MINI PROJECT REPORT

On

CALORIE BURNT PREDICTION

Submitted to

Department of Computer Science & Engineering

KESHAV MEMORIAL ENGINEERING COLLEGE

(Affiliated to Osmania University)

Submitted by

Ms. Garlapati Sai Vinny Reddy	**245521733147**
Mr. G karthik	**245521733144**
Ms. Vaddemoni Hari Priya	**245521733191**

Under the Guidance of

Dr.D.V.S.S.SUBRAHMANYAM

HOD, Dept. of CSE

KESHAV MEMORIAL ENGINEERING COLLEGE

(Approved by AICTE, New Delhi & Affiliated to Osmania University, Hyderabad)

D.No. 10 TC-111, Kachavanisingaram (V), Ghatkesar (M), Medchal-Malkajgiri, Telangana – 500088

(2023-2024)

KESHAV MEMORIAL ENGINEERING COLLEGE

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the project report titled **Calorie Burnt Prediction** that is submitted by Ms**.Garlapati Sai Vinny Reddy(245521733147), Mr. G Karthik (245521733144), Ms Vaddemoni Hari Priya(245521733191),**under the guidance of **Dr. D.V.S.S SUBRAHMANYAM , HOD Department of CSE,**as per the requirements of the course curriculum in VI Semester of **B.E (CSE) of Osmania University** is a bonafide work carried out by the above said group of students under my guidance and supervision..

Dr. D.V.S.S SUBRAHMANYAM
Internal Guide

Dr. D.V.S.S SUBRAHMANYAM
HoD

EXTERNAL EXAMINER

Submitted for Viva voce Examination held on _____

DECLARATION

This is to certify that the mini project titled **Calorie Burnt Prediction** is a bonafide work done by us in fulfillment of the requirements for the award of the degree **Bachelor of Engineering** in Department of Computer Science and Engineering, and submitted to the **Department of CSE, Keshav Memorial Engineering College, Hyderabad.**

We also declare that this project is a result of our own effort and has not been copied or intimated from any source. Citations from any websites are mentioned in the bibliography.This work was not submitted earlier at any other university for the award of any degree.

Ms. Garlapati Sai Vinny Reddy (245521733147)
Mr. G Karthik (245521733144)
Ms. Vaddemoni Hari Priya (245521733191)

ACKNOWLEDGEMENT

This is to place on our record my appreciation and deep gratitude to the persons without whose support this project would never been this successful.

We are grateful to **Mr. Neil Gogte,**Founder Director, for facilitating all the amenities required for carrying out this project.

It is with immense please that we would like to express our indebted gratitude to the respected **Prof. P.V.N Prasad, Principal, Keshav Memorial Engineering College,** for providing a great support and for giving us the opportunity of doing the project.

We express our sincere gratitude to **Mrs. Deepa Ganu,** Director Academics, for providing an excellent environment in the college.

We would like to take this opportunity to specially thank to **Dr.D.V.S.S.SUBRAHMANYAM,HoD, Department of CSE ,Keshav Memorial Engineering College,** for inspiring us all the way and for arranging all the facilities and resources needed for our project.

We would like to take this opportunity to thank our internal guide to**Dr.D.V.S.S.SUBRAHMANYAM, HOD, Department of CSE , Keshav Memorial Engineering College,** who has guided us a lot and encouraged us in every step of the project work. His valuable moral support and guidance throughout the project helped us to a greater extent.

We would like to take this opportunity to specially thank our Project Coordinator, **SHAILAJA, ASSISTANT PROFESSOR Department of CSE ,Keshav Memorial Engineering College,** who guided us in successful completion of our project.

Finally, we express our sincere gratitude to all the members of the faculty of Department of CSE , our friends and our families who contributed their valuable adviceand helped us to complete the project successfully.

Ms. Garlapati Sai Vinny Reddy (245521733147)
Mr.G Karthik (245521733144)
Ms.Vaddemoni Hari Priya (245521733191)

CONTENTS

PROBLEM STATEMENT

DOMAIN: Machine Learning

PROJECT NAME: Calories-Burned-Prediction Using Machine Learning

To address the growing demand for personalized fitness tracking, this project aims to develop a machine learning model that accurately predicts the number of calories burnt during various physical activities. By analyzing user-specific data such as age, weight, height, and activity levels, the model will provide tailored calorie predictions, enhancing the precision of fitness tracking. This approach not only improves user engagement but also promotes healthier lifestyle choices by offering actionable insights based on individual metrics. Ultimately, the project seeks to bridge the gap between general fitness guidelines and personalized health recommendations.

ABSTRACT

In today's health-conscious society, wearable fitness trackers and mobile applications have become indispensable tools for individuals aiming to monitor their physical activity and manage their health. Despite their popularity, these devices often provide generalized data, which may not accurately reflect individual variations in calorie expenditure during physical activities. To address this limitation, our project focuses on developing a machine learning model that predicts the number of calories burnt more precisely by considering user-specific characteristics and activity data.

