Final Year Project

HealthHarmony - A Comprehensive Intelligent Fitness and Diet Advisor via Metaverse

Jingyu Lu

Student ID: 20205767

A thesis submitted in part fulfilment of the degree of

BSc. (Hons.) in Software Engineering

Supervisors: Dr Catherine Mooney & Dr Ruihai Dong, Dr Hadi Tabatabaee

Beijing Dublin International College

University College Dublin & Beijing University of Technology

July 26, 2024

Table of Contents

Abstract

HealthHarmony is a comprehensive fitness and diet consulting platform designed to improve user health management through personalized data-driven services. The platform combines machine learning technology with advanced data analytics to provide users with customized diet and fitness plans. By dynamically adjusting these plans in real-time, HealthHarmony helps users optimize their daily lifestyle and health habits based on their specific needs and health goals.

The project adopts cutting-edge technologies, such as virtual reality, explainable artificial intelligence and real-time scenario algorithms, which are organically integrated with the fitness and diet modules to enhance the immersive experience of users and improve the personalization of services. In addition, the platform features a community support function that encourages users to share their health experiences, thereby building a positive and supportive health community.

Project area: Machine Learning, Deep Learning, Health Tech, Virtual Reality

Keywords: Personalized health management, machine learning, data visualization, fitness program customization, community interaction, 3D scene modeling, Explainable Artificial Intelligence

Chapter 1: **Introduction**

With the accelerated pace of modern life and increased work pressure, people are increasingly aware of the importance of health, however, most users do not have systematic knowledge of fitness and diet, and hiring a fitness consultant is usually expensive. Meanwhile, existing health apps on the market often lack personalization and fail to meet the specific needs of users. HealthHarmony was created to not only fill the gap in the market, but also improve user immersion and satisfaction through advanced technology, providing a truly comprehensive health management solution. The platform gives users an immersive experience by combining virtual reality and real-time feedback, making it a powerful tool for improving quality of life. In this way, HealthHarmony not only promotes the improvement of personal health, but also advances the health tech space, bringing new perspectives and technological innovations to the health tech sector.

HealthHarmony aims to create a comprehensive online health consulting platform focused on providing personalized fitness and diet solutions. This project belongs to the health technology sector, applying machine learning and big data analytics to provide customized dietary and fitness guidance based on physiological data and lifestyle habits of users.

HealthHarmony is a health management platform based on Three.js technology, aimed at providing users with an immersive exercise and dietary guidance experience. As an innovative application that integrates web pages and 3D metaverse, HealthHarmony fully utilizes the powerful features of Three.js. Through Three.js, HealthHarmony is able to build realistic 3D virtual environments in web browsers, allowing users to immerse themselves in exploring health knowledge and completing various health tasks.

For example, users can perform virtual exercises in a 3D simulated gym, observe their movements in real-time, and receive professional feedback and guidance. At the same time, users can also learn to cook healthy food in the 3D kitchen and understand the nutritional content of various ingredients through intuitive 3D displays. This immersive interactive experience not only better stimulates user interest, but also allows them to have a deeper understanding and mastery of health knowledge.

With the continuous development of technologies such as WebAssembly, HealthHarmony has improved the performance and scalability of 3D applications on web pages. With the support of the 3D metaverse concept, HealthHarmony has achieved rich features such as real-time rendering and virtual reality. The combination of 3D and 2D brings users an unprecedented health management experience.

Traditional AI models are often considered "black boxes" due to their complex and opaque internal decision-making processes. Explainable Artificial Intelligence (XAI) [1] addresses this issue by designing AI models and systems whose decisions can be understood and explained by humans. XAI transforms the landscape by making the decision-making processes of AI more transparent and comprehensible.

With this, HealthHarmony is focused on bringing new thinking to the software development space. In the current era of big models, as the number of parameters in a model increases exponentially, in many cases developers do not have a good understanding of how these models make decisions and simply use them in a packaged way. But this is irresponsible; users tend to trust the decisions given by models in areas they don't understand, but this can lead to serious problems if the model gives the wrong decision.

HealthHarmony triggers a reflection on developer responsibility by bringing in XAI, suggesting that developers should be responsible for the results given by models, and that users and developers are more likely to trust and rely on these systems when they are able to understand how an AI model makes a particular decision.

At the same time, given that HealthHarmony operates within the highly regulated medical and health sector, it is imperative that we strictly adhere to European regulations such as the General Data Protection Regulation (GDPR) [2] and the Medical Devices Regulation (MDR) [3], as well as U.S. regulations including the Health Insurance Portability and Accountability Act (HIPAA) [4] and the Food and Drug Administration's (FDA) AI policies [5]. These regulations not only ensure the privacy and security of patient data but also demand that AI applications meet high standards of safety and effectiveness. By ensuring our AI systems comply with these stringent regulations, we can protect user rights and enhance both the social acceptance and competitive position of our project.

Chapter 2: Project Specification

2.1 The main demands of the project

The functionality and evaluation of HealthHarmony will be organized around the following exhaustive core and advanced objectives:

2.1.1 Core goals

- **Deeply Personalized Fitness Advice:** The platform will synthesize the physiological parameters (e.g. weight, height, age), lifestyle and health goals of the user and other indicators to provide scientific and accurate health and fitness advice through machine learning models.
- **Intelligent Dietary Recommendation System:** The system will analyze the body parameters and fitness intensity of the user and generate personalized nutritional recipes using advanced machine learning techniques. Recipes will be displayed in a 3D view to enhance the user's interactive experience and understanding. It also allows users to customize the diet plan.
- **Customized Fitness Plans:** Based on the individual fitness level given by the machine learning model, a tailor-made fitness program is provided. The plan includes diverse exercise programs and video demonstrations to help users accurately master various exercise skills.

2.1.2 Advanced goals

- **Virtual Reality Scenes and Interaction:** Introducing multiple types of 3D scenes, such as gyms, kitchens and personal spaces, based on instant scene algorithms to ensure an immersive fitness experience for users. Meanwhile, using the coordinate movement between different scenes, instead of the traditional webpage jumping.
- **Explainable Artificial Intelligence:** Because our project is in the healthcare sector, which is subject to strict regulatory requirements, introducing XAI has helped us demonstrate how models can comply with regulations, especially when it comes to physical health and privacy protection. At the same time, when users are able to understand how AI models make specific decisions, they are more likely to trust and rely on these systems.
- **Community interaction and support:** Build a supportive community that allows users to share health and fitness experiences, seek advice, and interact with other users to form a positive community support network.

Chapter 3: User Documentation

3.1 Project Introduction

3.1.1 Problem Statement

In modern society, as the pace of life accelerates and work pressure increases, people are paying more attention to health and fitness. However, they face challenges in effectively managing and improving their personal health. At present, health and fitness applications on the market generally lack personalization and precision, which cannot fully meet individual needs. Therefore, we have developed a 3D intelligent platform that can integrate personal health data to provide customized nutrition and fitness recommendations. This project takes into account the specific personal circumstances of users, including their physical data, lifestyle, dietary preferences, and health goals, to help them find suitable diets and fitness plans. Users can obtain personalized, comprehensive, and scientific health management plans here to meet their specific needs and conditions.

3.1.2 How to Access our Project

Suggestion 1 Using Chrome browser to ensure a perfect experience

Suggestion 2 Starting with a decent internet connection environment

Table 3.1: Website

Website	http://csi420-01-vm9.ucd.ie/

Table 3.2: Login Mail And Password

Email	itislunalight@gmail.com
Password	Cc123456

3.2 Functional Detail

3.2.1 Login Portal

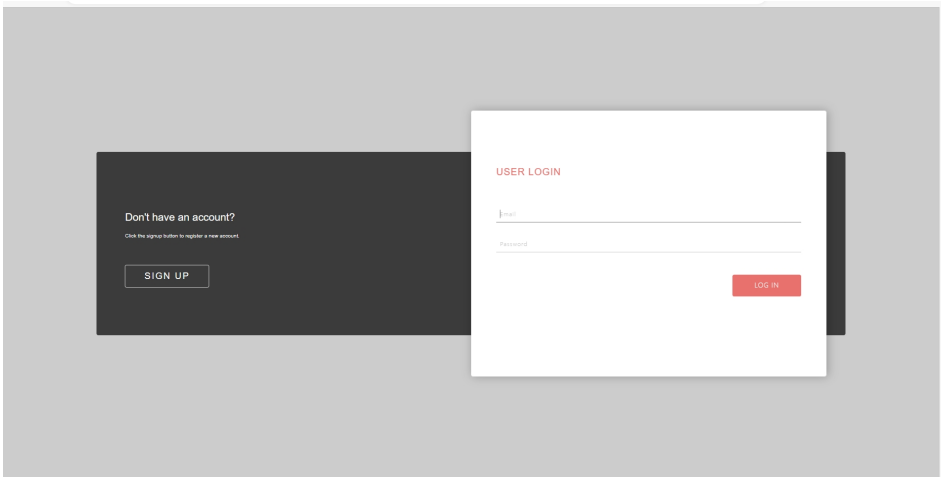

Figure 3.1: Login Portal

Once the user has logged in and registered, they will be taken to the home page. On the left side of the page, a 3D model is built and animated. On the right side is a menu list, divided into five parts: 3D mode, 2D mode, safe haven, community, and about us. Users can enter the corresponding module by clicking the button.

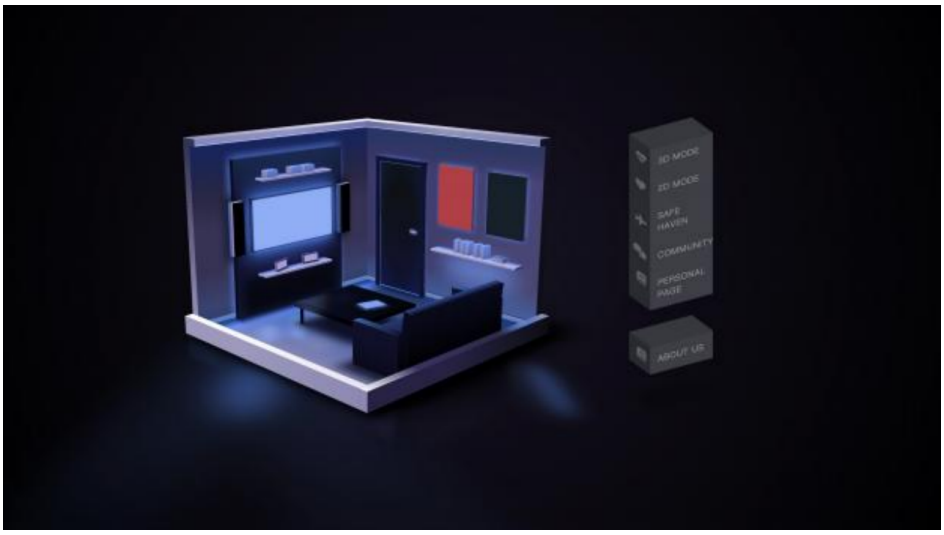

Figure 3.2: Home Page

3.2.2 3D Mode Page

User can enter the scene by clicking on the 3D mode link from the homepage. This will take a few seconds, waiting for all 3D models to be generated.

Figure 3.3: Select 3D Mode

Special Note For 3D World:

Due to performance optimization considerations, we have not set map collision boundaries. Therefore, players are advised not to cross walls to ensure a smooth gaming experience Due to network fluctuations, the waiting time has ended and the 3D scene has not been loaded. Please be patient and wait.

A. 3D Open World Lobby

This page uses advanced visualization technologies to construct a three-dimensional scene to enhance user experience. The scene consists of four parts: a lobby, a gym, a kitchen, and a personal room. After entering the scene, user can use the arrow keys on keyboard to control the perspective and move. The player's starting point is located in the lobby, the personal room is directly in front of the perspective, the kitchen is on the right, and the gym is on the left front.

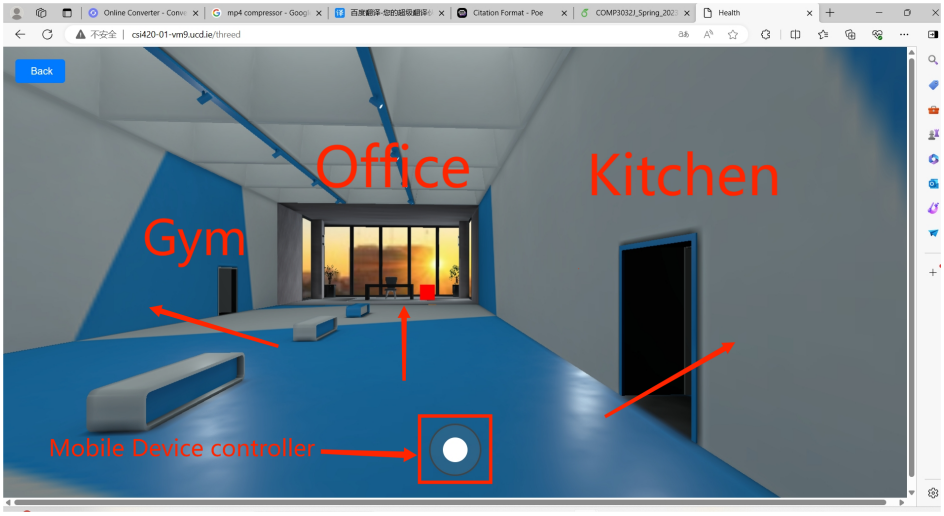

Figure 3.4: Lobby

B. Access To Fitness Plan Page

After entering other rooms from the lobby, user can drag the circle in the middle of the screen to control themselves to collide to the red cube in the scene to jump into corresponding module. Access the fitness plan page from the gym.

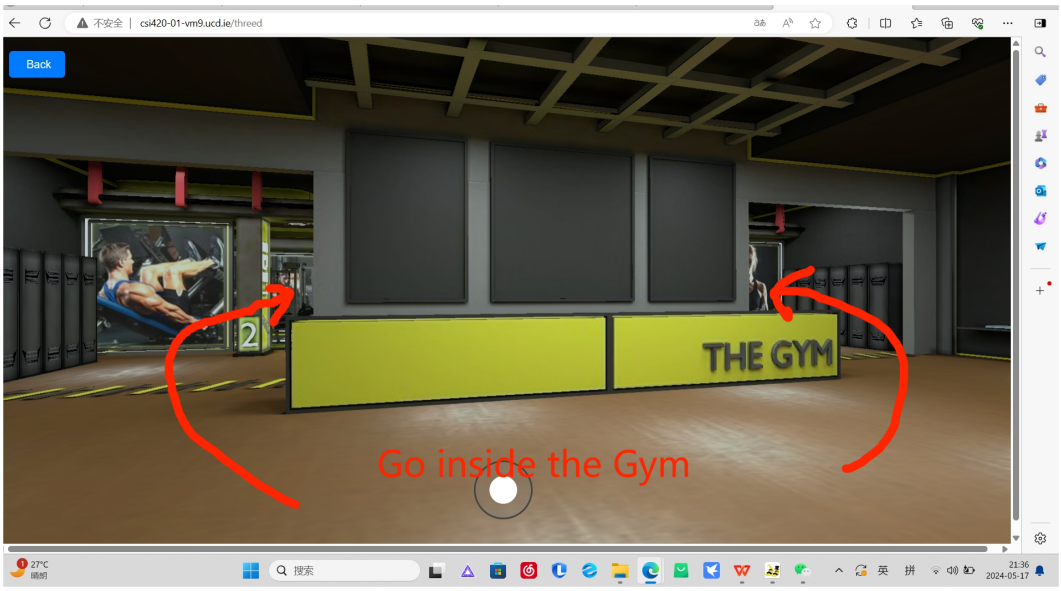

Figure 3.5: Gym

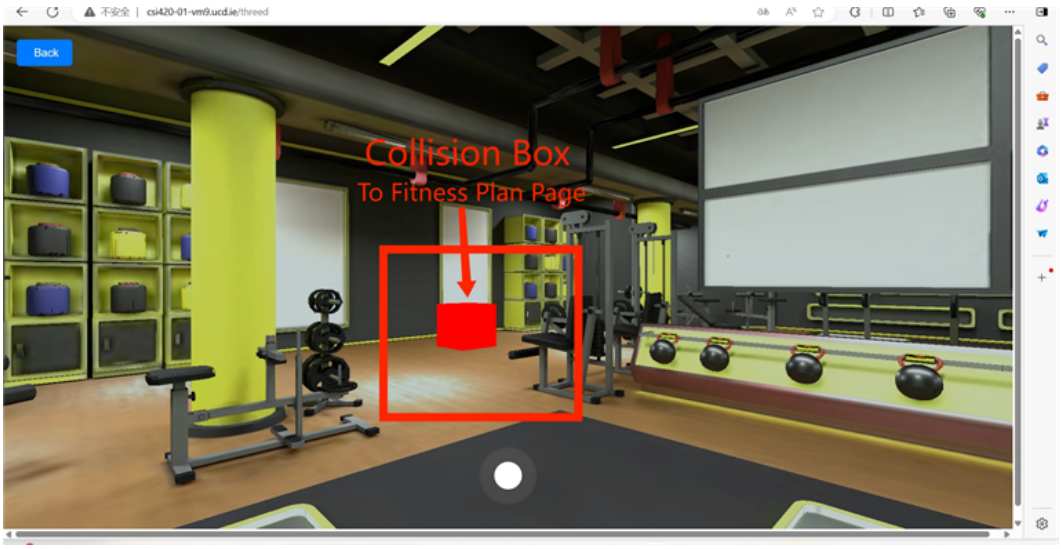

Figure 3.6: Gym

C. Access To Diet Plan Page

After entering other rooms from the lobby, user can drag the circle in the middle of the screen to control themselves to collide to the red cube in the scene to jump into corresponding module. Access to diet plan page from the kitchen.

Figure 3.7: Kitchen

D. Access To Personal Page

After entering other rooms from the lobby, user can drag the circle in the middle of the screen to control themselves to collide to the red cube in the scene to jump into corresponding module. Access to personal page from the office.

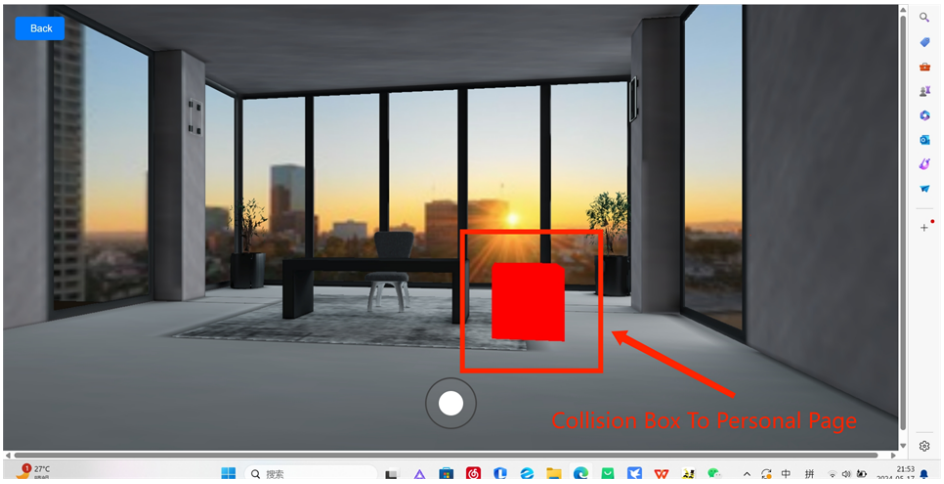

Figure 3.8: Personal Room

3.2.3 2D Mode Page

In addition to 3D perspective, user can also choose to experience the website in 2D mode. After clicking on the 2D mode link from homepage, user then clicks on the black start link in the middle of the page to enter website introduction page.

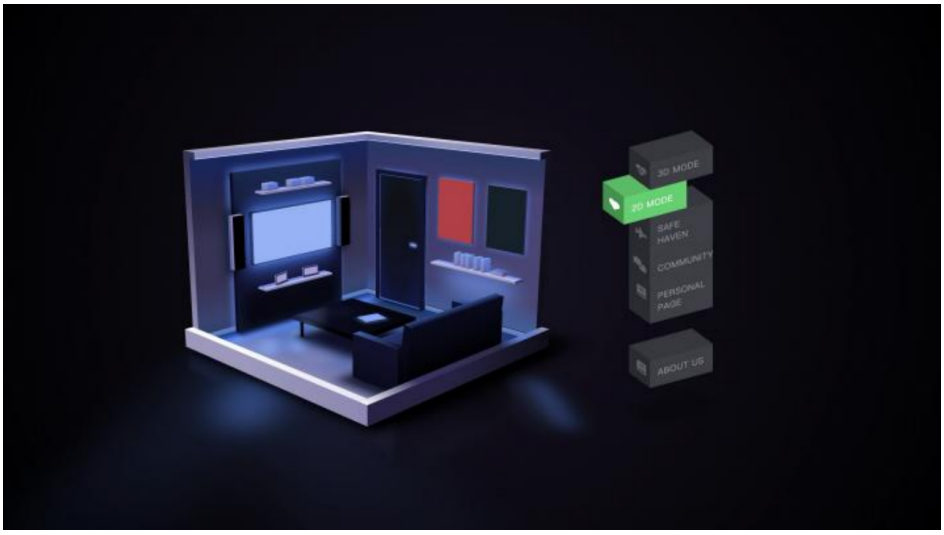

Figure 3.9: Select 2D Mode

Figure 3.10: 2D Mode Start Page

A. Project Introduction

We give an overview of the architecture, important technical points and uses of the website from five aspects: about, our services, professionalism, get started and side functions.

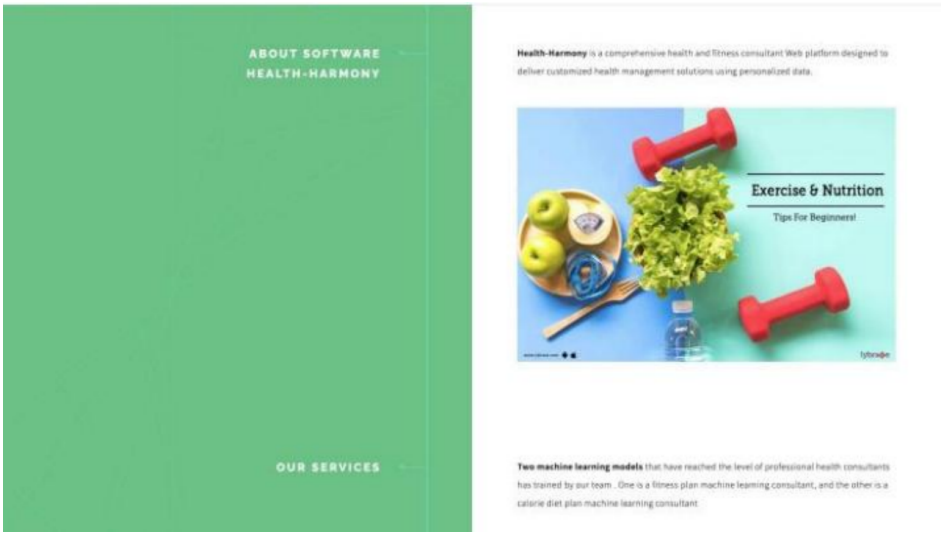

Figure 3.11: About Page

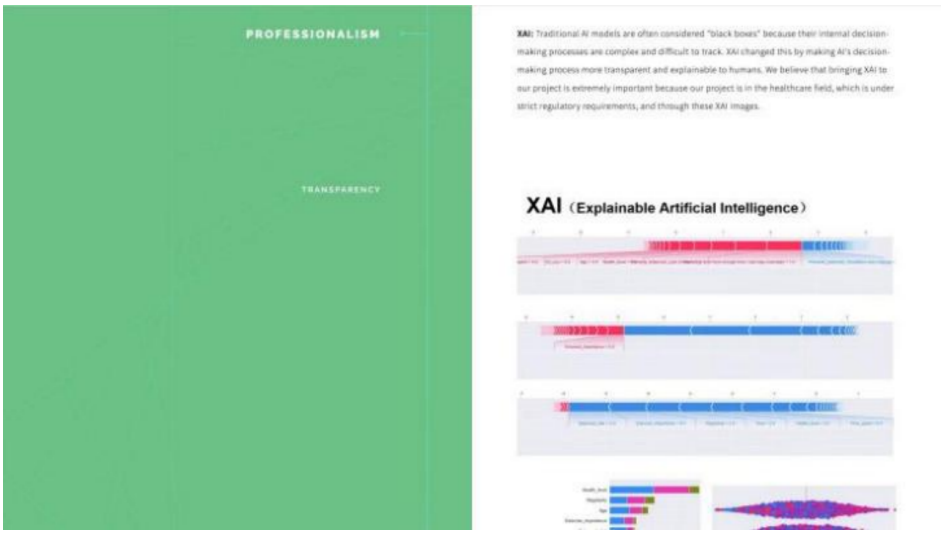

Figure 3.12: Professionalism Page

B. Enter Other Modules

In the get started section, user can click green button on the right side of the page to enter machine learning prediction section.

