RESEARCH ARTICLE OPEN ACCESS

Yog-Fit: Stretch Your Limits

Atharv Tambekar*, Shravan Ghorpade**, Siddharth Vhatkar***,Sarthak More****, Sagar Chavan*****

*(Computer Science and Engineering, Sanjay Ghodawat Institute, Atigre

Email: atharvtambekar28@gmail.com)

----------**********----------

Abstract:

While researching for the project "Yog-fit" we find out that there are multiple obstacles in the features of our project. We can't fetch the data from few smartwatches of user and also third-party fetching is not allowed in few brands of smartwatches. For development of our project first we understood user requirements and according to our collected research we have analysed the data and based on analysed data we will build our project. Still our project will achieve great results. We will be still be able to track user exercises and yoga. Users can still compete with each other. Awareness about yoga and fitness exercises are our key result. In conclusion our project will help everyone to keep them fit and help them for being healthy. Our project will be useful for the people who wants to live a healthy life. Our ranking and streak features will keep user stick to the routine exercise/yoga. At the end user is the one being beneficial in physical health/form and being fit. Yoga will also help user to keep their mental health in good state.

Keywords: Yog-fit, Track, fitness, yoga , exercises, healthyness

----------**********----------

INTRODUCTION

1.1 BACKGROUND

Yog-fit app will provide the articles about the exercise and yoga where people will get knowledge about the yoga and exercise in the brief. It will also track how they are performing Exercise/yoga tracking data. User will be able to select the levels of exercise and yoga according to users' choice.
User will also be able to compete with the friends and local users.

1.2 ENVIRONMENT

This is an Mobile Application so we require Mobile App Development Environment. There are multiple environments available

Paragraph 2

1.3 USE OF PROJECT

1. To learn Yoga/Exercises.
2. To get Fit and stay Fit.
3. Track your daily yoga/exercises.
4. Analysis of your workout.

1.4 PURPOSE

1. To introduce new Technology of tracking.
2. New way to learn Yoga/Exercises.
3. To keep users stay fit.
4. Making people aware of their fitness.
5. To keep users healthy.
6. To keep user motivated through competitive ranking and streaks.

LITERATURE REVIEW

2.1 EXISTING SYSTEM

We have researched on multiple apps/websites. Websites we have researched on only provide information and step by step procedures of yoga and exercise they do not provide that much information on why should you do this exercise/yoga and is this okay to perform for your body because multiple exercise/yoga requires specific age or height or not having heart disease. Which makes user doubtful to follow instructions. What if it doesn't suite their body. There are few exercise/fitness apps from mobile manufactures itself. But they are brand specific so features like competing with friend is not famous because chances of your friend having smartphone from same brand are very low.

2.1.1 Finding of Literature

1. Lack of Personalization in Fitness Apps.
2. Insufficient Information on Exercise Purpose.
3. Health Considerations and Exercise Suitability.
4. Brand-Specific Fitness Apps and Limited Social Features.

2.2 PROPOSED SYSTEM

App can boost fitness activities and assist in keeping fit. We are developing a fitness app that can help people for workouts and yoga. The main reason for selecting this problem statement is that in day-to-day life people are ignoring their health. Their lifestyle needs this improvement so our project is targeted towards them.

Adavantages

1. Users will able to learn exercises/yoga.
2. Users will be able track his exercise/yoga.
3. Users will be consistent with exercise/yoga.
4. Users will have healthy lifestyle.

SCOPE OF PROJECT

3.1. OBJECTIVE

1. Our comprehensive fitness platform combines yoga and exercise guidance with technology to track users' performance.
2. Motivation is fueled through competitive rankings and streaks.
3. Our mission is to keep users healthy by seamlessly integrating wellness into their daily lives, making fitness both accessible and enjoyable.
4. Provide a holistic approach to fitness, addressing physical and mental well-being.
5. Foster a sense of community through shared goals and achievements.

3.2. SCOPE OF PROJECT

1. Our project targets fitness enthusiasts, emphasizing the exercise and yoga domains.
2. It aims to provide a user-friendly platform offering tailored workout routines, personalized training programs, and valuable insights.
3. By fostering a dynamic and supportive community, the project seeks to meet the specific needs of individuals dedicated to enhancing their physical well-being in the fitness realm.
4. Encourage diversity in fitness routines to cater to different preferences and goals.
5. Implement gamification elements to enhance user engagement and enjoyment.

CORE TECHNOLOGY

1) 4.1. DESIGNING

4.1.1. React-Native

4.1.1.1. WHAT IS React-Native

React Native is an open-source framework developed by Facebook for building mobile applications using JavaScript and React. It allows developers to create cross-platform apps that run on both iOS and Android platforms with a single codebase. The framework leverages the React library, enabling developers to use a familiar and declarative programming style for building user interfaces.