This project utilizes a dataset containing information such as age, weight, height, gender, and details of various physical activities, including their duration and intensity. We preprocess this data to train a machine learning model capable of predicting calorie burn with higher accuracy compared to existing methods. The model employs algorithms like linear regression, decision trees, or more advanced techniques such as gradient boosting and neural networks to analyze and learn from the patterns in the data.

The key innovation in our approach lies in its ability to tailor predictions based on individual user profiles. By leveraging personalized data, the model can offer more relevant insights, enabling users to make informed decisions about their fitness routines. For instance, a person with a higher muscle mass may burn more calories during a particular activity compared to someone with less muscle mass, and our model accounts for such variations.

We also integrate the model into a user-friendly interface, allowing users to input their personal details and activity data to receive immediate calorie

burn predictions. This feature ensures that the model's benefits are accessible and practical for everyday use. Furthermore, the model's predictive capability is enhanced through continuous learning from new data, improving its accuracy over time.

The ultimate goal of this project is to bridge the gap between generic fitness guidelines and personalized health recommendations, promoting a more effective approach to health management. By providing more accurate calorie burn predictions, our model can help users optimize their workout routines, manage their diet more effectively, and ultimately achieve their fitness goals with greater precision. This innovation represents a significant step forward in the evolution of fitness tracking technology, offering users a more scientific and personalized way to monitor their health.

INTRODUCTION

1.1 Introduction about the Concept

In recent years, the proliferation of wearable fitness trackers and mobile applications has made it easier for individuals to monitor their physical activity and track calories burnt. However, these tools often rely on generalized algorithms that may not accurately account for individual differences in metabolism, body composition, and activity intensity. As a result, there is a growing need for a more personalized approach to predicting calorie expenditure, which can be achieved through the application of machine learning. By analyzing user-specific data, machine learning models can offer more accurate and tailored predictions, helping users make more informed decisions about their fitness and health routines.

1.2 Existing System and Disadvantages

Current fitness tracking systems primarily use heuristic-based methods to estimate calories burnt during physical activities. These methods typically consider only basic factors like time spent on activity, distance covered, and general intensity levels. While useful, these estimates often lack precision as they do not account for individual differences such as age, weight, height, and fitness level. Moreover, the existing systems are limited in their ability to adapt to new data or changes in a user's lifestyle. As a result, users receive generic feedback that may not be reflective of their actual calorie burn, leading to potential inaccuracies in their fitness tracking and health management efforts.

1.3 Literature Review

Several studies have explored the use of machine learning techniques to improve the accuracy of calorie burn predictions. Research indicates that incorporating personalized data, such as an individual's basal metabolic rate (BMR) and specific activity details, can significantly enhance prediction accuracy. Various machine learning models, including linear regression, support vector machines, and neural networks, have been tested for this purpose. However, the literature also highlights challenges such as the need for large datasets to train these models effectively and the importance of feature selection to ensure model accuracy.

1.4 Proposed System and Advantages

The proposed system aims to develop a machine learning model that predicts calorie expenditure based on a more comprehensive set of user-specific data. By incorporating factors such as age, weight, height, gender, and specific activity metrics (e.g., intensity, duration), the system can provide more accurate and personalized calorie predictions. The advantages of this approach include:

- Higher Accuracy: Improved precision in calorie burn estimates by considering a wider range of personal and activity-related factors.
- Personalization: Tailored feedback based on individual user profiles, leading to more relevant and actionable insights.
- Adaptability: Continuous learning from new data allows the model to refine its predictions over time, ensuring ongoing accuracy.
- User Engagement: Enhanced user satisfaction through more accurate and personalized health recommendations.

SYSTEM ANALYSIS

2.1 Feasibility Study

The feasibility study examines the technical, economic, and operational viability of developing the proposed machine learning-based calorie prediction system.

- Technical Feasibility: The system requires the integration of machine learning algorithms with user interfaces, which is technically feasible given current technologies.
- Economic Feasibility: The project is cost-effective as it leverages existing computational resources and does not require extensive hardware investments.
- Operational Feasibility: The system is user-friendly and can be easily adopted by individuals familiar with basic fitness tracking apps, making it operationally viable.

