gireesh Chiddarwar, et al. (2020), “AI-Based Yoga Pose Estimation for Android Application” International Journal of Innovative Science and Research Technology, Volume 5 - Issue 9 [5]

It surveys the various technologies that can be used for pose estimation and concludes the best method based on the usability for an android application. This paper ponders Pose Estimation models like OpenPose, DeepPose, and PoseNet.

OpenPose uses Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. The issue with OpenPose is that it necessitates specialized hardware and does not scale well on mobile devices. DeepPose makes use of deep learning models, yet it performs poorly due to weak generalization.

PoseNet endorses the PoseNet model as it is an open-sourced technology that allows extracting the 17 essential points natively and a skeleton of the human pose

is drawn with the help of these points, which is then used to derive angles between these points thus enabling us to effectively correct the user’s yoga poses. It is deployed using Tensorflow. But The Pose net model only returns 17 key points, none of which includes fingers, which could lead to hand position modifications being limited.

## M. Verma, S. Kumawat, Y. Nakashima and S. Raman, (2020), "Yoga-82: A New Dataset for Fine-grained Classification of Human Poses", IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) pp. 4472-4479 [6]

Yoga82 consists of 82 complex yoga pose images downloaded from the web using the Bing search engine. It includes both Sanskrit and English names of yoga poses that were used to search for images and the downloaded images were cleaned and annotated manually. Every image contains one or more people doing the same yoga pose. Furthermore, images have poses captured from different camera view angles. This dataset has a varying number of images in each class from 64 (min.) to 1133 (max.) with an average of 347 images per class. Some of the images are downloaded from a specific yoga website. Hence, they contain only yoga poses with a clean background.[23] However, there are many images with random backgrounds.

## Nuruldelmia Idris, Cik Feresa Mohd Foozy, and Palaniappan Shamala, (2020), “A Generic Review of Web Technology: DJango and Flask”, International Journal of Advanced Computing Science and Engineering ISSN 2714-7533 Vol. 2, No. 1 [8]

This paper is about Django and Flask python web frameworks. Django comes with a comprehensive MVC Framework that takes care of everything. While Django may be used to create a RESTful API on its own, it is one of the frameworks that sows fantastical creations. It is a feature-rich extension of the Django framework. On the other hand, Flask is a micro-framework that adheres to a set of rules: complete one task at a time effectively. It provided very little upfront, but it has a significant number of extensions that match Django's feature set.

Versioning – Django's flexibility makes the task less critical and multiple URL formats are supported and can be sent in as a request parameter.

Browsable API – Django Generates HTML pages to browse and execute all the endpoints where the users or developers can operate swiftly and simply.

Regular releases - the news version is always released twice a year to keep users up to date with the newest edition.

Speed – Flask is generally fast in performance compared to Django which might be due to Flask's simplistic design as it can handle hundreds of queries per second without slowing down the process.

NoSQL support –Flask is compatible with NoSQL databases such as MongoDB and DynamoDB.

## Camillo Lugaresi, et al. (2019), “MediaPipe: A Framework for Perceiving and Processing Reality”, Computer Vision for AR/VR at IEEE Computer Vision and Pattern Recognition (CVPR). [1]

Mediapipe is a cross-platform library developed by Google that provides ready-to-use customizable ML solutions for live and streaming media,for computer vision tasks. Alternatives to mediapipe include ARKit3,Microsoft Mixed Reality Toolkit,OpenPose. ARKit 3 Body Tracking produces several false positives similar to the ones in the picture, e.g.shadows on the walls, painting decorations being mistakenly identified as humans. Microsoft Mixed Reality Toolkit SDK was an abstraction for the capabilities of the different platforms not suitable for all platforms. OpenPose is based on DNN which requires a high-end computer Slight tradeoff between speed and accuracy. Current human pose performance metrics are based on keypoint accuracy. Failure cases still exist including foot and leg occluded and rare joint position.

Google MediaPipe’s accuracy of the solution was very high. MediaPipe is Fast is able to achieve its speed by the use of GPU acceleration and multi-threading. Mediapipe is Modular and Reusable. Its use of graphs, subgraphs, and calculators means that the work of one project can easily translate to the work of another. Mediapipe is Deployment Platform Friendly allowing to deploy the application not only to desktops but to mobile devices as well.

## T. K. K. Maddala, P. V. V. Kishore, K. K. Eepuri and A. K. Dande, (2019), "YogaNet: 3-D Yoga Asana Recognition Using Joint Angular Displacement Maps With ConvNets," IEEE Transactions on Multimedia, vol. 21, no. 10, pp. 2492-2503, Oct. 2019 [9]

Recognition is performed by integrating joint angular movements along with the joint distances in a spatiotemporal color-coded image called a joint angular displacement map(JADM). JADMs are trained with a single-stream CNN. The Steps include computing JADMs from 3D data then Color coding the JADMs then to train and test the CNN. A single 3D skeleton with J joints, there will be J×(J−1) / 2 unique joint pairs. Dataset prepared contains a total of 4200 3D yoga videos recorded for 42 poses and 100 variations per pose. Model consisting eight convolutional layers followed by two fully connected layers produced a accuracy of 90.01%

## Chen, H., He, Y., & Hsu, C., (2018), “Computer-assisted yoga training system”, Multimedia Tools and Applications, 77, 23969-23991. [2]

This paper proposed a computer vision-based yoga training system, termed Y-system, to analyze the postures of a practitioner and assist in rectifying incorrect postures. This system analyzes up to twelve yoga poses. Topological skeletons are generated from the practitioner’s body maps of front and side views. Then, the Y-system extracts postural features including dominant axes, skeleton-based feature points, and contour based feature points. A distance map is produced by applying distance transform to the body map. Explicit posture description models for the 12 yoga poses based on their respective training is implemented. The accuracy of the Y-system for each pose ranges from 76.22% to 99.87%.

**SYSTEM ANALYSIS**

## Existing System

**CHAPTER 3** **SYSTEM ANALYSIS**

Table 3.1 Comparative study of Existing Android Applications.

|  |  |  |  |
| --- | --- | --- | --- |
| **App name** | **Google play store rating** | **Advantages** | **Disadvantages** |
| JEFIT | 4.1 | Expert-led exercise demonstrations promote proper form and safety and Offers customized workout options | Most features require a paid subscription and advanced training reports, featured workout sets are costly |
| Freeletics | 4.2 | High-quality video and Audio coaching workouts ensure you train properly | Users say the stretching portion of the workout feels a bit rushed and not customizable. Exercise counts forced to do all |
| Strava | 4.3 | Track your workout distance, speed and endurance. Free monthly fitness challenges keep you motivated | Most features require a paid subscription and tracks wrong location |
| myYogaTe acher | 4.4 | Appeals to beginners to advanced Users and contains highly customizable sessions and Live/recorded online sessions | Limited focus on other workouts yoga,  meditation and  subscription required after free trial |
| Curefit Healthcare | 4.5 | Activities classified to different levels and live classes are helpful  track calorie consumed, | Workout intensity is not reliable. Tracks certain activity even if permissions are denied |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | calorie burnt | breach of data |
| Healthify Me | 4.5 | Track calorie consumed, calorie burnt, sleep tracker, water consumed | Inconsistent food database and health assistant bot solutions are available for pre-trained questions alone |
| Map My Run | 4.6 | Most-used features are free and can track 600+ activities in addition to running | Can drain your phone battery on longer runs and no longer supports music integration |
| The 7 Minute Workout | 4.7 | Beginner-friendly workouts available. Convenient for those with busy schedules | Subscription fee required for full access to workouts but limited to 7-minute format |
| Leap Fitness Group | 4.8 | 5 to 30-minute workouts for 4 weeks classified into 3 levels – beginner, intermediate, advanced | Exercises are not customizable. Sometimes free trial scammed to pay forcibly |
| Nexoft | 4.8 | Exercise plans according to energy level - easy, normal, hard plan each for 8,12,20 minutes respectively | No weight tracking and no progress tracking.  Yoga pose instructions are given after the countdown has started |

The apps mentioned in the above table mostly contain paid subscriptions for recorded or live sessions[21] and only few features which are free to use. These apps are not AI based and the least feature which uses AI is the chatbot which answers the users questions or gives recommendations for the user whereas Zyofisik is completely AI based. These apps do not provide any accuracy or precision on the user's work. Zyofisik provides the accuracy of the yoga pose and also in the exercise trainer only correct exercises are considered.

## Proposed System

The proposed system is an interactive web application that uses a webcam to cognize the user's yoga and exercise poses and estimates each pose to assist the user practicing those postures to infer whether or not practiced effectively. In the Yoga Assistant module, the CNN algorithm is deployed by using the Keras- Sequential Model that uses categorical cross entropy to calculate the loss function and adam optimizer to reduce the loss. The Exercise Trainer module takes the frame from a real-time webcam stream at specific intervals and gets all the keypoints using the pose estimation algorithm from the BlazePose Model in MediaPipe. In the Meditation Module the user specifies the duration of the meditation.Consequently, when the client seeks the server at regular intervals and checks for a change in the file name to switch its instruction, the server provides the audio file depending on the command at specified intervals such as breathe in, breathe out.

## Feasibility Study Social Feasibility

* + 1. Yoga aims at the development of all-round personality, it is the synchronization of mind, body, and spirit[26].
    2. Yoga can result in serious damage to one’s physique and brain if not performed correctly. Hence it demands proper training and assistance during the course of action.
    3. In ancient times, it used to be performed only under the supervision of an accomplished Guru (teacher). Therefore, it is essential to adopt the right Yoga procedure at the very beginning.
    4. Deep Learning models can be trained to detect Yoga postures and be able to provide feedback/corrections if needed and conveys if the pose performed in the image is correct.

## Economical Feasibility

Economical Feasibility study using COCOMO model was carried out to find the cost/ benefits analysis of the project .

Advanced COCOMO model with organic,semi-detached categories of software project modeling is used.

Table 3.2. Economical Feasibility

|  |  |  |  |
| --- | --- | --- | --- |
| **MODULE** | **CATEGORY** | **EFFORT (PM)** | **DEVELOPMENT**  **TIME (Months)** |
| **Meditation** | ORGANIC | 3.2(KLOC)1.05 | 2.5(Effort)0.38 |
| **Exercise** | SEMI-DETACHED | 3.0(KLOC)1.12 | 2.5(Effort)0.35 |
| **Yoga Pose Estimation** | SEMI-DETACHED | 3.0(KLOC)1.12 | 2.5(Effort)0.35 |

## Meditation:

**Estimation of development effort:**

Effort = 3.2(KLOC)1.05 PM = 3.2(0.15)1.05 = 3.2 x 0.1364 =0.43 PM

## Estimation of development time:

Tdev = 2.5(Effort)0.38 Months = 2.5(0.43)0.38 = 2.5 x 0.7256 =1.81 Months

## Exercise

Effort Adjustment Factor EAF = 1

## Estimation of development effort:

Effort = 3.0(KLOC)1.12 PM = 3 x 0.3583 = 1.07 PM

## Estimation of development time:

Tdev = 2.5(Effort)0.35 Months = 2.5 x 1.025 = 2.56 Months

## Pose estimation

Effort Adjustment Factor EAF = 1

## Estimation of development effort:

Effort = 3.0(KLOC)1.12 PM = 3 x 0.2596= 0.77 PM

## Estimation of development time:

Tdev = 2.5(Effort)0.35 Months = 2.5 x 0.9162= 2.29 Months

**Total effort** = 0.43+1.07+0.77 = 2.27 PM

**Total development time** = 1.81 + 2.56+2.29 = 6.6 Months

## Technical Feasibility Python:

* Since this is an artificial intelligence project using python gives us the advantage for various inbuilt libraries like TensorFlow, Scikit-Learn, and NumPy.
* Python is an object-oriented programming language that makes coding easier and less prone to syntax errors.
* Python being platform-independent will help in future up-gradation to deploy as software.

