# Web Programming
# Course Code: BCSE203E
# Mini Project
# Golds Gym

Suryanshu, 22BCE0820

April 19, 2024

# *Contents*

*Mini Project Documentation*

---

# *Golds Gym*

---

## 1.1 Abstract

"Golds Gym" is a React-based web application designed to elevate users' fitness experiences through personalized exercise recommendations and seamless integration with Golds Gym services.

Leveraging Rapid API integration, this platform offers users access to a vast database of exercises tailored to specific body parts, curated by Golds Gym's renowned fitness experts.

With an intuitive user interface, individuals can easily discover recommended exercises, view detailed instructions, and track their progress over time. Golds Gym Online prioritizes accessibility and convenience, ensuring a seamless user experience across various devices. Hosted on Netlify, this web application exemplifies Golds Gym's commitment to innovation in the digital fitness space, providing users with a comprehensive platform to achieve their health and fitness goals effectively.

Golds Gym Online represents a significant milestone in Golds Gym's digital transformation journey, empowering users to optimize their workouts and embark on a path to improved physical well-being.

## 1.2 Introduction

In today's fast-paced world, the integration of technology into various aspects of our lives has become increasingly prevalent. In the realm of fitness and health,

the utilization of web-based applications has emerged as a powerful tool for individuals seeking to optimize their workout routines and achieve their fitness goals. This project, titled "Golds Gym", aims to address the evolving needs of fitness enthusiasts by developing a web-based platform that provides personalized exercise recommendations tailored to specific body parts.

### 1.2.1 Objectives and Goals

The primary objective of Golds Gym is to create a user-friendly and intuitive web application that offers comprehensive exercise recommendations based on individual preferences and fitness levels. Specifically, the goals of the project include:

1. **Development of a React-based Web Application:** Utilizing the React JavaScript library, Golds Gym will be developed to ensure a responsive and dynamic user interface that enhances the overall user experience.

2. **Integration of Rapid API for Exercise Recommendations:** Golds Gym will leverage the Rapid API platform to access a vast database of exercises categorized by targeted muscle groups. This integration will enable users to receive personalized exercise recommendations based on their specific fitness goals and preferences.

3. **Deployment on Netlify:** Golds Gym will be deployed on the Netlify platform to ensure seamless accessibility and scalability, allowing users to access the application from any device with an internet connection.

By achieving these objectives and goals, Golds Gym seeks to revolutionize the way individuals approach fitness and exercise, providing them with a powerful tool to optimize their workout routines, track their progress, and ultimately achieve their health and wellness goals more effectively.

## 1.3 Technologies Used

The development of this project involved the utilization of several programming languages, frameworks, and tools to create a robust and feature-rich web application. The key technologies used in this project include:

1. **React:** The project was built using React, a popular JavaScript library for building user interfaces. React's component-based architecture and virtual

DOM provided a solid foundation for creating dynamic and interactive UI components.

2. **Material-UI:** Material-UI, a React UI framework based on Google's Material Design, was extensively used to design and style the user interface of the application. Components such as Box, Typography, Stack, Button, and TextField from Material-UI were utilized to create a cohesive and visually appealing UI design.

3. **API Integration:** The project incorporated API integration to fetch and manipulate data from external sources. By leveraging APIs, the application was able to provide users with real-time data and enhance the overall user experience.

4. **useState and useEffect Hooks:** React's built-in hooks, such as useState and useEffect, were employed to manage state and side effects within functional components. These hooks enabled the application to maintain stateful data and perform asynchronous operations efficiently.

The combination of these technologies and tools facilitated the development of a modern and responsive web application that meets the project requirements and delivers an engaging user experience.

## 1.4 Project Requirements

The goal of the project is to develop a web application named "Golds Gym" that provides users with personalized exercise recommendations based on their fitness goals and preferences. The application should meet the following requirements:

1. **Exercise Recommendations:**Users should be presented with a list of exercise recommendations based on their selected body part, excersise or equipment and preferred workout intensity.

2. **Exercise Database:** The application should integrate with an external API to access a database of exercises categorized by targeted muscle groups and workout types.