4.1.1.2. REACT-NATIVE TAGS USED IN OUR PROJECT

1. <SafeAreaView>: Wraps content excluding status bar of iphone. It doesn't work on android.
2. <View>: Wraps content and other tags.
3. <Text>: Displays text content.
4. <Image>: Renders images.
5. <ScrollView>: Enables scrolling for content that overflows the screen.
6. <TextInput>: Provides an input field for text entry.
7. <FlatList>: Efficiently renders flat lists of data.
8. <TouchableOpacity>: Creates a touchable element with opacity changes.
9. <ActivityIndicator>: Displays a loading indicator.

4.1.1.3. Vanilla Javascript

vanilla JavaScript typically refers to using pure JavaScript without relying heavily on external libraries or frameworks outside of what React Native provides. While React Native itself is built upon JavaScript, it abstracts away many of the complexities of native mobile development by allowing developers to write JavaScript code that renders native UI components.

When working with React Native, developers often use vanilla JavaScript to handle logic, state management, and business logic within their applications. This includes tasks such as fetching data from APIs, processing user input, managing application state, and performing calculations.

While React Native does provide its own set of APIs and components for building mobile applications, there are times when developers may need to resort to pure JavaScript for tasks that are not covered by React Native's built-in functionality. This could involve using native JavaScript functions, working with the Document Object Model (DOM), manipulating arrays and objects, or implementing custom logic specific to their application.

Overall, understanding vanilla JavaScript is crucial for developers working with React Native as it allows them to effectively build and extend the functionality of their mobile applications beyond what is provided by the framework itself. By leveraging the power of JavaScript, developers can create rich and interactive user experiences in their React Native applications.

4.1.2. TensorFlow

TensorFlow is an open-source software library developed by Google for machine learning and artificial intelligence. It's used across a range of tasks but has a particular focus on training and inference of deep neural networks.

4.1.2.1. FEATUERS OF TENSORFLOW

- **Computational Graphs**: TensorFlow uses data flow graphs where nodes in the graph represent mathematical operations, and edges represent the multidimensional data arrays (tensors) communicated between them. This

flexible architecture allows you to deploy computation to one or more CPUs or GPUs in a desktop, server, or mobile device with a single API.

- **Deep Learning Models**: TensorFlow is used to build and train deep learning models, facilitating the creation of computational graphs and efficient execution on various hardware platforms.
- **Versatility**: TensorFlow can be used for both research and production tasks, and it can run on various platforms, including desktops, servers, and mobile devices.

- **Tools and Libraries**: TensorFlow provides multiple tools to help consolidate, clean, and preprocess data at scale. It also offers libraries of high-level components that allow you to take powerful models and fine-tune them on new data or customize them to perform new tasks.

4.1.3. FIREBASE

Firebase is a comprehensive mobile and web application development platform, owned by Google, that offers a wide range of tools and services to streamline the development process. It provides backend-as-aservice (BaaS) capabilities, allowing developers to focus on building features and functionality without worrying about managing servers or infrastructure. Firebase includes features such as real-time database, authentication, cloud functions, hosting, and cloud messaging, among others. The real-time database is a NoSQL database that enables developers to build responsive and collaborative applications with synchronized data across clients in real time. Authentication services support various authentication methods, making it easy to implement secure user sign-ins. Cloud functions enable serverless computing, allowing developers to run custom backend code in response to events triggered by Firebase features or HTTP requests.

4.1.3.1. WHERE IS IT USE?

Database

Firebase is a product of Google that helps developers build, manage, and grow their apps easily1. It provides a variety of services to Android, iOS, web, and Unity.

Firebase is versatile and can be used for both research and production tasks. It can run on various platforms, including desktops, servers, and mobile devices. It's known for its real-time database that's still at its core a multi-node, key-value database optimized for synchronizing data. It's a backend platform for building web and mobile applications.

4.1.3.2. WHAT IS USED?

4.1.3.2.1. Firebase Authentication

Firebase Authentication is a service provided by Firebase, a mobile and web application development platform. The signIn With Email And Password method is used for email and password authentication in Firebase. Here's a brief overview:

Sign In With Email and Password Method:

Purpose: The signIn With Email and Password method is used to authenticate a user with Firebase using their email address and password.

4.1.3.2.2. Realtime Database

Firebase Realtime Database is a cloud-hosted NoSQL database provided by Firebase, a mobile and web application development platform. It allows developers to build real-time applications by providing a cloud-based backend infrastructure for storing and synchronizing data across clients in real time.

Purpose: We are going to store user information in Realtime Database of Firebase. It is best for quick realtime responses from database and synchronize information as quick as possible. All attributes related to User Account are to be stored here.

