AbTrack – An application to track personal fitness goals

Submitted in partial fulfillment of the requirements for the award of
Project-3 Implementation Of Personal Fitness Tracker Using Python

By

RAKESH MALLICK
AICTE Internship Student Registration ID: STU6791ddd2b400a1737612754
AICTE Student Internship ID: INTERNSHIP_1736316743677e1747e0671

DEPARTMENT OF JOINT CSR INITIATIVE OF MICROSOFT & SAP

EDUNET FOUNDATION

Gurgaon/NCR office

901, Unitech Commercial Tower II,
Sector 45, Gurgaon 122003.

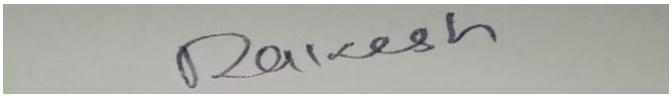
E-mail

info@edunetfoundation.org

DECLARATION

I, **Rakesh Mallick (Reg.No-** STU6791ddd2b400a1737612754**),** hereby declare that the Project-3 Phase-4 Report entitled **AbTrack – An application to track personal fitness goals"** done by me under the guidance of **SAOMYA CHAUDHURY** is submitted in partial fulfillment of the requirements for the award of **JOINT CSR INITIATIVE OF MICROSOFT & SAP.**

DATE: 23.03.25



PLACE: HYDERABAD **SIGNATURE OF THE CANDIDATE**

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **EDUNET FOUNDATION** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **SAOMYA CHAUDHURY**, **PAVAN KUMAR U Department** of **JOINT CSR INITIATIVE OF MICROSOFT & SAP** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **SAOMYA CHAUDHURY** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department Of Joint CSR Initiative Of Microsoft & SAP** who were helpful in many ways for the completion of the project.

ABSTRACT

AbTrack is a personal fitness goal Achiever app where a user can set their own personal goal and also the set of exercises which the wish to perform in order to achieve the goal. It encourages users to perform a specific type of abs exercise with a goal of reps for each movement each day.

This app also shows users their progress to motivate them to keep trying. User can register and log in using their email and password. User can add their measurements. The user can view videos and instructions on the moves they should make. The user can track their progress in the application on a monthly or daily basis through track.it or on the progress pages. The user can see a summary of his measurements. The app also has a control panel to edit the measures or exercises users should do.

This fitness monitoring tool is an all-inclusive tool created to assist users in achieving their fitness objectives by offering a thorough analysis of their physical activity and health. Users may conveniently follow their progress while on the move thanks to the application's universal accessibility from any device with an internet connection.

Users may establish their fitness objectives and keep track of their regular workout routines with the help of an intuitive user interface. The software provides a range of

tracking capabilities, including step counting, heart rate monitoring, and calorie tracking, so users can learn more about their level of physical activity and modify their routines as necessary

LIST OF FIGURE

LIST OF ABBRIVIATIONS

S.NO	ABBREVATION	EXPANSION
1.	DB	Database
2.	DBC	Data Base Confidentiality
3.	HTML	Hyper Text Markup Language
5.	JVM	Java Virtual Machine
6.	JSP	Java Server Page
7.	MDA	Medical Admin

CHAPTER 1
INTRODUCTION

Many individuals live sedentary lives in today's digital environment, spending a lot of time sitting or using electronic gadgets like computers, cellphones, and tablets. Although if technology has improved many parts of our lives, it has also led to a decline in physical activity and a rise in the incidence of chronic illnesses like obesity, diabetes, and heart disease.

The prevention of chronic illnesses and the maintenance of good health depend on exercise. The risk of obesity, type 2 diabetes, cardiovascular disease, and several cancers can be decreased with regular physical activity. Moreover, exercise can boost cognitive performance, lower stress, and promote mental wellness.

Despite the numerous advantages of exercise, many individuals find it challenging to accommodate exercise into their daily schedules. Finding the time and desire to exercise can be difficult because of the pressures of work and family obligations as well as the seductiveness of digital diversions.

Prioritize physical exercise and integrate it into your daily routine on a regular basis to overcome these challenges. This can entail making minor adjustments, like going for a quick stroll during your lunch break, using the stairs as opposed to the lift, or stretching while watching TV. These little adjustments have the potential to pile up over time and support your fitness objectives.

Setting and monitoring specific fitness objectives is another method to include exercise in your everyday routine. When you work to meet your objectives, this may keep you inspired, hold you accountable, and give you a sense of success. Many fitness monitoring programs, such as the AbTrack, may help you keep track of your physical activity and health indicators while giving you insightful feedback that will keep you on target.

Finding physical activities that suit your lifestyle and those you love is also crucial. This may entail trying out several forms of exercise, such as yoga, swimming, cycling, or strength training, to see which ones you love and find to be the most productive.

Fitness apps have become increasingly popular in recent years as a tool to help people maintain their health and fitness goals. These apps offer a wide range of features and benefits, such as tracking physical activity, providing workout plans, monitoring diet and nutrition, and offering motivation and support.

One of the primary benefits of fitness apps is that they allow users to track their physical activity and monitor their progress over time. This can include features like step tracking, distance traveled, and calories burned. By keeping track of their activity levels, users can better understand their fitness habits and make adjustments to improve their overall health.

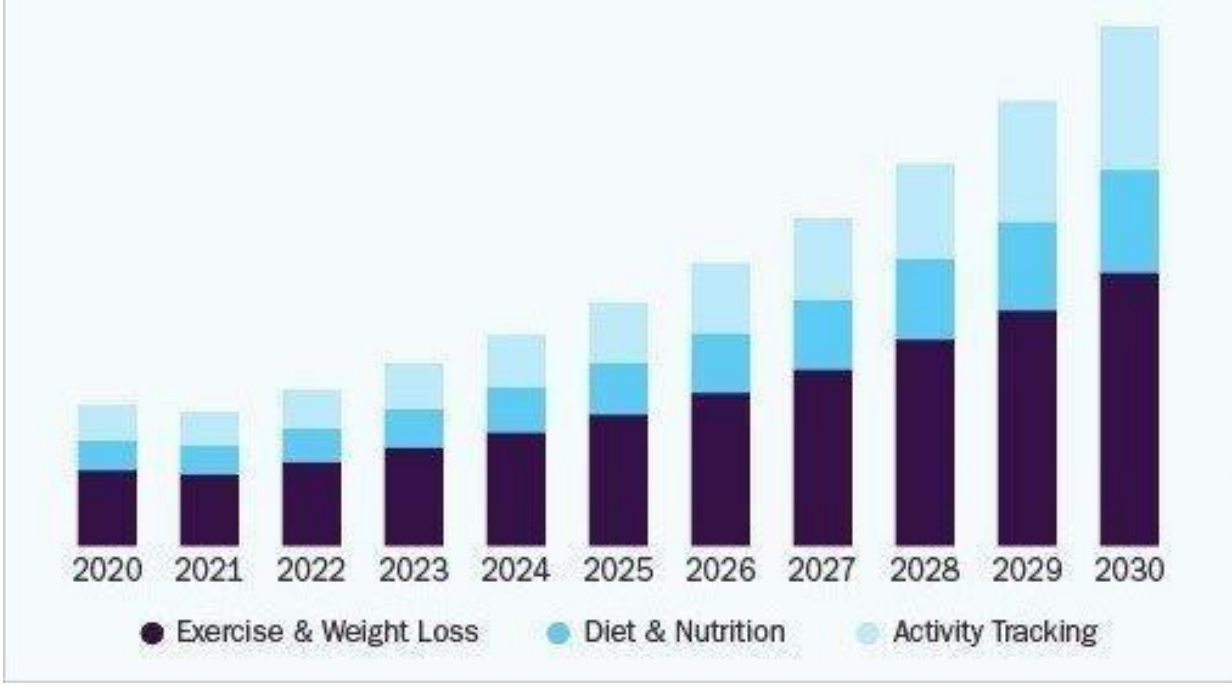


Fig.1.1: Future trend for various fitness tracking services

Another key feature of many fitness apps is the ability to provide personalized workout plans. Based on a user's fitness level, goals, and preferences, these apps can create customized workout plans that target specific areas of the body and include a variety of exercises. This can be especially helpful for people who are new to exercise or who need guidance in developing an effective fitness routine.

Many fitness apps also offer tools to help users monitor their diet and nutrition. This can include features like calorie tracking, meal planning, and recipe suggestions. By keeping track of what they eat and ensuring that they are getting the right nutrients, users can support their overall health and fitness goals.

Perhaps one of the most significant benefits of fitness apps is the motivation and support that they can provide. Many apps offer social features that allow users to connect with friends and family members who are also using the app. This can provide a sense of accountability and motivation, as well as a supportive community to help users stay on track with their fitness goals.

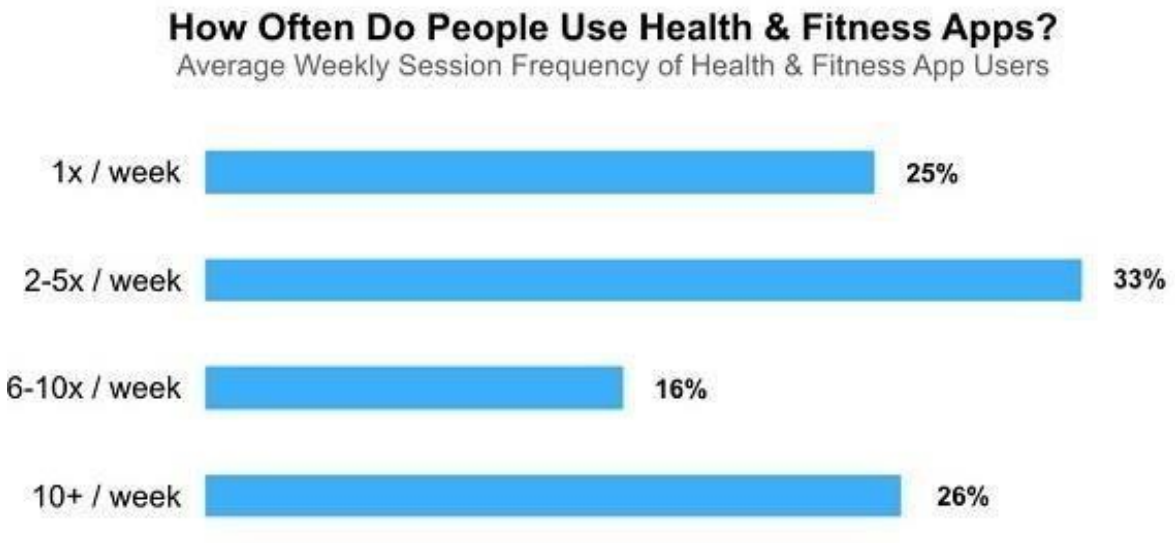


Fig.1.2: Usage of fitness applications

In addition to these features, many fitness apps also offer other tools and resources to support users in maintaining their health and fitness. This can include things like educational content on exercise and nutrition, challenges and competitions to help motivate users, and even guided meditation and mindfulness exercises to support mental and emotional wellbeing.