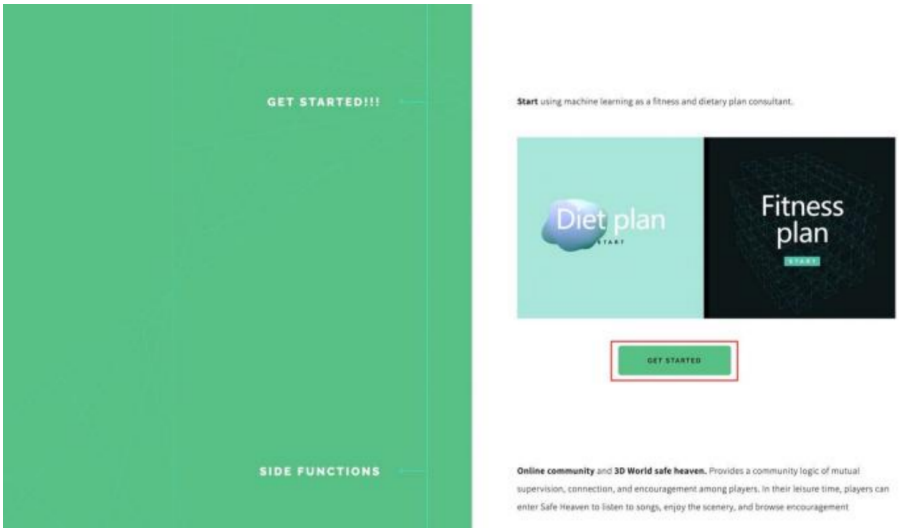

Figure 3.13: Get Started Page

3.2.4 Machine Learning Prediction Module

In 2D mode, user enter the diet and fitness survey questionnaire by clicking on the corresponding link. In 3D mode, user enter the same interface through the red cubes in the gym and kitchen.

A. Diet Plan

The user clicks on the start link in diet plan on the left side of the interface to enter the diet questionnaire survey page.

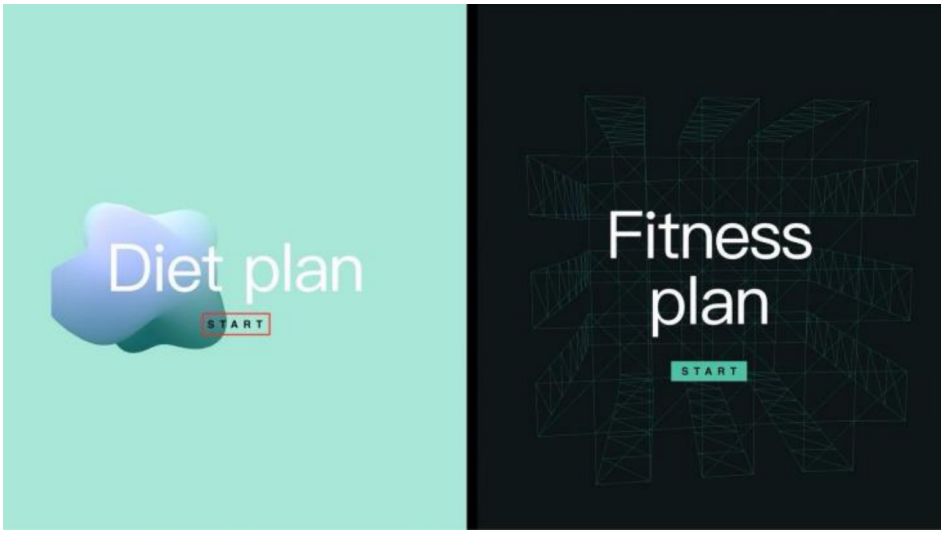

Figure 3.14: Questionnaire Start Page

A1. Questionnaire

User needs to answer 6 related questions, including multiple-choice and fill in the blank questions. After answering a question, users can click on the arrow at the edge of the page to switch between questions. After answering all the questions, the user clicks submit button to submit data to

machine learning model for processing. Subsequently, the page will redirect to the shop module, where users can view their daily calorie intake and select food. If the user has previously filled out a survey questionnaire, they will be redirected directly to the shop module.

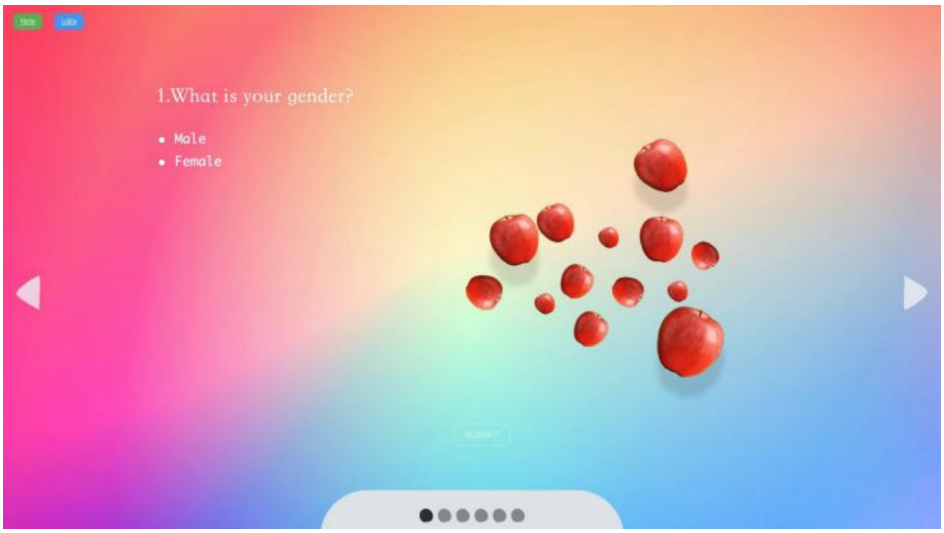

Figure 3.15: Diet Plan Questionnaire

A2. Shop

After entering the shop, user will first see the daily calorie intake customized by ML dietary health consultant based on their personal situation. User can scroll down the page to browse food list and their corresponding calories (every 100 grams).

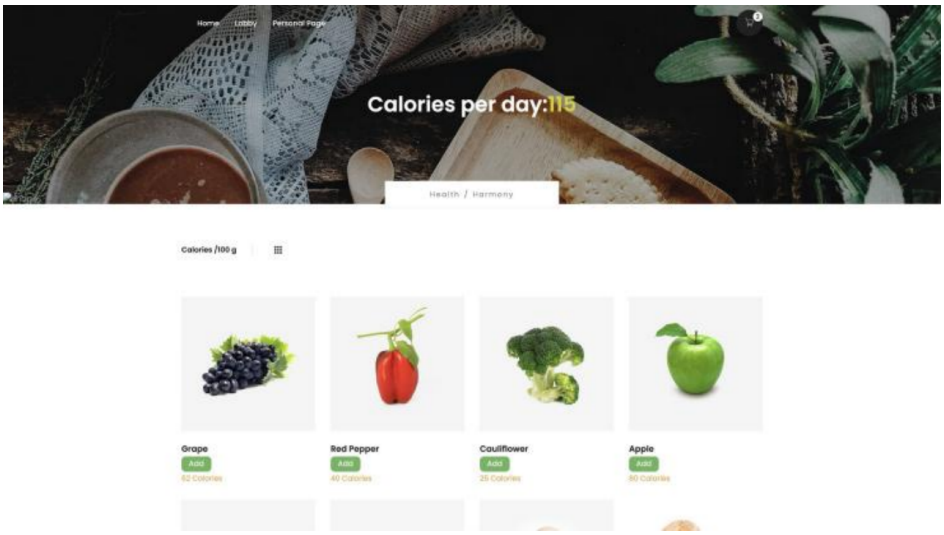

Figure 3.16: Shop Page

By clicking the add button, we can add our favorite foods to shopping cart. Then click on the icon in the upper right corner to enter the shopping cart, where information about the previously selected food is stored. Users can see the total calorie count of these foods below and click the submit button. The system will prompt whether the diet for the day is reasonable.

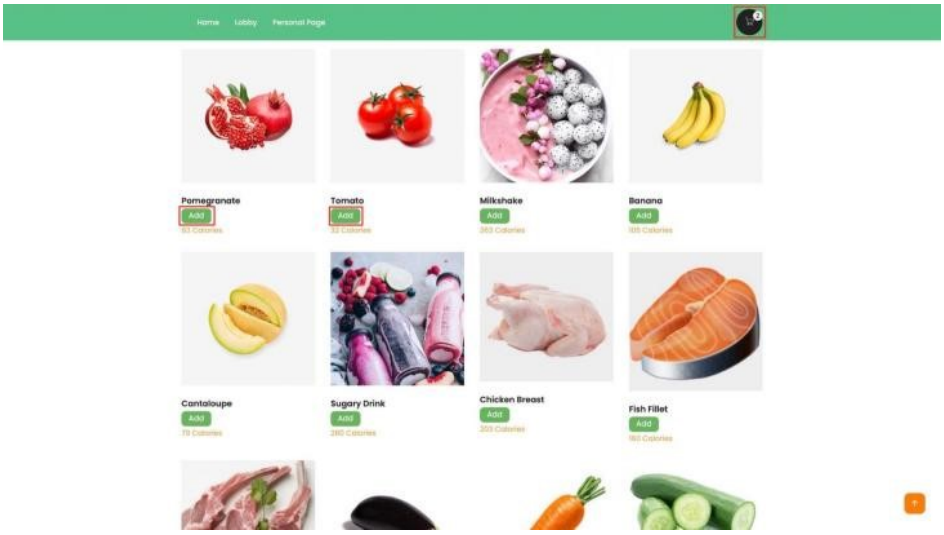

Figure 3.17: Food List

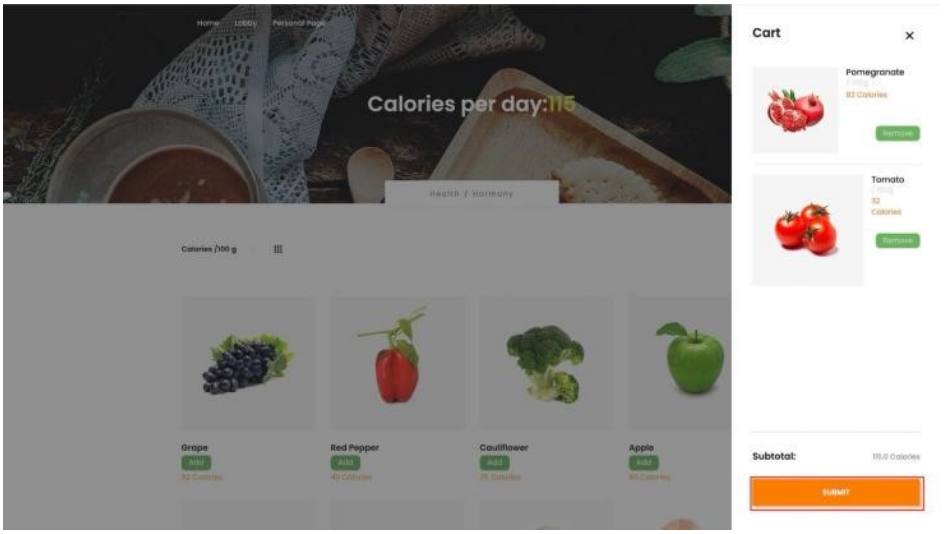

Figure 3.18: Shopping Cart

B. Fitness Plan

The user clicks on the start link in fitness plan on the right side of the interface to enter the fitness questionnaire survey page.

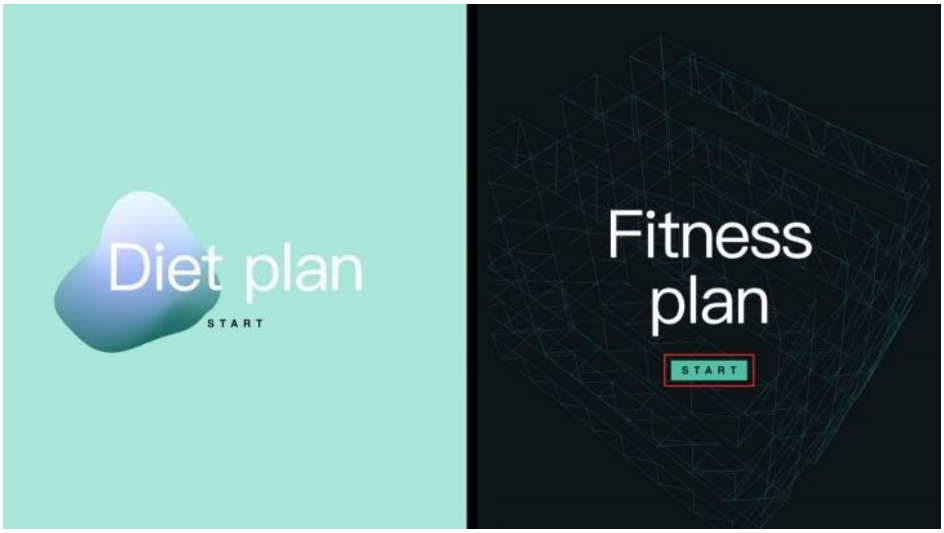

Figure 3.19: Questionnaire Start Page

B1. Questionnaire

User needs to answer 11 related questions, all of which are multiple-choice questions. After answering a question, user can click on the arrow at the edge of the page to switch between questions. After answering all questions, the user clicks the submit button to submit data to machine learning model for processing. Subsequently, the page will redirect to the ML Fitness Advisor module, providing user with fitness related information and targeted suggestions. If the user has previously filled out a survey questionnaire, they will be redirected directly to the ML Fitness Advisor module. Special attention: It takes 3-5 seconds to load the machine learning model results here. Please be patient and wait

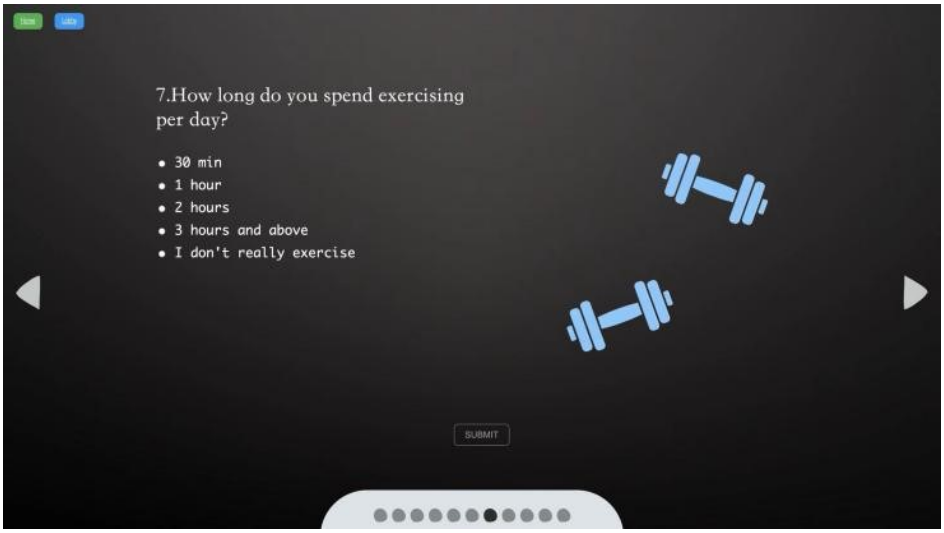

Figure 3.20: Fitness Plan Questionnaire

B2. ML Fitness Advisor

User can see information such as health levels, fitness frequency, and intelligent suggestions generated by machine learning technology on this page. By clicking on the navigation bar in the upper right corner of the page, user can also browse the structure of various tissues in the human body and understand how each exercise affects specific muscle groups.

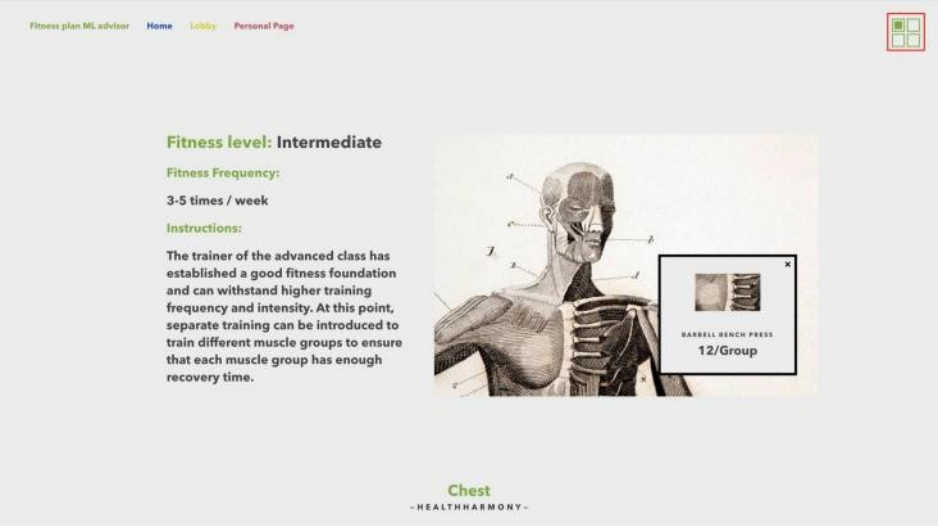

Figure 3.21: Fitness Advisor Page

Figure 3.22: Fitness Advisor Detail Page

3.2.5 Safe Haven

User can access this module by clicking on the Safe Haven link from homepage. The 3D scene here is designed for users to enjoy the scenery, listen to music, and browse encouragement during their leisure time.

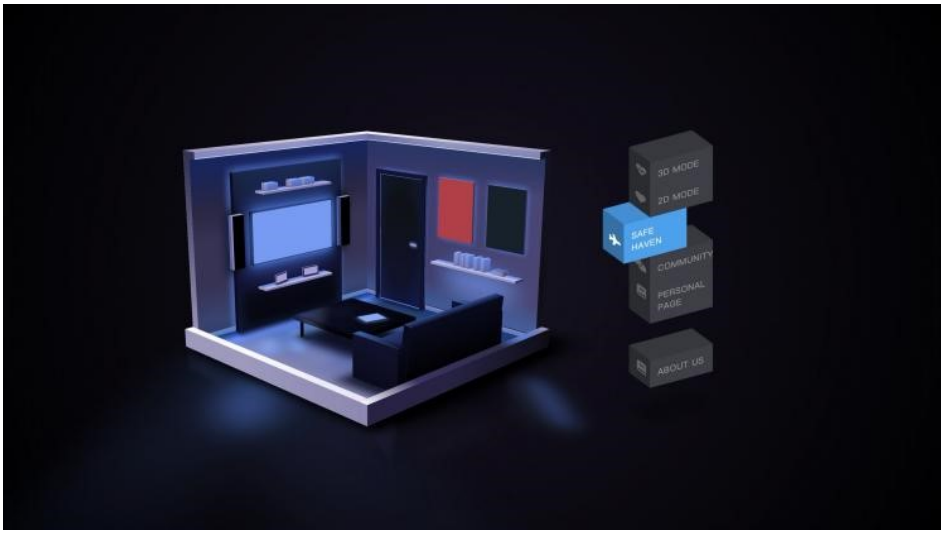

Figure 3.23: Select Safe Haven

Special Note For 3D World:

Due to performance optimization considerations, we have not set map collision boundaries. Therefore, players are advised not to cross walls to ensure a smooth gaming experience Due to network fluctuations, the waiting time has ended and the 3D scene has not been loaded. Please be patient and wait.

3D Safe Room + Postcard + Music Player

User can control their perspective to move around the room through the directional keys on the keyboard. There is a music player in the bottom right corner of the page, and user can select the music to play from the drop-down list. After selecting the music, we can start playing by clicking the play button and pause playing by clicking the stop button.

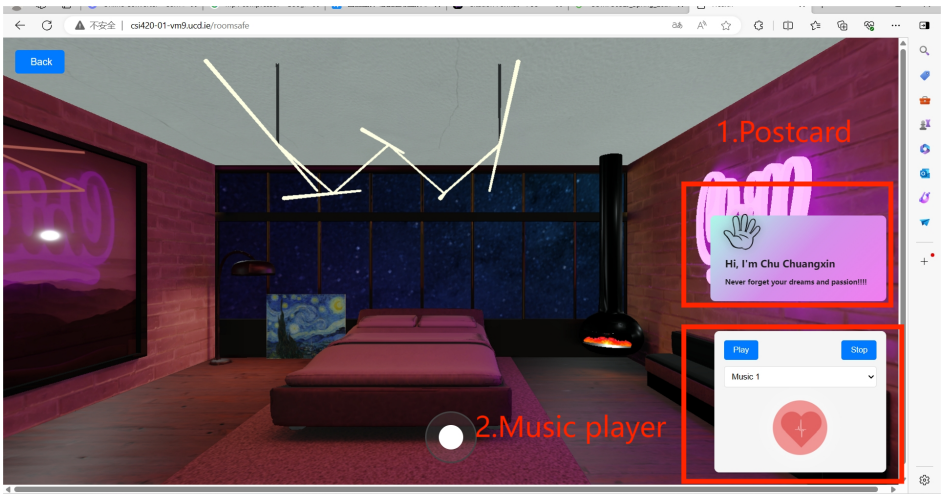

Figure 3.24: Safe Haven

3.2.6 Community Page

Users can see a list of posts on the community home page, which is presented as a sliding 3D grid. Above the list is a navigation bar that allows users to click Newest Post to sort posts by time, and click Top Post to sort posts by popularity (number of comments). Other links can be used to select categories and view posts under specific topics. Users can click the arrow at the bottom of the navigation bar to return to the homepage.

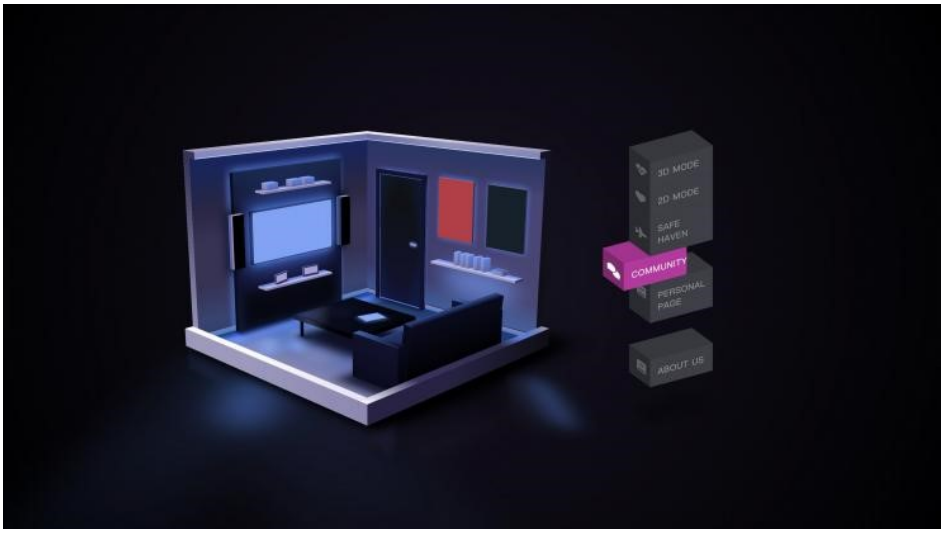

Figure 3.25: Community Entrance

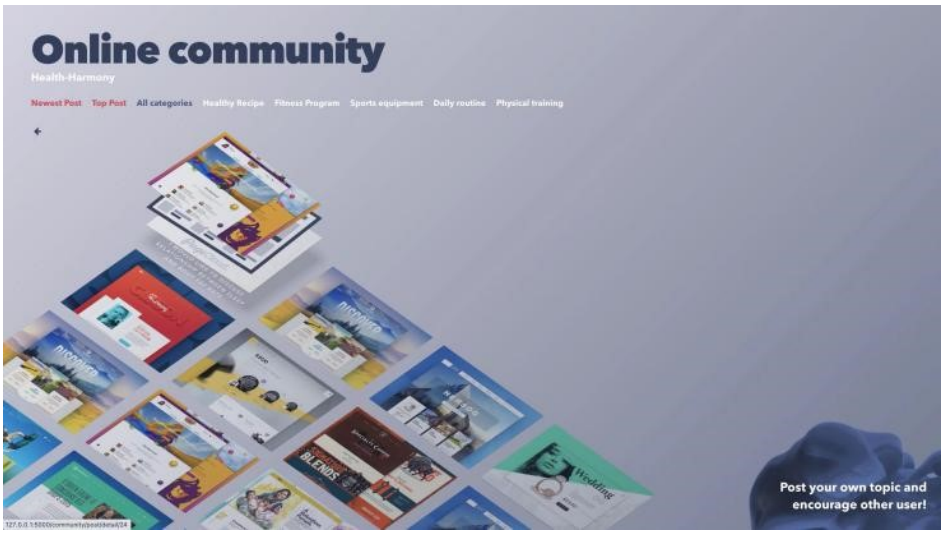

Figure 3.26: Community

A. Post Details

Users clicking on one of the grids will take them to the post details page. Here you can see the title, content, and comments of the post. Users can also comment below. By writing text in the input field and clicking the Send button, they can add their comments to the comment list. After viewing the post, the user clicks the Back button to return to the community home page.