4.1.3.2.3. Cloud Firestore

Firebase Cloud Firestore is a flexible and scalable NoSQL document database that allows developers to store, sync, and query data for their web, mobile, and server applications. Firestore organizes data into collections and documents, where each document contains key-value pairs. This schema-less structure provides versatility, allowing developers to work with data in a way that best suits their application's needs.
Purpose: We are going to store Yoga and Exercises Templates on Firestore. We are keeping Global Templates(All) different from user data and firestore offers perfect schema-less structure to do that.

4.1.3.2.3. E-R Diagram

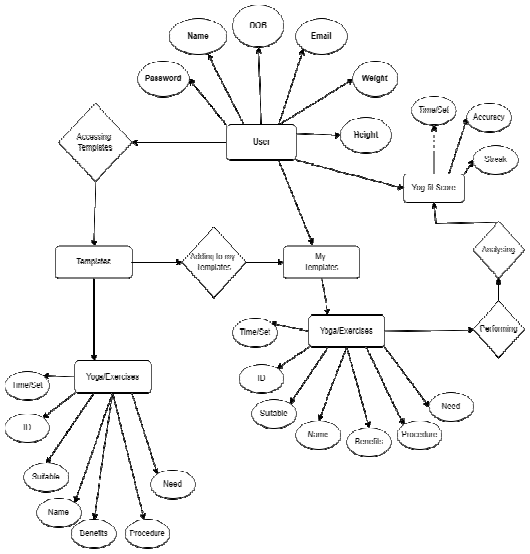


Figure 4.1.3.2.3.1 E-R Diagram

2) 4.2. DEVELOPMENT

4.2.1. EXPO Mobile App Development

4.2.1.1. WHAT IS EXPO ?

Expo providesthe building tools and services built around React Native and native platforms that help you develop, build, deploy, and quickly iterate on iOS, Android, and web apps. Here are some key features of Expo:

1. Expo CLI: A command fine app that is the main interface betweena developer and Expo tools.
2. Expo Client An app on your phone that lets you open your projectswhile you're working on them, without needing to go through XCode or Android Studio.

• COMMON USES OF EXPO

1. Cross-Platform Development: Expo is used for building apps that runnatively on Android, iOS, and the web.
2. Live Updates: It enables live updates, allowing developers to push updates to users without requiring them to update their apps.

4.2.1.2. WHERE IT IS USE?

Expo is used in the development of mobile applications that run natively on Android, iOS, and the web. It's an open-source framework that brings together the best of mobile and the web, enabling many important features for building and scaling an app such as live updates, instantly sharing your app, and web support.

HARDWARE AND SOFTWARE SPECIFICATION

5.1 SOFTWARE CONFIGURATION

OPERATING SYSTEM: Android 6+ or IOS 12.4+

FRONT-END LANGUAGES: React-Native
BACK-END LANGUAGES: Javascript(Node.js)
DATA SERVER: Firebase