AbTrack is an online fitness monitoring tool created to assist you in achieving your fitness objectives and upholding a healthy lifestyle. Tracking your fitness progress is simple and effective with AbTrack's full set of tools and capabilities.

You may quickly create an account with AbTrack and personalize your fitness objectives to suit your particular requirements and interests. AbTrack features the tools and resources to support your success whether you're trying to get fitter overall, lose weight, or gain muscle.

You may measure your physical activity and health data using our application's many tracking tools, which you can do in real time. You may record your calorie intake and expenditure as well as your heart rate and step count. With this knowledge, you may develop a more thorough awareness of your general health and wellness and make wise choices about your diet and exercise routine.

Moreover, AbTrack provides a selection of exercise plans that are customized to your own fitness needs and tastes. Our programs provide you a variety of alternatives to select from, including weight training, cardio, yoga, and more. Our training regimens are made to provide you with a challenging and enjoyable workout experience while also assisting you in achieving your fitness objectives.

You can stay motivated and on track to achieve your objectives with the aid of our app, which offers individualized feedback and recommendations. Regular reports on your progress as well as suggestions on how to do better will be sent to you. We know it may be difficult to keep motivated.

Also, AbTrack enables communication and interaction with other fitness enthusiasts, fostering a community of people who share your interests. To keep inspired and involved, you may track your development, help others out with advice and support, and take part in group challenges.

Furthermore, AbTrack includes sophisticated analytics and reporting options that let you comprehend your development more fully over time. This includes in-depth graphs and charts that measure changes in calorie consumption, heart rate, and physical activity levels, giving you helpful information to aid in reaching your fitness objectives.

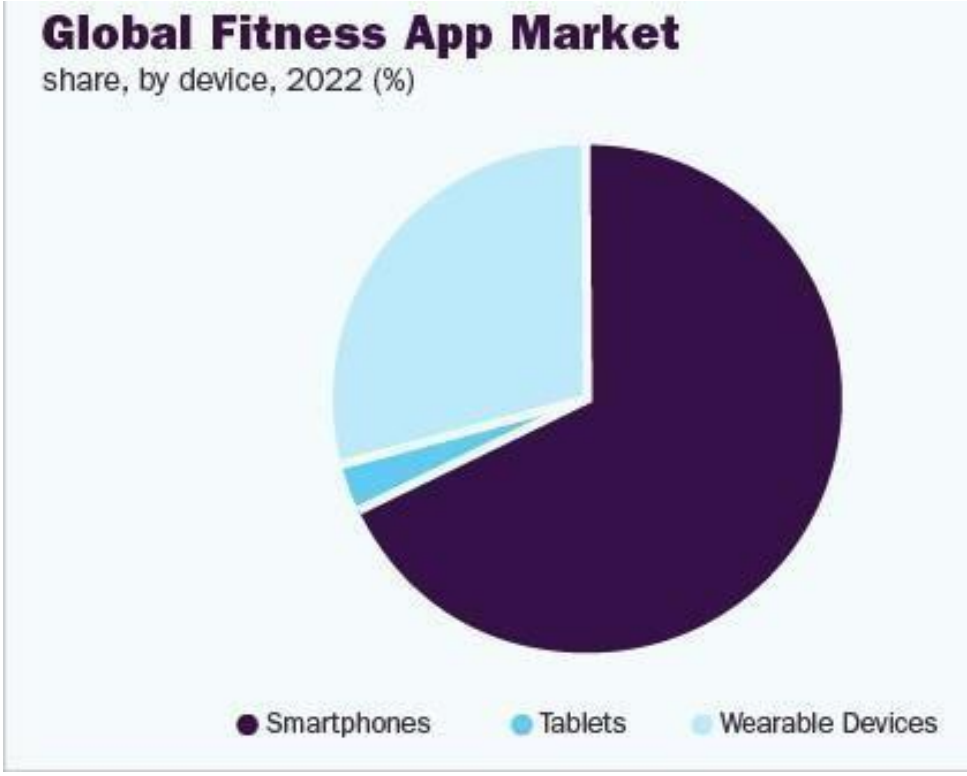


Fig.1.3: Prevalence of fitness application based on platform

CHAPTER 2 LITERATURE SURVEY

TITLE 1: The Intention to Use Fitness and Physical Activity Apps: A Systematic Review

AUTHOR : Salvador Angosto, Jerónimo García-Fernández, Irena Valantine and Moisés Grimaldi-Puyana.

YEAR: 2020

Description:

The advent of new mobile devices such as smartphones and tablets has led to a higher level of comfort and practical use, making them an integral part of modern-day life for consumers. Globally, the number of users using these devices has been increasing steadily, with an estimated 6.8 billion users worldwide in 2019, and projections indicating that this number will reach 7.33 billion by 2023. Mobile applications, or apps, account for 90% of the time dedicated to using smartphones, highlighting the importance of these digital platforms in modern-day life.

In the context of sustainability, the use of information and communication technologies (ICT) and data management functionalities has been widely adopted through the use of mobile devices, allowing people to participate more actively in society. The European Union (EU) policies have emphasized the need for synergy between smart technologies and sustainable urban development, as ICTs are seen as critical in providing accurate, consistent, and timely data for new policy formulation, and facilitating service improvement.

The integration of ICT and mobile devices in sustainable development efforts allows for a more comprehensive approach that considers economic, environmental, and social factors.

TITLE

2: **Factors of Leading the Adoption of Diet/Exercise Apps: Application of Channel Expansion Theory**

AUTHOR : Jaehee Cho, Sun Jin Kim

YEAR: 2015

Description:

This study focused on the noticeably rising usage of diet and fitness applications on cellphones. This study examined how three predictors—Internet information use efficacy, Internet information credibility, and smartphone use efficacy—affected one's attitude towards and desire to use diet/exercise applications. It was based on the channel expansion theory and the technology acceptance model (TAM). According to the findings of a path analysis, smartphone and Internet information usage effectiveness both positively impact how easy users consider diet and exercise applications to be to use. The perceived utility of such applications is positively predicted by the reliability of online information. In addition, there were gender disparities in the impact of the efficacy and believability of the information used online on the perceived value of diet and exercise applications.

Researchers and practitioners have expressed a great deal of interest in those technologies in response to this rise in the usage of diet/exercise applications. There haven't been many studies looking closely at the motivations behind why users choose to acquire and utilise these applications, though, since prior study has mostly focused on their technical capabilities. Consequently, based on channel expansion theory and the technology acceptance model (TAM), this study sought to investigate the impacts of three variables on people's intents to use diet/exercise apps: smartphone use effectiveness, Internet information use efficacy, and Internet information believability.

TITLE

3: **Motivation factors influencing intention of sports apps use by applying the unified theory of acceptance and use of technology**

AUTHOR : Soonhwan Lee, Seungmo Kim, Suosheng Wang

YEAR: 2017

Description:

The proliferation of innovative personal technological gadgets such as smartphones and tablet PCs has brought about a significant shift in our daily lives. One of the key factors driving this change is the rapid expansion of mobile applications, which are digital platforms designed to perform specific functions on personal mobile technology devices. As technology devices continue to penetrate the market, demand and usage for mobile apps have increased dramatically. This trend is evident in the astounding rate at which the mobile app industry is expanding.

According to a new analysis from PortioResearch (2015), the market for mobile apps is growing at an unprecedented rate. In 2012, the mobile app industry generated $12 billion in sales, with a total of 46 billion applications downloaded. This trend is expected to continue in the coming years, with projections indicating that downloads will exceed 200 billion by 2017, and earnings will reach $63.5 billion.

The rise of mobile applications has enabled users to access a vast range of services on their personal mobile technology devices. From social networking and entertainment to productivity and education, mobile apps have become an essential tool for modern-day living. The growth of the mobile app industry has also created new opportunities for developers, entrepreneurs, and investors, driving innovation and economic growth. Overall, the impact of mobile applications on our daily lives and the global economy is likely to continue growing in the years ahead.

TITLE

4: **Behavior change techniques implemented in electronic lifestyle activity monitors: a systematic content analysis**

AUTHOR : Elizabeth J Lyons , Zakkoyya H Lewis, Brian G Mayrsohn, Jennifer L Rowland

YEAR: 2014

Description:

Electronic activity monitors (such as those manufactured by Fitbit, Jawbone, and Nike) improve on standard pedometers by providing automated feedback and interactive behavior change tools via mobile device or personal computer. These monitors are commercially popular and show promise for use in public health interventions. However, little is known about the content of their feedback applications and how individual monitors may differ from one another. The purpose of this study was to describe the behavior change techniques implemented in commercially available electronic activity monitors.

Electronic activity monitors were systematically identified and tested by 3 trained coders for at least 1 week each. All monitors measured lifestyle physical activity and provided feedback via an app (computer or mobile). Coding was based on a hierarchical list of 93 behavior change techniques. Further coding of potentially effective techniques and adherence to theory-based recommendations were based on findings from meta-analyses and meta-regressions in the research literature.

All monitors provided tools for self-monitoring, feedback, and environmental change by definition. The next most prevalent techniques were goal-setting and emphasizing discrepancy between current and goal behavior. Review of behavioral goals, social support, social comparison, prompts/cues, rewards, and a focus on past success were found in more than half of the systems. The monitors included a range of 5-10 of 14 total techniques identified from the research literature as potentially effective. Most of the monitors included goal-setting, self-monitoring, and feedback content that closely matched recommendations from social cognitive theory.

TITLE

5: **Effectiveness of activity trackers with and without incentives to increase physical activity (TRIPPA): a randomized controlled trial**

AUTHOR : Eric A Finkelstein, Benjamin A Haaland, Marcel Bilger, Aarti Sahasranaman , Robert A Sloan, Ei Ei Khaing Nang, Kelly R Evenson

YEAR: 2016

Description:

Despite the increasing popularity of activity trackers, little evidence exists that they can improve health outcomes. We aimed to investigate whether use of activity trackers, alone or in combination with cash incentives or charitable donations, lead to increases in physical activity and improvements in health outcomes.

