| PROJECT REQUIREMENTS SPECIFICATION  Computer Vision Based Workout Application  UE19CS390A – Capstone Project Phase – 1  ***Submitted by:***   | **Tejas D R**  **Vishnu J G**  **Srujan A S**  **Pradeep V** | **PES1UG19CS537**  **PES1UG19CS574**  **PES1UG19CS509**  **PES1UG19CS330** | | --- | --- |   Under the guidance of   | **Prof. Savithri S.**  Designation  PES University | | --- |   **January–May 2022**  **DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**  FACULTY OF ENGINEERING  **PES UNIVERSITY**  (Established under Karnataka Act No. 16 of 2013)  100 Feet Ring Road, Bengaluru – 560 085, Karnataka, India |
| --- | --- | --- | --- |

TABLE OF CONTENTS

| 1. Introduction | 3 |
| --- | --- |
| 1.1 Project Scope | 3 |
| 1. Product Perspective | 3 |
| 2.1 Product Features | 4 |
| 2.2 Operating Environment | 4 |
| 2.3 General Constraints, Assumptions and Dependencies | 4 |
| 2.4 Risks | 5 |
| 1. Functional Requirements | 5 |
| 1. External Interface Requirements | 5 |
| 4.1 User Interfaces | 5 |
| 4.2 Hardware Requirements | 6 |
| 4.3 Software Requirements | 6 |
| 4.4 Communication Interfaces | 6 |
| 1. Non-Functional Requirements | 7 |
| 5.1 Performance Requirements | 7 |
| 5.2 Safety Requirements | 7 |
| 5.3 Security Requirements | 7 |
| Appendix A: Definitions, Acronyms and Abbreviations | 7 |
| Appendix B: References | 8 |

# Introduction

This document involves all the basic pre-requisites for the Computer Vision Based Workout application. It involves both functional and non-functional requirements, the usage, the compatability with different environments, the interface, software and hardware requirements, risks and safety and security measures. This document is to make the reader aware of the applications overall and workarounds in case of any failure. It also helps in troubleshooting and possible reasons for failure of application i.e., in case the device on which the application is being run lacks in any requirement.

# Project Scope

The project is a Computer Vision based workout application. It entails visual and audio guides for user to carry out exercises without having to come in contact with the devices such as mouse and keyboard to control the workout application.

The goal of this project is to make it possible for the user to workout without any human interation and bare minimum contact with the device.

The project will be able to give real time results pertaining to the user’s exercises poses which include corrections and analysis.

Few limitations would be pertaining to the side of the user facing the camera, for simplicity we have considered only one side of the user would be facing the camera at all points of time in a particular exercise. The camera would not be able percive body posture of the hidden side of the user.

Another limitation would be the present plan for the application can only support single user and does not share user data with other users, on the same device or otherwise.

# Product Perspective

The idea of the workout application comes from the popular gaming console XBox, which allows users to play without using any tactical device, it only requires a specialised infrared camera to percieve the user’s actions.

The idea of XBox is the get user’s current actions and not maintain any state of the same. But if some how we could store the state and some other information along with it, we will be able to extract insights from it.

# Product Features

The Computer Vision based Workout Application entails the following:

1. Use the camera of the user’s device to get and understand the user’s posture and map it with predefined postures, this is done to check errors in the user’s postures.
2. Give audio/visual feedback to the user in case of any errors. The results from the above function will helps us give appropriate correction feedbacks to the user.
3. Maintain state of the user’s exercise. This is for analysis purposes. Count the number of times exercise was executed, report weekly or time based results to the user.
4. For easing the user’s interaction with the application, use hand gestures to control the application so that the user need not have to come closer to the device, user mouse/keyboard to control the application. The gestures based actions include pause, play, next, exit, skip, etc,.

# Operating Environment

Presently the plan is to build with a computer with/without a GPU(non necessary). The reasons are majorly because of the models for each exercise that we may have to build and process in real time. For smoother working GPU is recommended.

Also to be able to control mouse and its actions currently aren’t available as per our literature survey. To combat this we have planned two approaches : Either to map gestures to actions rather than be able to control mouse OR build a software application that can be executed by the user on their system. The software system will be build on Windows and supports systems even without a GPU.

The only other hardware componenet in consideration is the camera, which need not necessary be high quality camera but just enough to identify the user without any blur or distortion.

# General Constraints, Assumptions and Dependencies

Since mouse control through website is perticularly considered not safe, we will have to stick to a software like application for ease of implementation. The other alternative would be to map the users gestures to specific functionalities so that we will not be needing to use mouse. The formulation of the gestures and mapping them to actions will be an overhead.

The application works smoother with a GPU, which means in the lack of which the results maybe delayed. To overcome this there are 2 options: Either drop some frames and select only those frames that are critical for analysis OR store all frames and give a summarised result after processing all of them(but in this we lose the ability to give real time results).