5.2 HARDWARE CONFIGURATION

RAM	: 6 GB MINIMUM
HARD-DISK	: 100 MB MAXIMUM

UML DIAGRAMS

6.1 FLOWCHART

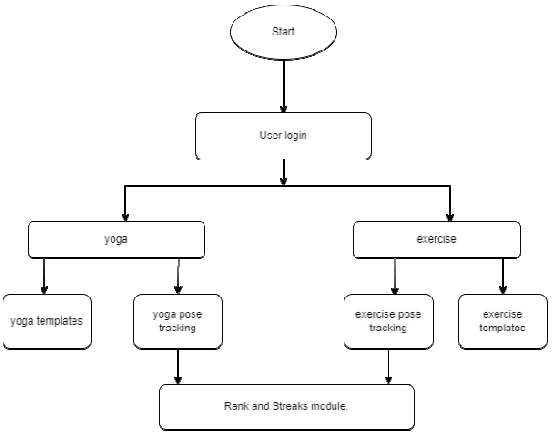


Figure 6.1 System Flowchart

6.2 USE CASE DIAGRAM

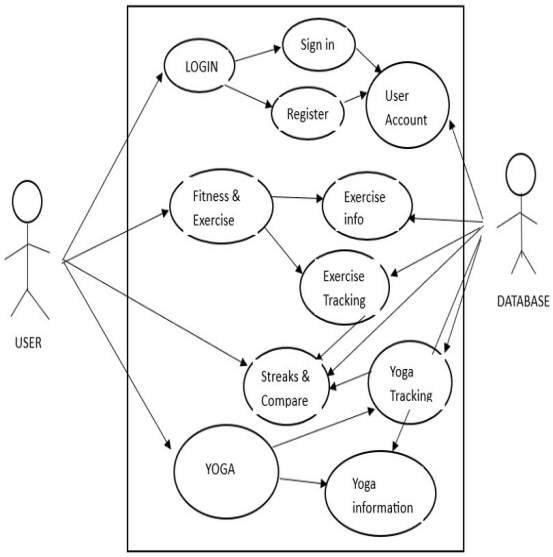


Figure 6.2.1 Use Case Diagram

6.3 Data Flow Diagrams

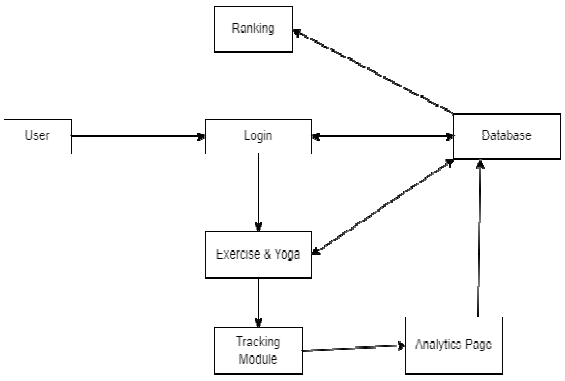


Figure 6.3.1 DFD level 0

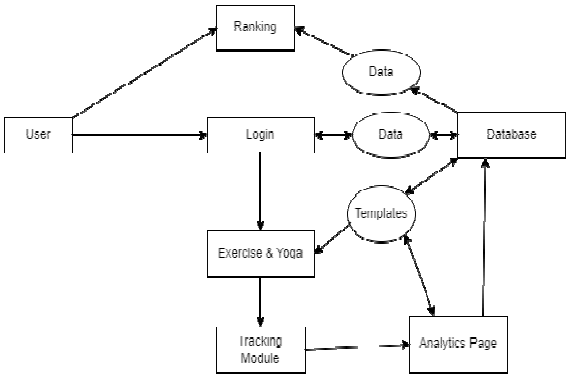


Figure 6.3.2 DFD level 1

6.4 SYSTEM CODING

App Navigator :-

```
// AppNavigator.js
import Ranking from "../pages/rank";
import React from "react";
import { createBottomTabNavigator } from
"@react-navigation/bottom-tabs";
import Home from "../pages/home";
import Profile from "../pages/profile";
import BottomNav from "./bottomNav";
import Tracker from "../pages/tracker";
import TemplatePage from
"../pages/templatesPage";
import LoginScreen from "../pages/login";
import EYPage from "../pages/eyPage";
import CreateAccount from
"../pages/createAccount";
import PersonalProfile from
"../pages/personalProfile";
import PublicProfile from
"../pages/publicProfile";
import Analytic from "../pages/analytic";

const Tab = createBottomTabNavigator();

const AppNavigator = () => {
  return (
<>
<Tab.Navigator
       options={{ headerShown: false }}
      // tabBar={() =><BottomNav />}
tabBar={() => null}
screenOptions={{tabBarStyle:
{showLabel:false}}}
      // tabBarOptions={{showLabel:false}}
initialRouteName="Login"

>
<Tab.Screen
        options={{ headerShown: false }}
        name="Login"
        component={LoginScreen}
      />
<Tab.Screen
        options={{ headerShown: false }}
        name="SignUp"
        component={CreateAccount}
      />
<Tab.Screen
        options={{ headerShown: false }}
        name="Home"
        component={Home}
      />
<Tab.Screen
        options={{ headerShown: false }}
        name="TemplatesPage"
        component={TemplatePage}
      />
<Tab.Screen
        options={{ headerShown: false }}
        name="Tracker"
        component={Tracker}
      />
<Tab.Screen
        options={{ headerShown: false }}
        name="Ranking"
        component={Ranking}
      />
<Tab.Screen
        options={{ headerShown: false }}
        name="EYPage"
        component={EYPage}
      />
<Tab.Screen
        options={{ headerShown: false }}
        name="Analytic"
        component={Analytic}
      />
<Tab.Screen
        options={{ headerShown: false }}
```



```
          name="Profile"
          component={Profile}
        />
<Tab.Screen
          options={{ headerShown: false }}
          name="PersonalProfile"
          component={PersonalProfile}
        />
<Tab.Screen
          options={{ headerShown: false }}
          name="PublicProfile"
          component={PublicProfile}
        />
</Tab.Navigator>
</>
  );
};

export default AppNavigator;
```