## Flask:

* The two mostly used web frameworks are flask and django with Flask being a lightweight framework.
* This project is deployed in HEROKU, which has limits to the size of the compressed file thus we made use of flask to reduce the lines of code. Also flask is best suited for SINGLE APPLICATIONS.

## MediaPipe:

* Mediapipe Human Pose Detection solution provides 33 key points each containing x,y,z coordinates along with the visibility score. The solution utilizes a two-step detector-tracker ML pipeline[20].
* Using a detector, the pipeline first locates the person/pose region-of-interest (ROI) within the frame. The tracker subsequently predicts the pose landmarks and segmentation mask within the ROI using the ROI-cropped frame as input.BlazePose accurately localizes more key points, making it uniquely suited for fitness applications.

## CNN:

* CNN a supervised learning model has given satisfactory results in identifying key features of the training data.

## Yoga-82 Dataset :

* It contains 82 classes of images with each yoga class containing an average of 200 images.
* The dataset is labeled with 3 levels of hierarchy and is obtained from the Internet.

## Hardware Environment

Processor - Intel core i3

RAM (Random Access memory) - 3GB Storage - 2.3GB

Network (Ethernet or Wi-Fi) - 2-5Mbps

## Software Environment

* + 1. Operating System - Windows 10
    2. Python - 3.8.7
    3. Tensorflow - 2.8.0
    4. Flask 2.1.1
    5. Mediapipe 0.8.9.1 6. OpenCV 4.5.5.64

7. Heroku

**SYSTEM DESIGN**

## 4.1. ER diagram

**CHAPTER 4** **SYSTEM DESIGN**

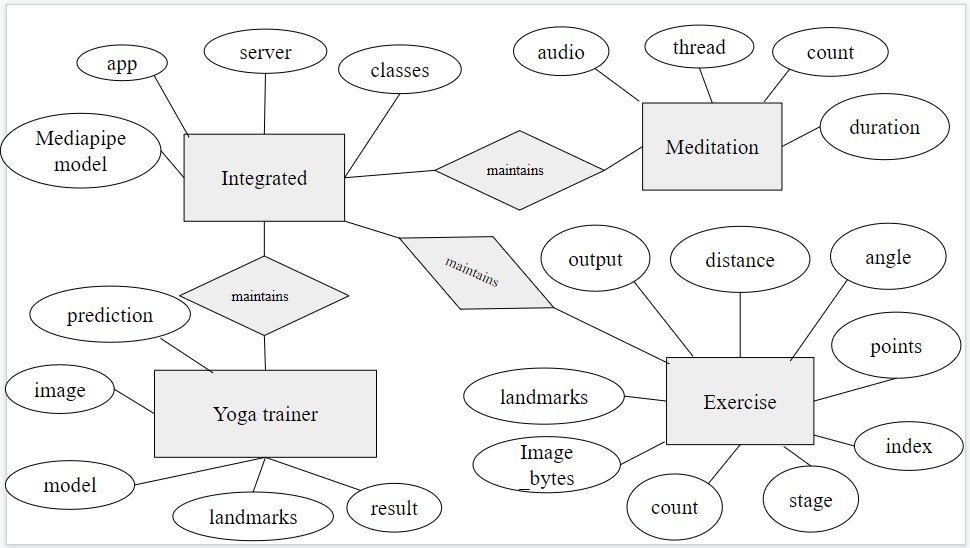
ER-Diagram is a pictorial representation of data that describes how data is communicated and related to each other.Entities are represented by means of rectangles.Attributes are the properties of entities. Attributes are represented by means of ellipses.Relationships are represented by diamond-shaped boxes.

Figure 4.1 ER diagram of Zyofisik

## Data dictionary

Data dictionary is a set of information describing the contents, format, and structure of a database and the relationship between its elements, used to control access to and manipulation of the database.

Table 4.1 Data dictionary

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entity** | **Attribute** | **Data Type** | **Data Size** | **Description** |
| Meditation | Audio | File |  | Contains path of audio file to play |
| Meditation | Count | int |  | Count of seconds |
| Meditation | Duration | int |  | Maximum value of count |
| Exercise | Exercise Image | numpy.array |  | Input image |
| Exercise | count | int |  | Correct Exercise count |
| Exercise | angle | float | 0 - 360 | Angle made by joints |
| Exercise | distance | float |  | Distance between two landmarks |
| Exercise | Exercise Landmark | <Landmark> | 33 x 2 | Landmarks of input image from mediapipe |
| Yoga | YogaImage | numpy.array |  | Input Image |
| Yoga | Model | <Sequential> |  | CNN model |
| Yoga | result | numpy.array | 1 x 82 | Prediction result |

## Data Flow Diagram

The flow of data of a system or a process is represented by DFD. It also gives insight into the inputs and outputs of each entity and the process itself. Input to output transformation in a system takes place because of process function. The symbols of a process are rectangular with rounded corners, oval, rectangle or a circle. Data flow describes the information transferring between different parts of the systems. The arrow symbol is the symbol of data flow.

## DFD - level 0

DFD - level 0 is designed to be an abstraction view, showing the system as a single process with its relationship to external entities.

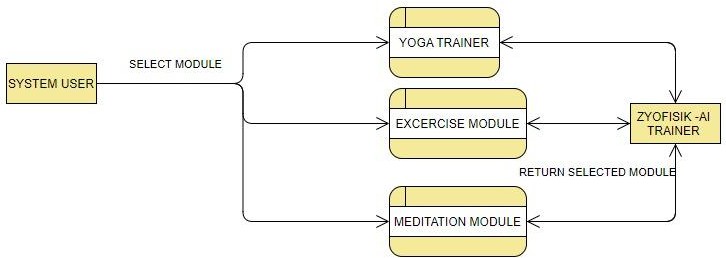


Figure 4.2 Data flow diagram of Zyofisik - level 0

## DFD - level 1

In 1-level DFD, the context diagram is decomposed into multiple bubbles/processes. In this level, we highlight the main functions of the system and break down the high-level process of 0-level DFD into subprocesses.

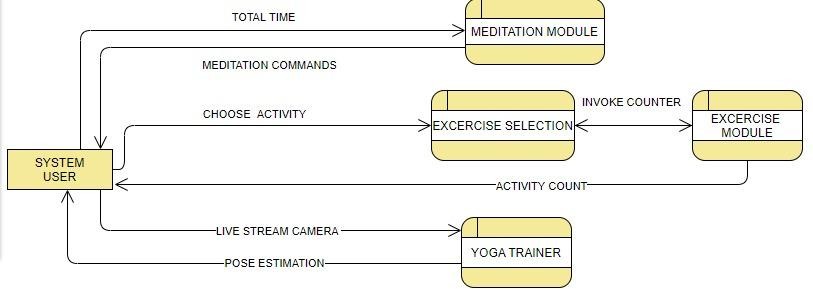


Figure 4.3 Data flow diagram of Zyofisik - level 1

## UML Diagrams

* + 1. **Use Case Diagram**

A use case diagram is used to represent the dynamic behavior of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their relationships. It models the tasks, services, and functions required by a system/subsystem of an application. It depicts the high-level functionality of a system and also tells how the user handles a system.