3. **Responsive Design:** Golds Gym should be designed with a responsive layout to ensure optimal viewing and usability across various devices, including desktops, tablets, and smartphones.

4. **Performance Optimization:** The application should be optimized for performance, with fast load times and efficient data retrieval to provide users with a seamless and responsive experience.

5. **Deployment:** The completed application should be deployed to a hosting platform such as Netlify for public access.

## 1.5 System Design

The system design of Golds Gym encompasses three main components: Architecture overview, Database schema, and User interface design.

### 1.5.1 Architecture Overview:

Golds Gym is follows only a client side architecture model. The front end is made entirely in React with the help of Material-UI. Rapid API's exerciseDB is used to load the exercise data dynamically. The web app is hosted with Netlify.

### 1.5.2 Database Schema:

No data is saved on the server side. The content is loaded dynamically using Restful API.

### 1.5.3 User Interface Design:

Golds Gym's user interface is designed with a focus on simplicity, intuitiveness, and responsiveness. The application features a clean and modern design aesthetic, with a consistent layout and visual hierarchy. Golds Gym's user interface is designed to be responsive, ensuring optimal viewing and usability across various devices and screen sizes, including desktops, tablets, and smartphones.

## 1.6 Implementation

The components of the app are divided into three major categories based on the functionality they serve:

- **Pages:** This breaks down the website into two pages, the Home and ExcerciseDetail page.

- **Utility:** This is the fetch script used to send requests to the API and recieve data.
- **Components:** There are 12 functional components being used to make our dynamic and interactive, like Footer, Loader, NavBar, ExcerciseCard etc.

### 1.6.1 Pages:

1. **Home:** Home.js calls on three components, the HeroBanner, SearchExcercises and Exercises themselves making up the entirety of the landing page of our site. We wrap all the three components in a ¡Box¿ tag from Material UI to set the flow of the page.

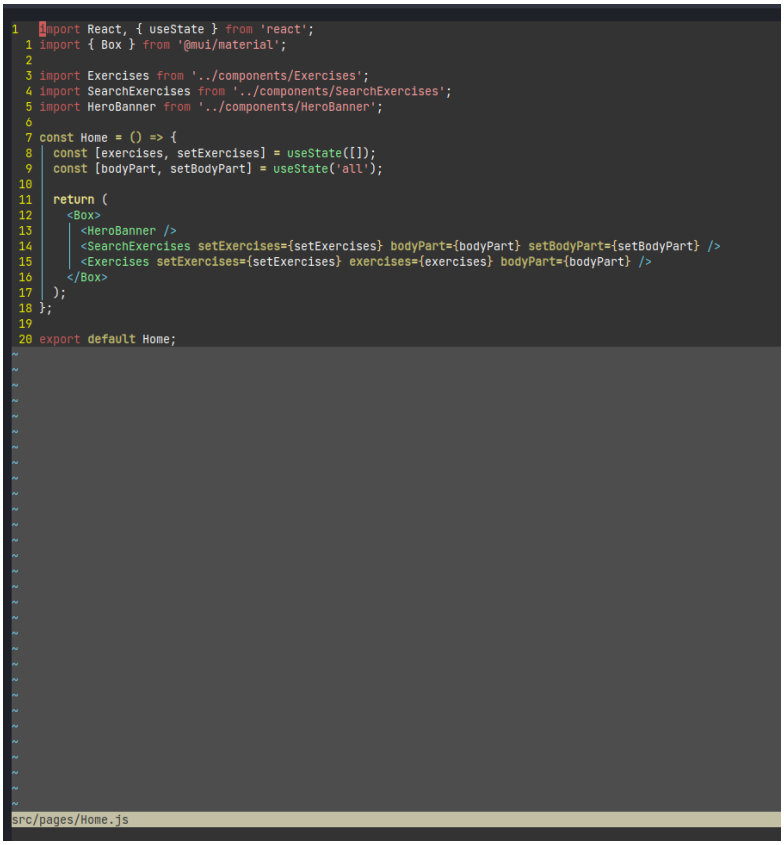


Figure 1.1: Home.js

2. **ExcerciseDetail:** ExcerciseDetail.js calls on two components, the Detail and SimilarExercises. It fetches data about exercises through the fetch function and API which we update using useEffect.