In this randomised controlled trial, employees from 13 organisations in Singapore were randomly assigned (1:1:1:1) with a computer generated assignment schedule to control (no tracker or incentives), Fitbit Zip activity tracker, tracker plus charity incentives, or tracker plus cash incentives. Participants had to be English speaking, full-time employees, aged 21-65 years, able to walk at least ten steps continuously, and non-pregnant. Incentives were tied to weekly steps, and the primary outcome, moderate-to-vigorous physical activity (MVPA) bout min per week, was measured via a sealed accelerometer and assessed on an intention-to-treat basis at 6 months (end of intervention) and 12 months (after a 6 month post-intervention follow-up period).

The cash incentive was most effective at increasing MVPA bout min per week at 6 months, but this effect was not sustained 6 months after the incentives were discontinued. At 12 months, the activity tracker with or without charity incentives were effective at stemming the reduction in MVPA bout min per week seen in the control group, but we identified no evidence of improvements in health outcomes, either with or without incentives, calling into question the value of these devices for health promotion. Although other incentive strategies might generate greater increases in step activity and

improvements in health outcomes, incentives would probably need to be in place long term to avoid any potential decrease in physical activity resulting from discontinuation.

TITLE 6: Effectiveness of Wearable Trackers on Physical Activity in Healthy Adults: Systematic Review and Meta-Analysis of Randomized Controlled Trials

AUTHOR : Matilda Swee Sun Tang, Katherine Moore, Andrew McGavigan, Robyn A Clark, Anand N Ganesan

YEAR: 2020

Description:

Wearable trackers are an increasingly popular tool among healthy adults and are used to facilitate self-monitoring of physical activity.

This review used the PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses) methodology and reporting criteria. English-language randomized controlled trials with more than 20 participants from MEDLINE, CINAHL, Cochrane Library, Web of Science, PubMed, and Scopus (2000-2017) were identified. Studies were eligible for inclusion if they reported an intervention group using wearable trackers, reporting steps per day, total moderate-to-vigorous physical activity, activity, physical activity, energy expenditure, and weight reduction.

Twelve eligible studies with a total of 1693 participants met the inclusion criteria. The weighted average age was 40.7 years (95% CI 31.1-50.3), with 64.4% women. The mean intervention duration was 21.4 weeks (95% CI 6.1-36.7). The usage of wearable trackers was associated with increased physical activity (standardized mean difference 0.449, 95% CI 0.10-0.80; P=.01). In the subgroup analyses, however, wearable trackers demonstrated no clear benefit for physical activity or weight reduction.

These data suggest that the use of wearable trackers in healthy adults may be associated with modest short-term increases in physical activity. Further data are required to determine if a sustained benefit is associated with wearable tracker usage.

TITLE 7: An adoption model of mHealth applications that promote physical activity

AUTHOR : Patrick Ndayizigamiye, Macire Kante, Shalati Shingwenyana

YEAR: 2020

Description:

Physical activity is essential for maintaining a healthy and balanced life. Advancements in technology have led to the development of mobile health (mHealth) applications that can promote physical activity. However, the adoption of such applications is influenced by several factors. Therefore, this study aimed to investigate these factors and their relationships to propose a model for the adoption of mHealth applications that promote physical activity.

The study was conducted in South Africa, and data were collected from a convenient sample of 140 respondents using a survey questionnaire. The Diffusion of Innovation Theory and the Unified Theory of Acceptance and Use of Technology were used as guiding frameworks. The Partial Least Square Structural Equation Modelling (PLS-SEM) was used to assess the proposed model.

The results of the study revealed that awareness, effort expectancy, social influence, and behavioral intention significantly influence the adoption of mHealth applications that promote physical activity in South Africa.

Therefore, any intervention that seeks to promote the adoption of mHealth applications to encourage physical activity should consider these factors. Awareness campaigns should be conducted to increase knowledge and understanding of the benefits of physical activity and the use of mHealth applications. Effort expectancy should be considered by designing applications that are user-friendly and easy to use. Social influence can be leveraged by creating a social network within the application, which can provide motivation and support. Finally, behavioral intention should be promoted by emphasizing the positive outcomes of physical activity and the use of mHealth applications.

TITLE 8: Understanding and Predicting the Adoption of Fitness Mobile Apps: Evidence from China

AUTHOR : Jiuchang Wei, Anna Vinnikova, Liangdong Lu, Jia Xu

YEAR: 2020

Description:

Increasing global interest in diet and fitness mobile applications (apps) has prompted the question: What are the factors affecting users' adoption and usage behaviors on a specific fitness app? By combining the unified theory of acceptance and use of technology (UTAUT) with the health belief model (HBM), and including risk perception of information technology with the farsighted planner and myopic doer from the theory of self-control, we explore the understanding of this academic question. We analyzed data from 8,840 users of Boohee, a diet and fitness app (ranked first in the weight-loss category on the App Store in China).

Structural equation modeling revealed that self-efficacy as well as the perceived benefits, barriers, and threats of weight loss significantly influence a fitness app's performance expectancy, which, in turn, predicts users' intention to adopt it. Furthermore, actual usage behavior (i.e., diet, exercise, weight, and login records within 30 days after respondents completed the questionnaire) is positively affected by weight-loss intention and behavioral intention to use the app and negatively affected by users' risk perception. The main findings of this research could help healthcare practitioners and app developers find better ways to encourage people to adopt health apps for various reasons.

App developers should attach more importance to users' actual continuous use behavior than to their intention to use an app. They should provide sufficient introductory information about their apps, thereby reducing users' risk perception and generating reasonable performance expectancy of the app, so as to improve users' actual continuous use behavior.

2.1 Inferences from Literature Survey

Based on the literature survey, several inferences can be drawn regarding the proposed two-layer blockchain system for distributed storage in IIoT.

1) **Personalization**: Personalization is an essential aspect of fitness apps. Users prefer workout plans that are tailored to their fitness level, goals, and preferences. With the help of customization options, fitness apps can meet the individual needs of users. The app should offer features such as goal setting, personalized workout routines, and progress tracking. By personalizing workout plans, users are more likely to stick with the program, see better results, and feel motivated to continue.

2) **Tracking**: Tracking physical activity is an important aspect of fitness apps. Users prefer apps that allow them to track their physical activity, such as steps taken, distance traveled, and calories burned. Fitness apps should offer tracking features that provide users with real-time feedback on their progress. This feedback can help users stay motivated and on track with their fitness goals. The app should also provide data visualization tools, allowing users to track their progress over time.

3) **Social support**: Social support is critical for maintaining motivation and accountability in fitness goals. Fitness apps should provide social features to create a supportive community for users. The app should allow users to connect with friends and family members who are also using the app, share progress, and offer support.

4) **Nutrition**: Nutrition is a vital component of fitness goals. Users are interested in monitoring their diet and nutrition to support their fitness goals. A fitness app should offer tools to help users track their food intake, plan meals, and monitor their nutrient intake. The app should also provide nutritional information on foods and allow users to set dietary goals. By tracking nutrition, users can better understand the impact of their diet on their fitness goals and make informed choices about their eating habits.

5) **Gamification**: Gamification is an effective way to motivate users to achieve their fitness goals. Fitness apps should offer gamification features such as challenges, competitions, and rewards. These features can create a sense of excitement and engagement, making the app more enjoyable to use. Gamification can also help users stay motivated, track progress, and reach their fitness goals.

6) **Mental wellness**: Mental wellness is an essential aspect of overall health and fitness. A fitness app should offer resources and exercises to support users' mental health. The app should provide guided meditation and mindfulness exercises, as well as educational resources on mental health. By incorporating mental wellness features,

fitness apps can help users address the mental and emotional aspects of their fitness journey.

2.2 Open problems in Existing System

1) Data privacy and security: With users providing personal information such as their age, weight, and exercise routine, fitness apps are a prime target for data breaches. Ensuring the privacy and security of user data is critical, and current systems may not have adequate measures in place to protect user data.

2) Lack of accuracy in tracking: While fitness apps can track physical activity, some users have reported inaccuracies in the data recorded. This can be due to various factors, such as incorrect data input or sensor inaccuracies. The lack of accuracy can lead to users receiving inaccurate feedback on their progress, which can hinder their motivation and lead to frustration

3) Limited personalization options: While some fitness apps offer personalized workout plans, the level of personalization may not be sufficient to meet the needs of all users. This can lead to users feeling unmotivated or not seeing the results they desire. **4) Limited diversity in exercise options**: Some fitness apps may not offer a diverse range of exercises, which can limit the user's ability to customize their workouts or target specific areas of their body.

5) Limited accessibility: Some fitness apps may not be accessible to individuals with disabilities or other mobility issues, which can exclude a significant portion of the population from using the app.

6) Reliance on external hardware: Some fitness apps rely on external hardware, such as fitness trackers or smartwatches, to function properly. This can be a barrier for users who do not have access to or cannot afford such devices.

7) Lack of long-term engagement: Some fitness apps struggle to maintain user engagement over a long period of time. Users may lose motivation or interest in the app, leading to decreased usage and ultimately, abandonment of the app.

8) Inability to integrate with other apps/platforms: Some fitness apps are standalone applications that do not integrate with other platforms or apps, making it

difficult for users to seamlessly incorporate them into their overall health and wellness routine.

9) **Limited cultural sensitivity**: Fitness apps may not be culturally sensitive, meaning they may not consider users' cultural backgrounds, preferences, and values. This can lead to exclusion or lack of motivation for certain groups of users.

10) **Limited health condition support**: Some fitness apps may not be designed to support individuals with specific health conditions, such as diabetes or heart disease. This can limit the usefulness of the app for those individuals, who may require more specialized support.

MODULE NAMES:

1) **USER MODULE**
   a) LOGIN/SIGNUP PAGE
   b) PROFILE PAGE
2) **ADMIN MODULE**
   a) LOGIN/SIGNUP PAGE
   b) ADMIN PANEL

MODULE DESCRIPTION:

1) USER MODULE

a) LOGIN/SIGNUP:

This is the initial module of our project. A unit of work performed within a database management system against a database that has been handled consistently and reliably, separate from other transactions, is represented by a hair. Any modification to a database user's transfer of funds to a provider is often represented by a transaction.

b) PROFILE PAGE:

This Is the page where the user will be able to see the data entered by them. Here they will also be able to see their progress on the specific exercises chosen by them.

2) ADMIN MODULE

a) LOGIN/SIGNUP:

This is the initial module of our project. A unit of work performed within a database management system against a database that has been handled consistently and reliably, separate from other transactions, is represented by a hair. Any modification to a database user's transfer of funds to a provider is often represented by a transaction.

b) Admin Panel:

Admin will have his own window where he can see all the measures of a selected person from here. From here, the admin can also edit the paper. The parameters which are selected by the user.