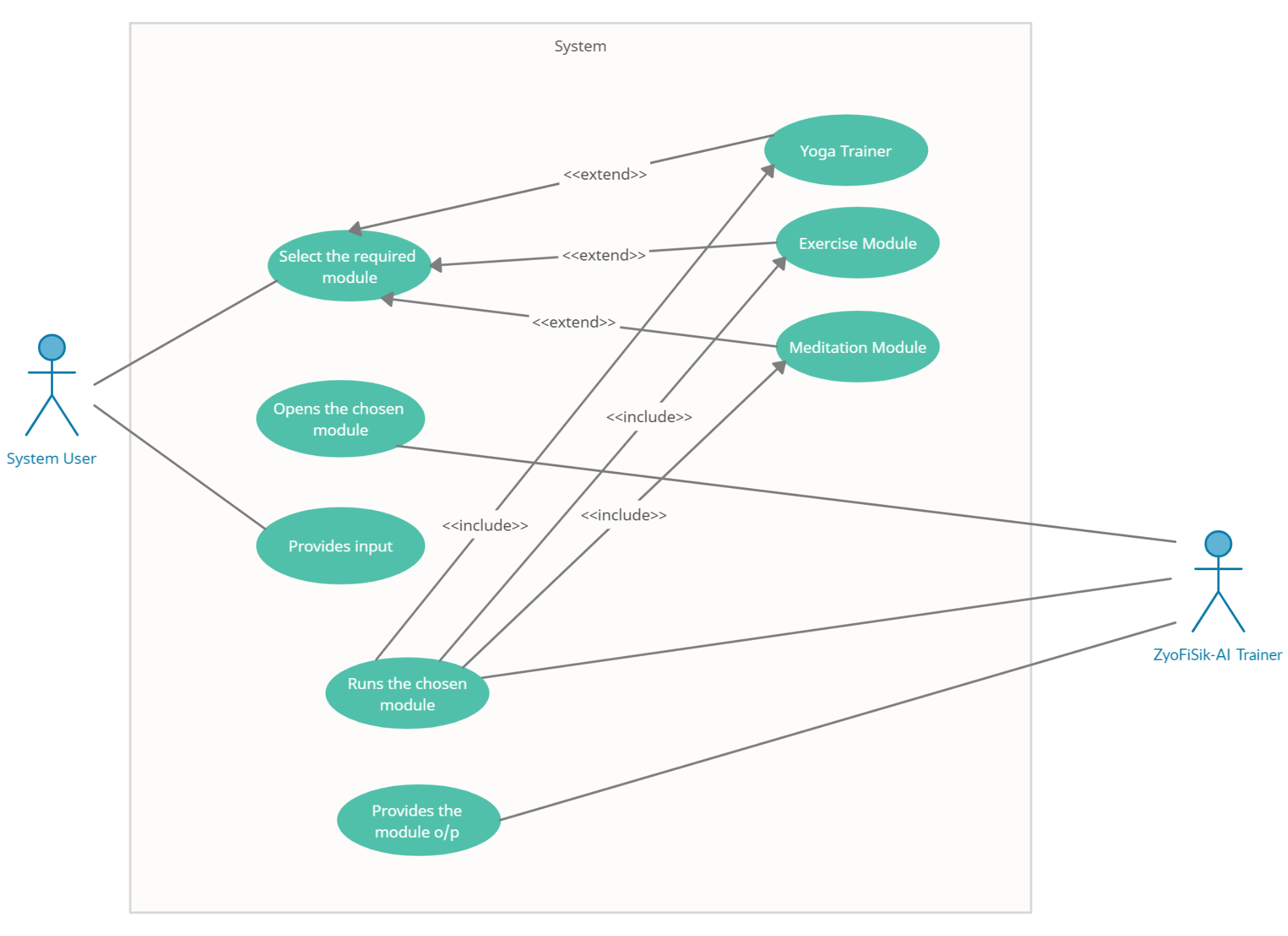


Figure 4.4 Use case diagram of Zyofisik

## Class Diagram

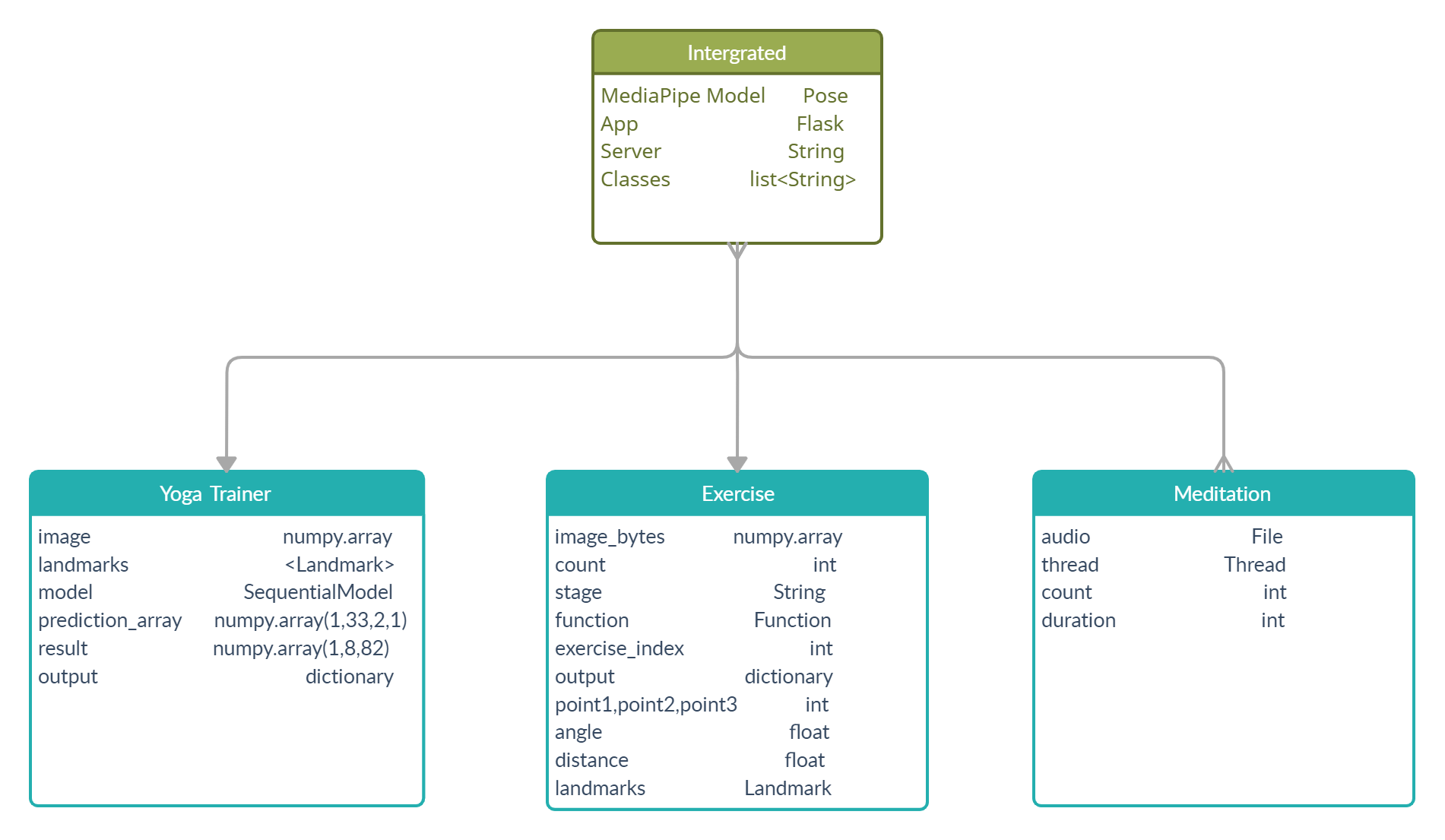
The class diagram depicts a static view of an application. It represents the types of objects residing in the system and the relationships between them. A class consists of its objects, and also it may inherit from other classes.

Figure 4.5 Class diagram of Zyofisik

## Activity Diagram

The activity diagram is used to demonstrate the flow of control within the system rather than the implementation. It models the concurrent and sequential activities. It encompasses activities composed of a set of actions or operations that are applied to model the behavioral diagram.

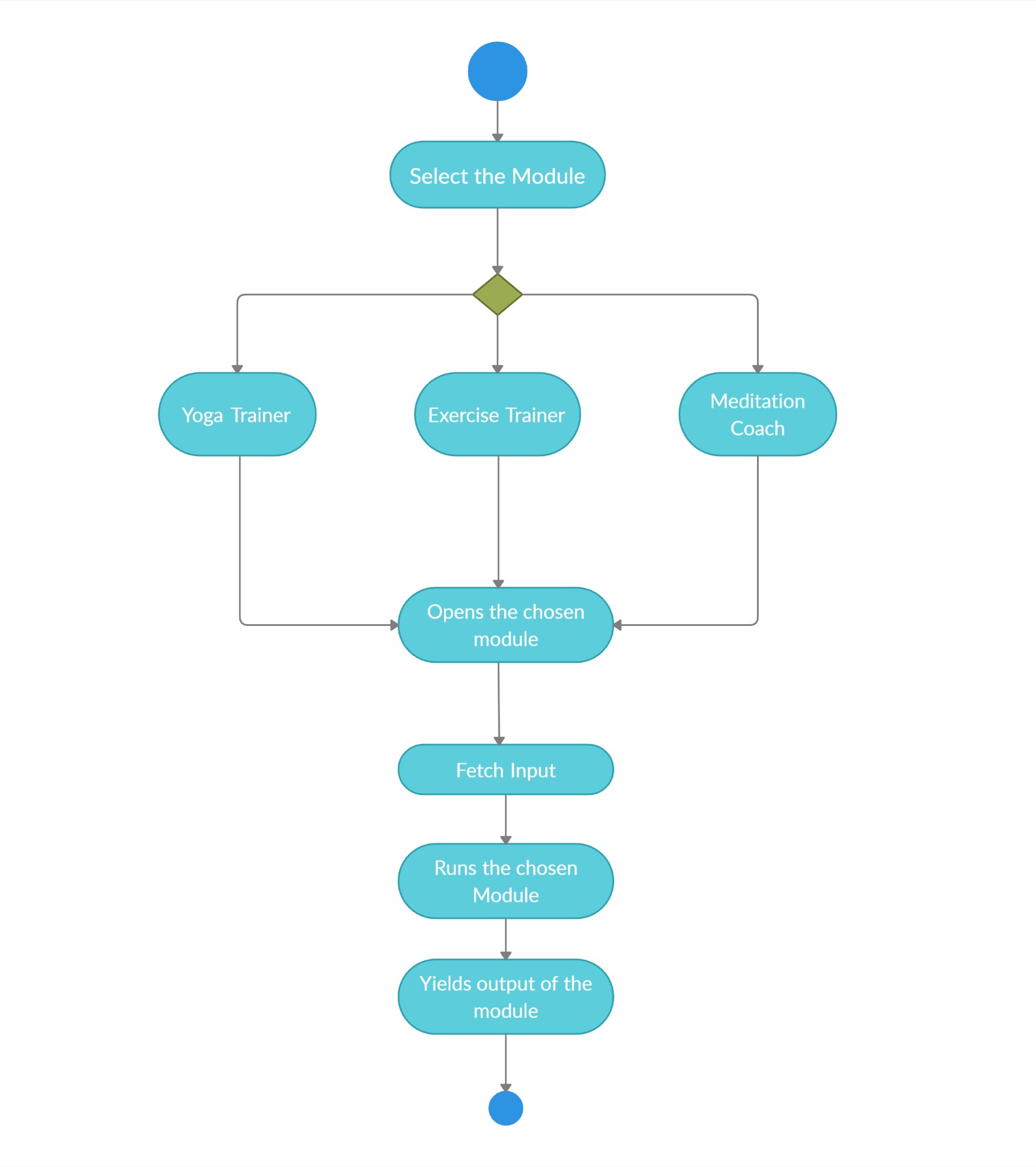


Figure 4.6 Activity Diagram of Zyofisik

## Sequence Diagram

The sequence diagram represents the flow of messages in the system and is also termed as an event diagram. It helps in envisioning several dynamic scenarios. It portrays the communication between any two lifelines as a time-ordered sequence of events, such that these lifelines took part at the run time.

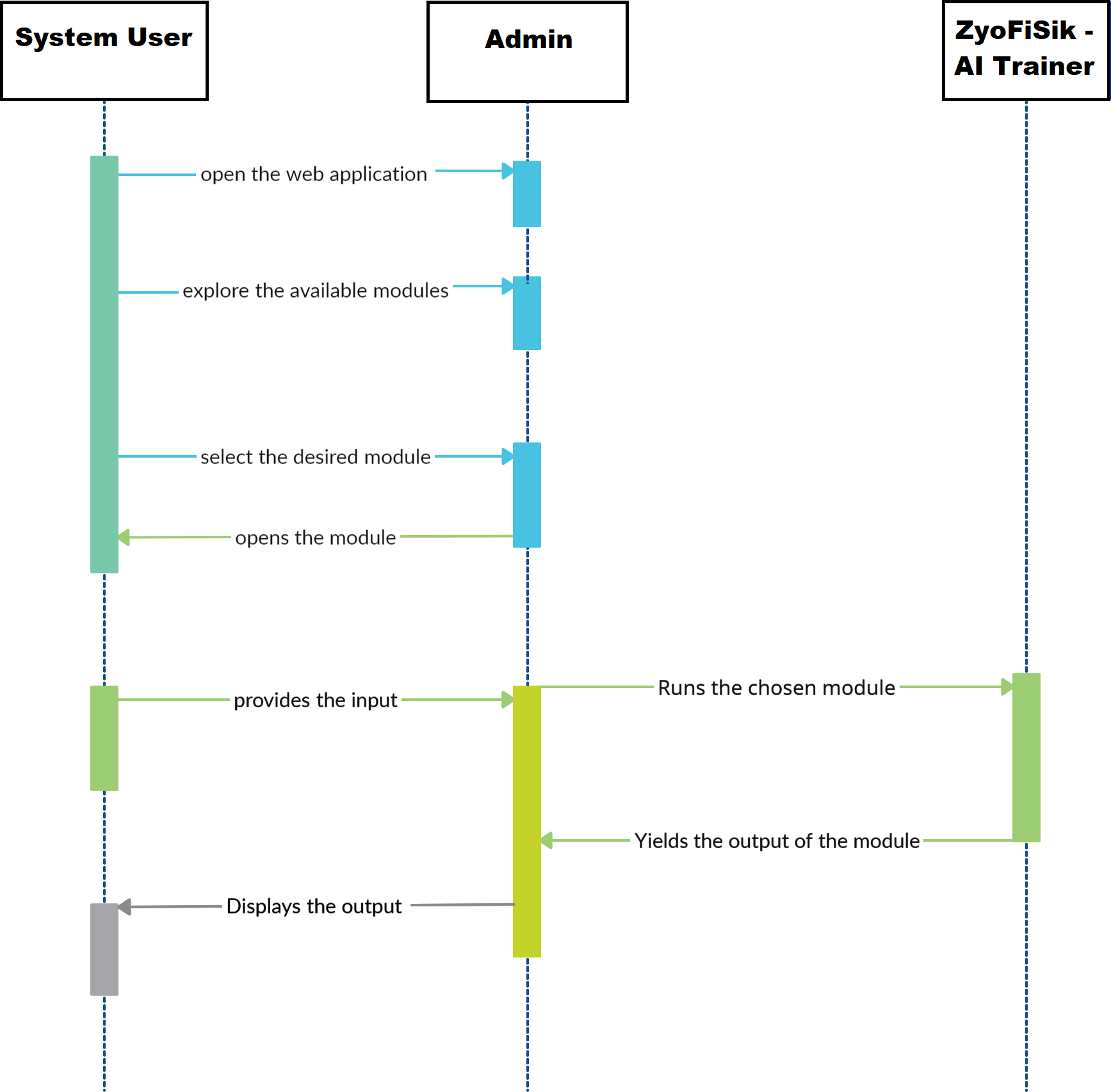


Figure 4.7 Sequence diagram of Zyofisik

**SYSTEM ARCHITECTURE**

## System Overview

**CHAPTER 5** **SYSTEM ARCHITECTURE**

Zyofisik contains three subsystems. Meditation coach, Exercise Trainer and Yoga assistant. All the three subsystems are integrated with the Flask web app. Mediapipe is used for retrieving landmarks of human pose which contains 33 keypoints.