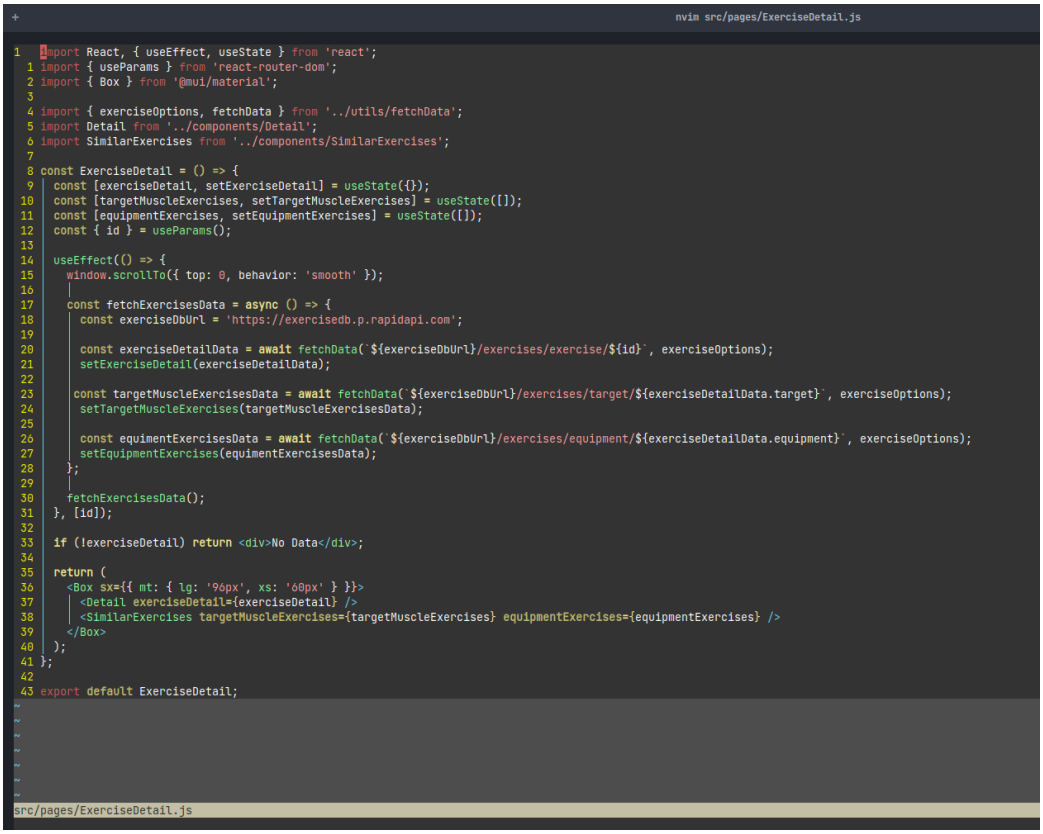


Figure 1.2: excersiseDetail.js

### 1.6.2 Utils:

**fetchData:** It is used to make the actuall asynchronous calls to fetch exercise details through api and then data is returned.

```
nvim src/utils/fetchData.js

export const exerciseOptions = {
  method: 'GET',
  url: 'https://exercisedb.p.rapidapi.com/exercises/bodyPart/back',
  params: {limit: '10'},
  headers: {
    'X-RapidAPI-Host': 'exercisedb.p.rapidapi.com',
    'X-RapidAPI-Key': process.env.REACT_APP_RAPID_API_KEY,
  },
};

export const youtubeOptions = {
  method: 'GET',
  url: 'https://youtube-search-and-download.p.rapidapi.com/channel/about',
  params: {
    id: 'UCE_M8A5yxnLfW0KghEeajjw',
  },
  headers: {
    'X-RapidAPI-Host': 'youtube-search-and-download.p.rapidapi.com',
    'X-RapidAPI-Key': process.env.REACT_APP_RAPID_API_KEY,
  },
};

export const fetchData = async (url, options) => {
  const res = await fetch(url, options);
  const data = await res.json();

  return data;
};

src/utils/fetchData.js
```

Figure 1.3: fetchData.js

### 1.6.3 Components

Since these components are the very basis of our web application, it also includes some very basic features like loading component, the navbar or the footer. Here we will only pay attention to the key components.