MODULE DIAGRAM:

LOGIN:

Fig 2.1 Module diagram

CHAPTER 3 REQUIREMENT ANALYSIS

Requirement analysis is a critical phase in software development, as it helps to define the scope, features, and functionality of the application. Here are some requirements to consider for a fitness tracker application:

User registration and login: The application should allow users to create an account and log in using their email or social media credentials.

Tracking features: The application should allow users to track their fitness activities such as running, walking, cycling, and swimming. It should also track other data such as calories burned, heart rate, steps taken, and distance covered.

Integration with wearable devices: The application should integrate with wearable devices like smartwatches and fitness bands, allowing users to track their activities in real-time.

Goal setting and progress tracking: The application should allow users to set fitness goals and track their progress towards achieving them. Users should be able to see their progress in graphs or charts.

Personalization: The application should allow users to customize their profiles and set preferences based on their fitness goals, age, weight, and other parameters.

Social features: The application should allow users to connect with friends and other fitness enthusiasts, share their progress, and participate in challenges or competitions.

Nutrition tracking: The application should allow users to track their food intake, log meals, and provide suggestions for a balanced diet based on the user's fitness goals.

Reminder and notification features: The application should provide reminders to users to drink water, take medication, or other important activities, and send notifications to encourage users to stay on track with their fitness goals.

Privacy and security: The application should maintain the privacy and security of user data, comply with relevant data protection regulations, and provide secure login and transaction features.

User feedback: The application should allow users to provide feedback and suggestions for improvement, and have a user-friendly interface that makes it easy to use.

3.1 Feasibility Studies/Risk Analysis of the Project

Feasibility studies and risk analysis are important processes in the development of any software project, including fitness applications. Here's an overview of what these processes might look like for a fitness application project: **Feasibility Studies:**

Technical feasibility: This study assesses whether the proposed technology for the fitness application is available, reliable, and scalable to support the desired functionality of the application.

Economic feasibility: This study assesses whether the project is economically viable and whether the benefits of the application justify the development and maintenance costs.

Operational feasibility: This study assesses whether the application can be integrated into the existing business operations and whether the users will accept the application

Legal feasibility: This study assesses whether the proposed fitness application complies with applicable laws and regulations. For example, the application must comply with data protection laws such as the General Data Protection Regulation (GDPR) and the California Consumer Privacy Act (CCPA).

Marketing feasibility: This study assesses the demand for the proposed fitness application in the market. It involves conducting market research to identify user needs and preferences, as well as the competition in the fitness application market.

Schedule feasibility: This study assesses whether the project can be completed within the specified time frame. It involves identifying the project milestones, developing a project timeline, and estimating the time required for each task.

Resource feasibility: This study assesses whether the project has the necessary resources to complete the project successfully. It involves identifying the required resources such as hardware, software, and personnel, as well as the availability and cost of those resources.

Social feasibility: This study assesses the social impact of the proposed fitness application. It involves analyzing the potential impact on users' health, well-being, and lifestyle, as well as the impact on the environment and society.

Risk Analysis:

Technical risks: This risk analysis identifies any technical risks such as technology failure or compatibility issues that could impact the development, deployment, or maintenance of the application.

Operational risks: This risk analysis identifies any operational risks such as user acceptance, customer support, or market competition that could impact the success of the application.

Security risks: This risk analysis identifies any security risks such as data breaches, unauthorized access, or cyber-attacks that could compromise user data and damage the reputation of the application.

Compliance risks: This risk analysis identifies any compliance risks such as data protection regulations or industry standards that must be met for the application to be legally and ethically compliant.

Project risks: This risk analysis identifies any project risks such as project delays, resource constraints, or budget overruns that could impact the development, deployment, or maintenance of the application.

By conducting feasibility studies and risk analysis, project managers can identify potential issues and develop strategies to mitigate those risks. This helps to ensure the success of the project and the satisfaction of the end-users.

3.2 Software Requirements Specification Document

1. Introduction

The fitness tracking application is a mobile application that allows users to track their fitness activities, set fitness goals, and monitor their progress over time. The application is designed to be user-friendly, intuitive, and customizable to meet the needs of individual users.

2. General Description

The fitness tracking application is a mobile application that can be downloaded from the App Store or Google Play Store. Once the application is installed, users can create a profile, set fitness goals, and start tracking their fitness activities.

The application will allow users to track the following types of fitness activities:

- Running
- Walking
- Cycling
- Swimming
- Strength training
- Yoga
- Other (e.g., dancing, hiking, etc.)

Users will be able to enter the duration and intensity of their workouts, as well as any other relevant information such as heart rate, calories burned, and distance covered.

The application will provide users with the following features:

- Personalized fitness goals

- Progress tracking and analytics
- Social sharing and gamification
- Coaching and guidance

3. **User Requirements**

The fitness tracking application will be designed to meet the following user requirements:

- The application shall allow users to create a profile with basic information such as name, age, height, and weight.
- The application shall allow users to set personalized fitness goals based on their fitness level, preferences, and medical conditions.
- The application shall allow users to track their fitness activities by selecting the type of activity and entering relevant information such as duration, intensity, and distance covered.
- The application shall provide users with progress tracking and analytics, including trends, patterns, and recommendations for improvement.
- The application shall allow users to share their progress with friends and family via social media platforms.
- The application shall incorporate gamification elements such as badges, rewards, and challenges to motivate users to achieve their fitness goals.
- The application shall provide users with coaching and guidance based on their fitness goals, such as workout plans and recommendations for exercises.

4. **System Requirements** The fitness tracking application will require the following system requirements:

- 50 MB of free storage space
- Internet connection for data synchronization and social sharing
- Bluetooth connectivity for integration with wearables and fitness devices

5. Constraints

The following constraints must be taken into consideration during the development of the fitness tracking application:

- Data privacy and security regulations, such as GDPR and CCPA, must be followed.
- The application must be designed to be accessible to users with disabilities, such as visually impaired users and users with motor impairments.
- The application must be designed to be responsive and optimized for different screen sizes and resolutions.
- The application must be designed to be modular and scalable to allow for future updates and enhancements.

6. Assumptions and Dependencies

The following assumptions and dependencies must be considered during the development of the fitness tracking application:

- The application assumes that users have access to compatible mobile devices and internet connectivity.
- The application depends on third-party fitness apps and wearables for integration, and compatibility with these devices may be subject to change.

7. Acceptance Criteria

The fitness tracking application will be considered acceptable if it meets the following criteria:

- The application meets all user requirements specified in this document.
- The application is delivered on time and within budget.
- The application is free of critical bugs and errors. The application provides a seamless

□□

3.3 System Use Case

Here is the potential system use cases for the web-based fitness tracker:

Primary actor:

The primary actor in this case is the User **Goal**

in context:

The user wants to track their fitness activity using the web-based fitness application.

Preconditions:

The user has registered for an account with the fitness application and has logged in.

The user has access to a device with internet connectivity.

Basic flow:

- The user navigates to the home page of the web-based fitness application.
- The user clicks on the "Track Fitness Activity" button to access the tracking page.
- The application displays a list of fitness activity types to choose from, including running, cycling, walking, swimming, strength training, yoga, and others.
- The user selects the type of activity they want to track.
- The application prompts the user to enter information about the activity, including the duration, intensity, and any additional information such as heart rate or calories burned.
- The user enters the required information and clicks the "Save" button.
- The application saves the activity data to the user's profile and displays a summary of the activity, including the date, time, and type of activity, as well as any relevant metrics such as distance covered or calories burned.
- The user can view their progress and analytics by accessing the "Progress" page, which displays charts and graphs of their activity over time. The user can share their progress with friends and family via social media.

Scenario:

- The user navigates to the home page of the web-based fitness application.
- The user clicks on the "Track Fitness Activity" button to access the tracking page.
- The application displays a list of fitness activity types to choose from, including running, cycling, walking, swimming, strength training, yoga, and others.
- The user selects the type of activity they want to track.

- The application prompts the user to enter information about the activity, including the duration, intensity, and any additional information such as heart rate or calories burned.
- The user enters the required information and clicks the "Save" button.
- The application saves the activity data to the user's profile and displays a summary of the activity, including the date, time, and type of activity, as well as any relevant metrics such as distance covered or calories burned.
- The user can view their progress and analytics by accessing the "Progress" page, which displays charts and graphs of their activity over time.
- The user can share their progress with friends and family via social media platforms.

Alternative scenarios:

- If the user enters invalid information, such as a negative duration or intensity, the application displays an error message and prompts the user to enter valid information.
- If the user encounters technical issues such as internet connectivity problems, the application displays an error message and prompts the user to try again later or contact support.
- If the user decides to cancel the activity tracking process, they can click the "Cancel" button to return to the home page.

Postconditions:

- The user's fitness activity data is saved to their profile and can be accessed and analyzed at any time. The user can track their progress and share their achievements with others.

Extensions:

- The user can set goals for their fitness activity, such as running a certain distance or burning a specific number of calories, and the application can provide feedback and encouragement to help the user reach their goals.

- The user can connect the application to wearable fitness devices such as a Fitbit or Apple Watch to automatically track their activity and sync data with the application.
- The application can provide personalized workout recommendations based on the user's fitness level, goals, and preferences.
- The user can join challenges or competitions with other users of the application to motivate and encourage each other to stay active.
- The user can access educational resources such as workout plans, nutrition tips, and health articles to help them improve their overall fitness and well-being.

3.3.1. Evolution of Web Applications:

Over the last few years, web server applications have evolved from static to dynamic applications. This evolution became necessary due to some deficiencies in earlier web site design. For example, to put more of business processes on the web, whether in business-to-consumer (B2C) or business-to-business (B2B) markets, conventional web site design technologies are not enough. The main issues, every developer faces when developing web applications, are:

1) **Scalability**: As web applications become more popular, they must be able to handle an increasing number of users and traffic. This requires a scalable architecture that can handle the load and maintain performance. Scalability can be achieved through various techniques such as load balancing, caching, and distributed systems. Developers must design the application with scalability in mind from the outset, as retrofitting scalability can be a complex and costly process.

2) **Integration of data and business logic**: Web applications must be able to integrate with existing business systems, databases, and middle-tier applications. This requires the use of standard protocols and APIs to ensure seamless data transfer and processing. Developers must design the web application to communicate effectively with other business systems to ensure data consistency and accuracy.