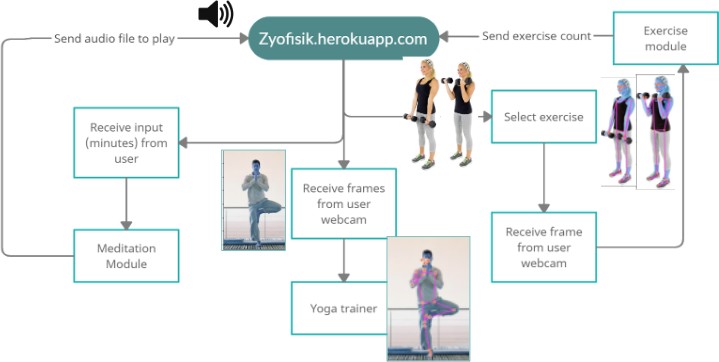


Figure 5.1 System Architecture of Zyofisik

## Module Design Specification

Zyofisik contains three modules Yoga assistant, Exercise Trainer and meditation coach all integrated into a web application.

## Yoga Assistant Module

* + - 1. The user can select the asana he/she needs assistance for or allow the system to predict.
      2. The frames from the client side are sent to the server side.
      3. The pre-built CNN model is loaded on receiving the first request.
      4. The landmarks of the input image are retrieved using the Mediapipe Blazepose model and the figure is drawn on the image.
      5. The landmarks are reshaped into a 33x2 shaped array.
      6. The reshaped array is passed to the CNN model which returns the result containing 82 values[18].
      7. The accuracy of the selected asana is returned or if no asana is selected the asana with highest accuracy is returned.
      8. For building the CNN model Yoga-82 dataset was used.
      9. The Yoga-82 dataset contains URLs to images for 82 asanas[28][29]. 10.Each URL was read and the image was downloaded to Google drive cloud

storage.

1. All the downloaded images were read and the landmarks[16] of the images were written to a CSV file.
2. For creating the training and test dataset the CSV was read and the data was reshaped to a 33x2 array.
3. The CNN model was trained on this data and produced a accuracy 0f 98.9%

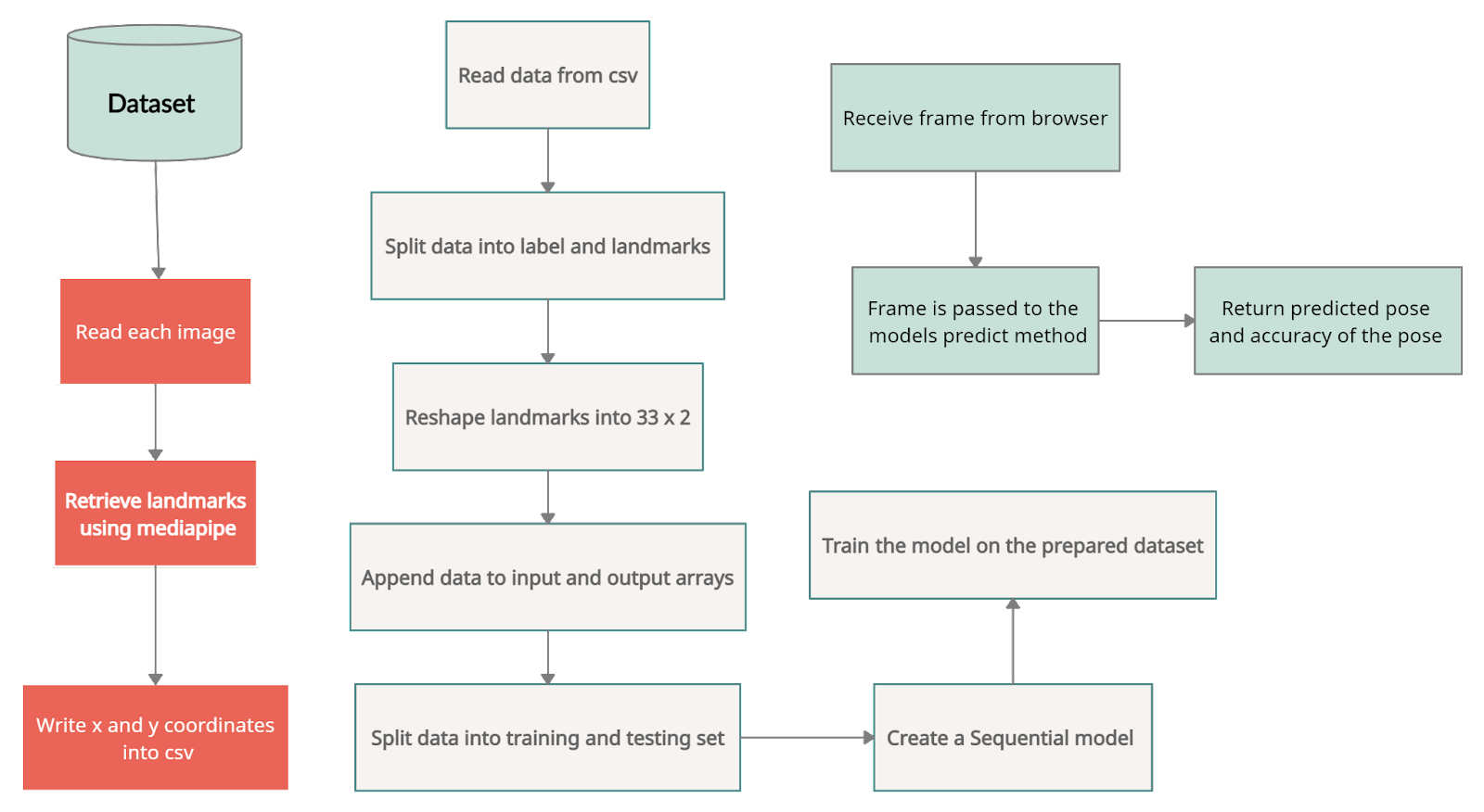


Figure 5.2 Yoga Assistant architecture

## Exercise Trainer Module

* + - 1. The user must select the exercise he/she is going to perform.
      2. Based on the selected exercise the web page is rendered on the client side with the steps to perform and a graphical representation for reference.
      3. On clicking the start button the frames captured on the clients webcam are sent to the server side.
      4. The landmarks of the input image are retrieved using the Mediapipe Blazepose model and the figure is drawn on the image[15].
      5. According to the selected exercise the angle and distance made by the joints are calculated. [Eg: Shoulder, Elbow, Wrist for Bicep curl]
      6. The calculated angle is checked with the coded limits and if it passes the conditions the count is increased and stage is changed .
      7. The changed values and modified image are returned to the Client side.
      8. For every 500ms the frame is sent from client to server.
      9. According to the selected exercise the angle and distance made by the joints are calculated. [Eg: Shoulder, Elbow, Wrist for Bicep curl]
      10. The calculated angle is checked with the coded limits and if it passes the conditions the count is increased and stage is changed .
      11. The changed values and modified image are returned to the Client side. 12.For every 500ms the frame is sent from client to server.

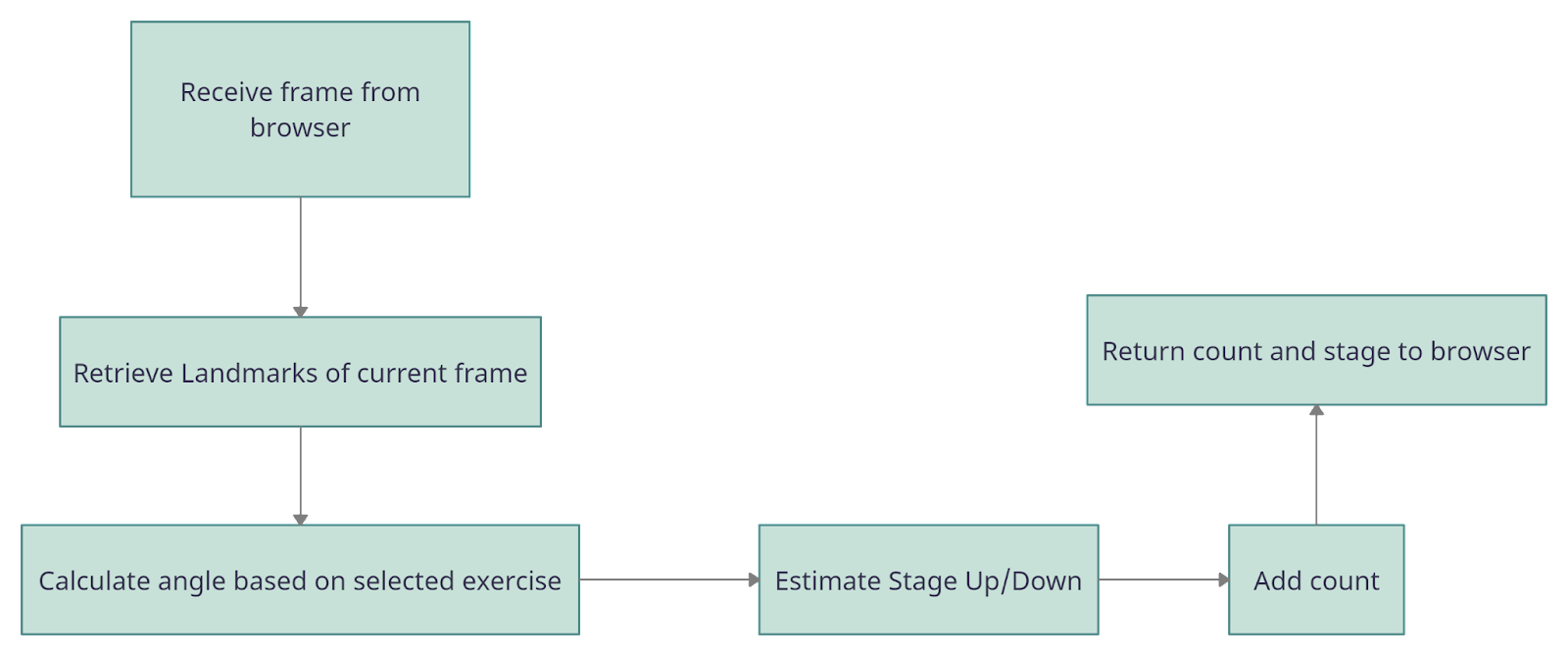


Figure 5.3 Exercise trainer architecture

## Meditation Module

* + - 1. Meditation module gets the duration as input from the user on the client side.
      2. On receiving the input a thread is created which runs for the specified duration.[24]
      3. Initially the “Close Your Eyes.mp3” file is written to an audio file.
      4. After 2 seconds intervals “breathIn.mp3” and “breathe Out.mp3” are written to the audio file at an interval of 3 seconds.
      5. The client checks for the current audio file every second and if there is a change it plays the new audio file.
      6. After the timer ends the thread stops and clears the audio file.