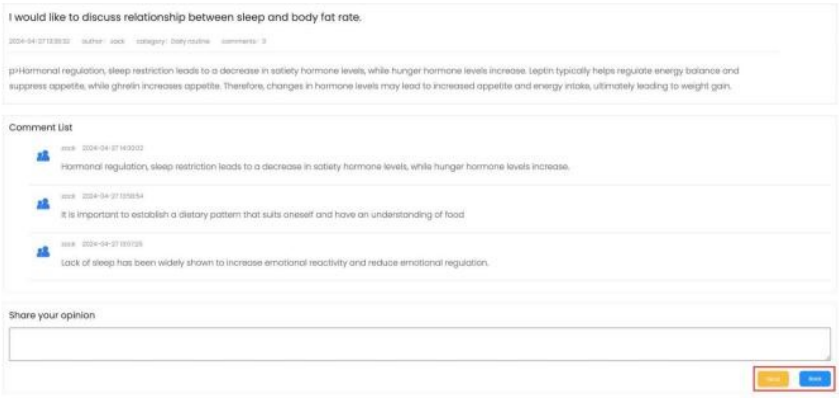

Figure 3.27: Post Details

B. New Post

Users can post new posts created by themselves by clicking on the link in the lower right corner of the community home page. After entering the title and content of the post and selecting the subject of the post, click the Send button to successfully send the post and enter the details page of the new post.

Figure 3.28: New Post

3.2.7 Personal Page

Users can see personal and program-related information on this page. Users can also return to the 3D Mode Page, ML Fitness Advisor page or Shop page via the navigation bar at the top of this page.

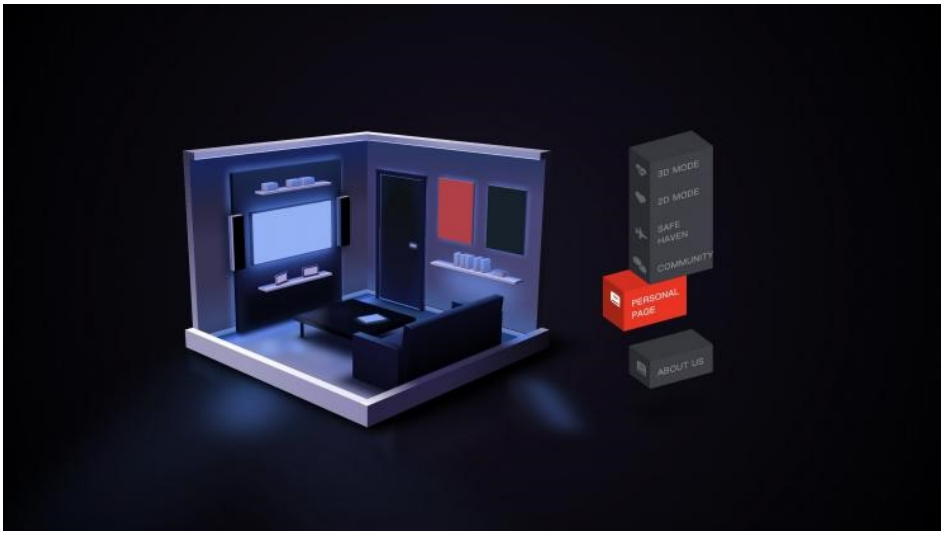

Figure 3.29: Personal Page

A. Personal Information

Here you can see the user name and the user's self-introduction.

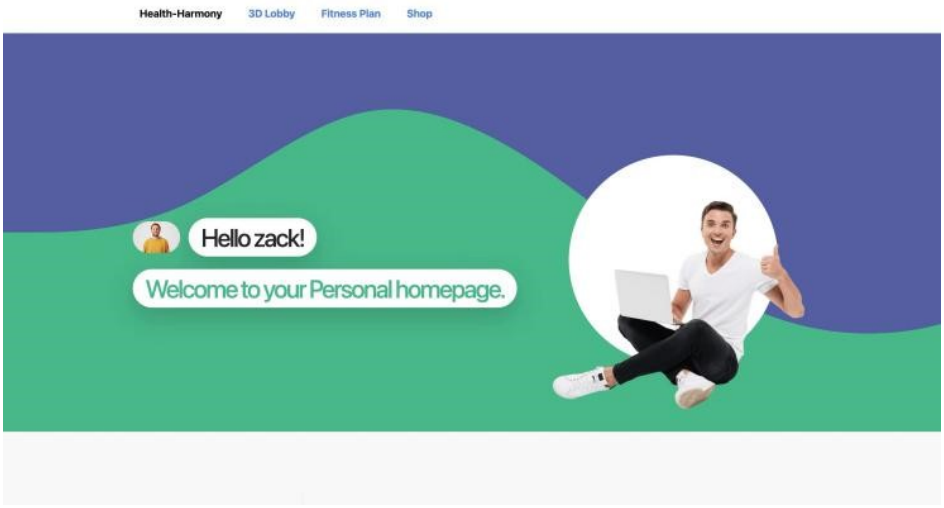

Figure 3.30: Personal Information

B. Fitness and Diet Plan Information

Users swipe down to see information about their own fitness level, recommended daily calorie intake, fitness plan and diet plan.

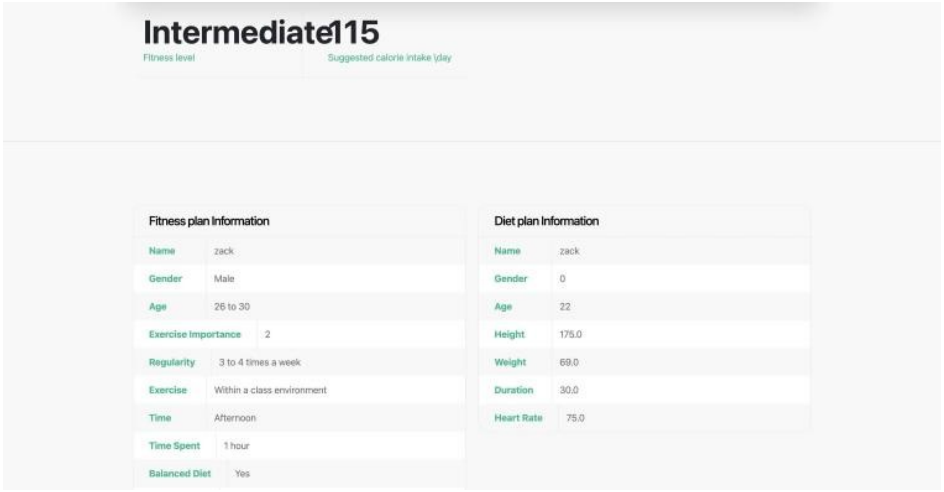

Figure 3.31: Fitness and Diet Plan Information

C. Update Plan

At the bottom of the page, users can update their fitness and diet plans by clicking on a link to fill out the questionnaire again. Users can also access the community module from the far right link.

Figure 3.32: Update Plan

3.2.8 About Us Page

This module is about the picture and video introduction of our project.

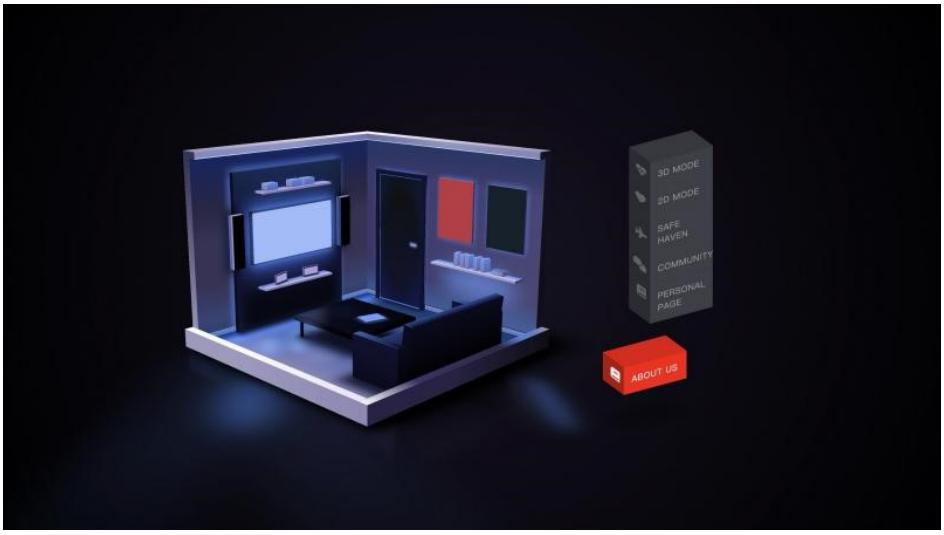

Figure 3.33: About Us Page

Special Note For 3D World:

Due to performance optimization considerations, we have not set map collision boundaries. Therefore, players are advised not to cross walls to ensure a smooth gaming experience Due to network fluctuations, the waiting time has ended and the 3D scene has not been loaded. Please be patient and wait.

A. 3D "About Us" Exhibition Hall

This page uses multiple models and light and shadow effects to build a 3D scene. Users can freely explore the room through the arrow keys on the keyboard, and view the screenshots of various parts of the website through the photos hanging on the wall, which makes it more convenient and intuitive to understand our projects.

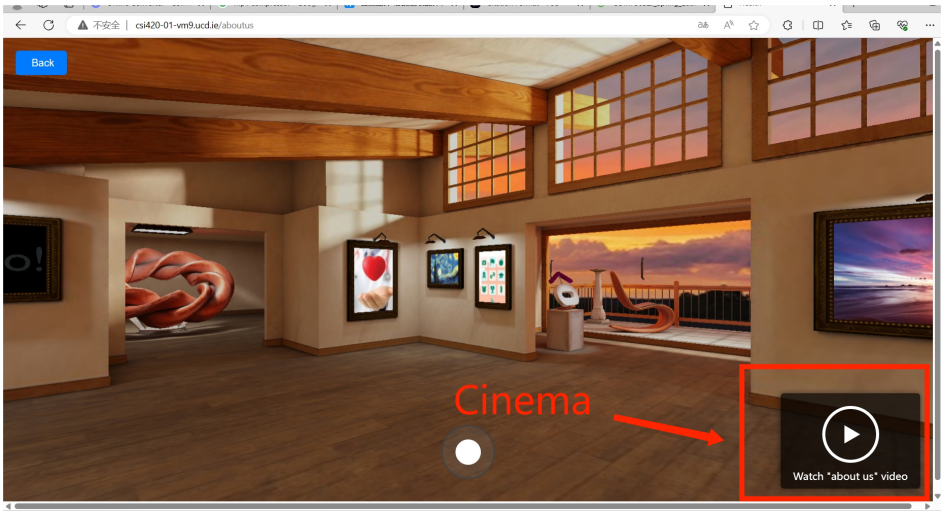

Figure 3.34: 3D "About Us" Exhibition Hall

B. Watch Introduction Video

Users can access the 3D cinema by clicking on the video in the bottom right corner of the page. Through the green link below, we can view the seating diagram of the theater and choose different seats to watch the video. Viewers can tap a button marked 180 degrees in the middle of the screen to unlock the camera, move the lens to the appropriate Angle and then tap the button to lock it. Users will then click the green play button on the theater screen to watch a video about the project. After the user finishes watching, click the Quit cinema button on the right side to jump to the home page.

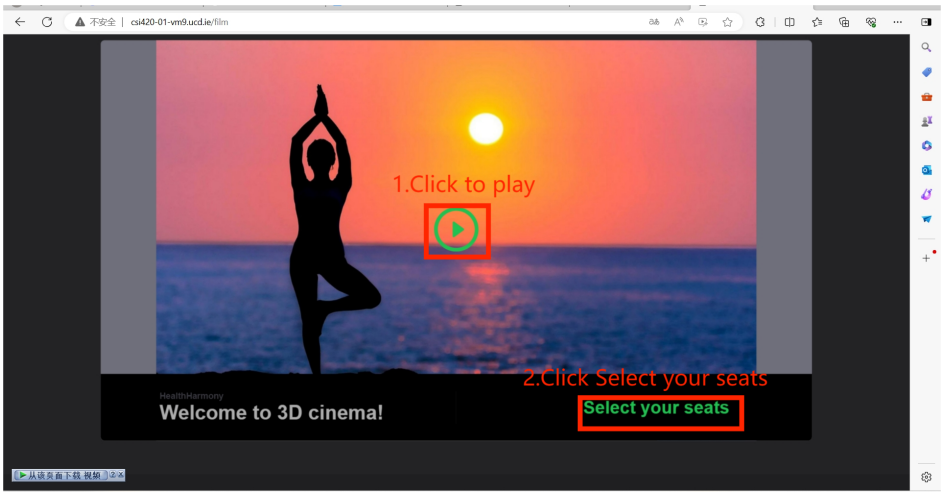

Figure 3.35: Watch Introduction Video

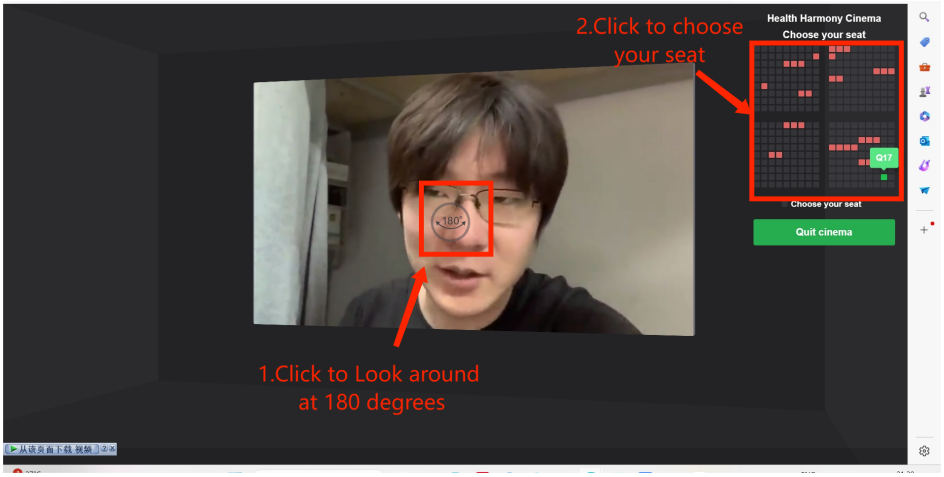

Figure 3.36: Look around and Choose Seat

Chapter 4: System Documentation

4.1 System Overview

4.1.1 System Architecture

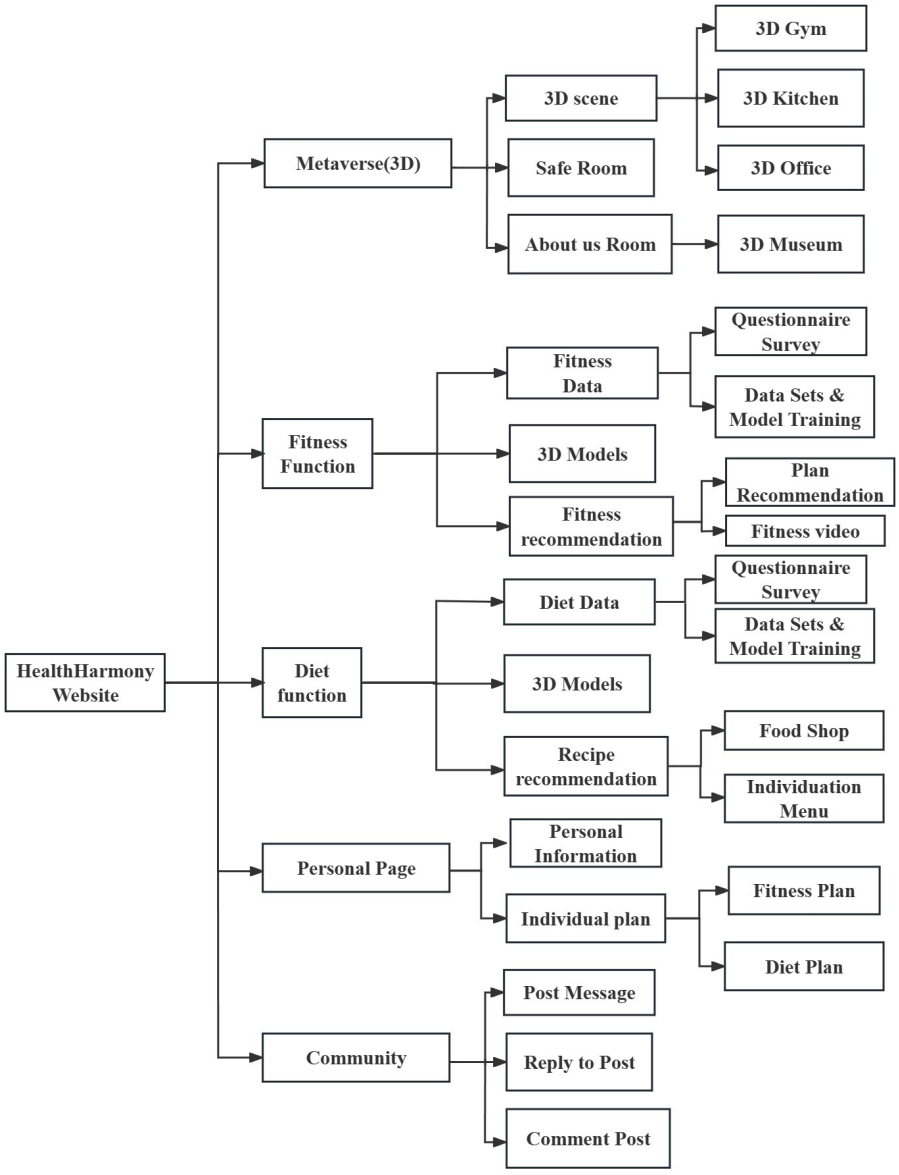

Figure 4.1: System Architecture

The HealthHarmony website presents a comprehensive architecture that integrates various cutting-edge technologies to provide personalised fitness and dietary recommendations, interactive 3D environments, and a community platform for users.

At the core of the website are the Fitness Function and the Diet Function. The Fitness Function collects Fitness Data through a Questionnaire Survey, which is then utilised for Data Sets and Model Training to generate personalised Fitness Recommendations tailored to each user. These recommendations are presented through immersive 3D Models and engaging Fitness Videos.

The Diet Function gathers Diet Data and leverages it for Data Sets and Model Training to provide personalised Recipe Recommendations. To enhance the user experience, these recommendations can be accompanied by interactive 3D Models of food items or realistic kitchen environments. Users can also access a Food Shop and an Individuation Menu, which could offer personalised meal planning, shopping recommendations, or other tailored dietary solutions.

A key feature is the Personal Page, where users can access their Individual Plan, a consolidated view of their Fitness Plan and Diet Plan recommendations based on their Personal Information. This centralised hub ensures a seamless and comprehensive experience.

The Community component enables users to engage with one another by posting messages, replying to posts, and commenting on shared content. This fostering of a supportive environment promotes knowledge sharing and encourages users on their health and wellness journeys.

Website integrates the Metaverse (3D) component, which offers immersive 3D environments. Users can experience a virtual 3D Gym for fitness activities, a 3D Kitchen for interactive cooking demonstrations or meal planning, and a 3D Office for work-related activities or wellness programs. These environments leverage technologies like Virtual Reality (VR) and 3D graphics to create engaging and realistic simulations.

The About Us Room provides comprehensive information about the platform and its features, while the 3D Museum serves as an interactive educational space, utilising 3D modelling and visualisation to showcase health and wellness-related exhibits and galleries.

Overall, the HealthHarmony website demonstrates the integration of cutting-edge technologies, including machine learning, 3D modelling, virtual reality, and social networking, to deliver a comprehensive and personalised solution for users seeking tailored fitness routines, dietary recommendations, and an engaging community experience.

4.1.2 Core Functions

A. User Account

Registration and login

All users who utilise the HealthHarmony website for fitness and dieting must have a unique account. To gain access, individuals are required to go through a registration process, providing essential details to create their personalised profiles. Once registered, users can log in securely, ensuring a personalised experience tailored to each individual's specific goals and needs. With a dedicated account, users can track their progress, log their meals, and monitor their exercise routines effortlessly.

Furthermore, the account system allows for secure data storage, protecting users' personal information and preserving their privacy. By maintaining an account, users gain access to a wealth of resources, including customised meal plans, workout regimens, and educational materials. This streamlined approach not only fosters accountability but also empowers individuals to take control of their health journey, ultimately promoting long-term success in achieving their desired fitness and wellness objectives.

Hash Function

In the realm of health and wellness, maintaining user trust and safeguarding personal data is of paramount importance. At our fitness website, we understand the sensitivity of the information our users entrust us with, including their dietary habits, workout routines, and potentially even health-related metrics. As such, we have implemented robust security measures, specifically secure hashing encryption, to ensure the utmost protection of our users' privacy.

Hashing is a cryptography technique that transforms sensitive data, such as passwords, into a fixed-length string of characters known as a hash value or digest. This process is one-way, meaning that it is computationally infeasible to reconstruct the original data from the hash value alone. Even the slightest alteration in the input data results in a completely different hash output, making it highly resistant to tampering or unauthorised modifications.

The secure hashing algorithm we employ is renowned for its strength and resilience against various crypt analytic attacks. It designed to be collision-resistant, meaning it is incredibly difficult to find two different input messages that produce the same hash value.

When a user creates an account on our fitness website or sets up a new password, their sensitive information is never stored in plaintext. Instead, we pass the data through the hashing algorithm, and only the resulting hash value is securely stored in our databases. This approach ensures that even in the unlikely event of a data breach, the original sensitive information remains protected and inaccessible to potential attackers.

During authentication processes, such as user logins, the entered password is hashed using the same algorithm, and the resulting hash value is compared against the stored hash. If the two hashes match, the user is granted access to their personalised fitness and dietary modules. This method eliminates the need to store or transmit plaintext passwords, significantly reducing the risk of unauthorised access or data exposure.

By implementing secure hashing encryption for user data, particularly passwords and potentially sensitive health information, we demonstrate our unwavering commitment to protecting the privacy and security of our users. This proactive measure not only enhances the overall trust and confidence in our fitness platform but also aligns with industry best practices and regulatory requirements for data protection in the health and wellness sector.

B. Fitness Function

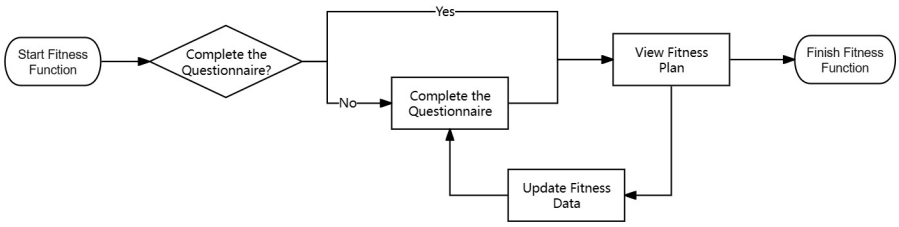

Figure 4.2: Fitness Flowchart

The fitness function on HealthHarmony follows a data-driven approach to assist users in achieving their fitness goals. The process begins when a user start the fitness function. At this point, the system checks the database to determine if the user has previously completed the questionnaire.