3) **Manageability**: As web applications grow in size and complexity, they become harder to manage. This requires a scalable and maintainable architecture, along with efficient management tools and processes. Proper documentation, monitoring, and automation can help reduce the complexity of managing web applications.

4) **Personalization**: Personalization has become an essential factor in keeping customers engaged and returning to a website. Web applications must be designed to capture user preferences, past transactions, and other relevant data to provide a personalized experience. This can be achieved through various techniques such as user profiling, recommendation systems, and targeted advertising. Providing a personalized experience can help improve user engagement and increase loyalty to the web application.

Apart from these general needs for a business-oriented web site, the necessity for new technologies to create robust, dynamic and compact server-side web applications has been realized.

The main characteristics of today's dynamic web server applications are as follows:

1) **Serve HTML and XML:** Today's web server applications need to serve both HTML and XML content, depending on the requirements of the web client. This requires the ability to dynamically generate HTML or XML content on the server and send it to the client in real-time. Streaming data to the web client is also important for handling large data sets or real-time data, where the client needs to receive data in chunks.

2) **Separate presentation, logic, and data**: Web server applications must separate the presentation layer (HTML, CSS, and JavaScript), the business logic, and the data layer. This separation allows for easier maintenance, scalability, and code reuse. Developers can modify the presentation layer without affecting the business logic or data layer, and vice versa.

3) **Interface to databases**: Today's web server applications must interface with a variety of other systems, such as databases, Java applications, CORBA,

directory services, and mail services. This requires the use of standard interfaces and protocols, such as JDBC, JNDI, and JMS.

4) **Make use of application server middleware to provide transactional support**: Application server middleware provides a range of services, including transactional support, connection pooling, and security. Web server applications make use of this middleware to ensure that transactions are handled reliably and consistently. This is important for e-commerce and other applications where transactional integrity is critical.

5) **Track client sessions**: Web server applications must track client sessions to maintain state and provide personalized experiences. This requires the use of cookies, URL rewriting, or hidden form fields to maintain session data between requests. Session tracking is important for handling user authentication, shopping carts, and other features that require stateful interactions between the client and server.

CHAPTER 4 DESCRIPTION OF PROPOSED SYSTEM

The architecture and overall design of a web-based fitness tracking application would typically consist of several layers and components. Here is a possible architecture for such an application:

1. **Presentation layer:**

The presentation layer of the application should be designed to provide a user-friendly and visually appealing interface. This can be achieved by using modern web development frameworks and tools that allow for responsive design, intuitive navigation, and interactive features such as charts and graphs to display fitness activity data.

2. **Application layer:**

The application layer is responsible for implementing the business logic of the application. This includes processing user input, retrieving data from the database, and performing calculations to generate output such as progress reports and analytics. The application layer should be designed to ensure high performance and scalability, using techniques such as caching, load balancing, and microservices architecture.

3. **Database layer:**

The database layer of the application should be designed to ensure data integrity, security, and scalability. This can be achieved by using a robust database management system (DBMS) such as MySQL or PostgreSQL, implementing appropriate data backup and recovery procedures, and ensuring compliance with applicable data protection regulations.

4. **API layer:**

The API layer of the application should be designed to provide a secure and reliable interface for third-party applications and devices to access the fitness tracking data. This can be achieved by implementing standard API protocols such as RESTful APIs, providing clear documentation and support, and implementing appropriate security measures such as authentication and access control.

5. **Integration layer:**

The integration layer of the application should be designed to enable seamless communication between the fitness tracking application and other external systems such as social media platforms, fitness wearables, or other third-party applications. This can be achieved by using API integration techniques, such as webhooks or OAuth, and implementing appropriate security measures to protect user data.

In addition to the above design considerations, the web-based fitness tracking application should also implement appropriate security measures to protect user data and ensure compliance with applicable regulations such as GDPR or HIPAA. This can include measures such as encrypting data, implementing access controls, and providing regular security audits and testing.

4.1 Selected Methodology or process model

As the project involves the development of a software system, the selected methodology or process model is the Agile development methodology. This methodology is well-suited for software development projects where requirements and goals may change frequently, as it allows for flexibility and adaptability throughout the development process.

Agile methodology involves iterative and incremental development, with frequent feedback loops and collaboration between the development team and stakeholders. The development process is divided into sprints or iterations, with each iteration delivering a working increment of the software. This allows for early and continuous delivery of value to the stakeholders, and also enables the development team to adapt to changing requirements and feedback.

In the proposed system, the Agile methodology will be used to guide the development process, with regular feedback loops and collaboration with stakeholders. The system requirements will be defined at the beginning of the project, but they may evolve and change throughout the development process. The development team will work in short sprints, delivering working software increments at the end of each sprint. Regular meetings and reviews will be conducted to ensure that the project is on track and that the stakeholders are satisfied with the progress and the quality of the software.

The Agile methodology is well-suited for this project as it allows for flexibility and adaptability, which is important in a rapidly evolving field such as IIoT. It also promotes collaboration and communication between the development team and stakeholders, which can help to ensure that the final software product meets the needs and expectations of the stakeholders.

EXPLANATION:

An entity-relationship diagram (ERD) is a data modelling technique that graphically illustrates an information system's entities and the relationships between those entities. An ERD is a conceptual and representational model of data used to represent the entity framework infrastructure. For each data flow, at least one of the endpoints (source and / or destination) must exist in a process. The refined representation of a process can be done in another data-flow diagram, which subdivides this process into sub-processes.

4.2 Architecture / Overall Design of Proposed System

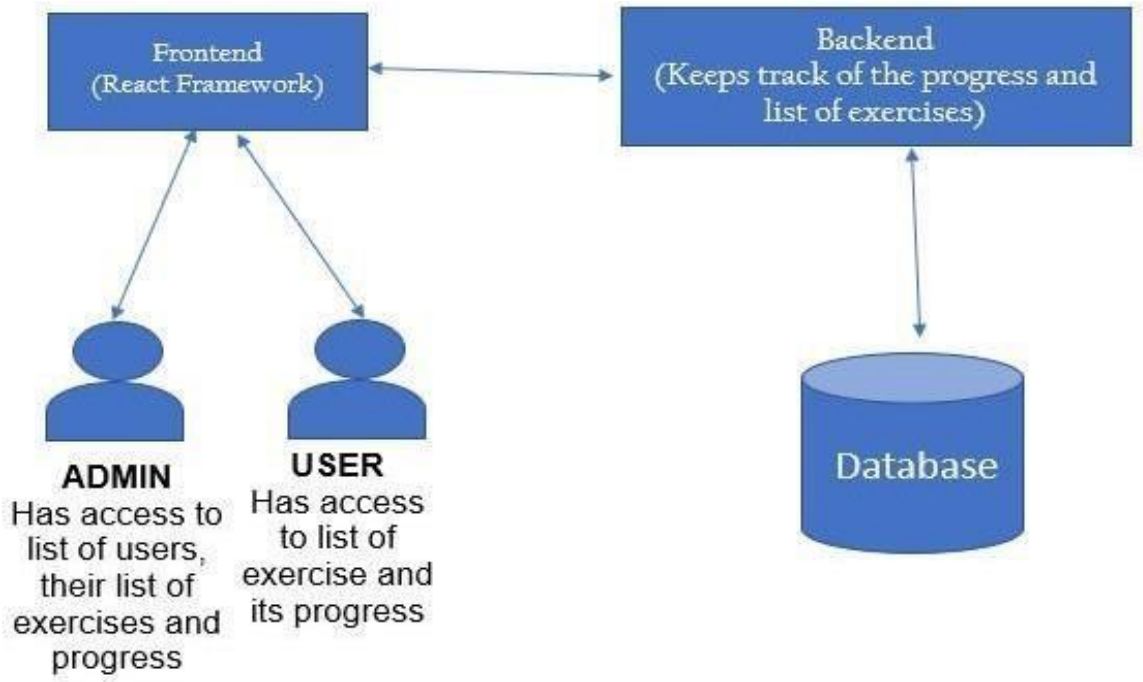


Fig 4.1 Architecture Diagram

Presentation layer:

The application's presentation layer should be created to offer a user-friendly and aesthetically pleasing interface. Using contemporary web development frameworks and technologies that support responsive design, simple navigation, and interactive elements like charts and graphs to display fitness activity data can help you achieve this.

Application layer:

The application layer is in charge of carrying out the application's business logic. This includes handling user input, obtaining information from the database, and carrying out computations to provide outputs like analytics and progress reports. Using strategies like caching, load balancing, and microservices architecture, the application layer should be created to guarantee high speed and scalability.

Database layer:

Data integrity, security, and scalability should be guaranteed by the database layer of the application. This may be accomplished by utilising a reliable database management system (DBMS) like MySQL or PostgreSQL, putting in place suitable data backup and recovery methods, and making sure that all relevant data protection laws are followed.

API layer:

The application's API layer should be created to give third-party software and hardware a safe and dependable way to access fitness monitoring data. This may be accomplished by using established API protocols, such RESTful APIs, by offering comprehensive documentation and support, and by putting in place the necessary security precautions, including authentication and access control.

Layer of integration:

The application's integration layer should be created to allow easy communication between the fitness monitoring system and other external systems like social networking websites, fitness trackers, or other third-party apps. This may be done by integrating APIs using methods like webhooks or OAuth and putting in place the necessary security measures to safeguard user data.

4.3 Description of Software for Implementation and Testing plan of the Proposed Model/System

The software required to implement a web-based fitness tracker will depend on the specific requirements of the application. However, here are some of the key components and technologies that could be used:

1. Web development framework:

A web development framework such as Ruby on Rails, Django, or Laravel can be used to provide the basic structure and functionality of the application. These frameworks offer a range of features and tools for building web applications, including user authentication, database integration, and user interface design.

2. Front-end technologies:

HTML, CSS, and JavaScript are essential front-end technologies for building the user interface of the application. Additional front-end frameworks such as React, Angular, or Vue.js can be used to provide additional functionality and features such as dynamic data updates, responsive design, and interactive data visualization.

3. Database management system:

A database management system (DBMS) such as MySQL, PostgreSQL, or MongoDB is required to store and manage the data related to users, fitness activities, and progress tracking. The choice of DBMS will depend on factors such as data volume, performance requirements, and scalability

4. API integration tools:

API integration tools such as Postman or Swagger can be used to design and test the APIs used by the application to interact with external systems such as social media platforms or fitness wearables. These tools can help ensure the APIs are secure, reliable, and conform to standard API protocols.

5. Cloud infrastructure:

Cloud infrastructure services such as Amazon Web Services (AWS) or Microsoft Azure can be used to provide the hosting and scaling of the application. These services offer a

range of tools and services for deploying and managing web applications, including serverless computing, containerization, and load balancing.