Tracker :-

```
import React, { useState, useEffect, useRef }
from 'react';
import { StyleSheet, Text, View,
ImageBackground, Image, TouchableOpacity }
from 'react-native';
import { Camera } from 'expo-camera';
import * as posenet from '@tensorflow-
models/posenet';
const Tracker = () => {
  const [hasPermission, setHasPermission] =
useState(null);
  const [net, setNet] = useState(null);
  const [detectedPose, setDetectedPose] =
useState(null);
  const [referencePoses, setReferencePoses] =
useState([]);
  const [poseCorrect, setPoseCorrect] =
useState(null);
  const [isPlaying, setIsPlaying] =
useState(false);
  const [timer, setTimer] = useState(0);
  const [cameraType, setCameraType] =
useState(Camera.Constants.Type.back);

  const timerRef = useRef(null);

useEffect(() => {
   const loadPoseNet = async () => {
     const net = await posenet.load();
setNet(net);
   };

loadPoseNet();
  }, []);

  const handleCameraPermission = async () =>
{
   const { status } = await
Camera.requestCameraPermissionsAsync();
setHasPermission(status === 'granted');
  };

useEffect(() => {
handleCameraPermission();
  }, []);

  const loadReferencePoses = async () => {
   const posesWithImages = await
Promise.all(
referencePoses.map(async (pose) => {
       const image = await
Image.resolveAssetSource({ uri:
pose.imageUri });
       const net = await posenet.load();
       const imageTensor = await
netUtil.fetch(image.uri, {}, { isBinary: true });
       const poseEstimation = await
net.estimateSinglePose(imageTensor, {
flipHorizontal: false,
       });

       return {
         name: pose.name,
keypoints: poseEstimation.keypoints,
       };
     })
   );

setReferencePoses(posesWithImages);
  };

useEffect(() => {
   // Load reference poses when the
component mounts
```



```
loadReferencePoses();
  }, []);

  const isYogaPoseCorrect = (detectedPose, referencePose) => {
    const distanceThreshold = 20;

    const areAllKeypointsCorrect = referencePose.keypoints.every((refKeypoint) => {
      const detectedKeypoint = detectedPose.keypoints.find((dKey) =>dKey.part === refKeypoint.part);

      if (!detectedKeypoint) {
        return false;
      }
      const distance = Math.sqrt(
Math.pow(detectedKeypoint.position.x - refKeypoint.position.x, 2) +
Math.pow(detectedKeypoint.position.y - refKeypoint.position.y, 2)
      );

      return distance <= distanceThreshold;
    });

    return areAllKeypointsCorrect;
  };

  const handleCameraStream = async (camera) => {
    const options = {
flipHorizontal: false,
maxDetections: 1,
scoreThreshold: 0.5,
    };

    const imageTensor = await camera.takePictureAsync({
      base64: true,
    });

    const pose = await net.estimateSinglePose(imageTensor.base64, options);
setDetectedPose(pose);
    // Check if the yoga pose is correct for each reference pose
    const correctPose = referencePoses.find((refPose) => isYogaPoseCorrect(pose, refPose)
    );

setPoseCorrect(correctPose ? correctPose.name : null);
  };

  const startTimer = () => {
setIsPlaying(true);
setTimer(0);
timerRef.current = setInterval(() => {
setTimer((prevTimer) =>prevTimer + 1);
    }, 1000);
  };

  const stopTimer = () => {
setIsPlaying(false);
clearInterval(timerRef.current);
  };

  const flipCamera = () => {
setCameraType(
cameraType === Camera.Constants.Type.back
        ? Camera.Constants.Type.front
        : Camera.Constants.Type.back
    );
  };

useEffect(() => {
    if (isPlaying&&poseCorrect) {
      // You can add your logic here when the pose is correct
console.log('Correct Pose!');
    }
  }, [isPlaying, poseCorrect]);

  if (hasPermission === null) {
    return <View />;
  }
  if (hasPermission === false) {
    return <Text>No access to camera</Text>;
  }
```