If the database search reveals the user has already completed the questionnaire, they can directly

view the fitness plan tailored to their specific needs and preferences based on the submitted questionnaire data. However, if the user has not yet completed the questionnaire, they need to provide essential information through a comprehensive questionnaire about their current fitness level, goals, dietary preferences, and any physical limitations or medical conditions.

Once the questionnaire is completed, the user's responses are stored in the database, allowing the system to generate a personalised fitness plan aligning with their unique circumstances. The user can then choose to update their fitness data, inputting progress, tracking workouts, and logging dietary intake. This data, combined with the initial questionnaire information, enables the system to continuously refine and adjust the fitness plan, ensuring its relevance and effectiveness as the user progresses on their fitness journey.

After updating their fitness data, users can view the comprehensive fitness plan, which includes tailored exercise routines, meal plans, and other relevant information to support their goals. The plan adapts dynamically based on the user's input and progress, providing a customised experience that evolves alongside their journey.

When the user achieves their desired fitness objectives or reaches a milestone, they can finish the fitness function by setting new goals, transitioning to a maintenance plan, or simply celebrating their accomplishments. Throughout the process, the website leverages the user's data stored in the database to provide personalised guidance and recommendations, empowering users to take control of their fitness journeys and achieve their desired outcomes effectively.

C. Diet Function

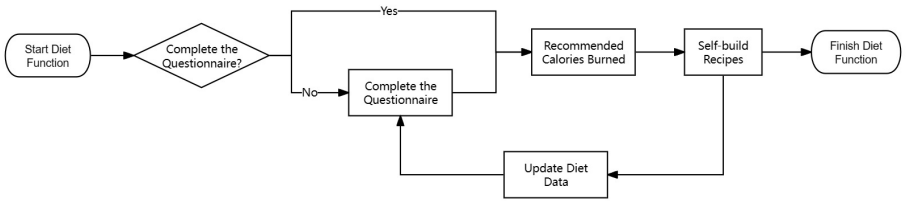

Figure 4.3: Diet Flowchart

HealthHarmony offers a Diet Function that follows a data-driven approach to assist users in achieving their dietary goals. The process begins when a user start diet function. The system then checks the database to determine if the user has previously completed the diet questionnaire.

If the database search reveals that the user has already completed the questionnaire, they can proceed directly to the Recommended Calories Burned section. This recommendation is tailored to their specific dietary needs and preferences based on the previously submitted questionnaire data.

However, if the user has not yet completed the diet questionnaire, they will be prompted to Complete the Questionnaire. This comprehensive questionnaire gathers essential information about the user's current weight, desired weight goals, dietary restrictions, food preferences, and any medical conditions that may impact their diet plan.

Once the questionnaire is completed, the user's responses are stored in the database, allowing the system to generate a personalised diet plan aligned with their unique circumstances. Users then have the option to Update Diet Data.

Users can self-build recipes according to the recommend calories burned, which provides tailored meal plans and recipes that complement their dietary goals and preferences. These recipes are

dynamic and adapt based on the user's input and progress, providing a customised experience that evolves alongside their journey.

Finally, when the user achieves their desired dietary objectives or reaches a milestone, they can Finish Diet Function. This step may involve setting new goals, transitioning to a maintenance plan, or simply celebrating their accomplishments.

Throughout the entire process, the Diet Function relies on the user's data stored in the database to provide personalised guidance and recommendations. By leveraging this data-driven approach, the website ensures that the diet plans and recommendations are tailored to each individual's unique needs and preferences, empowering users to take control of their dietary journeys and achieve their desired outcomes effectively.

D. Personal Page

HealthHarmony offers a Personal Page that provides users with a comprehensive overview of their individual information, fitness, and dietary data. This centralised hub serves as a one-stop destination for users to access and manage their personal details, progress, and preferences.

Upon accessing the Personal Page, users are presented with their profile information, including their name, contact details, and any relevant personal data they have provided.

Additionally, the Personal Page displays a detailed summary of the user's fitness and dietary data. This includes their current fitness levels, workout routines, meal plans, and any relevant metrics or milestones they have achieved. Users can easily review their progress, track their achievements, and identify areas for improvement or adjustment.

One of the key usages of the Personal Page is its ability to facilitate data updates. Users can conveniently log their workouts, record their dietary intake, and input any other relevant information directly within this feature. This streamlined process ensures that the system has access to the latest data, enabling it to provide more accurate and personalised recommendations for the user's fitness and dietary plans.

Overall, the Personal Page serves as a comprehensive hub for users to manage their personal information, monitor their progress, and maintain their fitness and dietary data.

E. Community

HealthHarmony offers a vibrant community tailored specifically for fitness and diet enthusiasts, serving as a dynamic platform to connect, engage, and support one another on their respective journeys. This interactive space fosters a sense of community, enabling users to share their experiences, seek advice, and draw inspiration from like-minded individuals pursuing similar health and wellness goals.

Within the community, users can create posts to share their fitness milestones, dietary challenges, and insights related to their unique journeys. These posts can include text, images, or even videos, allowing for a rich and multimedia-driven experience. Other users can interact with these posts by leaving comments, offering encouragement, providing helpful tips on exercises or meal plans, or sharing their own relevant experiences.

Additionally, users can engage in discussions on various fitness and diet-related topics, such as the latest workout trends, healthy recipes, or strategies for overcoming plateaus. Experienced community members can share their valuable insights and tips, while newcomers can seek guidance and learn from the collective wisdom of the group.

Overall, the HealthHarmony community enriches the website's offerings by fostering a supportive and engaging environment specifically designed for fitness and diet enthusiasts. It enables users to connect, learn, and grow together on their paths to better health and wellness, leveraging the power of shared experiences and collective knowledge.

4.2 3D Virtual Open World

4.2.1 Function Overview

A. 3D Open World Lobby

Users can freely explore in 3D scenes and experience an immersive experience. Users can interact with the 3D collision box, enter the healthy eating machine learning model interface in the 3D kitchen, enter the fitness plan machine learning model in the 3D gym, and enter the personal homepage interface in the 3D office. This greatly enhances the user's sense of participation.Figure 4.4 shows the model construction of 3D Open World Lobby.

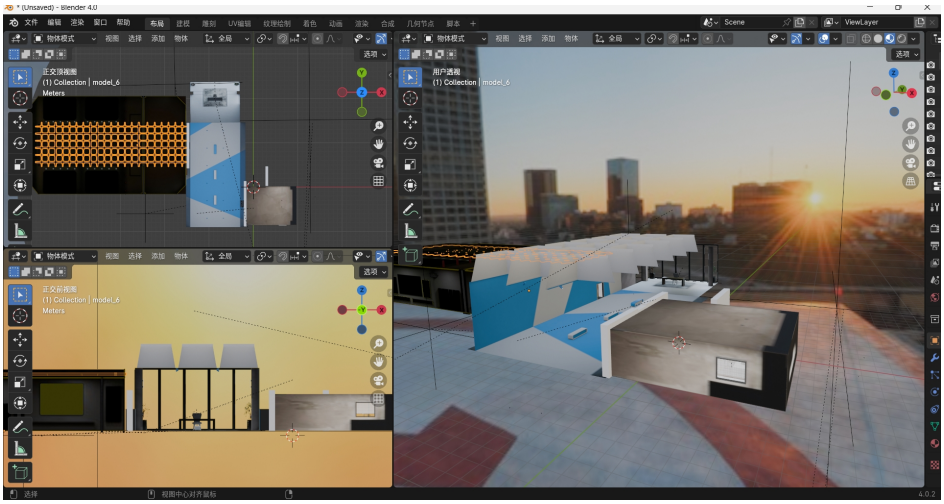

Figure 4.4: 3D Open World Lobby

B. 3D Personal Safe Room

Users can enter the Safe Haven 3D Metaverse Lounge to enjoy personalized entertainment and relaxation experiences. This personalized rest space helps alleviate user stress and promote physical and mental health. The 3D personal safe house is connected to the 2D community forum, providing players with information encouragement for strangers. Players can also listen to music to relax and enjoy the starry sky and sunset. Futhermore, 3D animations are imported so that users can enjoy the movement of the stars in the universe. Figure 4.5 shows the model construction of 3D Personal Safe Room.

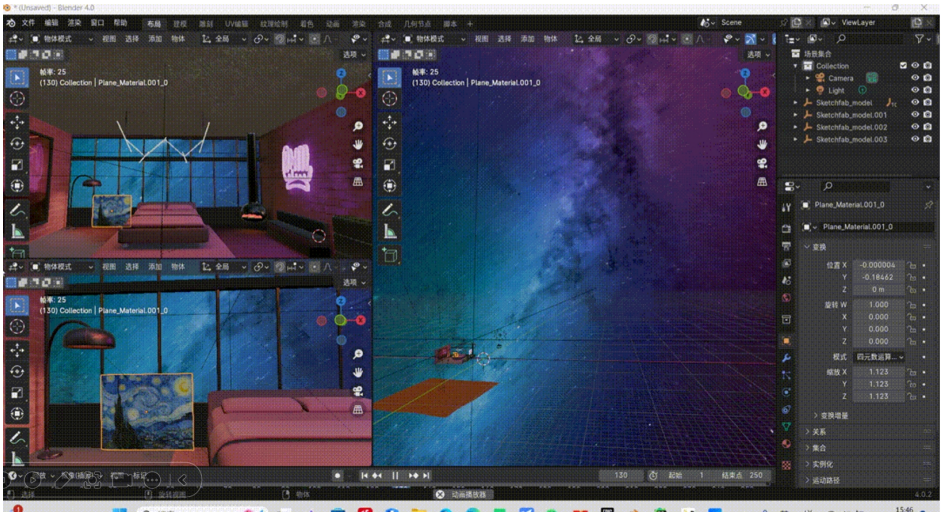

Figure 4.5: 3D Personal Safe Room

C. 3D Exhibition Hall

3D Exhibition Hall: Users can visit About Us 3D Exhibition Hall to learn about the company's profile and team information. This immersive display format enhances users' awareness and trust in the company. Figure 4.6 shows the model construction of 3D Exhibition Hall.

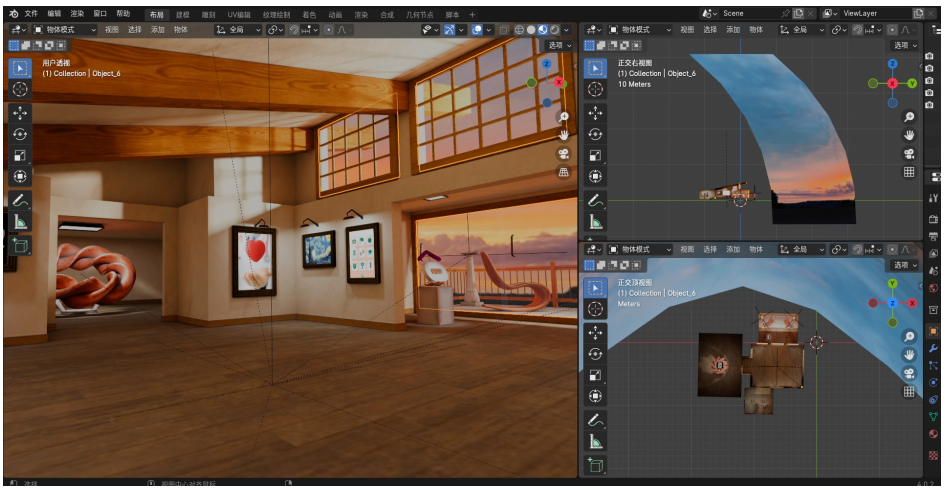

Figure 4.6: 3D Exhibition Hall

D. 3D "About Us" Cinema

Implement 3D cinema through Threejs to introduce our program healthharmony, breaking the boundaries between 2D and 3D. The front screen plays a video. Users can choose any seat on the right column and click on it to directly switch the player camera coordinate position to the seat coordinate position. The specific code implementation logic can be found in the appendix. 4.7 shows the 3D "about us" cinema.

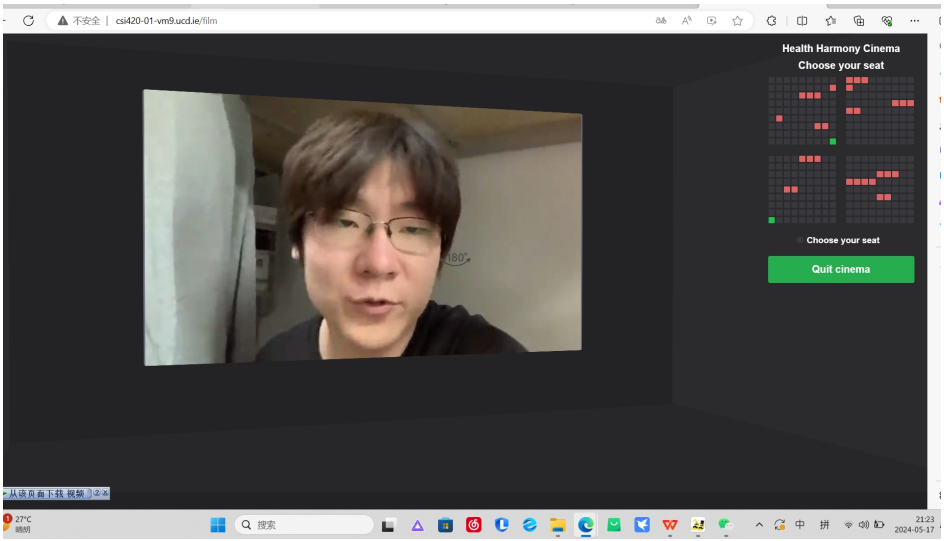

Figure 4.7: 3D Exhibition Hall

4.2.2 Technology Stack Application

Three.js + WebGL + Blender + Unreal Engine 5 + Photoshop

A. Three.js

Used for WebGL rendering and GLTF model import. Three.js is a JavaScript 3D library based on WebGL, which provides us with a series of rendering and interaction APIs, allowing us to easily import 3D assets made by Blender and Unreal Engine 5 into the browser, achieving high-performance 3D rendering.

B. WebGL

A low-level graphics API that provides hardware accelerated 3D rendering capabilities. WebGL, as an API based on OpenGL ES, provides hardware accelerated support for our 3D rendering, significantly improving performance.

C. Blender

Used for 3D scene modeling and animation production. Blender, as a powerful 3D modeling software, provides us with rich modeling tools and data export formats, such as GLTF, which provides a solid foundation for subsequent rendering and animation.Figure 4.8 shows the implementation of Blender.

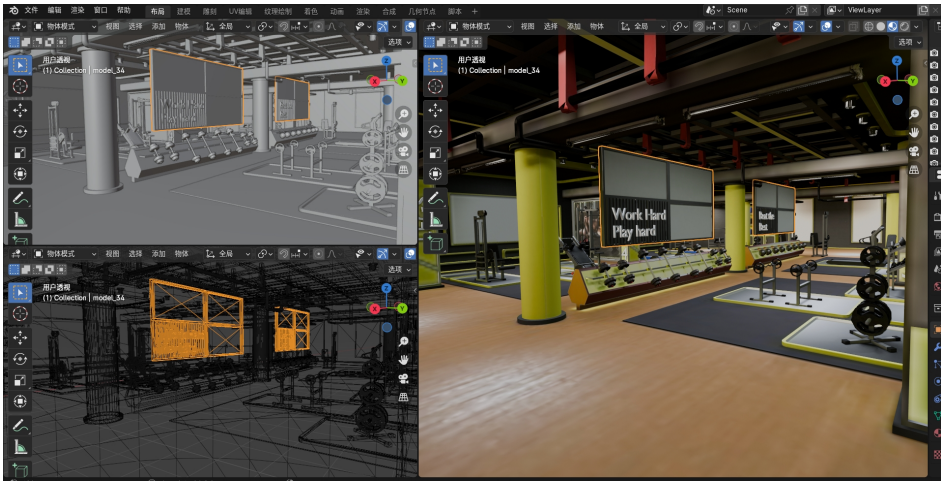

Figure 4.8: Blender

D. Unreal Engine 5

Used for realistic rendering, including ray tracing and baking techniques. Unreal Engine 5 is a leading game engine with powerful rendering features such as ray tracing and baking, adding realistic visual effects to our 3D scenes and greatly enhancing user immersion.Figure 4.9 shows The process of building scenes in Unreal Engine 5.

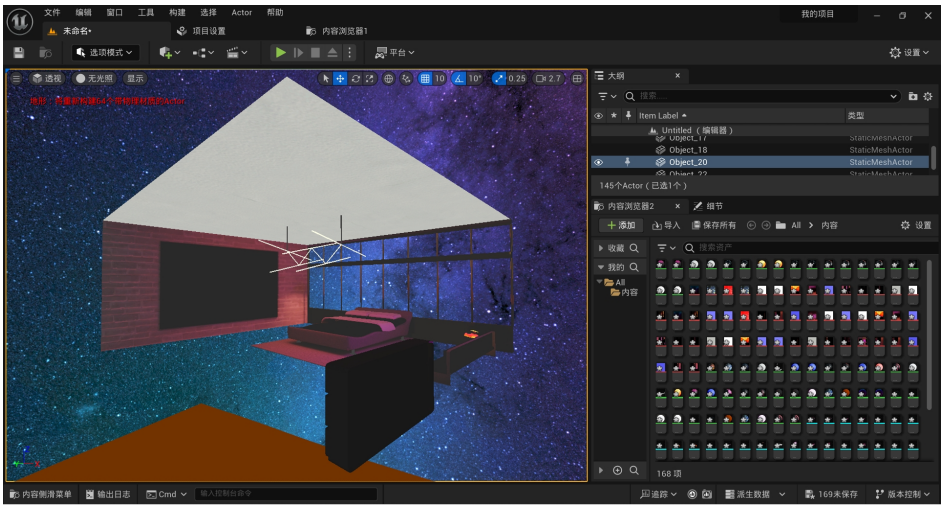

Figure 4.9: Unreal Engine 5

E. Photoshop

Photoshop provides powerful image editing capabilities that can help Blender users create and modify various types of textures, such as color maps, normal maps, metallicity maps, etc. It supports multiple image formats and provides rich tools such as painting tools, filter effects, etc., making it convenient for users to create and modify textures.

4.2.3 Technical Process Analysis

A. 3D Scene Construction

Technology: Blender + Unreal Engine 5

Using Blender and Unreal Engine 5 for modeling, increasing scene realism through ray tracing and baking techniques. The GLTFLoader of Three.js is used to import optimized. glb models. Blender, as a professional 3D modeling software, provides us with rich modeling tools and plugins, such as mesh editing, UV unfolding, material editing, etc., enabling us to quickly and efficiently create realistic 3D scene assets. Unreal Engine 5 has brought us more advanced rendering technologies, such as physics based ray tracing and baking functions, which greatly enhance the realism and visual impact of 3D scenes, enhancing the user's immersive experience. Figure 4.10 and Figure 4.11 shows 3D scene model construction and baking.

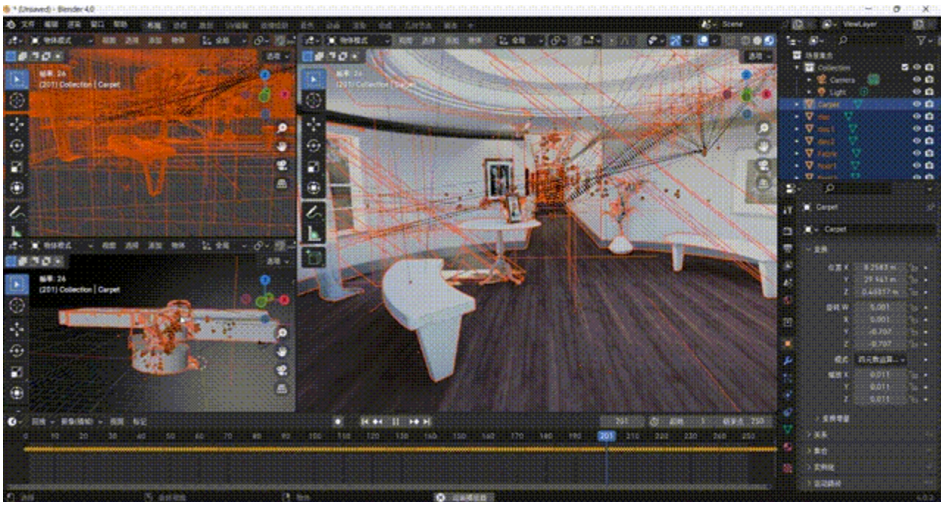

Figure 4.10: Ray tracing, Baking

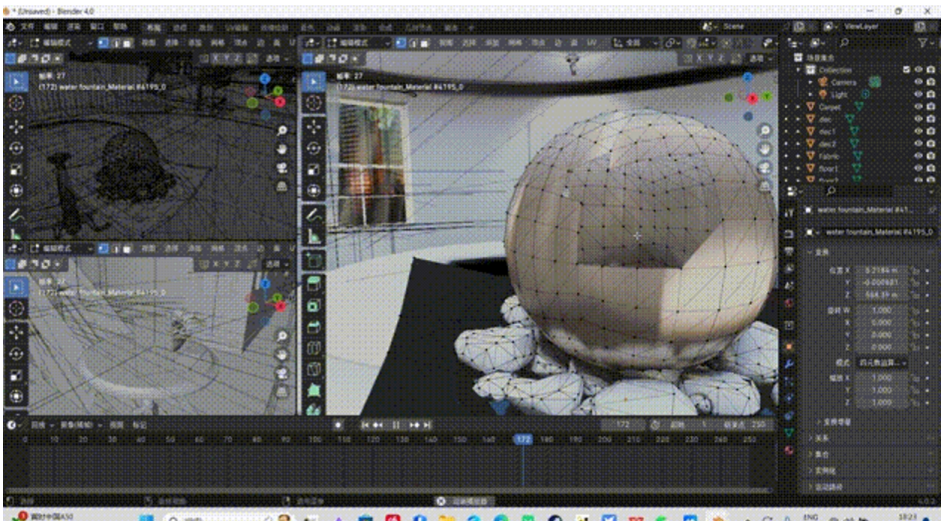

Figure 4.11: Vertex And Surface Editing Mode

B. Present 3D Models On Web Platform

Technology: Three.js + GLTFLoader

In order to seamlessly import the. glb model assets produced by Blender and Unreal Engine 5 into the 3D open world driven by Three.js, we adopted the GLTFLoader component of Three.js. GLTFLoader can efficiently parse and render optimized. glb format models. Create a Three.js scene and add cameras and lights. Instantiate GLTFLoader and use it to load GLB model files. Get the root node of the model in the loading callback function and add it to the Three.js scene. Finally, import the JavaScript scene as a .js file into the HTML file that needs to be rendered. Figure 4.12 shows 3D model scene on web platform.

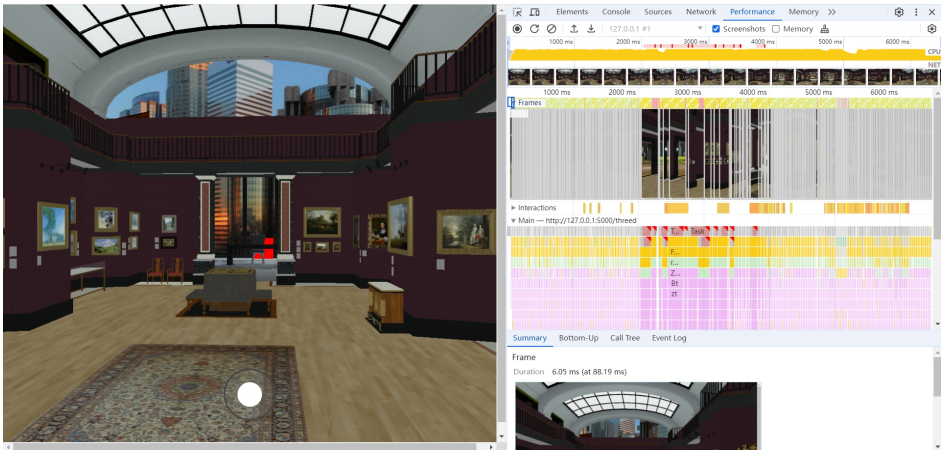

Figure 4.12: Present 3D Models On Web Platform

C. Present 3D Model Animation

Technology: Three.js + AnimationMixer

AnimationMixer is an important class in Three.js used to manage animations. AnimationMixer can play, pause, stop, and reset animations. It provides rich APIs to control the playback of animations. You can also listen to key events in the animation, such as the start, end, loop, etc., and trigger corresponding callback functions. For animation, in the loading callback function, obtain the animation array of the model. Create an instance of Three.js AnimationMixer and create an AnimationAction for each animation. Update AnimationMixer in the rendering loop to make the animation play. Figure 4.13 shows 3D model scene galaxy rotation animation, 180 second animation loop playback.