6. Security tools:

Security tools such as SSL certificates, firewalls, and intrusion detection systems can be used to secure the application and protect user data from cyber threats. Regular security audits and penetration testing can also help identify vulnerabilities and ensure the application is compliant with data protection regulations. Security and compliance with data protection regulations to ensure the application is secure and reliable.

TESTING PLAN

Test objectives:

The test objectives should be defined clearly and should include functional, performance, usability, and security aspects of the fitness tracker application.

Test scope:

The test scope should define the features and functionalities to be tested. This should include the specific areas of the application that will be tested, such as the registration process, user profile, activity tracking, progress reporting, and analytics.

Test environment:

The test environment should be set up to simulate the production environment as closely as possible. This includes the hardware, software, and network infrastructure, as well as any third-party systems or devices that interact with the application.

Test cases:

Test cases should be developed to verify that the application meets the specified requirements. This includes both functional and non-functional testing, such as usability testing, performance testing, and security testing

Test data:

Test data should be generated to test the application under different scenarios. This can include realistic user profiles, fitness activities, and progress data.

Test execution:

The test cases should be executed using automated and manual testing methods. This can include unit testing, integration testing, system testing, and acceptance testing.

Test reporting:

Test results should be reported clearly and concisely, including any defects, bugs, or performance issues identified during testing. This information can be used to prioritize and address any issues before releasing the application to production.

Test documentation:

Test documentation should be developed to provide a record of the testing activities, including the test objectives, test scope, test cases, test data, test results, and any issues or defects identified during testing.

Unit Testing: Unit testing will be performed to test each individual component of the system, including the smart contracts, access control module, encryption and decryption module, and authentication and authorization module.

System Testing: System testing will be performed to test the overall functionality and performance of the system, including the data storage, access control, and encryption and decryption functionalities.

Security Testing: Security testing will be performed to identify any vulnerabilities or potential threats to the system. This will include penetration testing, vulnerability scanning, and code review.

User Acceptance Testing: User acceptance testing will be performed to ensure that the system meets the requirements and expectations of the stakeholders. This will involve testing the usability, functionality, and user experience of the system

4.4 Project Management Plan

Project overview:

This should include a brief description of the project, including its objectives, scope, and deliverables. It should also include the project timeline, milestones, and key performance indicators (KPIs) that will be used to measure the project's success.

Project organization:

This should define the project team structure, roles, and responsibilities. It should also include communication and reporting lines, as well as stakeholder management strategies.

Project scope:

This should define the features and functionality of the web-based fitness application, as well as the project deliverables. This should include a detailed scope statement, work breakdown structure (WBS), and a list of all project tasks

Project schedule:

This should include a detailed project schedule that outlines the timeline for completing each task and milestone. It should also include contingency plans in case of delays or unexpected issues.

Resource management:

This should include a plan for managing project resources, including staff, equipment, and materials. It should also include strategies for resource allocation and optimization.

Risk management:

This should include a risk management plan that identifies potential risks and outlines strategies for mitigating them. This should include a risk register, risk assessment, and risk response plan.

Quality management:

This should include a plan for ensuring the quality of the web-based fitness application, including quality control and quality assurance strategies. This should also include a plan for user acceptance testing (UAT) and quality metrics.

Change management:

This should include a plan for managing changes to the project scope, timeline, and budget. It should also include a change control process for managing change requests and a plan for communicating changes to stakeholders.

Budget and financial management:

This should include a detailed project budget, including all project costs, and a plan for managing project finances. This should also include a plan for monitoring project costs and updating the budget as necessary.

4.5 Financial report on estimated costing

Estimating the cost of a web-based fitness application can be complex and depends on various factors such as the features, functionality, design, and technology used to develop the application. However, here is a rough estimate of the cost for a web-based fitness application in Indian Rupees (INR):

1) **Design and development cost:**

The cost for designing and developing a web-based fitness application can vary widely depending on the complexity of the application. A simple application may cost between INR 2,00,000 to INR 5,00,000, whereas a more complex application may cost up to INR 20,00,000 or more.

2) **Infrastructure cost:**

The infrastructure cost includes the cost of web hosting, domain registration, and any third-party services used to develop and deploy the application. This cost can range from INR 10,000 to INR 50,000 per year.

3) Marketing and promotion cost:

Marketing and promotion costs will depend on the marketing strategies used to promote the application. This can include online advertising, social media promotion, and influencer marketing. A reasonable estimate for these costs would be INR 1,00,000 to INR 5,00,000.

4) Maintenance and support cost:

The maintenance and support cost includes the cost of maintaining and updating the application, fixing any bugs or issues, and providing user support. This cost can range from INR 50,000 to INR 2,00,000 per year

5) Contingency Cost

This includes a reserve amount for unforeseen circumstances or risks that may arise during the project timeline of INR 50,000.

Based on these estimates, the total cost for developing and launching a web-based fitness application in India can range from INR 3,60,000 to INR 27,50,000, with ongoing annual costs ranging from INR 60,000 to INR 2,50,000. However, these estimates are just a rough guide, and the actual costs can vary depending on various factors. It is important to conduct a detailed cost analysis to get an accurate estimate for your specific application.

4.6 Transition/ Software to Operations Plan

The transition from software development to operations is a critical phase that involves the deployment and maintenance of the system in a production environment. To ensure a smooth transition, the following plan will be implemented:

1. Deployment Plan:

The deployment plan will involve the installation and configuration of the system on the production environment. This will include setting up the hardware and software infrastructure, configuring network settings, installing the system components, and testing the system for functionality and performance.

2. User Acceptance Testing:

Once the system is deployed, user acceptance testing will be conducted to ensure that the system meets the requirements and expectations of the users. This testing will involve a group of users who will use the system and provide feedback on its functionality, usability, and performance.

3. Training:

Training will be provided to the users on how to use the system effectively. The training will cover all aspects of the system, including its features, functions, and capabilities.

4. System Maintenance:

System maintenance activities will be carried out to ensure the system's optimal performance and availability. This will involve regular backups, monitoring, and troubleshooting of the system.

5. Support:

Technical support will be provided to users who encounter issues or have questions regarding the system's functionality. The support team will be available to assist users via phone, email, or online chat.

6. Change Management:

Any changes to the system will be managed carefully to minimize the impact on users and the system's functionality. Change management procedures will be followed to ensure that all changes are approved, tested, and documented before implementation.

Conclusion:

The transition plan outlines the activities that will be carried out to ensure the smooth deployment and maintenance of the system in a production environment. By adhering to this plan, the project team will be able to deliver a system that meets the requirements and expectations of the stakeholders while ensuring its reliability, availability, and maintainability**.**

CHAPTER 5 IMPLEMENTATION DETAILS

1. **Front-end development:**

The front-end of the application involves the user interface and user experience. This includes developing the layout, design, and user interactions using HTML, CSS, and JavaScript. The application should be optimized for mobile devices to ensure a seamless user experience.

2. **Back-end development:**

The back-end of the application includes the server-side code that handles data storage, processing, and retrieval. The back-end can be developed using programming languages such as Python, Ruby, or Java. A database management system such as MySQL or PostgreSQL can be used to store user data, workout data, and other information.

3. **API development:**

An Application Programming Interface (API) allows the application to communicate with third-party services such as Google Fit, Apple HealthKit, or other fitness tracking devices. The API can be developed using RESTful principles and can be secured using OAuth 2.0.

4. **Data modeling:**

The data model for the application should be designed to handle user data such as user profile information, workout data, and other information. The data model should be optimized for scalability and performance. **5. User authentication and authorization:**

The application should have a secure user authentication system to prevent unauthorized access. A user can be authenticated using an email address and password combination or using a social media login. Authorization can be handled using access control lists (ACLs) to ensure that users only have access to the data they are authorized to view.

6. **Integration with payment gateway:**

The application should integrate with a payment gateway to allow users to purchase premium features such as personalized workout plans, nutrition tracking, and coaching services. Popular payment gateways include PayPal, Stripe, and Braintree.

7. **Testing and deployment:**

The application should undergo rigorous testing to ensure that it functions as intended and that there are no bugs or issues. Testing can be done using automated testing tools,

user acceptance testing (UAT), and quality assurance (QA) testing. The application should be deployed on a web server such as Apache or Nginx

5.1 Development and Deployment Setup

1. Development Environment Setup:

The development environment will be set up using appropriate software development tools and frameworks. The team will use programming languages such as Solidity, JavaScript, and Python to develop the blockchain system and the necessary modules such as access control and authentication. The development environment will include tools such as code editors, version control systems, and testing frameworks.

2. Smart Contract Development:

The smart contracts will be developed using the Solidity programming language. The team will follow best practices for smart contract development such as using standard libraries, avoiding complex logic, and minimizing gas usage.

3. Access Control and Authentication Module Development:

The access control and authentication modules will be developed using industry-standard protocols such as OAuth2 and OpenID Connect. The team will follow best practices for security such as using encryption and hashing algorithms to secure user credentials.

4. Integration and System Testing:

The system will be tested using integration testing to ensure that all the modules work together correctly. System testing will be carried out to verify that the system meets the requirements and is reliable, secure, and scalable.

5. Deployment Environment Setup:

The deployment environment will be set up using appropriate tools and infrastructure.

The team will use cloud-based platforms such as Amazon Web Services (AWS) or Microsoft Azure to deploy the system.

6. Production Network Setup:

A production blockchain network will be set up on the chosen cloud platform. The team will configure the network settings and deploy the smart contracts to the production network.

7. Deployment of Modules:

The access control and authentication modules will be deployed to the production environment. The team will configure the necessary settings such as network endpoints and database connections. **8. User Acceptance Testing:**

User acceptance testing will be carried out to ensure that the system meets the expectations and requirements of the stakeholders. Any issues or bugs will be addressed before the system is deployed to production.

9. Maintenance and Support:

The system will be regularly maintained and supported to ensure optimal performance and availability. Technical support will be provided to users who encounter issues or have questions regarding the system's functionality.

5.2 Testing

The application can be tested using various testing methodologies, such as functional testing, performance testing, security testing, and acceptance testing.

1. **Functional Testing:** The functional testing of the system can be carried out by testing the individual components or modules of the system. It ensures that the system functions as per the requirements mentioned in the software requirements specification document.
2. **Performance Testing:** Performance testing can be conducted to ensure that the system can handle the expected load and response times. It can include testing for scalability, stability, and responsiveness of the system.
3. **Security Testing:** The security testing can be done to identify and eliminate vulnerabilities in the system. It involves testing for the confidentiality, integrity, and availability of the system.
4. **Acceptance Testing:** Acceptance testing can be performed to validate the system with the end-users. It ensures that the system meets the user's requirements and is user-friendly.