1. **Detail:** This component sets up a brief description of the exercise using Material-UI components.

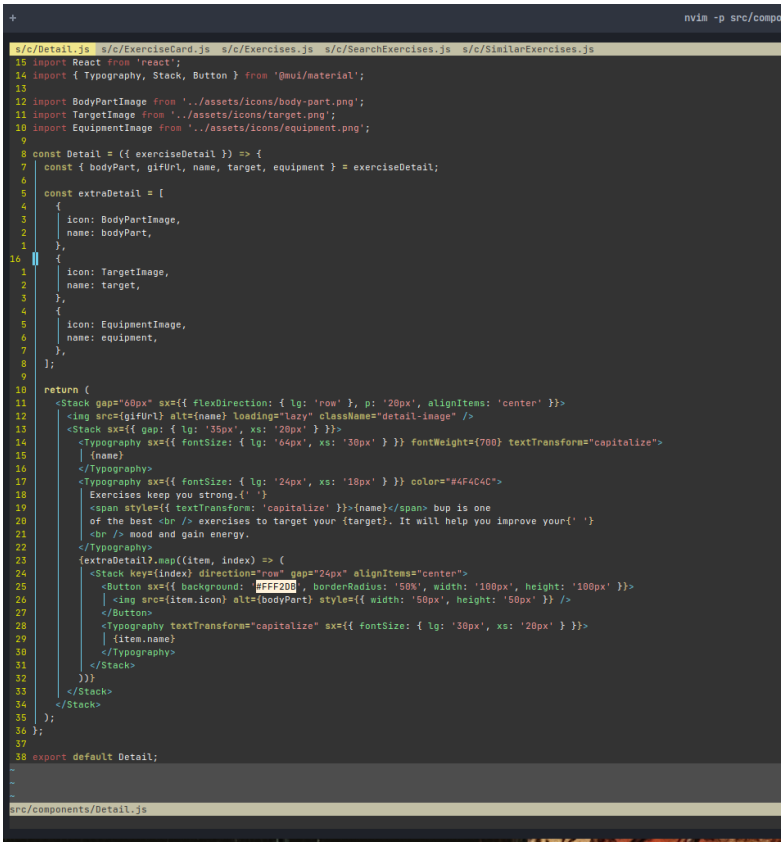


Figure 1.4: Detail.js

2. **ExerciseCard:** This component sets up the frame for three exercise components – body part, target and name.

```
nvim -p src/components/*js

s/c/Detail.js  s/c/ExerciseCard.js  s/c/Exercises.js  s/c/SearchExercises.js  s/c/SimilarExercises.js
import React from 'react';
import { Link } from 'react-router-dom';
import { Button, Stack, Typography } from '@mui/material';

const ExerciseCard = ({ exercise }) => (
  <Link className="exercise-card" to={`/exercise/${exercise.id}`}>
    <img src={exercise.gifUrl} alt={exercise.name} loading="lazy" />
    <Stack direction="row">
      <Button sx={{ ml: '21px', color: '#fff', background: '#FFA9A9', fontSize: '14px', borderRadius: '20px', textTransform: 'capitalize' }}>
        {exercise.bodyPart}
      </Button>
      <Button sx={{ ml: '21px', color: '#fff', background: '#FCC757', fontSize: '14px', borderRadius: '20px', textTransform: 'capitalize' }}>
        {exercise.target}
      </Button>
    </Stack>
    <Typography ml="21px" color="#000" fontWeight="bold" sx={{ fontSize: { lg: '24px', xs: '20px' } }} mt="11px" pb="10px" textTransform="capitalize">
      {exercise.name}
    </Typography>
  </Link>
);

export default ExerciseCard;
```

Figure 1.5: ExerciseCard.js

3. **Exercises:** Using Pagination here we construct a smooth user experience for scrolling through various exercises.