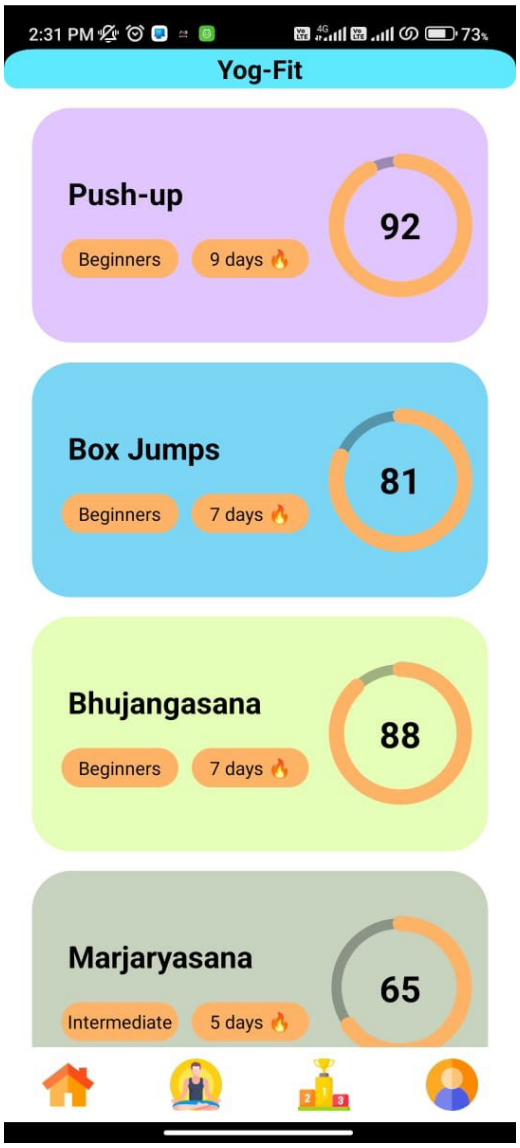


```
 return (
<View style={styles.container}>
<Camera
      style={styles.camera}
      type={cameraType}
onCameraReady={() => console.log('Camera
is ready')}
onMountError={(error)
=>console.log('Camera Error: ', error)}
      ratio={'16:9'}
>
<ImageBackground
        style={{ flex: 0.8, resizeMode: 'cover',
justifyContent: 'center',alignContent:"center",
opacity: 0.3 }}
        source={{ uri:
'https://cdn.yogajournal.com/wp-
content/uploads/2007/08/Cobra-
Pose_Andrew-Clark.gif?width=730' }}
>
       {detectedPose&& (
<View
           style={[
styles.poseContainer,
{ borderColor: poseCorrect ? '#00ff00' :
'#ff0000' },
           ]}
>
<Text style={styles.poseText}>
             Detected Pose:
{detectedPose.keypoints[0].part} (

{detectedPose.keypoints[0].position.x.toFixed
(2)},{' '}

{detectedPose.keypoints[0].position.y.toFixed
(2)})
</Text>
           {poseCorrect !== null && (
<Text style={styles.correctnessText}>
              {`Correct Pose: ${poseCorrect}`}
</Text>
           )}
           {/* Display coordinates for other
keypoints if needed */}
</View>
       )}
</ImageBackground>
</Camera>
<View style={styles.bottomContainer}>
<TouchableOpacity
        style={styles.button}
onPress={flipCamera}
>
<Image
         source={{uri:"https://cdn-icons-
png.flaticon.com/128/1829/1829373.png"}}
         style={styles.butImage}
        />
</TouchableOpacity>
<TouchableOpacity
        style={[styles.button,
{ backgroundColor: isPlaying ? '#ff0000' :
'#00ff00' }]}
onPress={isPlaying ? stopTimer : startTimer}
>
<Image
         style={styles.butImage}
         source={isPlaying? {uri:"https://cdn-
icons-
png.flaticon.com/128/4340/4340168.png"}:{u
ri:"https://cdn-icons-
png.flaticon.com/128/10109/10109952.png"}}
        />
</TouchableOpacity>

<Text style={styles.timerText}>{`Timer:
${timer}s`}</Text>
</View>
</View>
 );
};
const styles = StyleSheet.create({
 container: {
   flex: 1,
 },
 camera: {
   flex: 0.9,
 },
poseContainer: {
   position: 'absolute',
   bottom: 20,
   left: 20,
borderWidth: 2,
   padding: 10,
borderRadius: 10,
```