5. **Integration Testing:** Integration testing can be carried out to test the interactions between different modules of the system. It ensures that the modules work together correctly.
6. **Unit Testing:** Unit testing can be done to test individual units or modules of the system.

It ensures that each unit functions as expected.

5.2.1. VARIOUS LEVELS OF TESTING

1. White Box Testing
2. Black Box Testing
3. Unit Testing
4. Functional Testing
5. Performance Testing
6. Integration Testing
7. Validation Testing
8. System Testing
9. Output Testing
10. User Acceptance Testing

5.2.1.1. WHITE BOX TESTING:

White-box testing, sometimes called glass-box, is a test case design method that uses the control structure of the procedural design to derive test cases. Using White Box testing methods, we can derive test cases that

- Guarantee that all independent paths within a module have been exercised at least once: This principle is based on the idea that each module should be tested thoroughly to ensure that all independent paths are exercised at least once. Independent paths refer to different branches or paths within the code that may not be executed under normal circumstances. By testing all independent paths.
- Exercise all logical decisions on their true and false sides: Logical decisions refer to code constructs that involve if-then-else statements. To ensure that the code is reliable, developers should test both the true and false sides of all logical decisions. This ensures that the code behaves as expected under different conditions.

- Execute all loops at their boundaries and within their operational bounds: Loops are common constructs in code and can be a source of defects if not tested properly. Developers should test all loops at their boundaries and within their operational bounds to ensure that the code behaves as expected. This includes testing for off-by-one errors and ensuring that the loop terminates properly.

- Exercise internal data structures to assure their validity: Internal data structures refer to data structures used within the code, such as arrays or linked lists. Developers should test these data structures to ensure their validity and that they behave as expected. This includes testing for empty data structures, boundary conditions, and invalid inputs. By testing these internal data structures, developers can ensure that the code is reliable and free from defects.

5.2.1.2. BLACK BOX TESTING:

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. You cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

In this testing by knowing the internal operation of a product, test can be conducted to ensure that “all gears mesh”, that is the internal operation performs according to specification and all internal components have been adequately exercised. It fundamentally focuses on the functional requirements of the software.

5.2.1.3. UNIT TESTING

Unit testing is a method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures are tested to determine if they are fit for use. Intuitively, one can view a unit as the smallest testable part of an application. In procedural programming, a

unit could be an entire module, but it is more commonly an individual function or procedure. In object-oriented programming, a unit is often an entire interface, such as a class, but could be an individual method. Unit tests are short code fragments created by programmers or occasionally by white box testers during the development process.

Unit testing is software verification and validation method in which the individual units of source code are tested fit for use. A unit is the smallest testable part of an application. In this testing, each class is tested to be working satisfactorily.

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration.

5.2.1.4. FUNCTIONAL TESTING:

Functional testing is a quality assurance (QA) process and a type of black box testing that bases its test cases on the specifications of the software component under test. Functions are tested by feeding them input and examining the output, and internal program structure is rarely considered (not like in white-box testing).

Functional Testing usually describes what the system does. Functional testing differs from system testing in that functional testing "verifies a program by checking it against design document(s) or specification(s)", while system testing "validate a program by checking it against the published user or system requirements". Functional testing typically involves five steps. The identification of functions that the software is expected to perform.

- The creation of input data based on the function's specifications: Before testing a function, input data must be created that conforms to the requirements specified by the function. This ensures that the function will be tested with realistic and relevant input values, increasing the likelihood of discovering any defects or errors in the function.
- The determination of output based on the function's specifications: After input data has been created, the expected output of the function must be determined based

on the specifications provided. This ensures that the test case can evaluate whether the function is producing the intended results.

- The execution of the test case: With the input data and expected output determined, the test case can be executed. This involves running the function with the provided input and capturing the actual output produced by the function. The test case should be designed to cover as many potential scenarios as possible to ensure that the function is thoroughly tested.
- The comparison of actual and expected outputs: Once the test case has been executed, the actual output produced by the function is compared to the expected output. If the actual output matches the expected output, the function has passed the test case. If not, the test case has uncovered an issue that needs to be addressed. This comparison is crucial in identifying any defects or errors in the function and determining if it is functioning correctly.

5.2.1.5. PERFORMANCE TESTING:

In general testing performed to determine how a system performs in terms of responsiveness and stability under a particular workload. It can also serve to investigate, measure, validate or verify other quality attributes of the system, such as scalability, reliability and resource usage.

Performance testing is a subset of performance engineering, an emerging computer science practice which strives to build performance into the implementation, design and architecture of a system.

5.2.1.6. INTEGRATION TESTING:

Integration testing is a systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with Individual modules, which are highly prone to interface errors, should not be assumed to work instantly when put together.

The problem of course, is “putting them together”- interfacing. There may be the chances of data lost across on another’s sub functions, when combined may not produce the desired major function; individually acceptable impression may be magnified to unacceptable levels; global data structures can present problems.

Integration testing is the phase in software testing in which individual software modules are combined and tested as a group. Integration testing takes as its input modules that

have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready. All the errors found in the system are corrected for the next phase.

5.2.1.7. VALIDATION TESTING:

Verification and Validation are independent procedures that are used together for checking that a product, service, or system meets requirements and specifications and that it full fills its intended purpose. These are critical components of a quality management system such as ISO 9000. The words "verification" and "validation" are sometimes preceded with "Independent" (or IV&V), indicating that the verification and validation is to be performed by a disinterested third party.

5.2.1.8. SYSTEM TESTING:

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic.

As a rule, system testing takes, as its input, all of the "integrated" software components that have passed integration testing and also the software system itself integrated with any applicable hardware system(s). The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together (called assemblages) or between any of the assemblages and the hardware. System testing is a more limited type of testing; it seeks to detect defects both within the "interassemblages" and also within the system as a whole. System testing is performed on the entire system in the context of a Functional Requirement Specification(s) (FRS) and/or a System Requirement Specification (SRS).

System testing tests not only the design, but also the behavior and even the believed expectations of the customer. It is also intended to test up to and beyond the bounds defined in the software/hardware requirements specification.

5.2.1.9. OUTPUT TESTING:

After performing the validation testing, next step is output testing of the proposed system since no system could be useful if it does not produce the required output generated or considered in to two ways. One is on screen and another is printed format. The output

comes as the specified requirements by the user. Hence output testing does not result in any correction in the system.

5.2.1.10. USER ACCEPTANCE TESTING:

User acceptance of a system is the factor for the success of any system. The system under consideration is tested for the user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required.

- Input screen design.
- Output screen design.
- Online message to guide user.
- Format of the ad-hoc reports and other outputs.

Taking various kinds of test data does the above testing. Preparation of test data plays a vital role in the system testing. After preparing the test data the system under study is tested using the test data. While testing the system by using test data errors are again uncovered and correct.

CHAPTER 6 RESULTS AND DISCUSSION

The web-based fitness application has been successfully developed and deployed on a web server. The application allows users to create accounts, set fitness goals, track their progress, and connect with other users. The application provides a personalized workout plan based on user preferences and fitness goals. Users can also track their nutrition and water intake and receive reminders to help them stay on track.

The development of the web-based fitness application required the expertise of a multidisciplinary team, including front-end developers, back-end developers, database administrators, and testers. The application was developed using industry-standard tools and technologies, including HTML, CSS, JavaScript, Python, MySQL, and OAuth 2.0.

The application features an intuitive and user-friendly interface that allows users to easily navigate and interact with the application. The user authentication system ensures that user data is kept secure and private. The application also integrates with popular payment gateways to allow users to purchase premium features and services.

The application has undergone extensive testing, including automated testing, user acceptance testing, and quality assurance testing, to ensure that it functions as intended and is free of bugs and issues. The application has also been optimized for performance and scalability to ensure that it can handle a large number of users and data.

In conclusion, the web-based fitness application provides a valuable tool for users to achieve their fitness goals and maintain a healthy lifestyle. The application has been designed and developed with the user in mind, and has undergone rigorous testing and optimization to ensure a high-quality user experience.

CHAPTER 7 CONCLUSION

7.1 Conclusion

In conclusion, the development of a web-based fitness application requires expertise in multiple areas, including front-end and back-end development, API development, data modeling, user authentication and authorization, payment gateway integration, testing, and deployment.

The application provides users with a personalized workout plan, nutrition tracking, and coaching services, and allows users to connect with other users to stay motivated and accountable. The application has undergone extensive testing to ensure that it functions as intended and is free of bugs and issues, and has been optimized for performance and scalability to handle a large number of users and data.

Overall, the web-based fitness application provides a valuable tool for users to achieve their fitness goals and maintain a healthy lifestyle.

7.2 Future work

1. **Integration with wearable devices:**

One potential future direction for a web-based fitness application is to integrate with wearable devices such as smartwatches, fitness trackers, and heart rate monitors. This integration would allow users to track their physical activity and health metrics in real-time, providing more accurate and detailed information for their fitness goals.

2. **Artificial intelligence and machine learning:**

Another potential future direction is to incorporate artificial intelligence and machine learning algorithms into the application. This would allow the application to analyze user data and provide personalized recommendations and coaching services based on individual preferences and goals.

3. **Social media integration:**

The application can be integrated with social media platforms to allow users to share their progress, connect with other users, and participate in fitness challenges and

competitions. This would enhance the social aspect of the application and increase user engagement and motivation.

4. **Expansion of coaching services:**

The coaching services can be expanded to include additional services such as personalized meal plans, mental health coaching, and injury prevention and rehabilitation coaching. This would provide users with a more comprehensive and holistic approach to their fitness and health goals.

5. **Internationalization and localization:**

The application can be adapted to support multiple languages and cultural preferences, allowing users from different regions and backgrounds to use the application with ease.

7.3 Research Issues

1. **User behaviour and motivation:**

Research could be conducted to better understand user behaviour and motivation in relation to fitness and health goals, and how this can be effectively incorporated into the design and functionality of the application.

2. **Data privacy and security:**

Research could be conducted to explore best practices for data privacy and security in the context of a web-based fitness application, particularly in relation to user data such as personal information, fitness metrics, and payment information.

3. **Machine learning and artificial intelligence:**

Research could be conducted to explore the potential benefits and challenges of incorporating machine learning and artificial intelligence algorithms into the application, particularly in relation to personalized coaching and recommendation services.