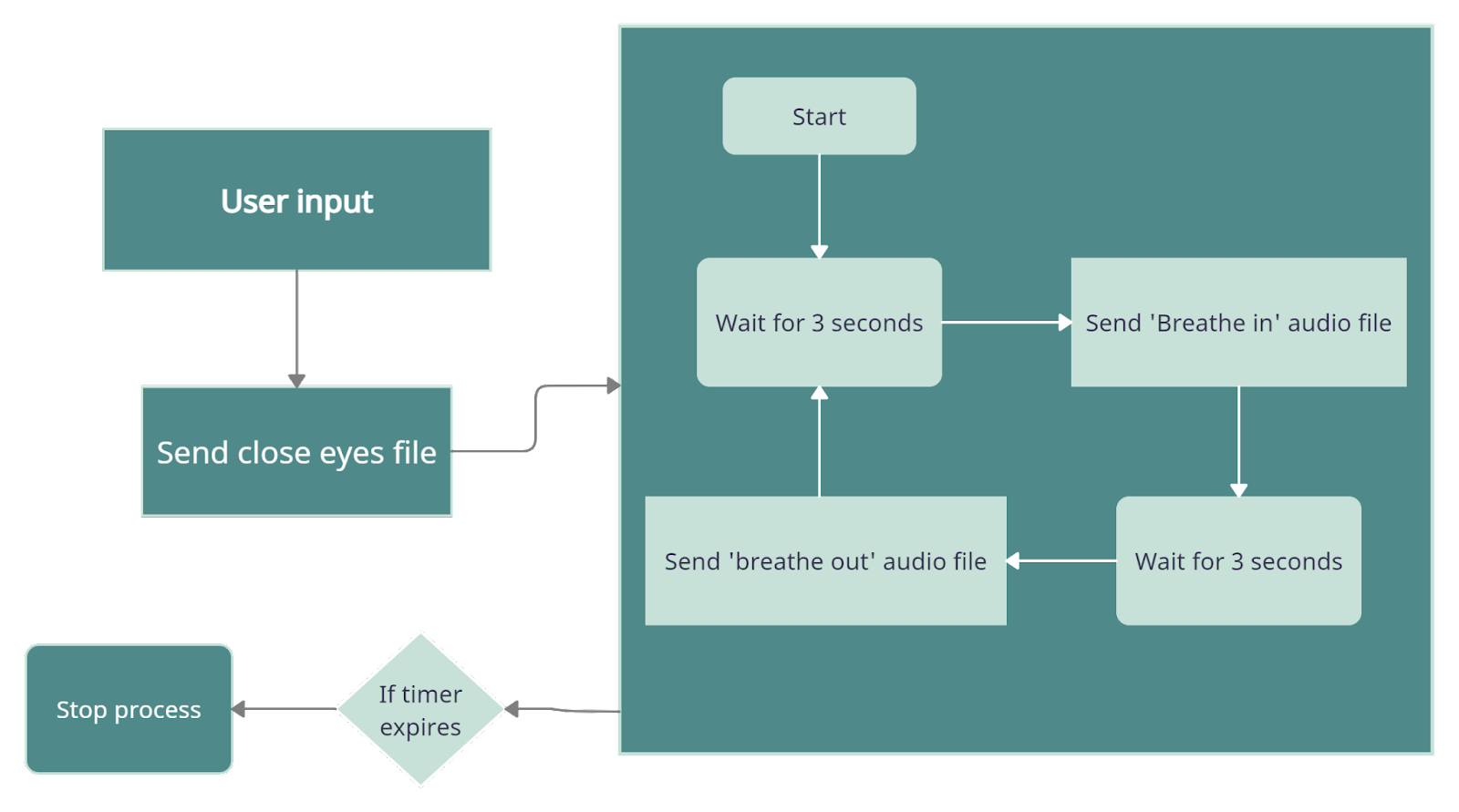


Figure 5.4 Meditation coach architecture

## ALGORITHMS

This application uses CNN for yoga assistant module with adaptive moment estimation algorithm and Euclidean algorithm

## Yoga Assistant Module

Step 1 : Reading URLs and download the images

Step 2 :Read each image and retrieve the landmarks using mediapipe.

Step 3 :The x an y coordinates of each landmark in each image are written to a CSV file

Step 4 :The data in the CSV file is read and reshaped to 33 x 2 arrays and appended to form the training and test data.

Step 5 : The labels for each image is written to a array of length 82 where only the index specifying the label is ‘1’ and all other elements are ‘0’[22]

Eg: For Pose of index 0 ⇒ [1,0,0,0, 0] (1 x 82)

Step 6 :The training and test data are split in the ratio of 8 **:** 2

Step 7 : CNN algorithm is used for prediction of Poses in this module

## Sequential Model + Layers

The sequential model[13] is built with two 2D Convolution layers with kernel size of (1,1) and Relu activation function. The layers within the sequential model are sequentially arranged. The input size of the first Convolution layer is (None,33,2,1). The model contains 3 Dense layers with a Dropout of 50%. The final Dense layer uses Softmax activation function. The model is compiled using Adam optimizer and categorical cross entropy loss function and 0.1 loss weight. The model is trained with the training data with 50 epochs and 10 steps per epoch. The trained model is converted to a tflite model and loaded in the web app.

The input layer of this model is a 2D convolution layer[12] with 1x33x2x1 input shape, 3x1x1 filter kernel and bias vector default to 3. The second layer is also a Conv2D layer with the same attributes.[25] The output after conv2D layers is of 1x33x2x3 shape. The output from the previous layer is flattened to a 1D array of size 1x198. The sequential model contains four fully connected dense layers. Fully Connected layers in a neural network are those layers where all the inputs from one layer are connected to every activation unit of the next layer. The flattened 1D array is passed to the first fully connected layer. The first fully connected layer contains 1000x198 activation weights and a bias value of 1000. The output of this layer is of shape 1x1000. The next fully connected layer contains 700x1000 activation weights and a bias value of 700. The output of this layer of shape 1x700 is passed to the next fully connected layer with 500x700 activation weights and bias of 500. The output is passed to the last fully connected layer with 82x500

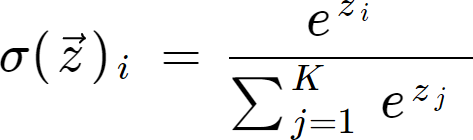
weights and bias of 82. This is the output layer and the 82 values represent the probability of each yoga asana[19]. All the fully connected layers use ReLu activation function, the final fully connected layer also the output layer uses softmax activation function. This model uses adam optimizer [17] for optimizing the loss of the model

ReLu activation function,

R(z)=> { z , z>0

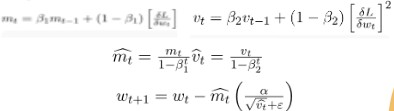
0 , otherwise}

Softmax activation to perform multiclass classification.



## Adaptive Moment(Adam) Estimation algorithm

It is used for optimization



## Exercise Trainer Module

Step 1: θ is the angle made by three points (x1,y1), (x2,y2), (x3,y3) Step 2: θ(𝑖𝑛 𝑑𝑒𝑔𝑟𝑒𝑒𝑠) = θ(𝑖𝑛 𝑟𝑎𝑑𝑖𝑎𝑛𝑠) ×  180

Π

Step 3: If θ is greater then defined angle set stage as down

Step 4: If θ is smaller then defined angle and previous stage was down then set

stage as up and increment counter

**SYSTEM IMPLEMENTATION**

## Client-Side Coding Meditation.html

<div class="menu">

## CHAPTER 6 SYSTEM IMPLEMENTATION

<input required type="number" class="duration" placeholder="Minutes" name="minutes" />

<button class="start" value="Start" onclick="meditateStart();"> Start

</button>

<button class="stop" value="Stop" onclick="meditateStop();"> Stop

</button>

</div>

<script type="text/javascript"> const meditateStop = () => { fetch("/meditation\_stop", {

method: "POST",

headers: { "Content-type": "application/json" },

}).then((r) => { clearInterval(iterator); clearInterval(getAudioFile);

});

};

const meditateStart = () => { audioFilePath = 0;

duration = document.getElementsByName("minutes")[0].value; fetch("/meditation\_start", {

method: "POST",

headers: { "Content-type": "application/json" }, body: JSON.stringify({ minutes: duration }),

})

.then((response) => response.text())

.then((responseText) => {

iterator = setInterval(secondsIterator, 1000); getAudioFile = setInterval(() => { fetch("/getAudio", {

method: "GET",

})

.then((response) => response.text())

.then((responseText) => { responseText = responseText.trim();

if (audioFilePath != responseText && responseText.length != 0) { audioFilePath = responseText; document.getElementById("audioFile").src = responseText;

}

});

}, 500);

});

};

</script>

## yoga.html

const startStreaming = () => {

if (navigator.getUserMedia) { navigator.getUserMedia(

{

video: true, audio: false,

},

(stream) => { videoRec = true;

videoElement = document.getElementById("video"); videoElement.srcObject = stream;

webcam = stream;

iterator = setInterval(sendFrame, interval);

},

(error) => {

errorElement.text = "not Supported";

});

}

};

const stopStreaming = () => { clearInterval(iterator); video.srcObject.getTracks()[0].stop();

context.clearRect(0, 0, canvas.width, canvas.height); fetch("/closemodel", {

headers: { "Content-type": "application/json" }, method: "GET",

}).then((response) => console.log(response));

};

const sendFrame = () => {

context.drawImage(video, 0, 0, canvas.width, canvas.height); inputImageData = canvas.toDataURL("image/png");

if (canFetch) { fetch(fetchUrl, { method: "POST",

headers: { "Content-type": "application/json" }, body: JSON.stringify({

data: inputImageData, pose: pose,

}),

})

.then((response) => response.json())

.then((responseData) => { if (videoRec) {

document.getElementById("landmark").src = "data:image/png;base64," + responseData.data; document.getElementById("score").innerText = responseData.score + "%"; document.getElementById("error").innerText = responseData.pose;

}

})

.catch((error) => { canFetch = true;

document.getElementById("landmark").src = "data:image/" + inputImageData;

});

}};

</script>

## exercise.html

<script type="text/javascript"> const startStreaming = () => { if (navigator.getUserMedia) {

navigator.getUserMedia(

{

video: true, audio: false,

},

(stream) => {

videoElement = document.getElementById("video"); videoElement.srcObject = stream;

webcam = stream;

iterator = setInterval(sendFrame, interval);

},

(error) => { errorElement.text = "not Supported"; });

}};

const stopStreaming = () => { video.srcObject.getTracks()[0].stop(); clearInterval(iterator);

context.clearRect(0, 0, canvas.width, canvas.height); document.getElementById("landmark").src =

"{{ url\_for('static',filename='images/detection\_default.png')}}";

};

const sendFrame = () => {

context.drawImage(video, 0, 0, canvas.width, canvas.height); imageInputData = canvas.toDataURL("image/png");

if (canFetch) { fetch("/trainerWeb", {

headers: { "Content-type": "application/json" }, method: "POST",

body: JSON.stringify({data: imageInputData,count: count,stage: stage,ex: exercise }),

})

.then((response) => response.json())

.then((responseData) => { canFetch = true;

if (videoRec) { document.getElementById("landmark").src = "data:image/png;base64," + responseData.data; var count = document.getElementById("score"); count.dataset.count = responseData.count; count.dataset.stage = responseData.stage;

}

})

.catch((err) => {

document.getElementById("landmark").src = imageInputData;

});

}};

</script>

## Server-side coding App.py

def breatheIn(): breatheInFilePath=url\_for("static",filename="audio/breathe\_in.mp3") audioData=open("static/audio.txt","w") audioData.write(breatheInFilePath)

time.sleep(3) def breatheOut():

breatheOutFilePath=url\_for("static",filename="audio/breathe\_out.mp3") audioData=open("static/audio.txt","w") audioData.write(breatheOutFilePath)

time.sleep(3)

def meditationStart(duartionMinutes): global secondsCounter,thread,audioData secondsCounter=0 duartionMinutes=int(duartionMinutes) durationSeconds=duartionMinutes\*60 secondsCounter=secondsCounter+2 isTimeOut=True while(isTimeOut==True and thread):

breatheIn() breatheOut()

if(secondsCounter>=durationSeconds): isTimeOut=False

@app.route("/getAudio") def getAudio():

global audioData

audioData=open("static/audio.txt","r") audioFilePath=audioData.read() audioData.close()

return audioFilePath @app.route("/meditation\_start",methods=["POST"]) def meditationStartRequest():

global thread if(request.method=="POST"):

durationInput=request.json["minutes"] thread=True

userThread=Thread(target=meditationStart,args=(durationInput,)) userThread.start()

return "Meditation Started"