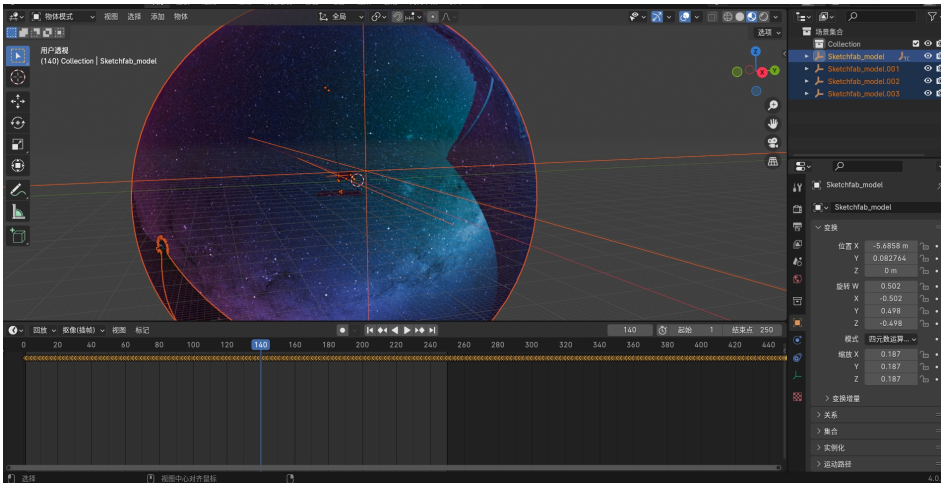

Figure 4.13: Galaxy Rotation Animation

D. Connect 3D World With 2D Interface

Technology: Three.js + Javascript

By setting the collision box, the distance between the player and the collision box is determined per unit time interval "dt". The 2D interface redirection is performed through JavaScript backend logic, and the code logic is included in the appendix's code section "Connect 3D World With 2D Interface". Figure 4.14 shows 3D model collision box.

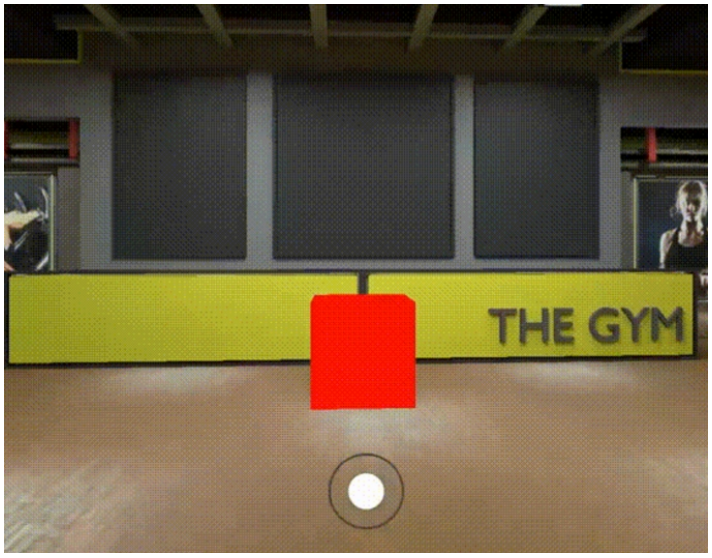

Figure 4.14: 3D Model Collision Box

E. Mobile Device Interaction Control

Technology: Three.js + Joystick

The mobile end adopts Joystick control method, mapping input events to character movement through Three.js. For mobile users, we have adopted the Joystick control method based on Three.js, which can map physical input devices such as touch screen gestures to the movement behavior of 3D characters, allowing users to more naturally control the behavior of characters in 3D space. This not only improves the intuitiveness and responsiveness of interaction, but also

reduces the learning cost for users and enhances the overall user experience.Figure 4.15 shows mobile device interaction control.

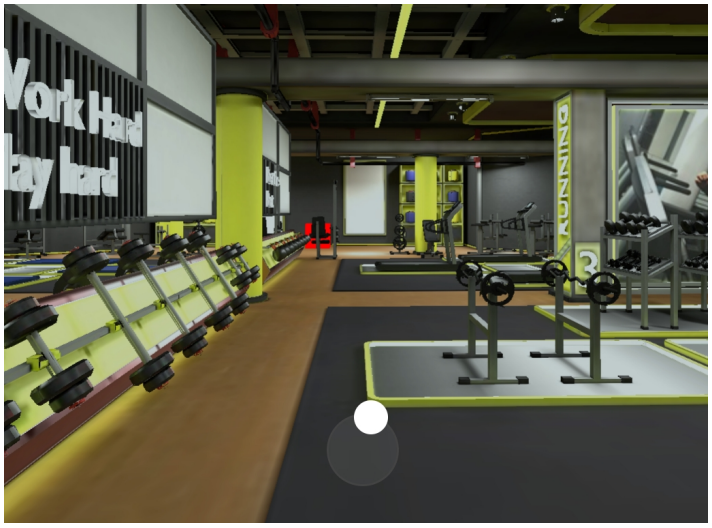

Figure 4.15: Mobile Device Interaction Control

F. PC Device Interaction Control

Technology: Three.js + KeyListener

After initializing the player's initial position and orientation. By listening to user input and binding the W A S D key on the keyboard, the initial speed is given in front, back, left, and right directions. Update the player's position and orientation based on user input. Update the camera position in the scene to ensure that players are within their field of view.

G. Performance Optimization

Significant optimization has been made in terms of running frame rate and server loading 3D scene response time, as high frame rates (60FPS and above) can provide smooth animation effects, making user experience more natural and reducing discomfort such as eye shaking. Table 4.1 shows the .glb scene model 1 optimization result, table 4.2 shows the .glb scene model 1 optimization result, table 4.3 shows the .glb scene model 1 optimization result while Table 4.4 shows the testing hardware standard.

Table 4.1: .glb Model 1 Optimization Result

Value	Original	Current
Average FPS	36 fps	147 fps
Load time	37s	7s

Table 4.2: .glb Model 2 Optimization Result

Value	Original	Current
Average FPS	42 fps	153 fps
Load time	21s	5s

Table 4.3: .glb Model 3 Optimization Result

Value	Original	Current
Average FPS	56 fps	162 fps
Load time	15s	4s

Table 4.4: Testing Hardware Standard

Hardware	Type
Graphics card:	NVIDIA GeForce RTX 3060
Processor:	11th Gen Intel (R) Core (TM) i7-11800H @ 2.30GHz
RAM:	16.0 GB (15.8 GB available)

1. **Baking:**
   - For static scene elements that do not require real-time changes, pre bake lighting has made in Blender, which greatly reduces the burden of Three.js in real-time rendering and improves the overall frame rate.
2. **Time interval:**
   - Appropriate time intervals have set to control the frequency of interface redirect detection, which can effectively reduce unnecessary computational overhead and further optimize performance.
3. **GLTF Model Compression:**
   - **The Importance of Compressing GLTF Model**
     - The smaller the download scene GLTF model package, the smaller the amount of data processed and transmitted by the server, which can reduce the load pressure on the server. Smaller data transfer means higher bandwidth utilization of the server, which can support more concurrent user access. This helps to improve the response speed and overall performance of the server. A small download package can reduce the amount of mobile network traffic consumed by users, especially for users with limited plans, which is very friendly. This helps to improve user satisfaction and stickiness.
   - **Composition of GLTF Model**
     - The size of a 3D scene model mainly consists of two parts. The first part is the number of objects and faces in the scene, as in computer graphics, the vertex coordinates of each geometric object are stored in the form of coordinate information. Irregular objects such as vegetation are very large. The second part is the texture image size. Figure 4.16 shows the two part of GLTF model composition.

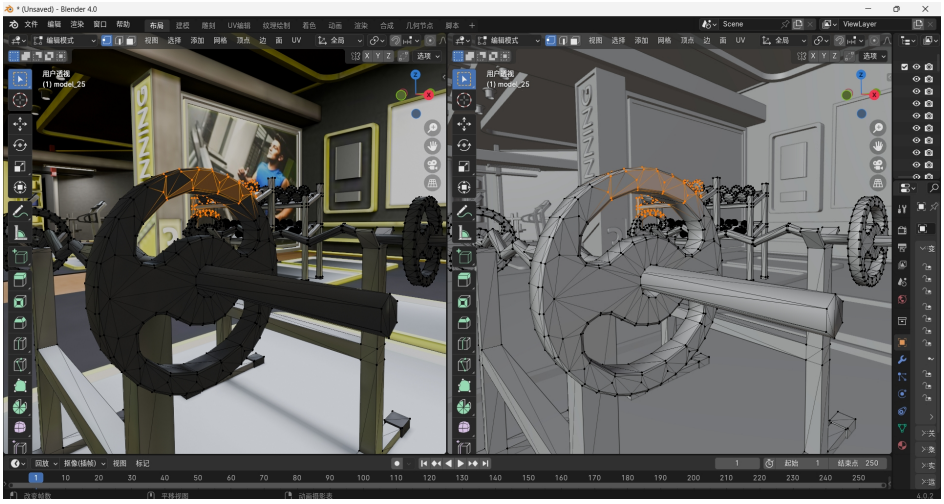

Figure 4.16: Composition of GLTF Model

- **Step 1: Blender Compression**
  - Use Blender to automatically compress the model, and then use DRACOLoader in Three.js to decompress it to the front-end of the web platform. The inner logic is to automatically merge similar materials and textures, reducing the number of materials. Automatically recognize adjacent patches that can be merged in the model. Identify adjacent vertices in the model that can be safely deleted. Table 4.5 shows the blender compression result.

Table 4.5: Blender Compression Result

Model Size	Original	Current
.glb Model 1 Size	137.07 MB	102.09 MB
.glb Model 2 Size	71.03 MB	50.05 MB
.glb Model 3 Size	17.19 MB	13.80 MB

- **Step 2: Model Faces Reduction**
  - By using Blender's built-in "Refine modifier", the ratio of individual model faces exceeding 5000 is set to 20 percent of the original, thereby reducing the large number of faces to 20 percent of the original. This significantly reduces the computational power required for rendering per unit scene. Table 4.6 shows the model faces reduction result. Figure 4.17 shows model objects, model faces and scene triangles reduction of .glb model 1.

Table 4.6: Model Faces Reduction Result

Model Size	Original	Current
.glb Model 1 Size	102.09 MB	52.70 MB
.glb Model 2 Size	50.05 MB	24.09 MB
.glb Model 3 Size	13.80 MB	6.72 MB

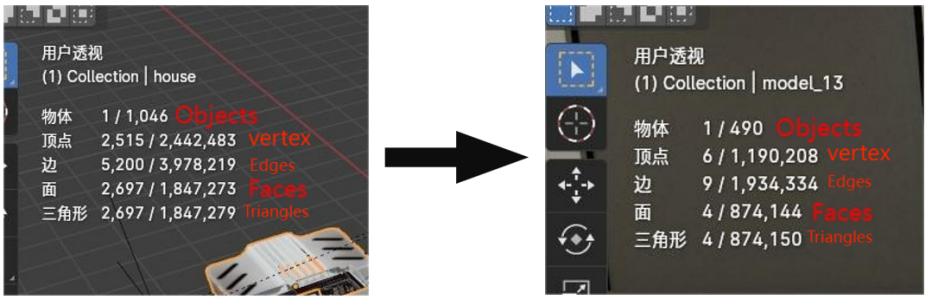

Figure 4.17: .glb Model 1 Faces Reduction

- **Step 3: Texture Compression**
  - Use Blender software to unpack all texture files in the model, and manually reduce the texture resolution to 20 percent of the original through drawing software and Photoshop. Table 4.7 shows the texture compression result

Table 4.7: Texture Compression Result

Model Size	Original	Current
.glb Model 1 Size	52.70 MB	19.10 MB
.glb Model 2 Size	24.09 MB	9.02 MB
.glb Model 3 Size	6.72 MB	3.12 MB

- **Achievement: Optimized .glb Scene Model Size**
  - Table 4.8 shows the overall compression result.

Table 4.8: Compression Result

Model Size	Original	Current
.glb Model 1 Size	137.07 MB	19.10 MB
.glb Model 2 Size	71.03 MB	9.02 MB
.glb Model 3 Size	17.19 MB	3.12 MB

4.3 Machine Learning

4.3.1 Fitness Plan Recommendation Model

Training Process

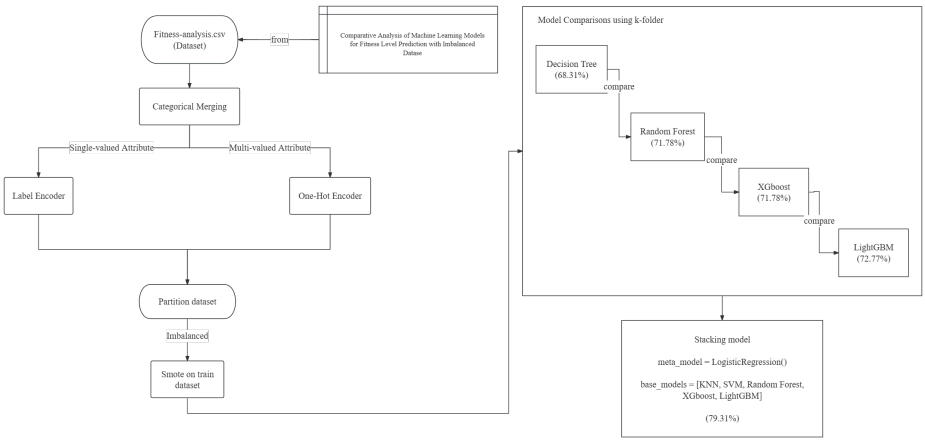

Figure 4.18: Fitness Model

A. Data Selection

Dataset called "Fitness Analysis" containing fitness related data was obtained from [6]. The dataset from [6] consisted of 1010 instances and 17 attributes. One attribute, which is the name of the respondent was removed, leaving the dataset with only 16 attributes. Table 4.9 shows the description of the 16 attributes in the dataset.

Table 4.9: Attributes Description

Attribute	Description
Gender	The gender of the respondent
Age	The age group of the respondent
Exercise_importance	How important is exercise to the respondent
Regularity	How often does the respondent exercise
Barriers	What barriers prevent the respondent from exercising more regularly
Exercises	What form(s) of exercises do the respondent participate in currently
Do you	Does the respondent exercise alone, with a friend or in a group or not at all
Time	What time of the day does the respondent prefer to exercise
Time_spent	How long does the respondent spend exercising per day
Balanced_diet	Does the respondent eat a healthy balanced diet
Prevents_balanced	What prevents the respondent from eating a healthy balanced diet, if any
Health_level	How healthy does the respondent consider himself/herself on a scale of 1 to 5
Recommend_fitness	Does the respondent recommend his/her friends to follow a fitness routine
Equipment	Have the respondent ever purchased a fitness equipment
Motivation	What motivates the respondent to exercise
Fitness_level	How does the respondent describe his/her current level of fitness

B. Data preprocessing

In the dataset, all variables were categorical in nature. Consequently, to facilitate computational analysis, each category within these variables was transformed into a numerical code, with distinct integers representing different categories. This process of numerical encoding was applied across all 16 attributes identified in the dataset.

Of these attributes, four of the attributes ("Barriers", "Exercises", "Prevents_balanced" and "Motivation") had multiple values. Each of these options was encoded separately to ensure detailed variable representation, leading to a total of 41 descriptive attributes in the dataset.

Additionally, the dataset featured a class attribute labeled "Fitness_level", which originally consisted of five categories: Unfit, Average, Good, Very Good, and Perfect. To enhance the model's ability to delineate classes and to address issues of class imbalance, the categories were consolidated. Specifically, the Average and Good categories were merged into a single category named Average, while Very Good and Perfect were combined under the label Fit. This modification reduced the number of classes in the class attribute to three distinct categories: Unfit, Average, and Fit. Table 4.10 in the document details the encoding scheme for each category across the attributes.

Table 4.10: Attribute Encoding

Attribute	Description
Gender	0: Female

Continued on next page

Table 4.10 – continued from previous page

Attribute	Description
	1: Male
Age	0: 15 to 18 1: 19 to 25 2: 26 to 30 3: 30 to 40 4: 40 and above
Exercise_importance	1: Not at all important 2: Not so important 3: Average 4: Important 5: Very important
Regularity	1: Never 2: 1 to 2 times a week 3: 3 to 4 times a week 4: 5 to 6 times a week 5: Everyday
Barriers	I don't have enough time: 0: No, 1: Yes I can't stay motivated: 0: No, 1: Yes I'll become too tired: 0: No, 1: Yes I have an injury: 0: No, 1: Yes I don't really enjoy exercising: 0: No, 1: Yes I exercise regularly with no barriers: 0: No, 1: Yes Barriers_Other: 0: No, 1: Yes
Exercises	Walking or jogging: 0: No, 1: Yes Gym: 0: No, 1: Yes Swimming: 0: No, 1: Yes Yoga: 0: No, 1: Yes Zumba dance: 0: No, 1: Yes Lifting weights: 0: No, 1: Yes Team sport: 0: No, 1: Yes I don't really exercise: 0: No, 1: Yes Exercises_Other: 0: No, 1: Yes
Do_you	1: Alone 2: With a friend 3: With a group 4: Within a class environment 5: I don't really exercise
Time	1: Early morning 2: Afternoon 3: Evening
Time_spent	1: 30 minutes 2: 1 hour 3: 2 hours 4: 3 hours and above 5: I don't really exercise
Balanced_diet	1: Yes 2: No 3: Not always
Prevents_balanced	Lack of time: 0: No, 1: Yes Cost: 0: No, 1: Yes Ease of access to fast food: 0: No, 1: Yes

Continued on next page

Table 4.10 – continued from previous page

Attribute	Description
	Temptation and cravings: 0: No, 1: Yes I have a balanced diet: 0: No, 1: Yes Prevents_balanced_Other: 0: No, 1: Yes
Health_level	1: Unhealthy 2: Below average 3: Average 4: Above average 5: Healthy
Recommend_fitness	0: No 1: Yes
Equipment	0: No 1: Yes
Motivation	I want to be fit: 0: No, 1: Yes I want to increase muscle mass and strength: 0: No, 1: Yes I want to lose weight: 0: No, 1: Yes I want to be flexible: 0: No, 1: Yes I want to relieve stress: 0: No, 1: Yes I want to achieve a sporting goal: 0: No, 1: Yes I am not really interested in exercising: 0: No, 1: Yes Motivation_Other: 0: No, 1: Yes
Fitness_level	1: Unfit 2: Average 3: Fit

Despite the reclassification of the class attribute, the dataset remained imbalanced. To address this, the Synthetic Minority Oversampling Technique (SMOTE) [7] was employed on training set to achieve a balanced dataset. SMOTE works by artificially generating new instances in the minority classes, thereby enhancing the dataset's balance. Empirical evidence suggests that using SMOTE to augment the minority class sizes can significantly improve the performance of classifiers. Table 4.11 details the distribution of the class attribute before and after the application of SMOTE.

Table 4.11: Distribution of the Class Attribute Before and After SMOTE

Attribute Value	Original	With SMOTE
Unfit	99	746
Average	746	746
Fit	165	746
Total	1010	2238

C. Model Evaluation

This project proposes a classification task, specifically a multi-class classification of three fitness levels: Unfit, average and Fit. In this case, each instance will be categorized as one of these three fitness levels. Six machine learning techniques were used to build the fitness level classification model: Decision Tree (DT), K-Nearest Neighbour (KNN), Support Vector Machine (SVM), Random Forest (RF), LightGBM (LGBM) and XGBoost (XGB).

K-fold Cross-Validation

In this project, machine learning models are evaluated using K-fold cross validation, which aims to fully utilize a limited dataset to evaluate model performance and reduce the risk of overfitting. In K-fold cross validation, the dataset is divided into k mutually exclusive subsets, called folds. The model is then trained k times, each time using k-1 of the folds as the training set and the remaining one as the validation set. After each training, the performance metrics of the model on the validation set, such as accuracy, precision, recall, etc., are calculated. Eventually, the performance metrics obtained from K times of training are averaged as the final performance evaluation metrics of the model. K-fold cross-validation helps to solve the chance problem that may be introduced by a single dataset division. By using different combinations of training and validation sets multiple times, the generalization ability of the model can be better assessed as the model is trained and validated on several different data subsets.

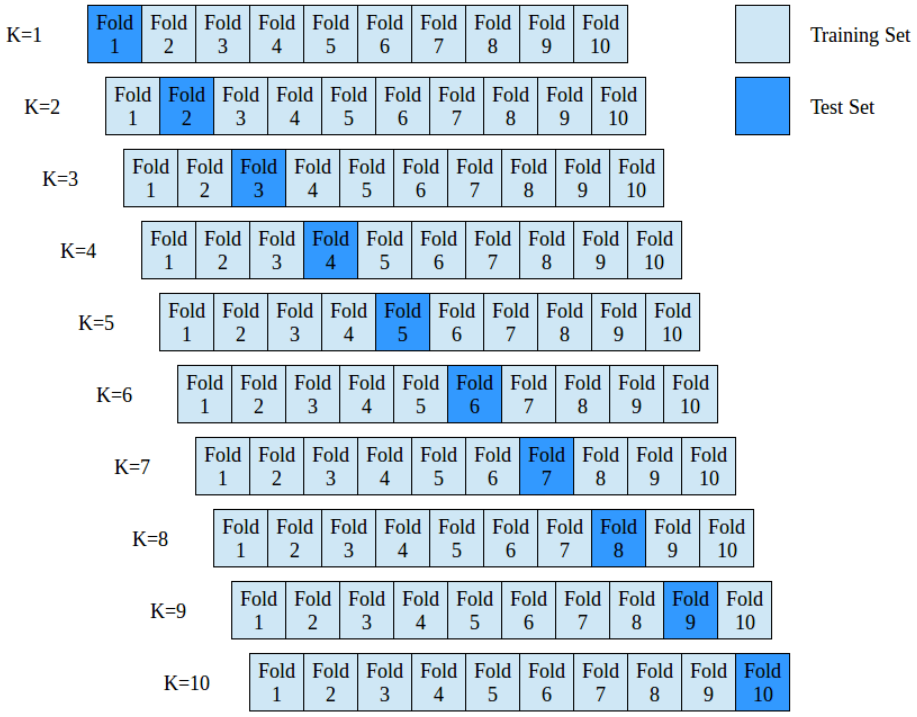

Figure 4.19: K-fold Cross-validation

Confusion Matrix

In this project, the performance of the multiclassification model is evaluated using the Confusion Matrix, which classifies the samples in the dataset into four categories based on the actual class (true value) and the class predicted by the model: true positive (TP), false positive (FP), true negative (TN), and false negative (FN).

- True Positive (TP): The number of samples correctly predicted as positive.
- False Positive (FP): The number of samples incorrectly predicted as positive.
- True Negative (TN): The number of samples correctly predicted as negative.
- False Negative (FN): The number of samples incorrectly predicted as negative.