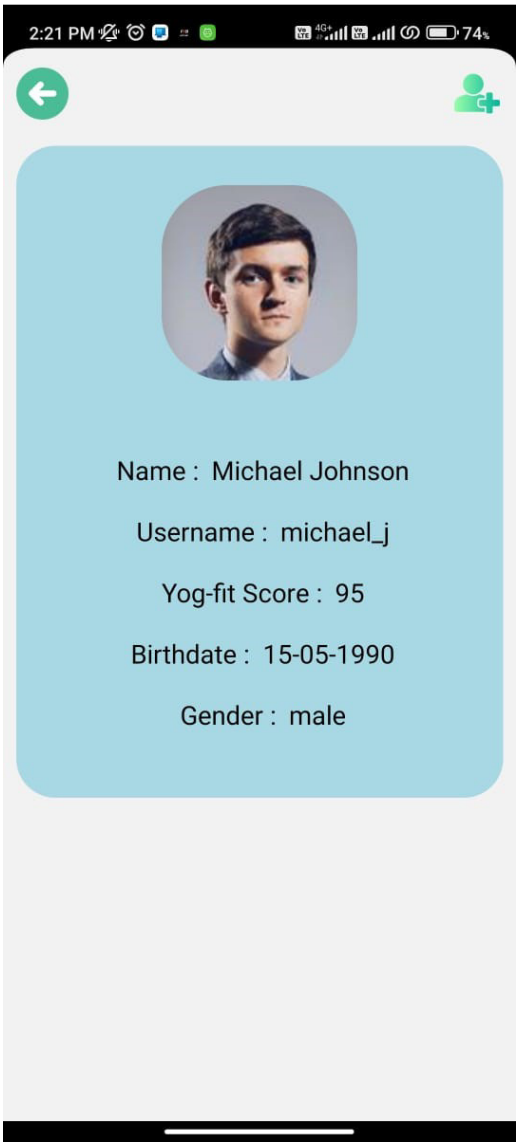


```
  },
poseText: {
fontSize: 16,
fontWeight: 'bold',
  },
correctnessText: {
fontSize: 16,
fontWeight: 'bold',
marginTop: 10,
  },
bottomContainer: {
    flex: 0.1,
flexDirection: 'row',
justifyContent: 'space-between',
alignItems: 'center',
paddingHorizontal: 20,
  },
butImage:{
    height:40,
    width:40,
  },
  button: {
borderRadius: 50,
  },
buttonText: {
color: '#fff',
fontSize: 16,
fontWeight: 'bold',
  },
timerText: {
fontSize: 16,
fontWeight: 'bold',
  },
});

export default Tracker;
```