## Exercise.py

def bicepCurl(self,image,stage=None,counter=0): try:

image = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)

With self.mp\_pose.Pose( min\_detection\_confidence = 0.5, min\_tracking\_confidence = 0.5 ) as self.pose\_model

results = self.pose\_model.process(image) landmarks = results.pose\_landmarks.landmark

shoulder = [landmarks[11].x, landmarks[11].y] elbow = [landmarks[13].x,landmarks[13].y] wrist = [landmarks[15].x,landmarks[15].y]

angle = self.calculateAngle(shoulder, elbow, wrist) if angle > 160:

stage = "down"

if angle < 30 and stage =='down': stage="up"

counter +=1

def calculateAngle(self,shoulder,elbow,wrist): shoulder = np.array(shoulder)#first elbow = np.array(elbow)#mid

wrist = np.array(wrist)#end

radians = np.arctan2(wrist[1] - elbow[1], wrist[0] - elbow[0])

-np.arctan2(shoulder[1] - elbow[1], shoulder[0] - elbow[0]) angle = np.abs(radians\*180.0/np.pi)

if angle > 180.0: angle = 360-angle

return angle

def calculateDistance(self,leftWrist,rightWrist): squareOfComponentsInX= (rightWrist[0]-leftWrist[0])\*\*2 squareOfComponentsInY= (rightWrist[1]-leftWrist[1])\*\*2

sumOfComponents = squareOfComponentsInX + SquareOfComponentsInY distance= math.sqrt(sumOfComponents)

return distance

## Yoga.py

def predict(self,i,inputImage): try:

inputImage = cv2.cvtColor(inputImage, cv2.COLOR\_BGR2RGB)

mediapipeResults = self.mediapipeModel.process(inputImage) landmarks = mediapipeResults.pose\_landmarks.landmark inputArray=[]

for landmark in landmarks: inputArray.append([landmark.x,landmark.y])

self.interpreter.set\_tensor(self.inputIndex, inputArray) self.interpreter.invoke()

predictedResult = self.interpreter.get\_tensor(self.outputIndex) if(i<0):

i= predictedResult[0].argmax()

return self.poses[i],predictedResult[0][i],inputImage except Exception as e:

return "none",0,False **modelTraining.py** model=Sequential()

model.add(Conv2D(filters=3,kernel\_size=(1,1),input\_shape=(33,2,1),activation="r elu"))

model.add(Conv2D(filters=3,kernel\_size=(1,1),activation="relu")) model.add(Flatten())

model.add(Dense(1000,activation='relu')) model.add(Dense(700,activation='relu')) model.add(Dropout(0.5)) model.add(Dense(500,activation='relu')) model.add(Dense(len(labels),activation="softmax")) model.summary()

model.compile(optimizer='adam',loss='categorical\_crossentropy',metrics=['categori cal\_accuracy'],loss\_weights=0.1)

hist = model.fit(train\_target, train\_output, steps\_per\_epoch=10, epochs=50, validation\_data=(test\_target,test\_output), validation\_steps=5)

**SYSTEM TESTING**

## 7.1 Unit Testing

**CHAPTER 7** **SYSTEM TESTING**

Unit testing involves the testing of each unit or an individual component of the software application. It is the first level of functional testing. The aim behind unit testing is to validate unit components with its performance. The purpose of unit testing is to test the correctness of isolated code. A unit component is an individual function or code of the application. In a testing level hierarchy, unit testing is the first level of testing done before integration and other remaining levels of the testing.

Table 7.1. Unit testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Module** | **Testing feature** | **Input** | **Acquired Output** | **Expected Output** |
| Meditation | Start meditation | Duration =1 | Change audio file at 3 seconds interval  for 1 minute | Audio file must be rewritten for every 3 seconds |
| Get audio path | /getAudio | Return “static/audio/breath e\_in.mp3” | Return audio file path |
| Exercise | Start stream | Video input | Count - 1 stage - up | Return count and stage of exercise |
| Error input | Incorrect video input | Count - 0 Stage - down | Return previous count and stage |
| Yoga | Image input | Tree pose image And random option selected | Pose - Tree\_Pose\_or\_Vrks asana\_  Accuracy - 98.67% | Return pose Tree pose |

## Integration Testing

Integration testing is the second level of the software testing process comes after unit testing. In this testing, units or individual components of the software are tested in a group. The focus of the integration testing level is to expose defects at the time of interaction between integrated components or units. Once all the components or modules are working independently, then we need to check the data flow between the dependent modules is known as integration testing.

Table 7.2. Integration testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Module** | **Testing feature** | **Input** | **Acquired Output** | **Expected Output** |
| App | Select meditation coach feature | Get request  /meditation | Meditation page rendered | Render meditation page |
| Select exercise assistant feature | Get request  /exercise | Exercise main page rendered | Render exercise selection page |
| Select yoga trainer feature | Get request  /video | Yoga trainer page rendered | Render yoga trainer page |
| Meditation | Meditation start | Post request  /meditation\_start  {duration:1} | Thread started | Thread start |
| Meditation Stop | Post Request  /meditation\_stop | Thread stopped | Thread stop |
| Exercise | Selecting Exercise | Get request  /trainer?exercise  =curl&index=1 | Page for Bicep curl is rendered | Render Bicep curl html page |
| Yoga | Selecting yoga pose | Post request  /pose  {pose: 76} | Standard image of Warrior\_I returned | Return Warrior\_I image |

## Test Cases & Reports

Table 7.3 Test Cases

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test case Id** | **Description** | **Steps** | **Input data** | **Actual Output** | **Expected Output** | **Pas s/Fa il** |
| 1 | Starting meditation for 1 minute | 1. Select meditation icon 2. Enter duration in given input box 3. Click Start button | Duration - 1 | Audio “Breathe in” 3 seconds pause “breathe out” for 1 minute in loop | Audio “Breathe in” 3 seconds pause “breathe out” | Pass |
| 2 | Do Bicep curl exercise correctly twice | 1. Select exercise icon 2. Select waiter curl image 3. Click next button and start button | Webcam video    Figure 7.1 Exercise input | Counter - 2 Stage - down    Figure 7.2 Exercise output | Counter - 2 Stage - down | Pass |
| 3 | Do Waiter curl exercise wrongly twice | 1. Select exercise icon 2. Select waiter curl image 3. Click next button and start button | Webcam video | Counter - 0 Stage - down | Counter - 0 Stage - down | Pass |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 4 | No person in input of exercise trainer | 1. Select exercise icon 2. Select waiter curl image 3. Click next button and start button | Webcam video | Counter - 0 Stage - down | Counter - 0 Stage - down | Pass |
| 5 | Perform Tree yoga pose correctly | 1. Select yoga icon 2. Select warrior\_I from list 3. Click start button | Webcam video    Figure 7.3 Yoga input | Pose - Tree\_Pose Accuracy - 98.9%    Figure 7.4 Yoga output | Pose - Tree\_Pose Accuracy - 100%  (>85%) | Pass |
| 6 | Perform Warrior\_I yoga pose incorrectly | 1. Select yoga icon 2. Select warrior\_I from list 3. Click start button | Webcam video | Pose - Warrior-I Accuracy  -24.9% | Pose - Warrior\_I Accuracy - 0% (<30%) | Pass |
| 7 | No person in input of yoga assistant | 1. Select yoga icon 2. Select warrior\_I from list 3. Click start button | Webcam video | Pose - Warrior-I Accuracy - 0% | Pose - Warrior\_I Accuracy - 0% | Pass |

## 7.4 Results & Discussions

Zyofisik is a Flask web app containing a Meditation Coach, Exercise Trainer and Yoga assistant. The system provides accurate scores for the yoga asanas with 100 as maximum score. The exercise counter takes into account only if the exercise is done correctly which helps the user to do the exercise properly. The exercise module currently contains 5 exercises counter. The number of exercises will be increased in future advancement. The web app is deployed in heroku cloud. The client sends the frame to the server for every 500 milliseconds in the exercise trainer and for every second in the yoga assistant when the application is deployed in the cloud server. While running the application as software or in localhost the frames will be sent for every 100 milliseconds in the exercise trainer and 300 milliseconds in yoga trainer.

The exercise trainer contains 5 exercises - Dumbbell curl, Waiter curl, Hammer curl, Squats, Push-ups. The instructions to be followed for each exercise along with a tutorial will be provided to the user. The application only counts the exercise if it is done correctly and it also shows which position the user is currently in.



Figure 7.5 Dumbbell curl

Figure 7.6 Waiter curl

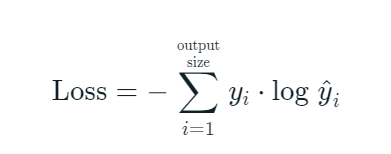


Figure 7.7 Hammer curl



Figure 7.8 Push-ups

The yoga trainer can give accuracy for 82 yoga poses using a CNN model. The CNN model is built using a Sequential model. The model gives an accuracy of 98.9% on test data. The yoga 82 dataset used for training and testing contains an average of 347 images per asana. The data is split as 80:20 ratio for training and testing respectively. The below figures shows the categorical accuracy and loss of the CNN model. The categorical crossentropy loss function calculates the loss of an example by computing the following sum:



The accuracy is calculated by dividing the total number of correct predictions by the total number of data in the test dataset passed for validation which is 20% of the yoga-82 dataset

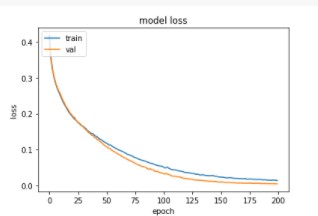
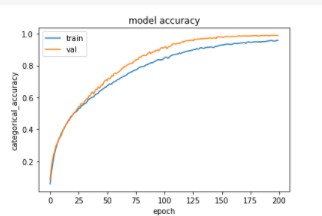


Figure 7.9 CNN model accuracy graph Figure 7.10 CNN model loss graph

The landmarks of the user input is retrieved which is passed to the CNN model which is previously trained. The user is also provided with reference images with the points which are considered for performing the asana correctly. The below figures show the reference images of a few yoga asanas.



Figure 7.10 Warrior - II pose



Figure 7.11 Tree pose

**CONCLUSION**

# CHAPTER 8 CONCLUSION

## 8.1 Conclusion and Future Enhancements

Transitioning to Modern fitness, Zyofisik suggests a system that can classify 82 yoga poses with an accuracy of 98.98%, accurately assesses the exercises performed to aid the user performing the workouts as they can revise their posture accordingly, and instructs the user to maintain their breathing pace using a programmed meditation coach. Currently there are 5 exercises added in the application. In the future more exercises will be added for various muscles. The current size of the application is restricted to 500MB due to the cloud limit in cloud platform. The application will be deployed as a software to install in desktops and also as mobile application in the future to increase the ease of use.