Specific Calculations

- **Average Class**
  - **TP**: Samples with true labels of "Average" correctly classified as "Average".
  - **FP**: Samples with true labels other than "Average" incorrectly classified as "Average".
  - **FN**: Samples with true labels of "Average" incorrectly classified as other categories ("Fit" or "Unfit").
  - **TN**: Samples with true labels other than "Average" correctly classified as other categories ("Fit" or "Unfit").
- **Fit Class**
  - **TP**: Samples with true labels of "Fit" correctly classified as "Fit".
  - **FP**: Samples with true labels other than "Fit" incorrectly classified as "Fit".
  - **FN**: Samples with true labels of "Fit" incorrectly classified as other categories ("Average" or "Unfit").
  - **TN**: Samples with true labels other than "Fit" correctly classified as other categories ("Average" or "Unfit").
- **Unfit Class**
  - **TP**: Samples with true labels of "Unfit" correctly classified as "Unfit".
  - **FP**: Samples with true labels other than "Unfit" incorrectly classified as "Unfit".
  - **FN**: Samples with true labels of "Unfit" incorrectly classified as other categories ("Average" or "Fit").
  - **TN**: Samples with true labels other than "Unfit" correctly classified as other categories ("Average" or "Fit").

The confusion matrix provides insight into the performance of model on different classes and allows for the evaluation of performance metrics such as accuracy, precision, recall, and F1-score. These metrics help in assessing the overall performance of the model.

Evaluation Metrics

Experiments were then conducted,using ten-fold cross-validation. The generated models were evaluated using accuracy, precision, recall, and F1 scores based on the confusion matrix values for true cases (TP), false positive cases (FP), true negative cases (TN), and false negative cases (FN). Table 4.19 shows the formulas used to evaluate the metrics. Since this model is a triple classification problem, the final score is taken as an average

Table 4.12: Formula For Evaluation Metrics

Technique	Parameters
Accuracy	(TP + TN) / (TP + TN + FP + FN)
Precision	TP / (TP + FP)
Recall	TP / (TP + FN)
F1-score	2 * (Precision * Recall) / (Precision + Recall)

Receiver Operating Characteristic curve

The ROC curve is used in this project to to evaluate the performance of a classification model across different decision thresholds. It evaluates the model by comparing the proportion of true positives identified (True Positive Rate) against the proportion of false positives (False Positive Rate). An ideal model's ROC curve approaches the top left corner, indicating efficient recognition of positives with a low rate of misclassification. The area under the ROC curve (AUC) closer to 1 signifies better overall model performance.

For multiclass classification problems, the ROC curve can be plotted by considering the problem as multiple binary classification tasks. There are generally two approaches:

1. **One-vs-All:** A ROC curve is drawn for each category by treating one category as the positive class and all other categories as the negative class.
2. **One-vs-One:** A ROC curve is drawn for each pair of categories. In the case of multiple categories, this will produce a large number of ROC curves.

Decision Tree

- **Confusion Matrix for Decision Tree**

The confusion matrix displayed in the Fig. 4.20 represents the performance of a decision tree classifier across three classes: "Average", "Fit", and "Unfit". Each row of the matrix corresponds to the true class labels, and each column corresponds to the predicted labels.

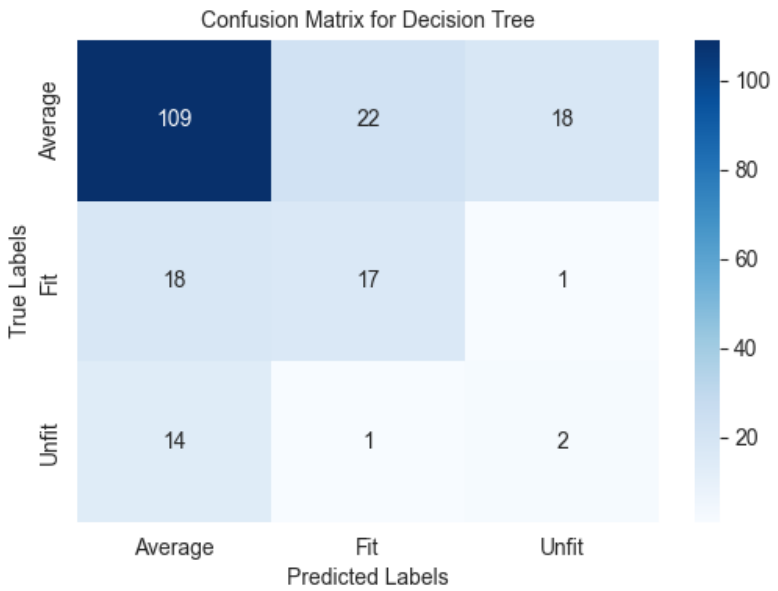

Figure 4.20: Confusion Matrix for Decision Tree

- **Average:** Out of 149 data points truly labeled as "Average", 109 are correctly predicted as "Average", 22 are incorrectly predicted as "Fit", and 18 as "Unfit".
- **Fit:** Out of 36 data points truly labeled as "Fit", 18 are incorrectly predicted as "Average", 17 are correctly predicted as "Fit", and 1 as "Unfit".
- **Unfit:** Out of 17 data points truly labeled as "Unfit", 14 are incorrectly predicted as "Average", 1 as "Fit", and 2 are correctly predicted as "Unfit".

- **Evaluation Metrics for Decision Tree**

Table 4.13: Evaluation Metrics for Decision Tree

Model	Accuracy	Precision	Recall	F1-score
DT	0.72	0.56	0.72	0.72

- **ROC Curve for Decision Tree**

When examining the Fig. 4.21 ROC curves of decision tree model for a multi-class classification problem, the graph displays three curves representing three classes: "Average," "Fit," and "Unfit." The x-axis represents the False Positive Rate (FPR), while the y-axis represents the True Positive Rate (TPR), both commonly used metrics for assessing classifier performance.

- The blue curve represents the ROC curve for the "Average" class, with an Area Under the Curve (AUC) of 0.56. This value indicates that the model's performance in distinguishing the "Average" class from other classes is slightly better than random guessing (0.5).
- The red curve represents the ROC curve for the "Fit" class, with an AUC of 0.67. This suggests that the model performs well in identifying the "Fit" class, significantly outperforming random guessing.
- Finally, the green curve represents the ROC curve for the "Unfit" class, with an AUC of 0.51. This value is very close to random guessing, indicating that the model has little advantage in identifying the "Unfit" class.

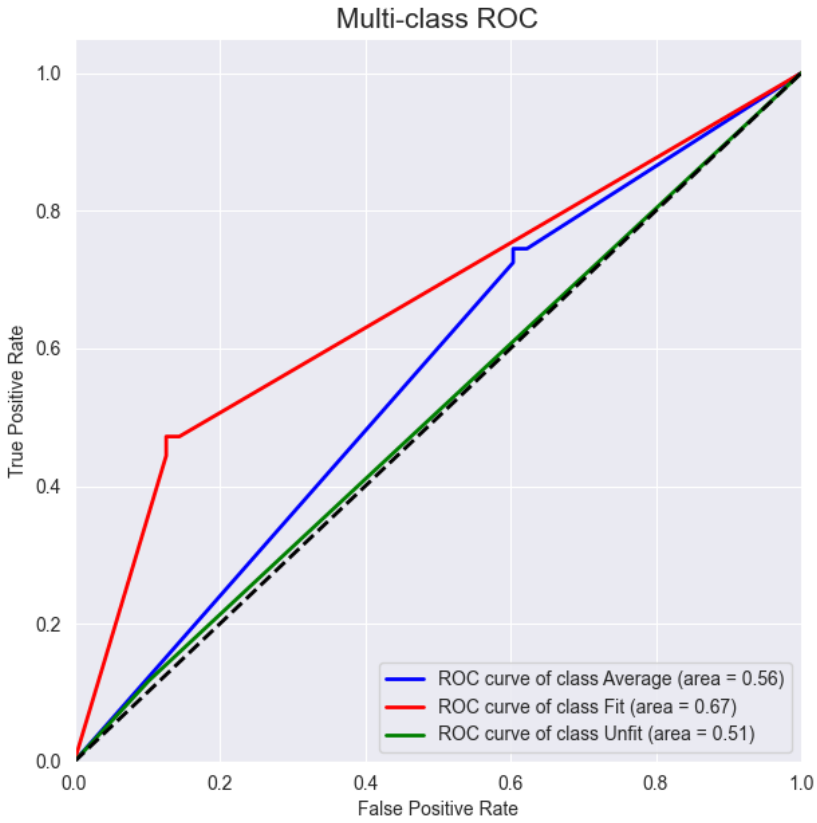

Figure 4.21: ROC Curve for Decision Tree

LightGBM

- **Confusion Matrix for LightGBM**

The confusion matrix displayed in the Fig. 4.22 represents the performance of a decision tree classifier across three classes: "Average", "Fit", and "Unfit". Each row of the matrix corresponds to the true class labels, and each column corresponds to the predicted labels.

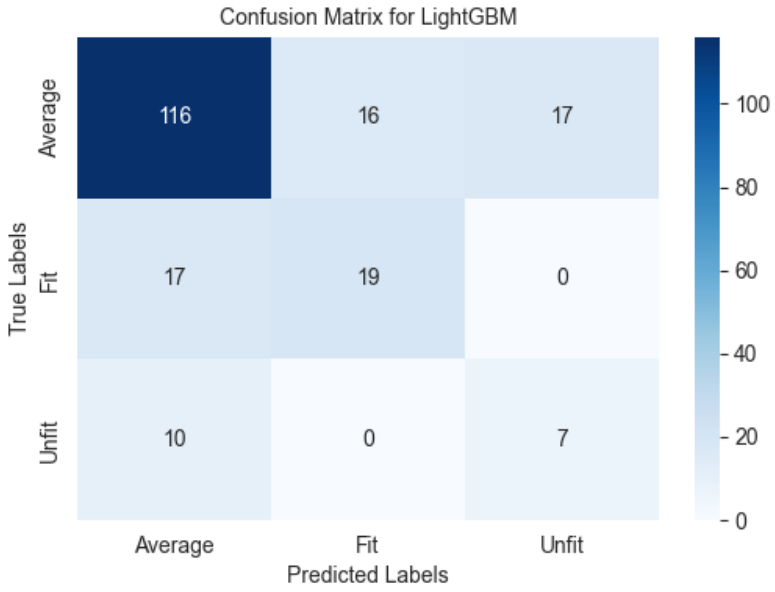

Figure 4.22: Confusion Matrix for LightGBM

- **Average:** Out of 149 data points truly labeled as "Average", 116 are correctly predicted as "Average", 16 are incorrectly predicted as "Fit", and 17 as "Unfit".
- **Fit:** Out of 36 data points truly labeled as "Fit", 17 are incorrectly predicted as "Average", 19 are correctly predicted as "Fit", and 0 as "Unfit".
- **Unfit:** Out of 17 data points truly labeled as "Unfit", 10 are incorrectly predicted as "Average", 0 as "Fit", and 7 are correctly predicted as "Unfit".

- **Evaluation Metrics for LightGBM**

Table 4.14: Evaluation Metrics for LightGBM

Model	Accuracy	Precision	Recall	F1-score
LGBM	0.78	0.67	0.78	0.77

- **ROC Curve for LightGBM**

When examining the Fig. 4.23 ROC curves of LightGBM model for a multi-class classification problem, the graph displays three curves representing three classes: "Average," "Fit," and "Unfit." The x-axis represents the False Positive Rate (FPR), while the y-axis represents the True Positive Rate (TPR), both commonly used metrics for assessing classifier performance.

- The blue curve represents the ROC curve for the "Average" class, with an Area Under the Curve (AUC) of 0.68. This value indicates that the model's performance in distinguishing the "Average" class from other classes is better than random guessing (0.5).
- The red curve represents the ROC curve for the "Fit" class, with an AUC of 0.82. This suggests that the model performs well in identifying the "Fit" class, significantly outperforming random guessing.

- Finally, the green curve represents the ROC curve for the "Unfit" class, with an AUC of 0.82, indicating that the performance of the model in identifying the " unfit" class has a significant advantage.

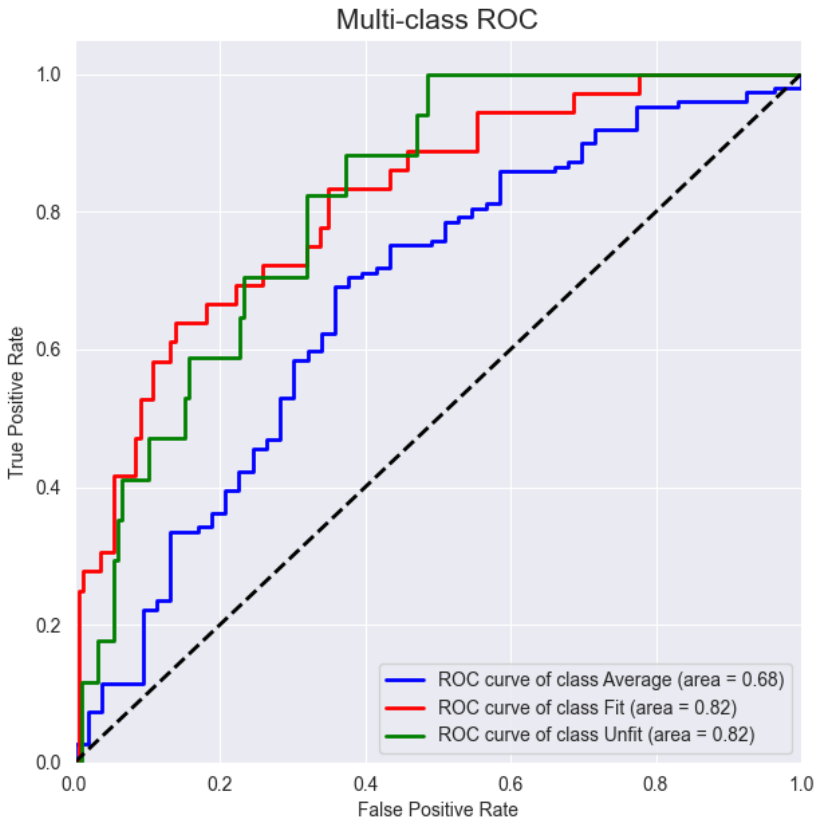

Figure 4.23: ROC Curve for LightGBM

K-Nearest Neighbors

- **Confusion Matrix for KNN**

The confusion matrix displayed in the Fig. 4.24 represents the performance of a KNN classifier across three classes: "Average", "Fit", and "Unfit". Each row of the matrix corresponds to the true class labels, and each column corresponds to the predicted labels.

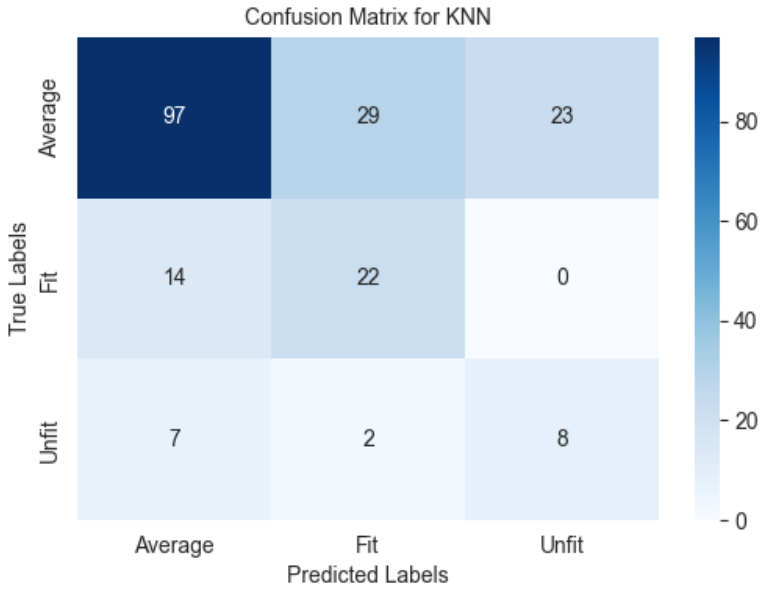

Figure 4.24: Confusion Matrix for KNN

- **Average:** Out of 149 data points truly labeled as "Average", 97 are correctly predicted as "Average", 29 are incorrectly predicted as "Fit", and 23 as "Unfit".
- **Fit:** Out of 36 data points truly labeled as "Fit", 14 are incorrectly predicted as "Average", 22 are correctly predicted as "Fit", and 0 as "Unfit".
- **Unfit:** Out of 17 data points truly labeled as "Unfit", 7 are incorrectly predicted as "Average", 2 as "Fit", and 8 are correctly predicted as "Unfit".

- **Evaluation Metrics for KNN**

Table 4.15: Evaluation Metrics for KNN

Model	Accuracy	Precision	Recall	F1-score
KNN	0.80	0.76	0.80	0.77

- **ROC Curve for KNN**

When examining the Fig. 4.25 ROC curves of decision tree model for a multi-class classification problem, the graph displays three curves representing three classes: "Average," "Fit," and "Unfit." The x-axis represents the False Positive Rate (FPR), while the y-axis represents the True Positive Rate (TPR), both commonly used metrics for assessing classifier performance.

- The blue curve represents the ROC curve for the "Average" class, with an Area Under the Curve (AUC) of 0.64. This value indicates that the model's performance in distinguishing the "Average" class from other classes is better than random guessing (0.5).
- The red curve represents the ROC curve for the "Fit" class, with an AUC of 0.76. This suggests that the model performs well in identifying the "Fit" class, significantly outperforming random guessing.

- Finally, the green curve represents the ROC curve for the "Unfit" class, with an AUC of 0.71, indicating that the model has significant advantage in identifying the "Unfit" class.

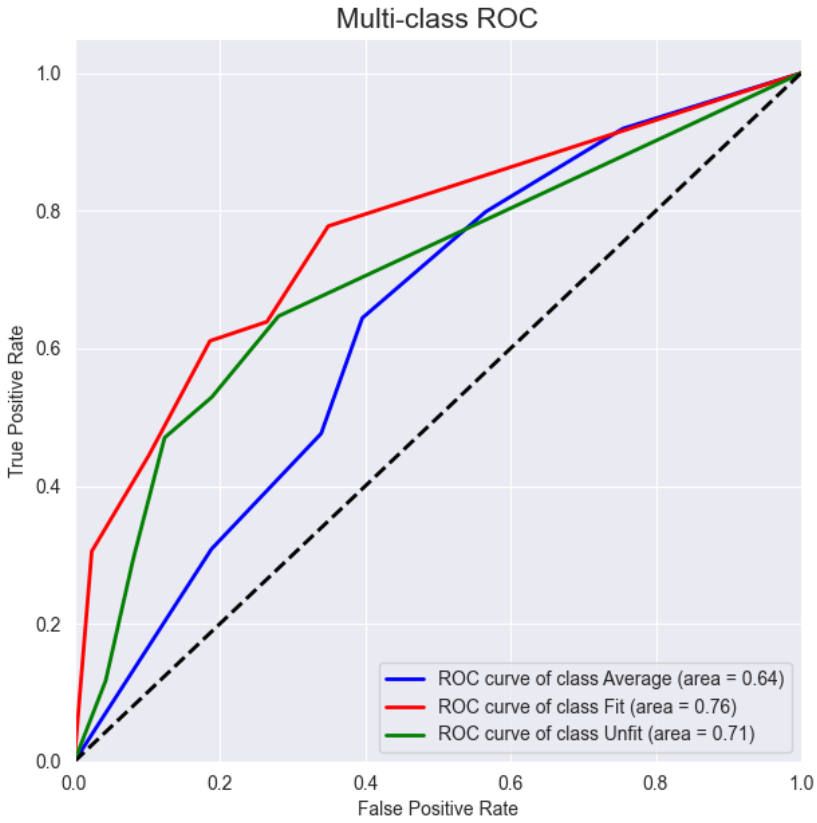

Figure 4.25: ROC Curve for KNN

Random Forest

- **Confusion Matrix for Random Forest**

The confusion matrix displayed in the Fig. 4.26 represents the performance of a decision tree classifier across three classes: "Average", "Fit", and "Unfit". Each row of the matrix corresponds to the true class labels, and each column corresponds to the predicted labels.

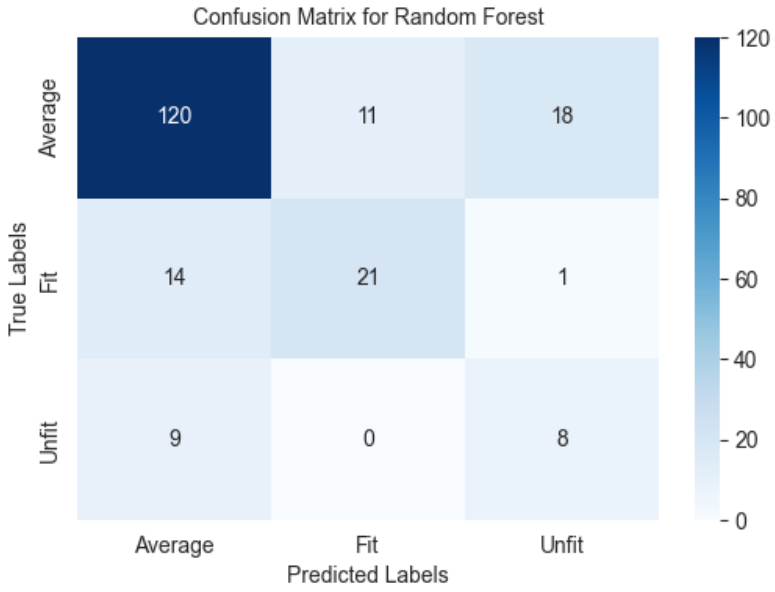

Figure 4.26: Confusion Matrix for Random Forest

- **Average:** Out of 149 data points truly labeled as "Average", 120 are correctly predicted as "Average", 11 are incorrectly predicted as "Fit", and 18 as "Unfit".
- **Fit:** Out of 36 data points truly labeled as "Fit", 14 are incorrectly predicted as "Average", 21 are correctly predicted as "Fit", and 1 as "Unfit".
- **Unfit:** Out of 17 data points truly labeled as "Unfit", 9 are incorrectly predicted as "Average", 0 as "Fit", and 8 are correctly predicted as "Unfit".

- **Evaluation Metrics for Random Forest**

Table 4.16: Evaluation Metrics for Random Forest

Model	Accuracy	Precision	Recall	F1-score
RF	0.81	0.74	0.81	0.79

- **ROC Curve for Random Forest**

When examining the Fig. 4.27 ROC curves of random forest model for a multi-class classification problem, the graph displays three curves representing three classes: "Average," "Fit," and "Unfit." The x-axis represents the False Positive Rate (FPR), while the y-axis represents the True Positive Rate (TPR), both commonly used metrics for assessing classifier performance.

- The blue curve represents the ROC curve for the "Average" class, with an Area Under the Curve (AUC) of 0.75. This value indicates that the model's performance in distinguishing the "Average" class from other classes is significantly better than random guessing (0.5).
- The red curve represents the ROC curve for the "Fit" class, with an AUC of 0.87. This suggests that the model performs well in identifying the "Fit" class, significantly outperforming random guessing.

- Finally, the green curve represents the ROC curve for the "Unfit" class, with an AUC of 0.85. indicating that the model has a considerable advantage in identifying the "Unfit" class.

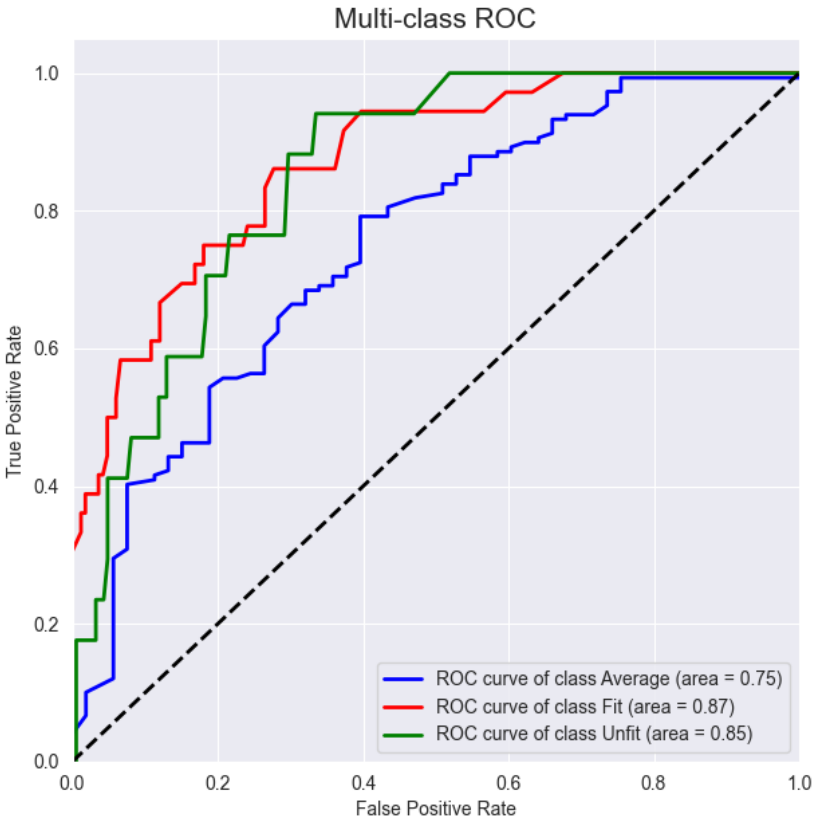

Figure 4.27: ROC Curve for Random Forest

Support Vector Machine

- **Confusion Matrix for SVM**

The confusion matrix displayed in the Fig. 4.28 represents the performance of a decision tree classifier across three classes: "Average", "Fit", and "Unfit". Each row of the matrix corresponds to the true class labels, and each column corresponds to the predicted labels.