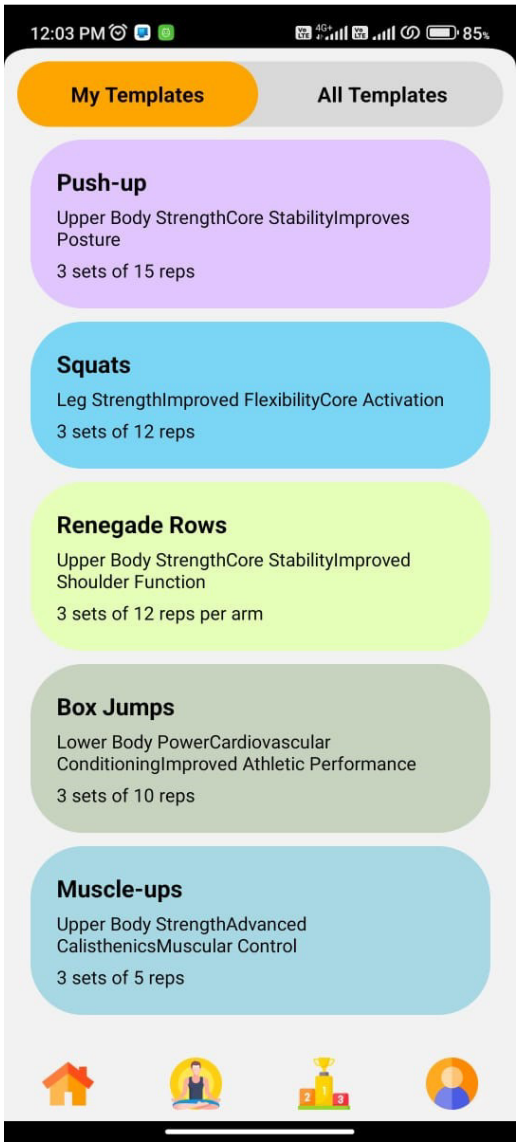
7. SNAPSHOTS OF YOG-FIT



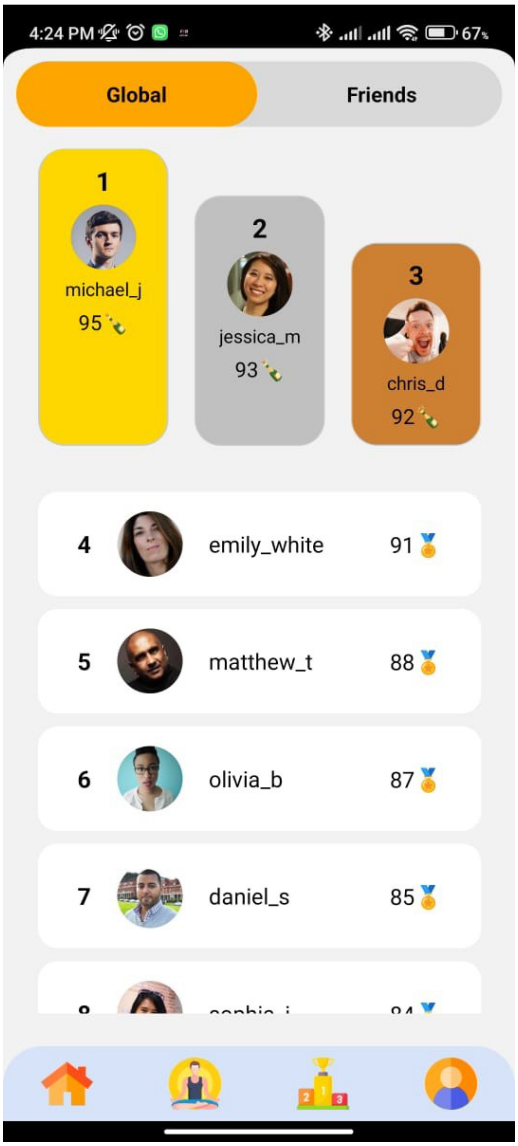
Snapshot 6.2.1 Home Page



Snapshot 6.2.2 Global Account Page



Snapshot 6.2.2 My Template Page



Snapshot 6.2.2 Ranking Page

TESTING AND RESULTS OF PROJECT

7.1 TESTING METHODOLOGIES

1. Black box Testing
2. White box Testing

7.2 LEVELS OF TESTING

1. Unit Testing
2. Integration Testing
3. System Testing

7.3 TESTING METHODOLOGIES

1. Black box Testing

It is the testing process in which tester can perform testing on an application without having any internal structural knowledge of application.
Usually Test Engineers are involved in the black box testing.

2. White box Testing

It is the testing process in which tester can perform testing on an application with having internal structural knowledge. Usually The Developers are involved in white box testing.

7.4 LEVELS OF TESTING

1. Unit Testing

Unit Testing concentrates on the verification of the smallest element of the program i.e. Module. In this testing all control paths are tested to identify errors within the bounds of the module. The important goal of unit testing is to isolate each part of the program and show individual parts are correct. It is very easy to perform and requires less amount of time because the modules are smaller in size. In unit testing it is possible that the outputs produced by one unit become

input for another unit hence, if incorrect output produced by one unit is provided as input to the second unit then it also produces wrong output. If this process is not corrected, the entire software may produce unexpected outputs.To avoid this, all the units in the software are tested independently using unit – testing. In unit testing, the units are tested to ensure that they operate correctly. In software engineering the unit testing is not just performed once during software development, but repeated whenever the software is modified.

2. Integration Testing

When unit testing is complete, integration testing begins. In integration testing the tested units are combined together to form system as whole. The aim of this testing is to ensure that all modules are working properly according to user's requirements when they are combined. The integration test takes all tested individual modules, integrate them, test them again and develop the software. It ensures that all modules work together properly and transfer accurate data across their interfaces.
Integration testing contains:-

1. **Non –Incremental integration**: The entire program is tested as a whole and all errors are identified. We have tested whole app while integrating main module. All the dependency errors and unoptimized extra code has been removed.

2. **Incremental integration**: The program is constructed and tested in small segments, to find out errors.We have performed this testing while developing pages and components.

3. System Testing

System testing is the next level in the testing and tests the system as a whole. Once, all the components are integrated, the application as a whole is tested to see that it meets Quality Standards. This type of testing is performed by a specialized testing team. System testing can be defined as"a testing conducted on a complete, integrated system to ensure that the system is according to its specified requirement".

4.Test Development

- Test cases have been developed for each module.
- Test cases are developed considering negative testing.
- Testing is performed on 4 different devices for compatibility.

5.Test Execution

- There were few bugs in production app, which has been fixed.
- We noticed our app's response time is slow(1.5s to be precise).
- Our Error messages/warnings do not work as expected.

6.Result Analysis • Expected Result: Apps behaviour should be normal and performance should be optimal.

- Actual value: Apps behaviour is normal but performance is not the best.

7.Bug Tracing

- Bugs are in Camera and Tracking module.

8.Reporting

• Inprocess...

9.Test scope

• Test coverage is provided for the "Search and Friends" module ofYOG-FIT:"Stretch Your Limits"

CONCLUSION AND FUTURE ENHANCEMENT

8.1. **CONCLUSION**

We have combined exercise and yoga for better fitness preference to user. Our project will help you learn new exercise and yoga with tracking so users can see what they are doing wrong. Plus the templates we provide for exercise and yoga and the difficulty level. Streaks and ranking with friends are the competitive features. At the end user will be benefited a lot and have a healthy lifestyle.

8.2. FUTURE ENHANCEMENT

- In the next update we can add few more templates.
- We can definitely optimize our tracking module.
- The accuracy for results will be increased as we push updates.

REFERENCES

1. https://docs.expo.dev/faq/
2. https://firebase.google.com/
3. https://www.tensorflow.org/lite/models
4. https://firebase.google.com/docs/web/learn-more?hl=en&authuser=0#modular-version
5. https://firebase.google.com/docs/hosting/manage-hosting-resources