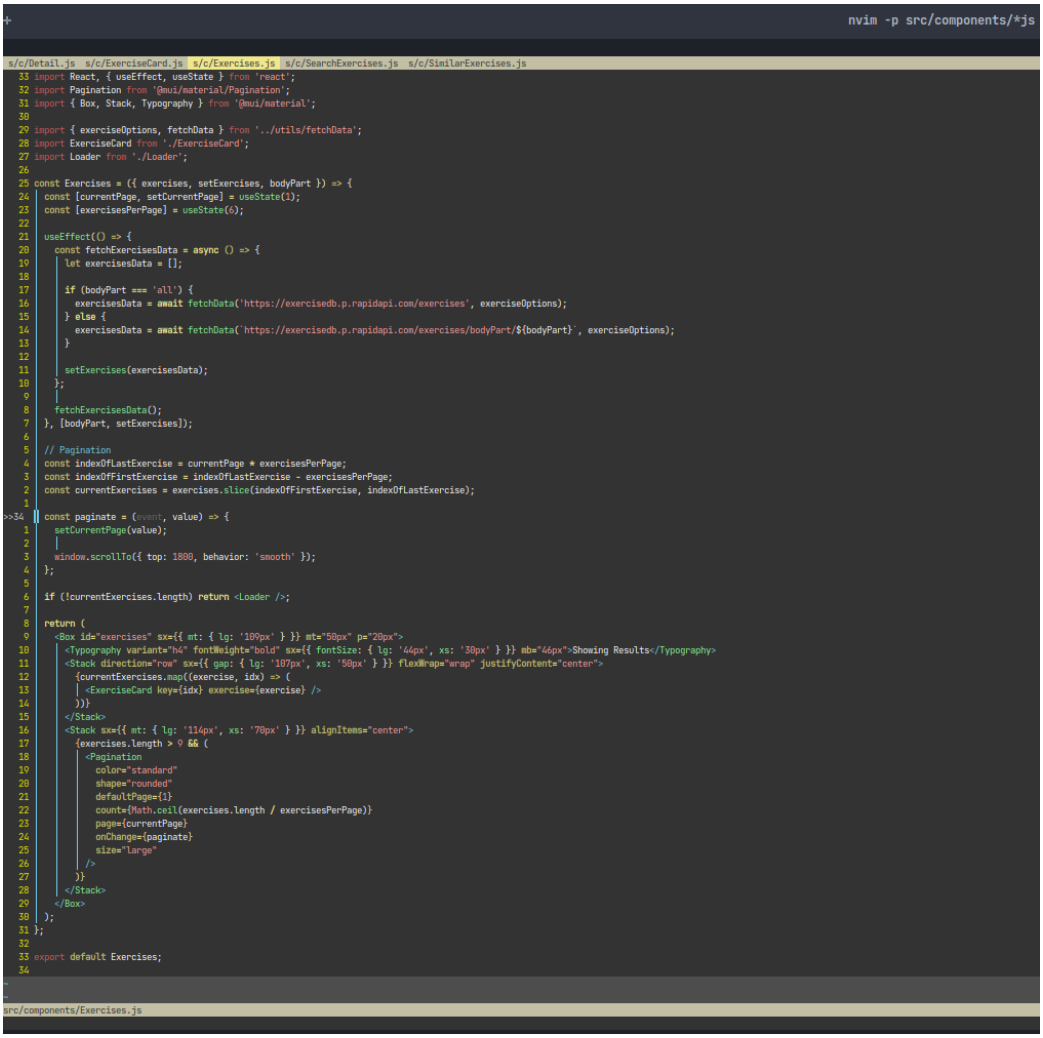


Figure 1.6: Exercises.js

4. **SearchExcercises:** This handles the search query put in by the user and returns it in a HorizontalScrollBar.


Figure 1.7: SearchExcercises.js

5. **SimilarExercises:**Finds the similar exercises and returns the result in a HorizontalScrollBar.