The application runs 2 operations in parallel at all times, one which check for errors and gives feedbacks, the other which just stores states and use this to generate insightful analytical results to the user.

One important assumption made is that the user always faces one side through out one exercise. Other assumption is that only one user is using the application and if new users have to use the app, they may have to reinstall the application.

# Risks

The major risk in the application is the firewall or device security which may not allow us to control the mouse pointer and camera access. Since the application does capture, display and analysis at the same time the system should be able to run all these in parallel. If in case the system reverts the access to one of the hardwares required for the application, the whole workflow will be aborted abruptly.

# Functional Requirements

The workflow goes as follows:

* The camera sends in stream of frames(video frames) to the application
* The application analyses this creates a mapping of body parts and returns the cordinates of important landmarks of the body in the frame
* This data is compared with the predefined data points and constrains, such as angle between limbs, inclination, etc,.
* The frames with the cordinates mapping is displayed to the user
* In case of an error detected, a visual/ audio feedback is returned
* If no errors detected then the appliation stores state for further analysis, state such as count or time, and discards the current frame information
* At any point during the whole execution of the program the user can control the application using hand gestures.
* Other errors such as frames with no user detected are ignored
* The application displays the user with his own frame, which acts like a mirror, and also frames showing how to go about doing the exercise accurately.

# External Interface Requirements

# User Interfaces

We have planned the application frontend as follows:

* The video of the user will be displayed on the side of another recorded video(either another professional’s video or another skeletal structure pre-recorded doing the same exercise)
* The count/timer will be displayed on one corner for the user’s reference
* The user can browse through all the available exercises using hand gestures, select, play, pause the exercises.
* The reference video will show, by some kind of indication, if the user’s poses are accurate or not.
* This analysis is expected to occur in real time. With maybe 1 sec delay at max.
* The application can also communicate through voice to guide or correct the user

# Hardware Requirements

The only hardware the is basic necessity of this application is a webcam. This hardware is used to capture the user’s actions. The connection and transfer of frames from the camera to the application is carried out by the OpenCV module of python. MediaPipe is another module which recognizes the hand gestures and gives the cordinates for few specific points on the user’s hand. OpenCV allows python programs to access the webcam of the device for as long as the program runs.

# Software Requirements

* **OpenCV2:** The version of OpenCV2 used in our application is 4.5.5, this version is compatible with python versions >2.7
* **Python :**  The main programming language used is python and the versions of the same used are 3.7.9 and 3.9.6. Although all the development plans are in 3.9.6 version, testing will be done in both the versions.
* **Mediapipe**
* Other software components may be included as and when necessary. Some planned software components include PoseNet, etc,.

# Communication Interfaces

Since our application is restricted to work in an single user configuration, we are not presently working with network transfer of any information between devices. But if our application works out to be a web based application then we may have to be able to tranfer frames from the user’s device/browser to the site where the program is running, this may cause some latency but we can have a work around where in the basic computations can be done on the user’s device.

# Non – Functional Requirements

# Performance Requirement

In a condition where high computational powered GPU is used, users can find bare minimum latency, the time to process each frame is few milli seconds. But in the case of network transfers, we may need to take into consideration the upload and download speed of the user. Besides the computation for each exercise has 2 tasks, one guiding and correction and the other storing the data for further analysis tasks. The latency of storing the data is pertaining to accessing the file where the previous data was stored and append the new data to the file. Where as the guiding and correction requires the program to access the pre-defined poses and match them with the user’s frames, post which the program has to identify which correction or guide to load in and output appropriately.

# Safety Requirements

The basic priciple of our application is that we are not saving any frames, rather just extracting important and useful information from each frame and give results. Due to this reason there are no worries of usage of excessive storage. Since the application can support only one user per device, new users have to seperately install the application or use the application on another device.

# Security Requirements

No critical information of the user is used or stored. The application uses frames to get data such as count, timing and number of exercises performed. The user data is specific to a single user and the application at present is not planned to handle multiple users on the same device.

# Appendix A: Definitions, Acronyms and Abbreviations

**MediaPipe** : MediaPipe offers cross-platform, customizable ML solutions for live and streaming media.

**OpenCV** : OpenCV is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel, it was later supported by Willow Garage then Itseez. The library is cross-platform and free for use under the open-source Apache 2 License.

**PoseNet** : Posenet is a real-time pose detection technique with which you can detect human beings' poses in Image or Video. It works in both cases as single-mode(single human pose detection) and multi-pose detection(Multiple humans pose detection).

**GPU** : Graphics processing unit, a specialized processor originally designed to accelerate graphics rendering. GPUs can process many pieces of data simultaneously, making them useful for machine learning, video editing, and gaming applications.

# Appendix B: References

* <https://pypi.org/project/mediapipe/>
* <https://opencv.org/releases/>
* <https://github.com/tensorflow/tfjs-models/tree/master/posenet>
* <https://youtu.be/brwgBf6VB0I>
* <https://youtu.be/NZde8Xt78Iw>