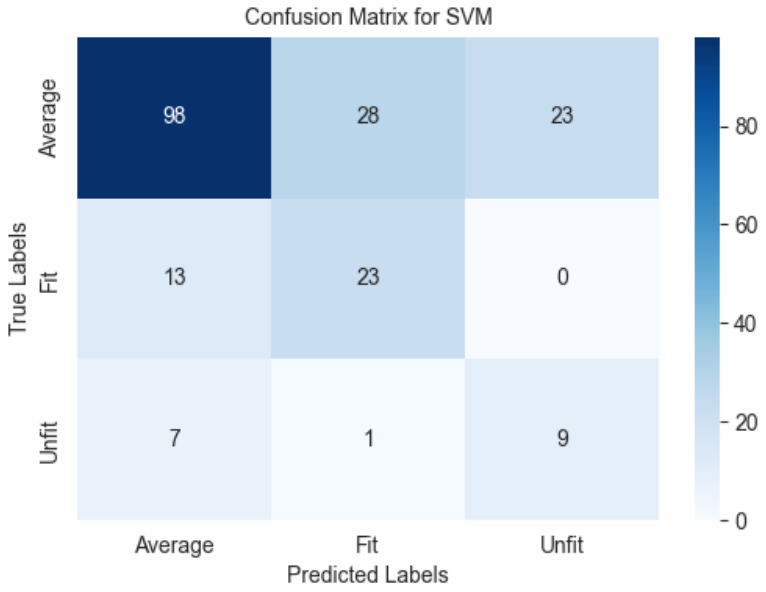

Figure 4.28: Confusion Matrix for SVM

- **Average:** Out of 149 data points truly labeled as "Average", 98 are correctly predicted as "Average", 28 are incorrectly predicted as "Fit", and 23 as "Unfit".
- **Fit:** Out of 36 data points truly labeled as "Fit", 13 are incorrectly predicted as "Average", 23 are correctly predicted as "Fit", and 0 as "Unfit".
- **Unfit:** Out of 17 data points truly labeled as "Unfit", 7 are incorrectly predicted as "Average", 1 as "Fit", and 9 are correctly predicted as "Unfit".

- **Evaluation Metrics for SVM**

Table 4.17: Evaluation Metrics for SVM

Model	Accuracy	Precision	Recall	F1-score
SVM	0.80	0.74	0.80	0.76

- **ROC Curve for SVM**

When examining the Fig. 4.29 ROC curves of SVM model for a multi-class classification problem, the graph displays three curves representing three classes: "Average," "Fit," and "Unfit." The x-axis represents the False Positive Rate (FPR), while the y-axis represents the True Positive Rate (TPR), both commonly used metrics for assessing classifier performance.

- The blue curve represents the ROC curve for the "Average" class, with an Area Under the Curve (AUC) of 0.70. This value indicates that the model's performance in distinguishing the "Average" class from other classes is significantly better than random guessing (0.5).
- The red curve represents the ROC curve for the "Fit" class, with an AUC of 0.85. This suggests that the model performs well in identifying the "Fit" class, significantly outperforming random guessing.

- Finally, the green curve represents the ROC curve for the "Unfit" class, with an AUC of 0.83. indicating that the model has significant advantage in identifying the "Unfit" class.

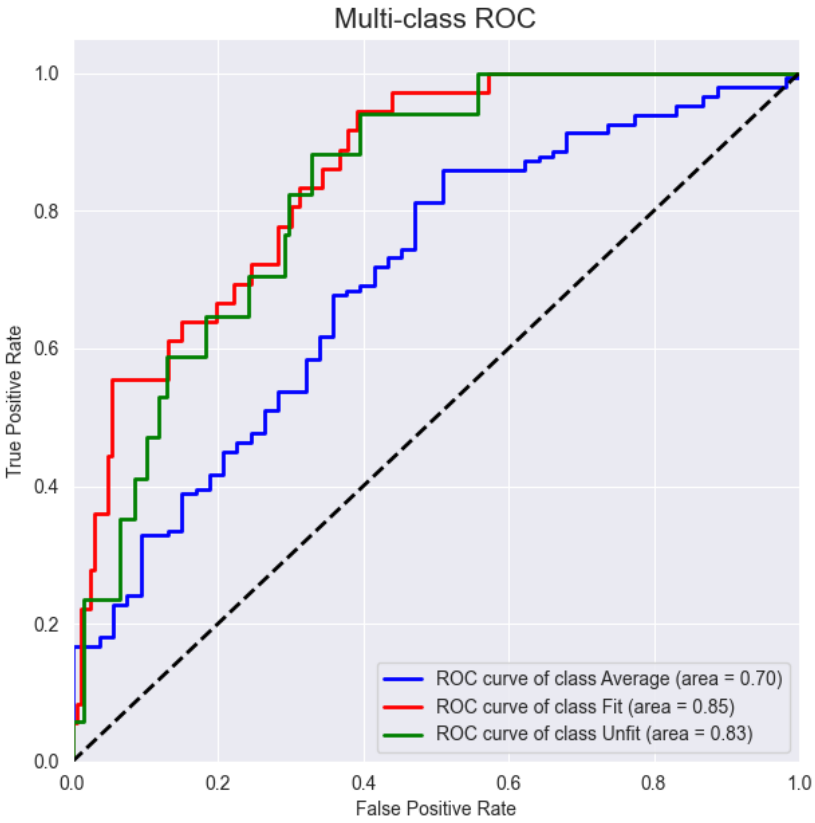

Figure 4.29: ROC Curve for SVM

XGBoost

- **Confusion Matrix for XGBoost**

The confusion matrix displayed in the Fig. 4.30 represents the performance of a XGBoost classifier across three classes: "Average", "Fit", and "Unfit". Each row of the matrix corresponds to the true class labels, and each column corresponds to the predicted labels.

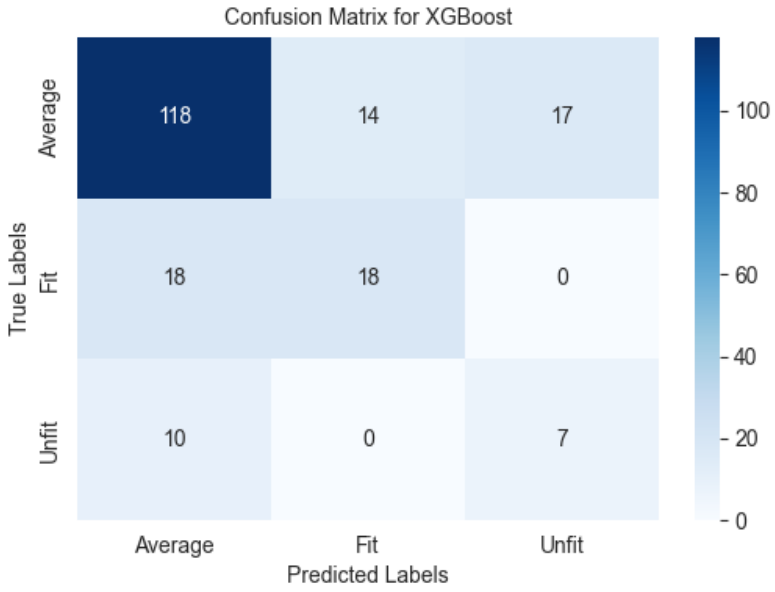

Figure 4.30: Confusion Matrix for XGBoost

- **Average:** Out of 149 data points truly labeled as "Average", 118 are correctly predicted as "Average", 14 are incorrectly predicted as "Fit", and 17 as "Unfit".
- **Fit:** Out of 36 data points truly labeled as "Fit", 18 are incorrectly predicted as "Average", 18 are correctly predicted as "Fit", and 0 as "Unfit".
- **Unfit:** Out of 17 data points truly labeled as "Unfit", 10 are incorrectly predicted as "Average", 0 as "Fit", and 7 are correctly predicted as "Unfit".

- **Evaluation Metrics for XGBoost**

Table 4.18: Evaluation Metrics for XGBoost

Model	Accuracy	Precision	Recall	F1-score
XGB	0.78	0.65	0.78	0.77

- **ROC Curve for XGBoost**

When examining the Fig. 4.31 ROC curves of XGBoost model for a multi-class classification problem, the graph displays three curves representing three classes: "Average," "Fit," and "Unfit." The x-axis represents the False Positive Rate (FPR), while the y-axis represents the True Positive Rate (TPR), both commonly used metrics for assessing classifier performance.

- The blue curve represents the ROC curve for the "Average" class, with an Area Under the Curve (AUC) of 0.68. This value indicates that the model's performance in distinguishing the "Average" class from other classes is better than random guessing (0.5).
- The red curve represents the ROC curve for the "Fit" class, with an AUC of 0.81. This suggests that the model performs well in identifying the "Fit" class, significantly outperforming random guessing.

- Finally, the green curve represents the ROC curve for the "Unfit" class, with an AUC of 0.80, indicating that the performance of the model in identifying the " unfit" class has a significant advantage..

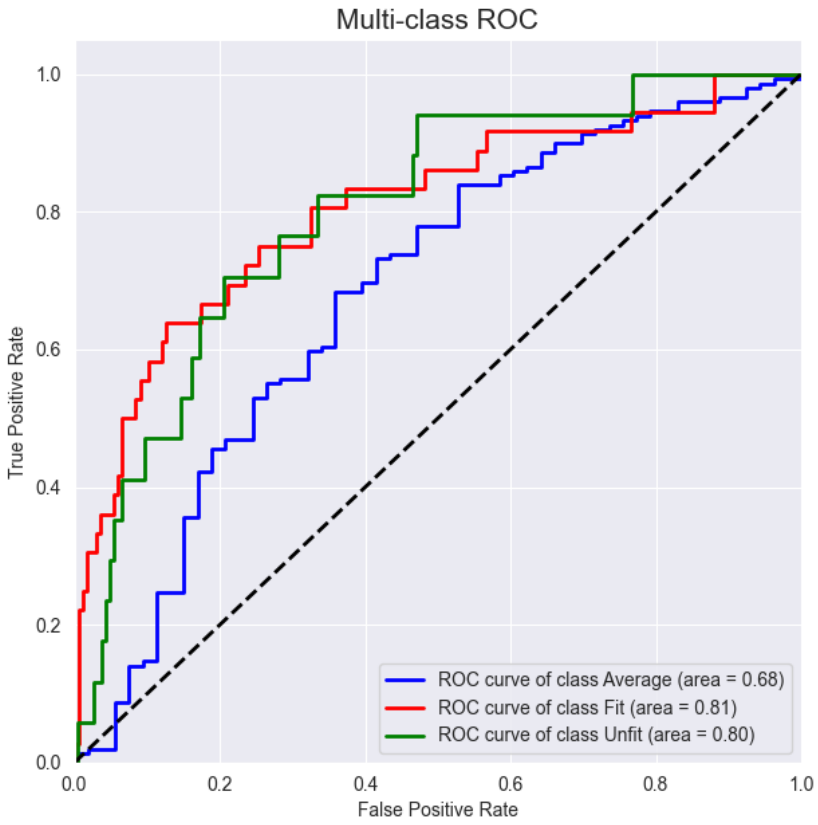

Figure 4.31: ROC Curve for XGBoost

D. Model Comparison

From the data in the Table 4.19, we can observe that there are variations in the performance of different machine learning models in this three-classification task:

Accuracy: The random forest (RF) model leads with an accuracy of 0.81, followed by the SVM and KNN models, both around 0.80. This indicates that these models are effective in classifying samples, but RF has a slightly higher accuracy.

Precision: The KNN model tops the precision with 0.76, followed by the RF and SVM models, both at 0.74. This suggests that the KNN model performs more accurately in predicting positive classes, effectively reducing misclassifications of negative classes.

Recall: Both the DT and KNN models achieve a recall rate of 0.72, showing relatively good performance. The RF and SVM models have higher recall rates, reaching 0.81 and 0.80, respectively. This indicates that these models effectively capture positive class samples.

F1-score: In terms of F1-score, the random forest (RF) model performs the best with a score of 0.79, followed by the KNN and SVM models at 0.77 and 0.76, respectively. F1-score combines precision and recall, providing an evaluation of the overall performance of the models.

Taking all metrics into consideration, the RF model excels in this task, demonstrating superior overall performance. Following are the SVM and KNN models. The DT, LGBM, and XGB models show relatively stable but not outstanding performance across metrics. The decision tree model's performance is slightly lower than the others, while LGBM and XGB are comparable in accuracy and F1-score but slightly lower in precision and recall. It is important to note that even though the accuracy of the DT also performs well, because the evaluation of the model is based on an unbalanced dataset and the precsion and recall of the DT is very low, it can be concluded that the accuracy of the DT is falsely high, and therefore the DT is considered to be underperforming.

Table 4.19: Performance Comparison of Six Machine Learning Models

Model	Accuracy	Precision	Recall	F1-score
DT	0.72	0.56	0.72	0.72
KNN	0.80	0.76	0.80	0.77
RF	0.81	0.74	0.81	0.79
SVM	0.80	0.74	0.80	0.76
LGBM	0.78	0.67	0.78	0.77
XGB	0.78	0.65	0.78	0.77

E. Model Stacking

Based on the previous analysis of the performance of different machine learning models, we have decided to utilize a stacked model to fully leverage the strengths of multiple base models (KNN, RF, SVM, LGBM, XGB), combining their predictions to enhance overall predictive performance. Additionally, stacked models can address bias and variance issues by integrating predictions from multiple models, thus reducing the risk of overfitting and improving generalization capabilities. Moreover, stacked models are suitable for complex tasks and diverse data features, as they can combine various types of models to adapt to different scenarios. Furthermore, stacked models can enhance robustness and reliability by mitigating the impact of individual model idiosyncrasies on predictions, resulting in more stable and dependable forecasts.

In this stacked model, the predictions from multiple base models are used as features input to the meta-learner (Logistic Regression), which then makes the final prediction. Cross-validation is employed to train and evaluate the stacked model, reducing overfitting and enhancing generalization capabilities. Considering the performance of the base models and their correlations, the stacked model can effectively classify data and provide improved performance across multiple metrics.

- **Evaluation Metrics for Stacked Model**

The Table 4.20 indicates that the stacked model performs excellently across evaluation metrics such as accuracy, precision, recall, and F1-score. Compared to the individual base models, the stacked model demonstrates superior performance. This suggests that the stacked model effectively leverages the advantages of multiple models, enhancing overall predictive capability. Consequently, it delivers superior performance when considering multiple metrics simultaneously.

Table 4.20: Evaluation Metrics for Stacked Model

Model	Accuracy	Precision	Recall	F1-score
DT	0.72	0.56	0.72	0.72
KNN	0.80	0.76	0.80	0.77
RF	0.81	0.74	0.81	0.79
SVM	0.80	0.74	0.80	0.76
LGBM	0.78	0.67	0.78	0.77
XGB	0.78	0.65	0.78	0.77
Stacked Model	0.81	0.77	0.81	0.79

4.3.2 Diet Plan Recommendation Model

Training Process

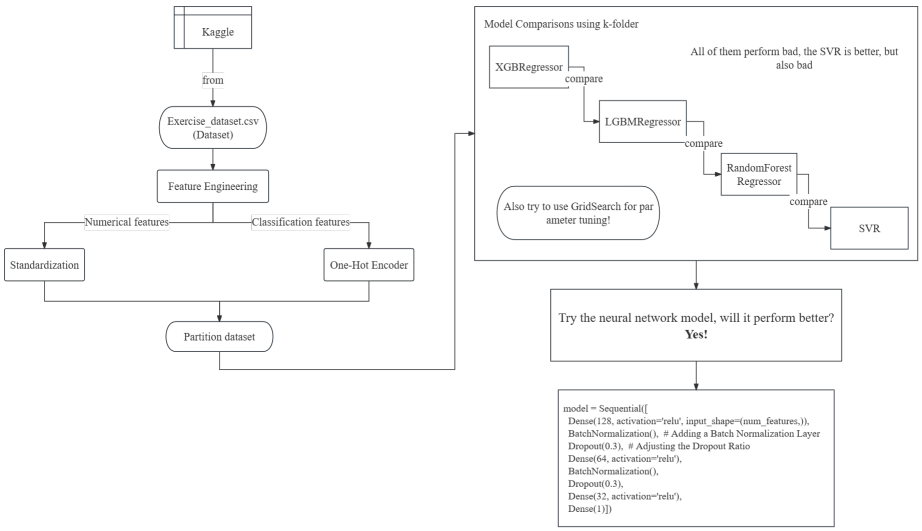

Figure 4.32: Diet Model

A. Data Selection

Dataset called "Calories Dataset" containing calories and exercise related data was downloaded from Kaggle[8]. The dataset from Kaggle consisted of 15000 instances and 9 attributes. Two attributes, user id and body temperature, were removed, leaving the dataset with only eight attributes.Table 4.21 shows the description of the 7 attributes in the dataset.

Table 4.21: Attributes Description

Attribute	Description
Gender	The gender of the respondent
Age	The age of the respondent
Height	The height of the respondent
Weight	The weight of the respondent
Duration	The amount of time the respondent exercised
Heart_Rate	The average heart rate of the respondent during exercise
Calories	Calories burned by respondents exercising

B. Data preprocessing

1. **Feature Scaling:** Using StandardScaler(), these continuous features including Age, Height, Weight, Duration, Heart Rate, Body Temp, and Calories were standardized. The standardization process involves subtracting the mean and dividing by the standard deviation of each feature, scaling the values of the features to have a mean of 0 and a standard deviation of 1. The purpose is to eliminate the scale effects among different features, ensuring that the training of the model is not dominated by certain feature scales.
2. **Categorical Encoding:** The gender feature (Gender) in the dataset was transformed using encoding, with "male" encoded as 0 and "female" encoded as 1. This conversion transforms categorical variables into numerical variables, enabling better understanding and processing by machine learning models.

C. Neural network model

1. **Model Architecture:**
   - The model is constructed as a Sequential model, consisting of multiple layers of neurons arranged sequentially.
   - The input layer is a Dense layer with 128 neurons and ReLU activation function, accepting the input features.
   - BatchNormalization layers are added to improve stability and convergence speed.
   - Dropout layers are included to mitigate overfitting by randomly dropping out a fraction of neurons during training.
   - Two intermediate Dense layers with 64 and 32 neurons, respectively, are employed with ReLU activation function.
   - The output layer is a Dense layer without activation function, suitable for regression tasks.
2. **Model Compilation:**
   - The model is compiled using the Adam optimizer with a learning rate of 0.001.
   - Mean Squared Error (MSE) is chosen as the loss function.
   - Mean Absolute Error (MAE) and Mean Squared Error (MSE) are selected as evaluation metrics.
3. **Early Stopping:** EarlyStopping callback is employed to monitor validation loss, halting training if no improvement is observed for 10 consecutive epochs, and restoring the best weights.
4. **Model Training:**
   - The model is trained using the fit() function with scaled training data and validation split of 20%, a batch size of 32, and a maximum of 100 epochs.
5. **Model Prediction and Evaluation:**
   - The trained model is utilized to predict target values on the test set.
   - Evaluation metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and R^2 score are computed.

- **Evaluation Metrics for NN**

Table 4.22: Evaluation Metrics for NN

Model	MSE	RMSE	MAE	R^2 score
NN	0.00217	0.04656	0.035257	0.9979

4.4 Explainable Artificial Intelligence (XAI)

XAI is dedicated to enhancing the transparency and explainability of traditional AI models. These traditional models are often seen as "black boxes" due to their complex internal decision-making processes. XAI aims to design and implement AI systems whose decisions are not only efficient but also understandable and explainable to users, thereby increasing their trust and acceptance of AI systems.

In this project, the XAI objective is achieved using the important tool Tree SHAP (Shapley Additive exPlanations) [9], which is based on the Shapley values from game theory.The SHAP technique quantifies the average contribution of each input feature to the model's prediction results by considering all possible combinations of features. In this way, the SHAP value of each feature provides a clear indication of how it affects the final prediction outcome, either positively or negatively.

4.4.1 Case Studes via SHAP Force Plot

The internal decision-making principle of the model is explained here by applying the SHAP force plot to analyze the process of how a particular user gets the predicted result through the model's decision-making process.

SHAP force plots used to interpret the contributions of different features to individual prediction results in the three-class fitness level prediction model. SHAP values quantify the actual impact of each feature on the predicted output, where red indicates that a feature is pushing the prediction towards a category, and blue indicates that a feature is pulling the prediction away from that category.

- **Predicted Category:** Unfit
- **Key Features:**
  - **Age (Age = 3.0):** The user is between the ages of 30 and 40, which means that he may be facing more life and career pressures, making it difficult to maintain an active fitness routine.
  - **Health Level (Health_level = 3.0):** The lower the health level of the user, the lower the predicted fitness level, reflecting the direct impact of health status on fitness activity.
  - **Time Spent (Time_spent = 0.0):** User do not take the time to participate in fitness activities, suggesting that participation is low and is a significant factor in lower fitness level.
  - **Prevents Balanced (Lack of time):** This user lacks time for a healthy diet, which leads to a lower fitness level.

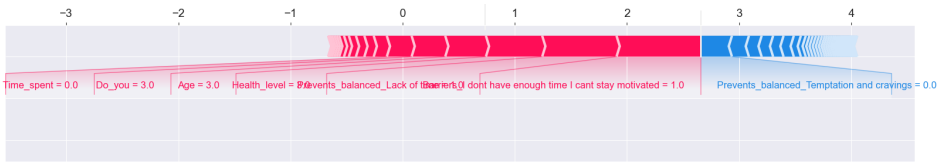

Figure 4.33: SHAP force plot - Unfit

- **Predicted Category:** Average
- **Key Feature:**
  - **Fitness is recommended (Recommend_fitness = 1):** The user recommends her friends to also do a fitness routine, which means that the user herself may be a fitness enthusiast and doing a planned fitness routine, and therefore it is judged that the user may have a better level of fitness.
  - **Wear equipment (Equipment = 1.0):** Users wear equipment for fitness, which indicates a preference or knowledge of fitness, which pushes the user to a higher level of fitness.
  - **Female (Gender = 0):** The user is female, and all female groups are likely to have better fitness levels.
  - **Exercise Importance (Exercise_importance = 4.0):** The user has a high evaluation of exercise, which indicates that the user recognizes the importance of exercise, but other constraints (e.g., time or resources) may prevent she from reaching a higher level of fitness, so it is judged to be a average fitness level.
  - **1 to 2 times a week (Regularity = 2.0):** The user engages in regular fitness, but not very frequently. This contributes to the fitness level of the user, but is limited.
  - **Health Level (Health_level = 2.0):** The user perceives her current fitness level to be below average, which will somewhat discourage the user from reaching higher level of fitness.
  - **Unbalanced Diet (Balanced_diet = 2.0):** The user does not follow a balanced diet, which will somewhat prevent the user from reaching high level of fitness

Figure 4.34: SHAP force plot - Average

- **Predicted Category:** Fit
- **Key Features:**
  - **Wear equipment (Equipment = 1.0):** Users wear equipment for fitness, which indicates a preference or knowledge of fitness, which pushes the user to a higher level of fitness.
  - **Female (Gender = 0):** The user is female, and all female groups are likely to have better fitness levels.