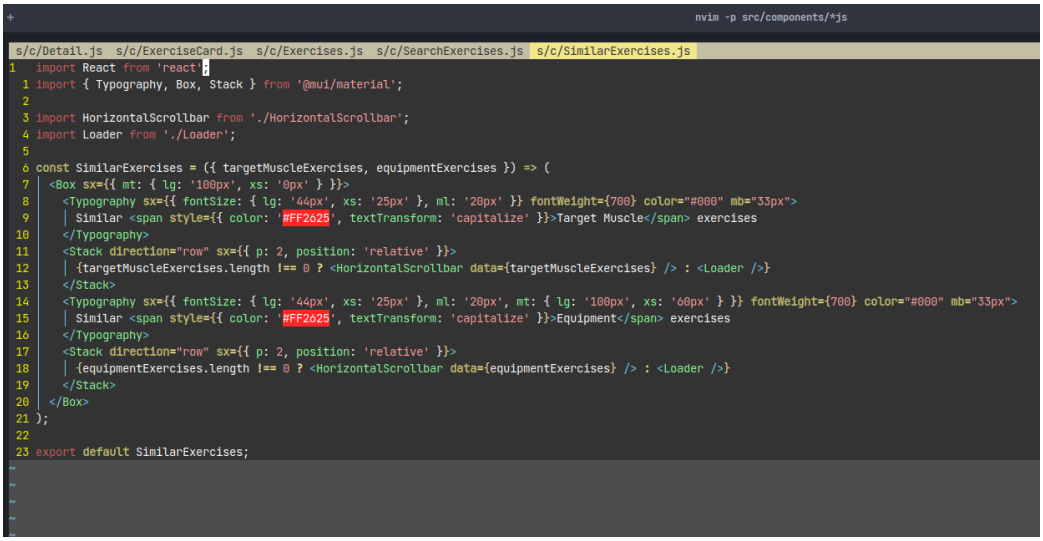


Figure 1.8: SimilarExercises.js

## 1.7 Deployment

### 1.7.1 Description of Deployment Environment

Golds Gym is deployed using Netlify, a cloud-based hosting platform that provides continuous deployment, serverless functions, and other features for hosting static websites and frontend applications. Netlify offers a simple and intuitive deployment process, allowing developers to deploy websites quickly and easily without the need for complex server configurations.

### 1.7.2 Deployment Process

1. **Build Process:** Before deploying Golds Gym, the project is built using the npm run build command. This command generates optimized production-ready assets, including HTML, CSS, JavaScript, and any other static files required for the application.

2. **Static Files:** Once the build process is complete, the generated static files are located in the build directory of the project.

3. **Deployment to Netlify:** The next step involves deploying the static files to Netlify. This can be done manually through the Netlify web interface or via continuous integration (CI) pipelines such as GitHub Actions or GitLab CI. When deploying manually, developers can drag and drop the build directory or use the Netlify CLI to deploy the project from the command line. If using CI pipelines, developers can configure automatic deployments triggered by changes to the project's repository. Netlify integrates seamlessly with popular version control systems like Git, allowing for automatic deployments whenever changes are pushed to the repository.

4. **Continuous Deployment:** Netlify supports continuous deployment, meaning that whenever changes are made to the project's repository, Netlify automatically rebuilds and deploys the updated version of the application. This ensures that the deployed application is always up-to-date with the latest changes made by developers.

By leveraging Netlify for deployment, Golds Gym benefits from a fast, reliable, and scalable hosting environment, ensuring that the application is accessible to users worldwide with minimal downtime and maximum performance.

## 1.8 Future Enhancements

1. **Integration of YouTube API for Video Recommendations:**

    - Consider integrating the YouTube API to fetch recommended YouTube videos related to fitness, exercise tutorials, and motivational content.
    - Use the YouTube API to customize video recommendations based on user preferences, workout history, and fitness goals.
    - Display recommended YouTube videos within the Golds Gym application interface, providing users with valuable additional resources for their fitness journey.