## APPENDICES

## Sample Screens



Figure A1.1 Meditation Coach Sample Screen

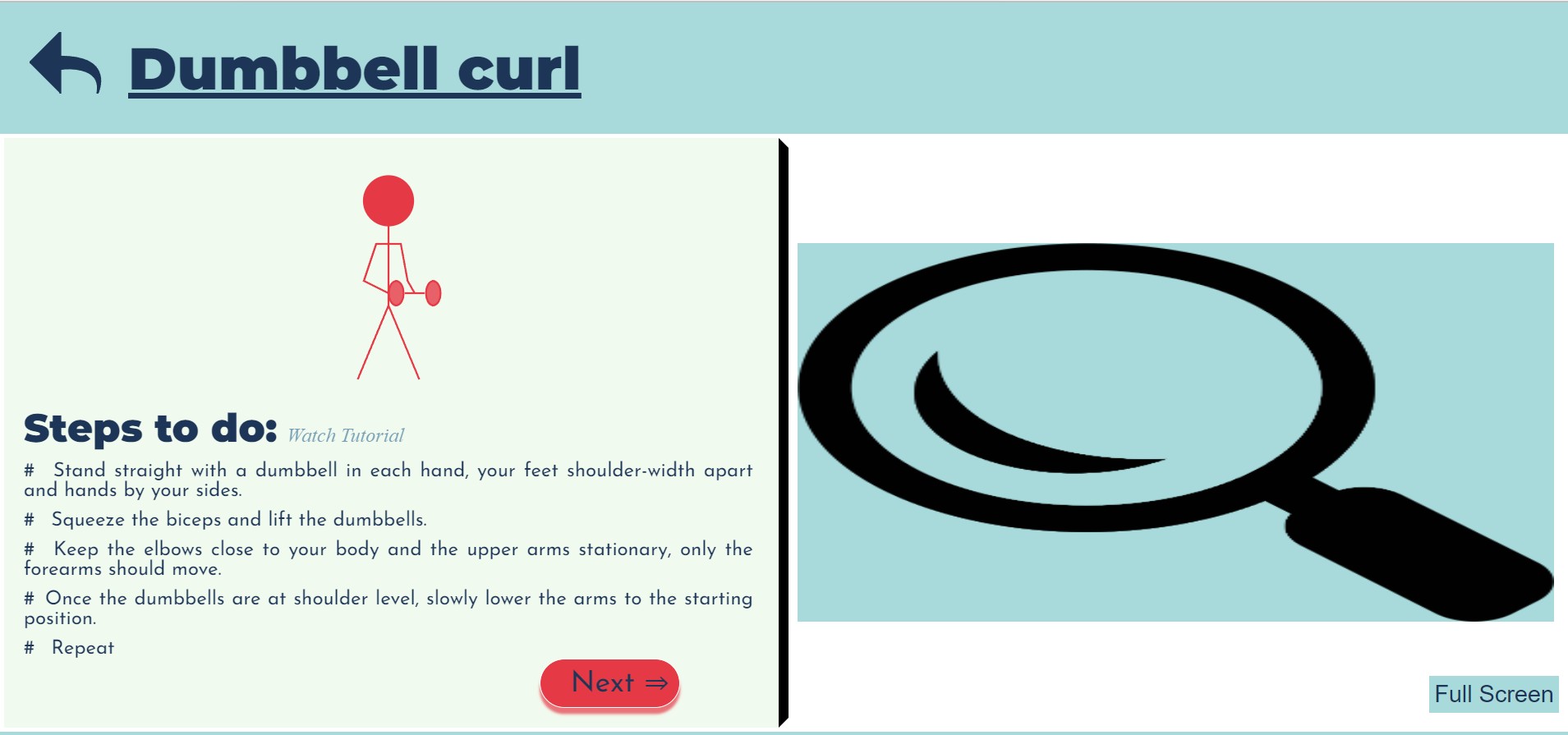
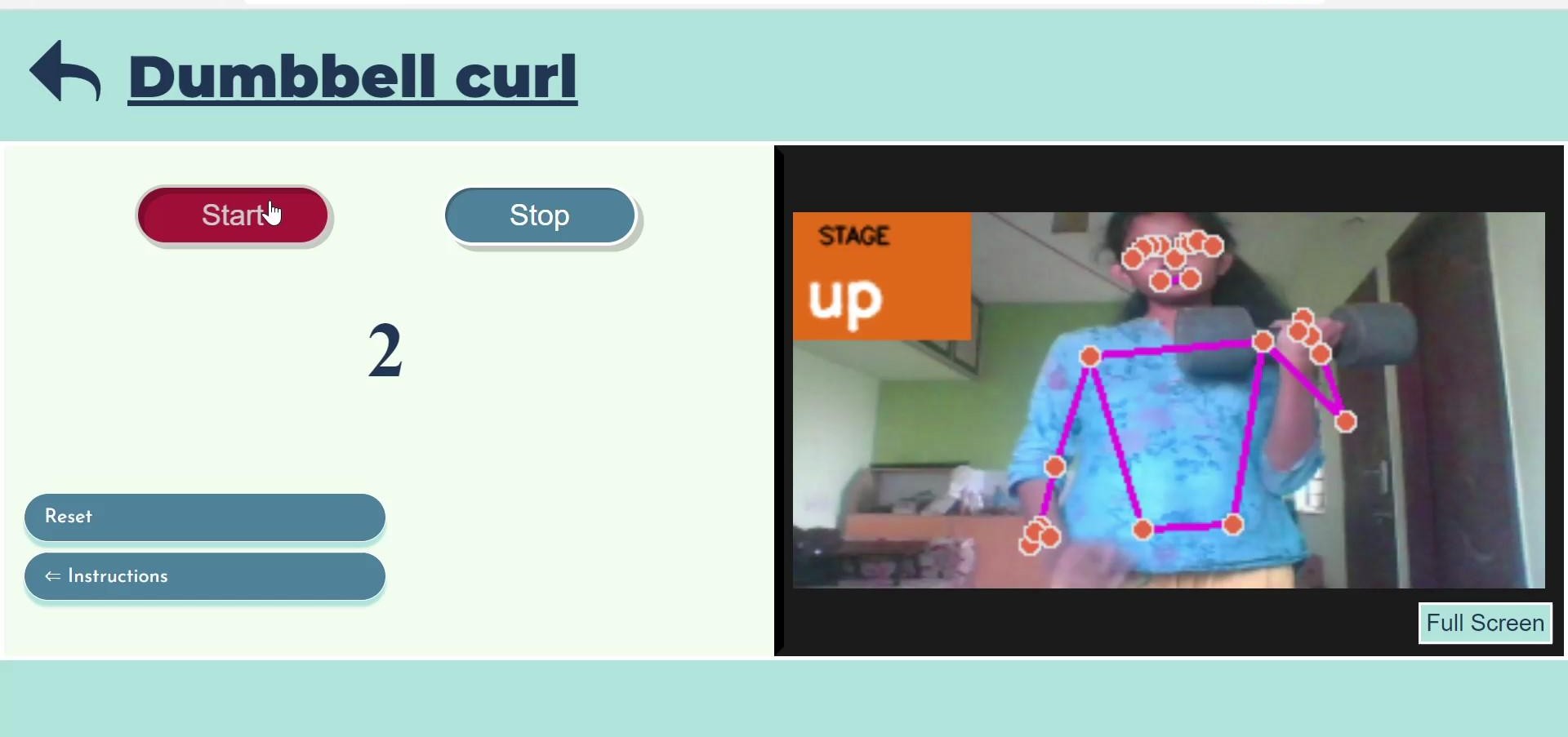


Figure A1.2 Exercise Trainer Sample Screen -1



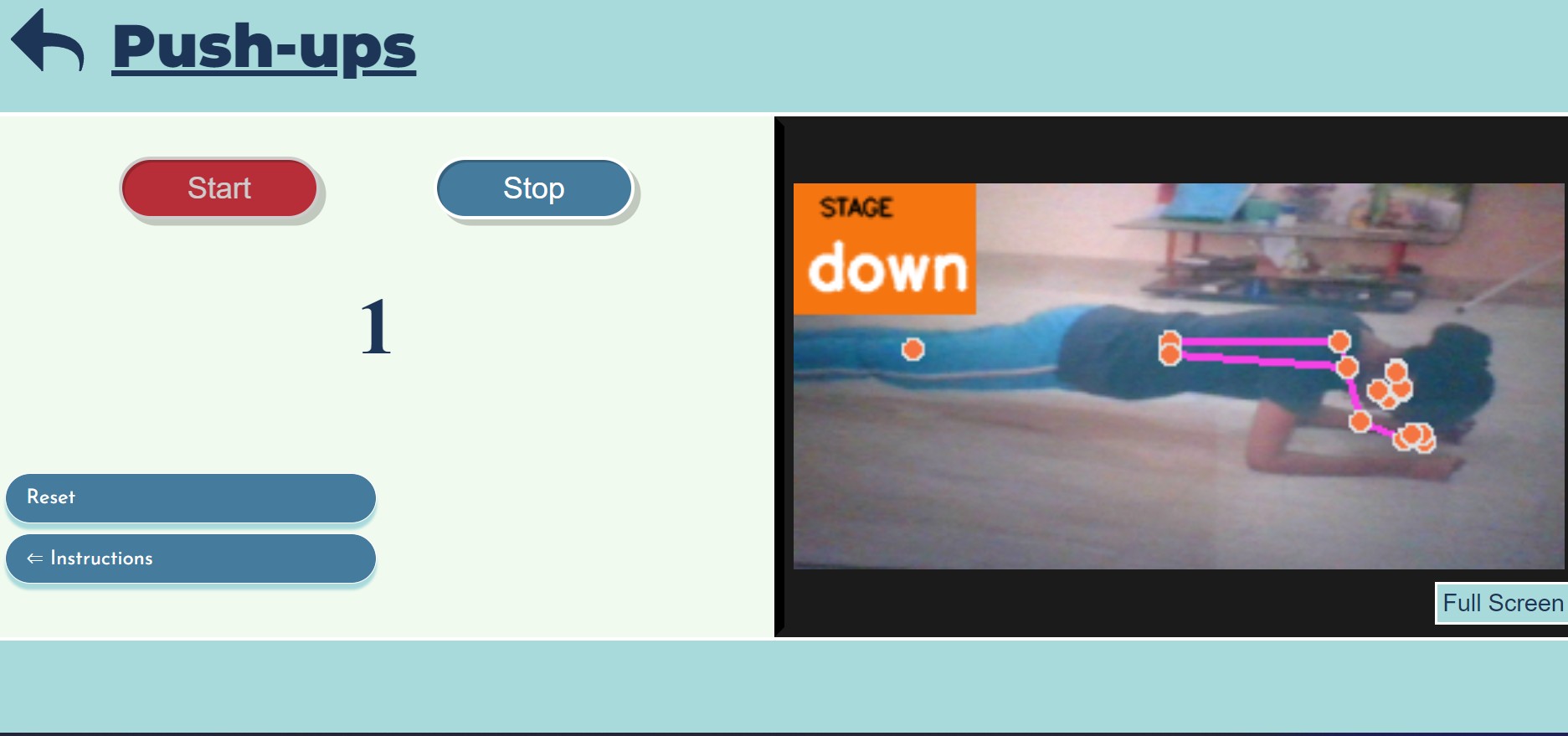
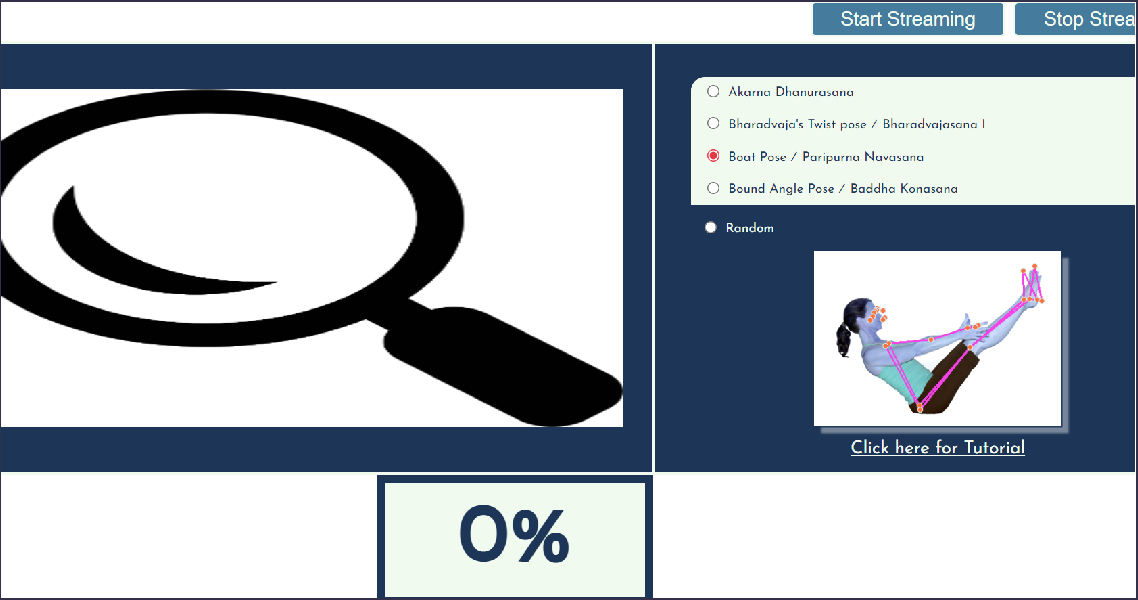
Figure A1.3 Exercise Trainer Sample Screen -2