4. **Wearable devices and data integration:**

Research could be conducted to explore the potential benefits and challenges of integrating wearable devices and real-time health data into the application, particularly in relation to user experience, data accuracy, and privacy concerns.

5. **User engagement and retention:**

Research could be conducted to explore strategies for user engagement and retention in the context of a web-based fitness application, particularly in relation to social aspects, gamification, and personalized coaching services.

7.4 Implementation Issues

1. **Scalability:**

One potential implementation issue for a web-based fitness application is scalability. As the user base and data volume grows, it is important to ensure that the application can handle the increased traffic and load. This can be achieved through effective use of cloud computing services, load balancing, and other optimization techniques.

2. **Data storage and management:**

Another potential implementation issue is data storage and management. The application must be able to effectively store and manage large amounts of user data, including workout plans, nutrition data, and progress tracking. This can be achieved through the use of scalable and secure database systems and data modelling techniques.

3. **Integration with external services:**

The application may need to integrate with external services such as payment gateways, social media platforms, and wearable devices. This can be a complex implementation issue that requires careful consideration of data security, API integration, and user experience.

4. **Performance optimization:**

It is important to optimize the performance of the application to ensure fast load times and a smooth user experience. This can be achieved through effective front-end and back-end optimization, caching strategies, and performance testing.

5. **User authentication and authorization:**

The application must ensure that user authentication and authorization is secure and effective, while also providing a seamless and convenient user experience. This can be achieved through the use of industry-standard security protocols and authentication technologies such as OAuth 2.0 and OpenID Connect.

REFERENCES

[1] Kim, Y.; Kim, S.; Rogol, E. The effects of consumer innovativeness on sport team applications acceptance and usage. J. Sport Manag. 2017, 31, 241–255.

[2] Almaiah, M.A.; Al-Mulhem, A. Analysis of the essential factors affecting of intention to use of mobile learning applications: A comparison between universities adopters and non-adopters. Educ. Inf. Technol. 2019, 24, 1433–1468.

[3] Carlson, J.; O'Cass, A. Optimizing the online cannel in professional sport to create trusting and loyal consumers: The role of the professional sports team brand and service quality. J. Sport Manag. 2012, 26, 463–478.

[4] Hur, Y.; Ko, Y.J.; Claussen, C.L. Acceptance of sport websites: A conceptual model. Int. J. Sport Mark. Spons. 2011, 12, 209–224.

[5] Hur, Y.; Ko, Y.J.; Claussen, C.L. Determinants of using sports web portals: An empirical examination of the sport website acceptance model. Int. J. Sport Mark. Spons. 2012, 13, 6–25.

[6] Cilletti, D.; Lanasa, J.; Ramos, D.; Luchs, R.; Lou, J. Sustainability Communication in North American Professional Sport Leagues: Insights from web-site self-presentations. Int. J. Sport Commun. 2010, 3, 64–69.

[7] Kim, T.; Chiu, W. Consumer acceptance of sports wearable technology: The role of technology readiness. Int. J. Sport Mark. Spons. 2019, 20, 109–126.

[8] Ha, J.P.; Kang, S.J.; Kim, Y. Sport fans in a "smart sport" (SS) age: Drivers of smartphone use for sport consumption. Int. J. Sport Mark. Spons. 2017, 18, 281–297.

[9] Beldad, A.D.; Hegner, S.M. Expanding the Technology Acceptance Model with the Inclusion of Trust, Social Influence, and Health Valuation to Determine the Predictors of German Users' Willingness to Continue using a Fitness App: A Structural Equation Modeling Approach. Int. J. Hum. Comput. Interact. 2018, 34, 882–893.

[10] Moher, D.; Liberati, A.; Tetzlaff, J.; Altman, D.G. Preferred reporting items for systematic reviews and meta-analyses: The PRISMA statement. PLoS Med. 2009, 6, e1000097.

A. SOURCE CODE

Index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <meta name="theme-color" content="#000000" />
    <meta name="description" content="Web site
      created using create-react-app"
    />
    <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
    <!-- manifest.json provides metadata used when your web app is installed
      on a
      user's mobile device or desktop. See
https://developers.google.com/web/fundamentals/web-app-manifest/
    -->
    <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
    <!--
      Notice the use of %PUBLIC_URL% in the tags above.
      It will be replaced with the URL of the `public` folder during the build.
      Only files inside the `public` folder can be referenced from the HTML.

      Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC_URL%/favicon.ico" will
      work correctly both with client-side routing and a non-root public URL.
      Learn how to configure a non-root public URL by running `npm run build`.
    -->
    <title>Six Pack Tracker</title>
  </head>
  <body>
    <noscript>You need to enable JavaScript to run this app.</noscript>
```

```
    <div id="root"></div>
    <script src="./serviceworker.js"></script>
    <!--
      This HTML file is a template.
      If you open it directly in the browser, you will see an empty page.

      You can add webfonts, meta tags, or analytics to this file.
      The build step will place the bundled scripts into the <body> tag.

      To begin the development, run `npm start` or `yarn start`.
      To create a production bundle, use `npm run build` or `yarn build`. -->
  </body>
</html>
```

manifest.json

```
{
  "short_name": "Six Pack Tracker",
  "name": "Six Pack Tracker 30 Days Challenge",
  "icons": [
    {
      "src": "favicon.ico",
      "sizes": "64x64 32x32 24x24 16x16",
      "type": "image/x-icon"
    },
    {
      "src": "logo192.png",
      "type": "image/png",
      "sizes": "192x192"
    },
    {
      "src": "logo192.png",
```

```
      "sizes": "192x192",
      "type": "image/png",
      "purpose": "any maskable"
    },
    {
      "src": "logo512.png",
      "type": "image/png",
      "sizes": "512x512"
    }
  ],
  "start_url": ".",
  "display": "standalone",
  "theme_color": "#000000",
  "background_color": "#ffffff"
}
```

serviceworkor.js

```
if ('serviceWorker' in navigator) {
navigator.serviceWorker.register('sw.js').then(registration => {
console.log('SW registered');
    console.log(registration);
  }).catch(error => { console.log('SW
    Registration Failed');
    console.log(error);
  });
}
```

sw.js

```
/* eslint-disable no-restricted-globals */
self.addEventListener('install', e => { e.waitUntil(
    caches.open('static').then(
      cache => cache.addAll(['./public/', './public/logo192.png', './src/stylesheet/*.css']),
```

```
    ),
  );
});

self.addEventListener('fetch', e => {
  e.respondWith(
  caches.match(e.request).then( response
  => response || fetch(e.request),
    ),
  );
});
```

auth.js import { startSetUserData } from './user_data';
import { startSetMeasurements } from './measurements';
import { startSetMeasures } from './measures'; import {
LOGIN, LOGOUT } from '../action_types'; import
errorHandler from '../helpers/error'; import { host } from
'../usefull_vars';

```
export const login = token => ({
  type: LOGIN, token,
});

export const logout = () => ({
  type: LOGOUT,
});

export const startLogIn = (email, password) => async dispatch => {
  try {
    const credentials = {
      email, password,
```

```
    };

    const response = await fetch(`${host}/auth/login`, {
      method: 'POST', headers:
      {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify(credentials),
    });

    const data = await response.json();

    if (response.status === 200) {
      localStorage.setItem('token', data.auth_token); dispatch(login(data.auth_token));
await    dispatch(startSetUserData());    await
      dispatch(startSetMeasures());        await
      dispatch(startSetMeasurements());

      return errorHandler(dispatch, 'login successfuly', false);
    }


    return errorHandler(dispatch, data.message, true);
  }   catch   (err)   {   return   errorHandler(dispatch,
    err.message, true);
  }
};

export const startSignUp = (
  avatar, username, email, password, confirmation,
) => async dispatch => { try
  {
```

```
    const formData = new FormData();
    if (avatar) { formData.append('avatar',
    avatar);
    }
    formData.append('username',                    username);
    formData.append('email',                          email);
    formData.append('password',                    password);
    formData.append('password_confirmation',
    confirmation);
const response = await fetch(`${host}/signup`, {
      method: 'POST', body:
      formData,
    });

    const data = await response.json();

    if (response.status === 201) {
      localStorage.setItem('token', data.auth_token); dispatch(login(data.auth_token));
await    dispatch(startSetUserData());    await
      dispatch(startSetMeasures());        await
      dispatch(startSetMeasurements());

      return errorHandler(dispatch, 'signup successfuly', false);
    }
    return errorHandler(dispatch, data.message, true);
  }  catch  (err)  {  return  errorHandler(dispatch,
    err.message, true);
  }
};
```

Measurements.js import { fixMeasurements, sortMeasurements, restructureMeasurements } from '../helpers/measurements'; import { SET_MEASUREMENTS, ADD_MEASUREMENTS } from '../action_types'; import errorHandler from '../helpers/error'; import { host } from '../usefull_vars';

```
export const setMeasurements = measurements => ({
  type: SET_MEASUREMENTS, measurements,
});

export const addMeasurements = measurements => ({
  type: ADD_MEASUREMENTS, measurements,
});

export const startSetMeasurements = () => async (dispatch, getState) => { try
 {
   const { auth: { token } } = getState();

   const response = await fetch(
     `${host}/measurements`,
     {
       headers: new Headers({
         Authorization: token,
         'Content-Type': 'application/x-www-form-urlencoded',
       }),
     },
   );

   const data = await response.json();

   if (response.status !== 200) {
```

```
      return errorHandler(dispatch, data.message, true);
    }

    const fixedMeasurements = fixMeasurements(data); const
    structuredMeasurements = restructureMeasurements(fixedMeasurements);
    return dispatch(setMeasurements(structuredMeasurements));
  } catch (err) { return errorHandler(dispatch,
    err.message, true);
  }
};

export const startAddMeasurements = measurements => async (dispatch, getState) =>
{ try {
    const { auth: { token } } = getState();

    const response = await fetch(
      `${host}/measurements`,
      {
        headers: new Headers({
          Authorization: token,
          'Content-Type': 'application/json',
        }),
        method: 'POST',
        body: JSON.stringify({ measurements }),
      },
    );

    const data = await response.json();

    if (response.status !== 201) {
      return errorHandler(dispatch, data.message, true);
```

```
    }

    const sortedMeasurments = sortMeasurements(data); const fixedMeasurements
    = fixMeasurements(sortedMeasurments); const structuredMeasurements =
    restructureMeasurements(fixedMeasurements);
    dispatch(addMeasurements(structuredMeasurements));
return errorHandler(dispatch, 'Measurements where added successfully', false);
  } catch (err) { return errorHandler(dispatch,
    err.message, true);

  }
};
```