2. **Personalized Planner Using AI:**

    - Explore the possibility of implementing a personalized workout planner powered by artificial intelligence (AI) algorithms.
    - Utilize machine learning techniques to analyze user data, including fitness goals, workout preferences, and historical workout patterns.
    - Develop AI models that can generate personalized workout plans tailored to each user's individual needs, optimizing exercise selection, intensity, and frequency.
    - Integrate the personalized planner seamlessly into the Golds Gym application, providing users with dynamically generated workout schedules that adapt and evolve based on their progress and feedback.

3. **Social Features and Community Engagement:**

    - Enhance the social aspects of Golds Gym by adding features such as user profiles, social sharing, and community forums.
    - Enable users to connect with each other, share their fitness achievements, and provide support and motivation to fellow community members.

- Implement features for users to create and join workout groups, participate in challenges and competitions, and track their progress alongside friends and peers.
- Foster a sense of community and camaraderie among Golds Gym users, encouraging collaboration and mutual support in achieving their fitness goals.

4. **Advanced Analytics and Insights:**

- Expand the analytics capabilities of Golds Gym to provide users with detailed insights into their fitness progress and performance.
- Develop advanced analytics dashboards that visualize key metrics such as workout duration, intensity, calories burned, and muscle groups targeted.
- Use data visualization techniques to identify trends, patterns, and areas for improvement in users' workout routines, helping them make informed decisions and optimize their training regimen.
- Provide personalized recommendations and actionable insights based on data analysis, empowering users to make positive changes and achieve better results in their fitness journey.

These future enhancements aim to enrich the Golds Gym experience, providing users with additional features, capabilities, and insights to support their fitness journey and promote a healthier lifestyle. Each enhancement has the potential to enhance user engagement, satisfaction, and long-term adherence to fitness goals.

## 1.9 Conclusion

### 1.9.1 Summary of Achievements:

Throughout the development of Golds Gym, significant accomplishments have been achieved, resulting in a robust and user-friendly fitness application. Key achievements include:

- Successful implementation of personalized exercise recommendations based on user preferences and fitness goals.
- Integration with external APIs, such as Rapid API, for accessing exercise databases and enriching user experience.
- Seamless deployment on Netlify, ensuring accessibility and scalability of the application.
- Creation of a modern and responsive user interface using React and Material-UI components.
- Implementation of user authentication and profile management functionalities to enhance security and user engagement.

### 1.9.2 Lessons Learned:

The development of Golds Gym has been a valuable learning experience, providing insights into various aspects of software development and project management. Key lessons learned include:

- Importance of thorough planning and requirement analysis to ensure clarity and alignment with project goals.
- Effective use of version control systems and deployment tools to streamline development workflows and ensure code integrity.
- Significance of user feedback and iterative development processes in refining features and enhancing user experience.
- Continuous learning and adaptation to new technologies and best practices to stay current and deliver high-quality software solutions.

### 1.9.3 Future Outlook:

As Golds Gym continues to evolve, there is immense potential for further enhancements and expansion. Future plans include:

- Integration of additional features, such as YouTube API for video recommendations and AI-powered personalized planners.
- Enhancements to social features and community engagement to foster a supportive and interactive fitness community.
- Implementation of advanced analytics and insights to provide users with deeper understanding and optimization of their fitness journey.
- Collaboration with wearable devices and health trackers to offer seamless integration and real-time monitoring of fitness activities.

In conclusion, the development of Golds Gym has been a rewarding journey, marked by significant achievements and valuable lessons. As we look towards the future, we remain committed to delivering innovative solutions that empower users to lead healthier and more active lifestyles.

## 1.10 References

1. Github Link for Golds Gym:

   `https://github.com/suryanshu-09/gym_web_app`

2. Golds Gym Website Application link:

   `https://662232882253cea00e88c0af--musical-nougat-ce70d3.netlify.app`

3. React Documentation. Retrieved from https://reactjs.org/docs/getting-started.html
4. Material-UI Documentation. Retrieved from https://material-ui.com/getting-started/installation/
5. Rapid API Documentation. Retrieved from

   `https://rapidapi.com/justin-WFnsXH_t6/api/exercisedb`

6. Netlify Documentation. Retrieved from https://docs.netlify.com/