- **Age (Age = 3.0):** Combined with the user being a female, it can be inferred that the female population in the age group of 30-40 years is more dedicated to their fitness consciousness, and under the influence of this consciousness, the user may spend more effort on fitness, which may push them to a high level of fitness (Fit).
- **Exercise Importance (Exercise_importance = 4.0):** The user has a high evaluation of exercise, which indicates that the user recognizes the importance of exercise, but other constraints (e.g., time or resources) may prevent she from reaching a higher level of fitness, so it is judged to be a average fitness level.
- **1 to 2 times a week (Regularity = 2.0):** The user engages in regular fitness, but not very frequently. This contributes to the fitness level of the user, but is limited.
- **Health Level (Health_level = 2.0):** The user perceives her current fitness level to be below average, which will somewhat discourage the user from reaching higher level of fitness.
- **Unbalanced Diet (Balanced_diet = 2.0):** The user does not follow a balanced diet, which will somewhat prevent the user from reaching high level of fitness

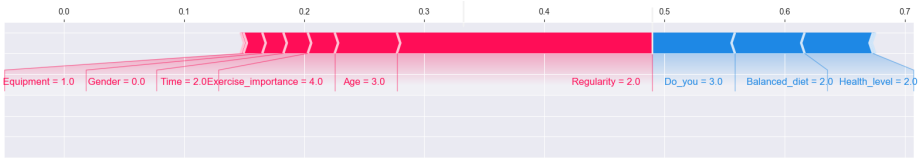

Figure 4.35: SHAP force plot - Fit

4.4.2 Global View via SHAP Summary Plot

The SHAP summary plot 4.36 shows the average impact (as measured by the SHAP value) of different features on the three-category fitness level prediction model. Each row in the plot represents a feature, each color represents a category (0: Average, 1: Unifit, 2: Fit), and the length of the color bar indicates the average magnitude of the feature's impact on the model output.

Feature Impacts

- **Health Level:** Shows significant influence across all three classes, especially prominent in Class 2 (Fit), indicating health level as a critical determinant in achieving higher fitness levels.
- **Regularity:** Strong impact on Class 2, underscoring regular exercise as crucial for high fitness.
- **Exercise Importance:** Major positive impact on Class 2, highlighting the importance of valuing physical activity for better fitness outcomes.
- **Age and Time Spent:** Significant impacts on Classes 1 and 2, suggesting that age and dedicated exercise time are key factors in fitness attainment.
- **Balanced Diet and Social Exercise Context (Do_you):** Notable effects on Classes 1 and 2, emphasizing the role of diet and exercising with others in influencing fitness levels.
- **Less Influential Features:** Time, Gender, Equipment, and Fitness Recommendation show smaller impacts, affecting certain classes slightly.

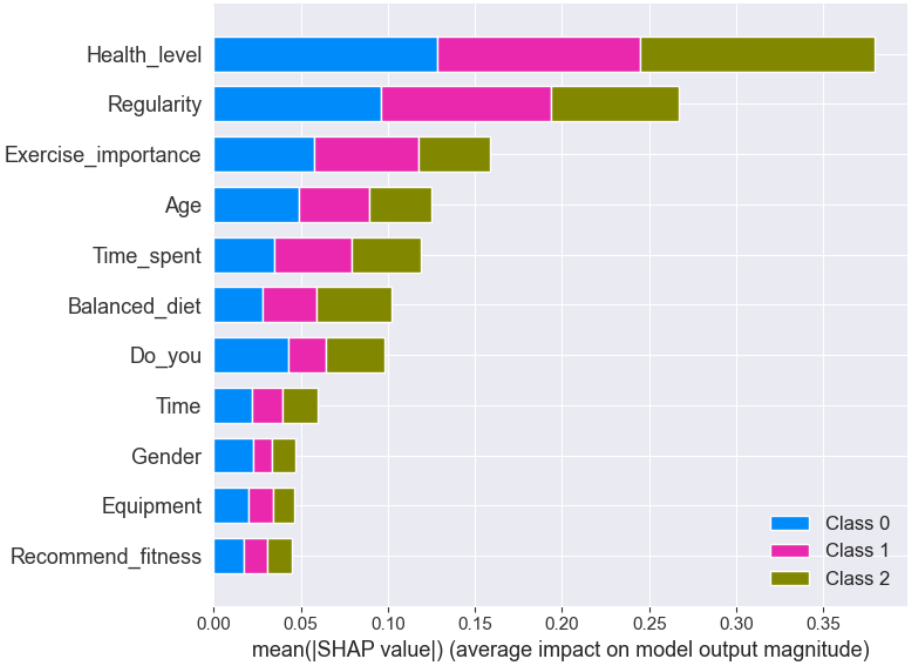

Figure 4.36: SHAP Summary Plot

4.5 Docker Deployment

4.5.1 Docker

Docker is an open source containerization platform that allows developers to package an application and its runtime environment into a lightweight, portable container. A container is a standardized unit of software that encapsulates an application and all of its dependencies, ensuring that the application runs consistently in any Docker-enabled environment.

- **Images:** A Docker image is a static snapshot of the application and its dependencies.
- **Containers:** A runtime instance of a Docker image.
- **Repositories:** Places where Docker images are stored. Docker Hub is the most popular public repository.

4.5.2 Docker Compose

Docker Compose is a tool for defining and running multi-container Docker applications. Through a configuration file (named docker-compose.yml), users can define a set of associated application containers to be started and managed as a single unit of service.Then create and start all the services from your configuration with a single command.

Core Features

- **Service Configuration:** Define services, networks, and volumes in a docker-compose.yml file.
- **Multi-service Management:** Simplifies the process of starting, stopping, and rebuilding services.
- **Development Environment Consistency:** Ensures consistency across development, testing, and production environments.

4.5.3 Benefits of deploying with Docker and Docker Compose

- **Environment Consistency:** Applications within containers will always run under the same operating system configuration and library version, This ensures consistency from development to testing to production. Performance is consistent regardless of the environment in which it is run.
- **Efficient resource utilization:** Docker containers share resources with the rest of the system, rather than requiring a fixed number of resources to be allocated, as is the case with virtual machines.
- **Isolation:** Different applications can run in isolated containers on the same physical machine without interfering with each other, increasing security.

Appendix A: Related Code

A.1 Confusion Matrix

```
def plot_confusion_matrix(y_true, y_pred, classes,
filename='confusion_matrix.png', title='Confusion_Matrix_for'):
    cm = confusion_matrix(y_true, y_pred)
    plt.figure(figsize=(6, 4))
    sns.heatmap(cm, annot=True, fmt="d", cmap='Blues',
    xticklabels=classes, yticklabels=classes)
    plt.xlabel('Predicted Labels')
    plt.ylabel('True Labels')
    plt.title(title, size = 10)
    plt.savefig(filename)
    plt.show()
```

Listing A.1: Confusion Matrix

A.2 Evaluation Metrics

```
def evaluate_classifier(model, X, y, n_splits=10):
    scoring = {
        'accuracy': make_scorer(accuracy_score),
        'precision': make_scorer(precision_score, average='macro
    '),
        'recall': make_scorer(recall_score, average='weighted'),
        'f1_score': make_scorer(f1_score, average='weighted')
    }

    kf = KFold(n_splits=n_splits, shuffle=True, random_state=42)

    cv_results = cross_validate(model, X, y, cv=kf, scoring=
   scoring)

    avg_accuracy = np.mean(cv_results['test_accuracy'])
    avg_precision = np.mean(cv_results['test_precision'])
    avg_recall = np.mean(cv_results['test_recall'])
    avg_f1_score = np.mean(cv_results['test_f1_score'])

    return avg_accuracy, avg_precision, avg_recall, avg_f1_score
```

Listing A.2: Evaluation Metrics

A.3 ROC Curve (One-vs-All)

```
def plot_multiclass_roc(y_true, y_scores, classes, filename=''):
    y_true = label_binarize(y_true, classes=[0, 1, 2])
    n_classes = y_true.shape[1]

    fpr = dict()
    tpr = dict()
    roc_auc = dict()
    for i in range(n_classes):
        fpr[i], tpr[i], _ = roc_curve(y_true[:, i], y_scores[:, i])
        roc_auc[i] = auc(fpr[i], tpr[i])

    colors = cycle(['blue', 'red', 'green'])
    plt.figure(figsize=(7, 7))
    for i, color in zip(range(n_classes), colors):
        plt.plot(fpr[i], tpr[i], color=color, lw=2,
                 label='ROC curve of class {0}
                 (area ={1:0.2f})'.format(classes[i], roc_auc[i]))

    plt.plot([0, 1], [0, 1], 'k--', lw=2)
    plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.05])
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('Multi-class ROC', size = 15)
    plt.legend(loc="lower right")
    plt.savefig(filename)
    plt.show()
```

Listing A.3: ROC Curve

A.4 DT Model

```
dt_classifier = DecisionTreeClassifier(random_state=42,criterion='gini')
dt_classifier.fit(X_train, y_train)
y_pred_dt = dt_classifier.predict(X_test)
accuracy_dt = accuracy_score(y_test, y_pred_dt)
y_scores_dt = dt_classifier.predict_proba(X_test)
accuracy, precision, recall, f1 = evaluate_classifier(dt_classifier,
X, y_encoded)
print(f"Accuracy: {accuracy:.2f}, Precision: {precision:.2f},
Recall: {recall:.2f}, F1-Score: {f1:.2f}")
plot_confusion_matrix(y_test, y_pred_dt, classes=label_encoder.classes_,
title='Confusion Matrix for Decision Tree')
plot_multiclass_roc(y_test, y_scores_dt, label_encoder.classes_,
```

```
filename='dt_roc_curve.png')
```

Listing A.4: Decision Tree Model

A.5 LightGBM Model

```
lgbm_classifier = LGBMClassifier(random_state=42)
lgbm_classifier.fit(X_train, y_train)
y_pred_lgbm = lgbm_classifier.predict(X_test)
accuracy_lgbm = accuracy_score(y_test, y_pred_lgbm)
y_scores_lgbm = lgbm_classifier.predict_proba(X_test)
accuracy, precision, recall, f1 = evaluate_classifier
(lgbm_classifier, X, y_encoded)
plot_confusion_matrix(y_test, y_pred_lgbm,
classes=label_encoder.classes_, title='Confusion Matrix for
    LightGBM')
plot_multiclass_roc(y_test, y_scores_lgbm,
label_encoder.classes_, filename='lgbm_roc_curve.png')
```

Listing A.5: LightGBM Model

A.6 KNN Model

```
knn_classifier = KNeighborsClassifier(n_neighbors=15)
knn_classifier.fit(X_train, y_train)
y_pred_knn = knn_classifier.predict(X_test)
accuracy_knn = accuracy_score(y_test, y_pred_knn)
y_scores_knn = knn_classifier.predict_proba(X_test)
accuracy, precision, recall, f1 =
evaluate_classifier(knn_classifier, X, y_encoded)
plot_confusion_matrix(y_test, y_pred_knn,
classes=label_encoder.classes_, title='Confusion Matrix for KNN'
    )
plot_multiclass_roc(y_test, y_scores_knn,
label_encoder.classes_, filename='knn_roc_curve.png')
```

Listing A.6: KNN Model

A.7 RF Model

```
rf_classifier = RandomForestClassifier(random_state=42,criterion
    ='gini')
rf_classifier.fit(X_train, y_train)
y_pred_rf = rf_classifier.predict(X_test)
accuracy_rf = accuracy_score(y_test, y_pred_rf)
```

```
y_scores_rf = rf_classifier.predict_proba(X_test)
accuracy, precision, recall, f1 =
evaluate_classifier(rf_classifier, X, y_encoded)
plot_confusion_matrix(y_test, y_pred_rf, classes=label_encoder.
    classes_,
title='Confusion Matrix for Random Forest')
plot_multiclass_roc(y_test, y_scores_rf,
label_encoder.classes_, filename='rf_roc_curve.png')
```

Listing A.7: Random Forest Model

A.8 SVM Model

```
svm_classifier = SVC(probability=True,random_state=42, kernel='
    rbf')
svm_classifier.fit(X_train, y_train)
y_pred_svm = svm_classifier.predict(X_test)
accuracy_svm = accuracy_score(y_test, y_pred_svm)
y_scores_svm = svm_classifier.predict_proba(X_test)
accuracy, precision, recall, f1 =
evaluate_classifier(svm_classifier, X, y_encoded)
plot_confusion_matrix(y_test, y_pred_svm,
classes=label_encoder.classes_, title='Confusion Matrix for SVM'
    )
plot_multiclass_roc(y_test, y_scores_svm,
label_encoder.classes_, filename='svm_roc_curve.png')
```

Listing A.8: SVM Model

A.9 XGBoost Model

```
xgb_classifier = XGBClassifier(use_label_encoder=False,
eval_metric='mlogloss', random_state=42)
xgb_classifier.fit(X_train, y_train)
y_pred_xgb = xgb_classifier.predict(X_test)
accuracy_xgb = accuracy_score(y_test, y_pred_xgb)
y_scores_xgb = xgb_classifier.predict_proba(X_test)
accuracy, precision, recall, f1 = evaluate_classifier(
    xgb_classifier, X, y_encoded)
plot_confusion_matrix(y_test, y_pred_xgb, classes=label_encoder.
    classes_, title='Confusion Matrix for XGBoost')
plot_multiclass_roc(y_test, y_scores_xgb, label_encoder.classes_
    , filename='xgb_roc_curve.png')
```

Listing A.9: XGBoost Model

A.10 Stacked Model

```
base_models = [
    ('knn', KNeighborsClassifier(n_neighbors=7)),
    ('random_forest', RandomForestClassifier
    (max_depth=15, random_state=42)),
    ('svm', SVC(C=10, gamma='scale', probability=True,
    random_state=42)),
    ('lgbm', LGBMClassifier(num_leaves=63, max_depth=5,
    random_state=42)),
    ('xgb', XGBClassifier(max_depth=5, learning_rate=0.05,
    eval_metric='logloss',
    random_state=42))
]
meta_model = LogisticRegression(C=1, penalty='l2', random_state
   =42)
cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
stacking_model = StackingClassifier(estimators=base_models,
   final_estimator=meta_model, cv=cv)
stacking_model.fit(X_train, y_train)
y_pred = stacking_model.predict(X_test)
```

Listing A.10: Stacked Model

A.11 Neural Network Model

```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
num_features = X_train_scaled.shape[1]
model = Sequential([
    Dense(128, activation='relu', input_shape=(num_features,)),
    BatchNormalization(),
    Normalization Layer
    Dropout(0.3),
    Dense(64, activation='relu'),
    BatchNormalization(),
    Dropout(0.3),
    Dense(32, activation='relu'),
    Dense(1)
])
model.compile(optimizer=tf.keras.optimizers.Ad
am(learning_rate=0.001),
              loss='mean_squared_error',
              metrics=['mae', 'mse'])
early_stopping = tf.keras.callbacks.EarlyStopping(monitor='val_
loss', patience=10, restore_best_weights=True)
history = model.fit(X_train_scaled, y_train, validation_split
   =0.2,
epochs=100, batch_size=32, callbacks=[early_stopping])
y_pred_nn = model.predict(X_test_scaled)
```

```
mse_nn = mean_squared_error(y_test, y_pred_nn)
rmse_nn = mean_squared_error(y_test, y_pred_nn, squared=False)
mae_nn = mean_absolute_error(y_test, y_pred_nn)
r2_nn = r2_score(y_test, y_pred_nn)
```

Listing A.11: Neural Network Model

A.12 THREE.AnimationMixer

```
const scene = new THREE.Scene();
const camera = new THREE.PerspectiveCamera(75, window.innerWidth
    / window.innerHeight, 0.1, 1000);
const renderer = new THREE.WebGLRenderer();
renderer.setSize(window.innerWidth, window.innerHeight);
document.body.appendChild(renderer.domElement);

const loader = new THREE.GLTFLoader();
loader.load('path/to/model.gltf', (gltf) => {
  const model = gltf.scene;
  const animations = gltf.animations;

  const mixer = new THREE.AnimationMixer(model);
  clips.forEach((clip) => {
    const action = mixer.clipAction(clip);
    action.play();
  });

  scene.add(model);

  const clock = new THREE.Clock();
  function animate() {
    requestAnimationFrame(animate);
    const delta = clock.getDelta();
    mixer.update(delta);
    renderer.render(scene, camera);
  }
  animate();
});
```

Listing A.12: THREE.AnimationMixer

A.13 3D "About Us" Cinema

```
const scene = new THREE.Scene();

const ambientLight = new THREE.AmbientLight(0x404040);
scene.add(ambientLight);
const directionalLight = new THREE.DirectionalLight(0xffffff);
```

```
directionalLight.position.set(1, 1, 1);
scene.add(directionalLight);

const screenGeometry = new THREE.PlaneGeometry(16, 9);
const screenMaterial = new THREE.MeshBasicMaterial();
const screen = new THREE.Mesh(screenGeometry, screenMaterial);
screen.position.set(0, 2, -5);
scene.add(screen);

const video = document.createElement('video');
video.src = 'static/healthharmony-video.mp4';
video.loop = true;
video.play();

const videoTexture = new THREE.VideoTexture(video);
screenMaterial.map = videoTexture;

const camera = new THREE.PerspectiveCamera(75, window.innerWidth
    / window.innerHeight, 0.1, 1000);
camera.position.z = 10;

const renderer = new THREE.WebGLRenderer();
renderer.setSize(window.innerWidth, window.innerHeight);
document.body.appendChild(renderer.domElement);

function animate() {
  requestAnimationFrame(animate);
  renderer.render(scene, camera);
}
animate();
```

Listing A.13: 3D "About Us" Cinema

A.14 THREE.DRACOLoader

```
import * as THREE from 'three';
import { DRACOLoader } from 'three/examples/jsm/loaders/
   DRACOLoader';
import { GLTFLoader } from 'three/examples/jsm/loaders/
   GLTFLoader';

const scene = new THREE.Scene();
const camera = new THREE.PerspectiveCamera(75, window.innerWidth
    / window.innerHeight, 0.1, 1000);
const renderer = new THREE.WebGLRenderer();
renderer.setSize(window.innerWidth, window.innerHeight);
document.body.appendChild(renderer.domElement);

const dracoLoader = new DRACOLoader();
dracoLoader.setDecoderPath('path/to/draco/');
dracoLoader.setDecoderConfig({ type: 'js' });
```

```

const loader = new GLTFLoader();
loader.setDRACOLoader(dracoLoader);

loader.load('path/to/model.gltf', (gltf) => {
  const model = gltf.scene;
  scene.add(model);

  camera.position.z = 5;
  function animate() {
    requestAnimationFrame(animate);
    renderer.render(scene, camera);
  }
  animate();
});
```

Listing A.14: THREE.DRACOLoader

A.15 Time Interval

```
function checkCollisions() {
  const currentTime = performance.now() / 1000;

  if (currentTime - player.lastJumpTime < player.jumpCooldown) {
    return;
  }

  for (const box of collisionBoxes) {
    if (
      player.position.x > box.position.x - box.size.x / 2 &&
      player.position.x < box.position.x + box.size.x / 2 &&
      player.position.y > box.position.y - box.size.y / 2 &&
      player.position.y < box.position.y + box.size.y / 2
    ) {
      player.position.y = box.position.y + box.size.y / 2 + 0.1;
      player.velocity.y = 0;
      player.lastJumpTime = currentTime;
      break;
    }
  }
}


function gameLoop(delta) {
  updatePlayer(delta);
  requestAnimationFrame(gameLoop);
}
```

Listing A.15: Time Interval

A.16 Player Movement Update

```
const playerSpeed = 5;

function controlPlayer(player) {
  document.addEventListener('keydown', (event) => {
    switch (event.key) {
      case 'w':
        player.translateZ(-playerSpeed);
        break;
      case 's':
        player.translateZ(playerSpeed);
        break;
      case 'a':
        player.translateX(-playerSpeed);
        break;
      case 'd':
        player.translateX(playerSpeed);
        break;
    }
  });

  function animate() {
    requestAnimationFrame(animate);
    renderer.render(scene, camera);
  }
  animate();
}
```

Listing A.16: Player Movement Update

A.17 Connect 3D World With 2D Interface

```

function updatePlayer(dt) {
  const clickableGeometry = new THREE.BoxGeometry(1, 1, 1);
  const clickableMaterial = new THREE.MeshBasicMaterial({ color:
    0xff0000 });
  const clickableBlock = new THREE.Mesh(clickableGeometry,
   clickableMaterial);
  clickableBlock.position.set(-52, 1, -10);
  scene.add(clickableBlock);

  const clickableBlockPos = clickableBlock.position.clone()

  const distance1 = pos.distanceTo(clickableBlockPos)

  if (distance1 < 1.3 ) {
    window.location.href = 'http://127.0.0.1:5000/fit'
  }
```

```
}
```

Listing A.17: Connect 3D World With 2D Interface

A.18 Import GLTF Scene Model

```
import * as THREE from 'three';
import { GLTFLoader } from 'three/examples/jsm/loaders/
    GLTFLoader';


const scene = new THREE.Scene();
const camera = new THREE.PerspectiveCamera(75, window.innerWidth
    / window.innerHeight, 0.1, 1000);
const renderer = new THREE.WebGLRenderer();
renderer.setSize(window.innerWidth, window.innerHeight);
document.body.appendChild(renderer.domElement);


const loader = new GLTFLoader();
loader.load(
  'path/to/model.gltf',
  (gltf) => {
    const model = gltf.scene;
    scene.add(model);

    model.position.set(0, 0, 0);
    model.rotation.set(0, 0, 0);


    function animate() {
      requestAnimationFrame(animate);
      model.rotation.y += 0.01;
      renderer.render(scene, camera);
    }
    animate();
  },
  (progress) => {
    console.log(`Loading file: ${(progress.loaded / progress.
    total * 100)}% loaded`);
  },
  (error) => {
    console.error('An error happened', error);
  }
);
```

Listing A.18: Import GLTF Scene Model

A.19 Docker Deployment

```
FROM python:3.9
WORKDIR /app
COPY . /app
RUN pip install --upgrade pip
RUN pip install --no-cache-dir -r requirements.txt
EXPOSE 5000
CMD ["python", "app.py"]
```

Listing A.19: DockerFile

Bibliography

1. *Explainable Artificial Intelligence. First World Conference, xAI 2023, Lisbon, Portugal, July 26–28, 2023, Proceedings, Part I* (ed Longo, L.) XXV, 692. ISBN: 978-3-031-44063-2 (Springer Cham, Oct. 2023).
2. European Parliament and the Council of the European Union. *Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation)* Accessed: 2024-05-16. 2016. https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679&from=EN.
3. European Parliament and the Council of the European Union. *Regulation (EU) 2017/745 of the European Parliament and of the Council of 5 April 2017 on medical devices, amending Directive 2001/83/EC, Regulation (EC) No 178/2002 and Regulation (EC) No 1223/2009 and repealing Council Directives 90/385/EEC and 93/42/EEC* Accessed: 2024-05-16. 2017. https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32017R0745&from=EN.
4. U.S. Congress. *Health Insurance Portability and Accountability Act of 1996 (HIPAA)* Public Law 104-191, Accessed: 2024-05-16. 1996. https://www.congress.gov/104/plaws/publ191/PLAW-104publ191.pdf.
5. U.S. Food and Drug Administration. *Artificial Intelligence and FDA: Publications* Accessed: 2024-05-16. 2024. https://www.fda.gov/science-research/artificial-intelligence-and-medical-products/artificial-intelligence-fda-publications.
6. Chua, S., Sii, C. I. & Nohuddin, P. N. E. *Comparative Analysis of Machine Learning Models for Fitness Level Prediction with Imbalanced Dataset* in *2022 International Conference on Digital Transformation and Intelligence (ICDI)* (2022).
7. Chawla, N. V., Bowyer, K. W., Hall, L. O. & Kegelmeyer, W. P. SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research* **16,** 321–357 (2002).
8. Figo. *Calories Dataset* Retrieved from https://www.kaggle.com/datasets/figolm10/calories-dataset?select=calories.csv. 2023.
9. McFall, G. P. *et al.* Identifying key multi-modal predictors of incipient dementia in Parkinson's disease: a machine learning analysis and Tree SHAP interpretation. *Frontiers in Aging Neuroscience* **15.** This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY)., 1124232. https://doi.org/10.3389/fnagi.2023.1124232 (2023).