Figure A1.4 Exercise Trainer Sample Screen - 3



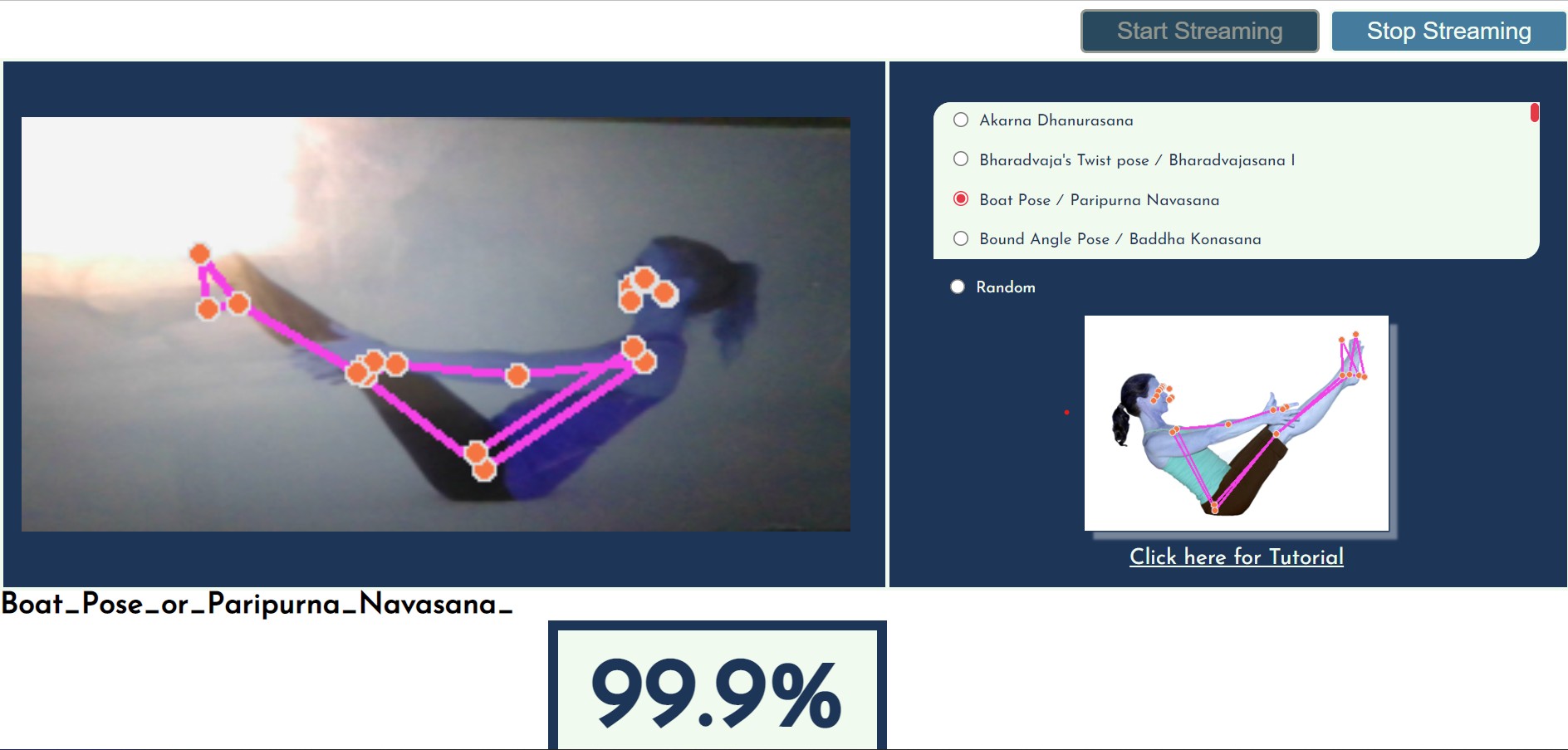
Figure A1.5 Yoga Assistant Sample Screen - 1

Figure A1.6 Yoga Assistant Sample Screen - 2

## REFERENCES

* + 1. Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays,Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, Wan-Teh Chang, Wei Hua,Manfred Georg and Matthias Grundmann, (2019), “MediaPipe: A Framework for Perceiving and Processing Reality”, Computer Vision for AR/VR at IEEE Computer Vision and Pattern Recognition (CVPR).
    2. Chen, H., He, Y., & Hsu, C., (2018), “Computer-assisted yoga training system”, Multimedia Tools and Applications, 77, 23969-23991.
    3. Chhaihuoy Long, Eunhye Jo, and Yunyoung Nam, (2022), “Development of a yoga posture coaching system using an interactive display based on transfer learning” - J Supercomputer 78, 5269–5284
    4. Deepak Kumar, and Anurag Sinha, (2020), “Yoga Pose Detection and Classification Using Deep Learning” International Journal of Scientific Research in Computer Science, Engineering and Information Technology, Volume 6 Issue 6 Page: 160-184.
    5. Girija Gireesh Chiddarwar, Abhishek Ranjane, Mugdha Chindhe, Rachana Deodhar, and Palash Gangamwar, (2020), “AI-Based Yoga Pose Estimation for Android Application” International Journal of Innovative Science and Research Technology, Volume 5 - Issue 9.
    6. M. Verma, S. Kumawat, Y. Nakashima and S. Raman, (2020), "Yoga-82: A New Dataset for Fine-grained Classification of Human Poses", IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) pp. 4472-4479
    7. Nagalakshmi Vallabhaneni and Dr. P. Prabhavathy, (2021), “The Analysis of the Impact of Yoga on Healthcare and Conventional Strategies for Human

Pose Recognition”, Turkish Journal of Computer and Mathematics Education (TURCOMAT) .,vol. 12 no. 6, pp. 1772–1783

* + 1. Nuruldelmia Idris, Cik Feresa Mohd Foozy, and Palaniappan Shamala, (2020), “A Generic Review of Web Technology: DJango and Flask”, International Journal of Advanced Computing Science and Engineering ISSN 2714-7533 Vol. 2, No. 1
    2. T. K. K. Maddala, P. V. V. Kishore, K. K. Eepuri and A. K. Dande, (2019), "YogaNet: 3-D Yoga Asana Recognition Using Joint Angular Displacement Maps With ConvNets," IEEE Transactions on Multimedia, vol. 21, no. 10, pp. 2492-2503, Oct. 2019
    3. Wu, Y., Lin, Q., Yang, M., Liu, J., Tian, J., Kapil, D.P., & Vanderbloemen, L. (2021). “A Computer Vision-Based Yoga Pose Grading Approach Using Contrastive Skeleton Feature Representations.” Healthcare (Basel), 10(1):36
    4. Yash Agrawal, Yash Shah, and Abhishek Sharma, (2020), “Implementation of Machine Learning Technique for Identification of Yoga Poses”, 9th IEEE International Conference on Communication Systems and Network Technologies, pp. 40-43
    5. L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. V. Gool, (2016) “Temporal segment networks: Towards good practices for deep action recognition,” in Computer Vision – ECCV 2016. Springer International Publishing, pp. 20–36
    6. Y. Zhang, B. Zhang, H. Yang, M. Norambuena and J. Rodriguez, (2019) "Generalized Sequential Model Predictive Control of IM Drives With Field-Weakening Ability," in IEEE Transactions on Power Electronics, vol. 34, no. 9, pp. 8944-8955
    7. E. Visceglia and S. Lewis, (2011) “Yoga therapy as an adjunctive treatment for schizophrenia: A randomized, controlled pilot study,” The Journal of Alternative and Complementary Medicine, vol. 17, no. 7, pp. 601–607
    8. D. Weinland, R. Ronfard, and E. Boyer, (2011) “A survey of vision-based methods for action representation, segmentation and recognition”, Computer Vision and Image Understanding, vol. 115, no. 2, pp. 224–241
    9. C. Zhang and Y. Tian, (2012) “Rgb-d camera-based daily living activity recognition,” Journal of Computer Vision and Image Processing, vol. 2, no. 4, p. 12
    10. S.Mehta, C.Paunwala and B.Vaidya, "CNN based Traffic Sign Classification using Adam Optimizer," (2019) International Conference on Intelligent Computing and Control Systems (ICCS), 2019, pp. 1293-1298
    11. S. Ji, W. Xu, M. Yang, and K. Yu, (2013) “3d convolutional neural networks for human action recognition,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 1, pp. 221–231
    12. M.-Y. Wang, G. A. Greendale, S. S.-Y. Yu, and G. J. Salem, (2016) “Physical Performance outcomes and biomechanical correlates from the 32-week yoga empowers seniors study,” Evidence-Based Complementary and Alternative Medicine, vol. 2016, pp. 1–10
    13. V. Chunduru, M. Roy, D. R. N. S and R. G. Chittawadigi, (2021) "Hand Tracking in 3D Space using MediaPipe and PnP Method for Intuitive Control of Virtual Globe," IEEE 9th Region 10 Humanitarian Technology Conference (R10-HTC), 2021, pp. 1-6.
    14. Patil S, Pawar A, Peshave A et al (2011) “Yoga tutor: visualization and analysis using SURF algorithm.”, Proceedings of 2011 IEEE control system graduate research colloquium, ICSGRC 2011, pp 43–46
    15. Poppe R (2010) “A survey on vision-based human action recognition.” Image Vis Comput 28:976–990.
    16. H. Chen, Y. He, C. Chou,(2013) “Computer assisted self-training system for sports exercise using kinetics”, IEEE Intl. Conf. Multimedia and Expo Work.
    17. T. G. Mattson, T. A. Anderson and G. Georgakoudis, (2021) "PyOMP: Multithreaded Parallel Programming in Python," in Computing in Science & Engineering, vol. 23, no. 6, pp. 77-80, 1
    18. Manap MS, Sahak R, Zabidi A, Yassin I, Tahir NM (2015) “Object detection using depth information from Kinect sensor.”, 2015 IEEE 11th International Colloquium on Signal Processing & Its Applications. pp 160–1
    19. Reddy MS, Venkatramana SA, Ramji BR (2015) “e-Yoga prescription designed for computer users using e-Yoga environment for posture.”, Proceedings of the 3rd International Congress on Sport Sciences Research and Technology Support, pp. 48–52
    20. Santosh Kumar Yadav, Amitojdeep Singh, Abhishek Gupta, and Jagdish Lal Raheja(2019) “Real-time yoga recognition using deep learning” Neural Comput. Appl., pp1–13
    21. Wikipedia: List of asanas. https://en.wikipedia.org/wiki/List\_of\_asanas. 29.Yoga journal. https://www.yogajournal. com/poses