2.2 System Requirements

2.2.1 Hardware Requirements

- Processor: Intel Core i5 or higher
- RAM: 8 GB or more
- Storage: 256 GB SSD or higher for fast data processing

2.2.2 Software Requirements

- Operating System: Windows 10, macOS, or Linux
- Programming Languages: Python (for model development and data processing)
- Development Environment:Jupyter Notebook, PyCharm, or VS Code

- Machine Learning Libraries: TensorFlow, Scikit-learn, Keras, Pandas, NumPy
- Database: MySQL or PostgreSQL for storing user data
- Web Framework (if applicable): Django or Flask for integrating the model with a web-based interface

2.2.3 Functional Requirements

- User Authentication: Secure login system to manage user profiles and protect data privacy.
- Data Input:Ability to input and update user-specific data (e.g., age, weight, activity details).
- Calorie Prediction: Core functionality to predict calories burnt based on user input and the trained machine learning model.
- Result Display: Visualization of predicted calorie burn along with relevant recommendations.

2.2.4 Non-Functional Requirements

- Performance: The system should provide predictions in real-time or near real-time.
- Scalability: Capable of handling an increasing number of users without significant degradation in performance.
- Security: Ensure data encryption and secure storage of sensitive user information.
- Usability: The interface should be intuitive and easy to navigate, ensuring a positive user experience.

SYSTEM DESIGN

3.1 Introduction

System design is a critical phase in the development of a machine learning-based calorie prediction model, as it involves translating the project requirements into an organized and structured architecture. This phase covers the creation of modules, block diagrams, UML diagrams, and database design, all of which are essential for implementing and maintaining the system efficiently. The design phase ensures that the system is scalable, reliable, and easy to use, laying the foundation for a robust application.

3.2 Modules and Description

The system is divided into several key modules, each responsible for a specific aspect of the calorie prediction process:

User Profile Management:

Manages user data, including personal details (age, weight, height, gender) and activity preferences.

Handles user authentication and secure storage of user information.

Data Input Module:

Allows users to input details about their physical activities, such as type, duration, and intensity.

Facilitates the input of continuous user updates to refine predictions.

Machine Learning Model:

Core module responsible for predicting calorie expenditure based on user data and activity details.

Implements algorithms such as linear regression, decision trees, or neural networks.

Continuously updates its prediction model with new data.

Prediction and Feedback Module:

Provides users with predicted calorie burn data.

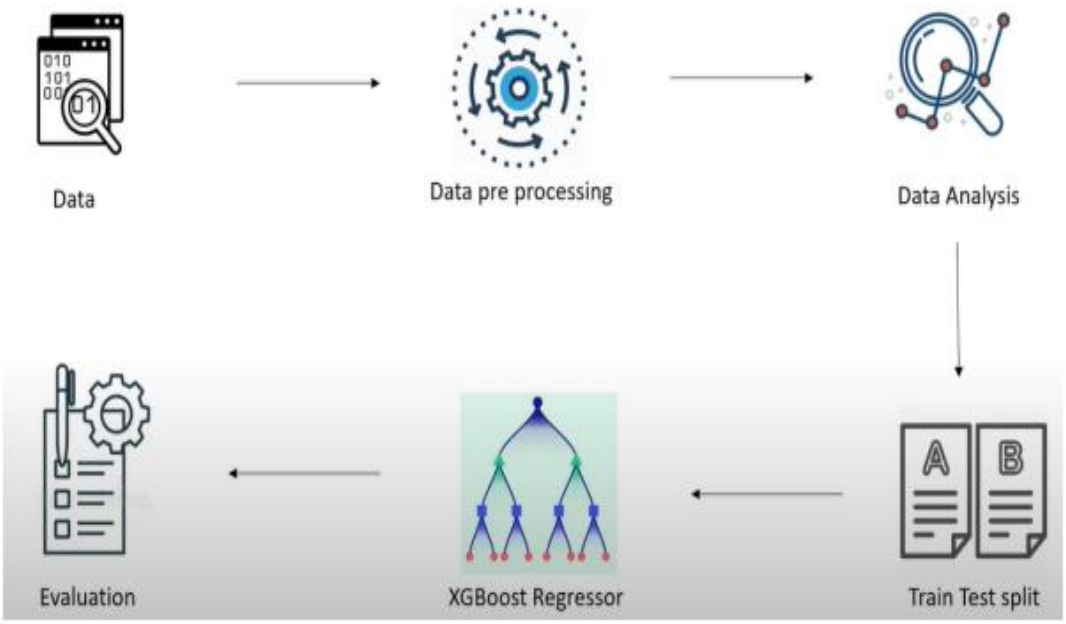
Offers personalized feedback and recommendations based on the predictions, helping users adjust their activities to meet fitness goals.

User Interface Module:

Ensures an intuitive and user-friendly interface for data input and results display.

Visualizes the calorie prediction results and additional insights in an accessible format.

3.3 Block Diagram



The block diagram provides a high-level overview of the system's architecture, illustrating the interaction between the different modules:

User Interface (UI): Interacts with the user for input and displays results.
Data Input: Receives user data and forwards it to the ML Model.
Machine Learning Model: Processes data and predicts calories burnt.
Prediction Feedback: Sends results back to the UI for display.
User Profile Management: Stores and retrieves user data securely.

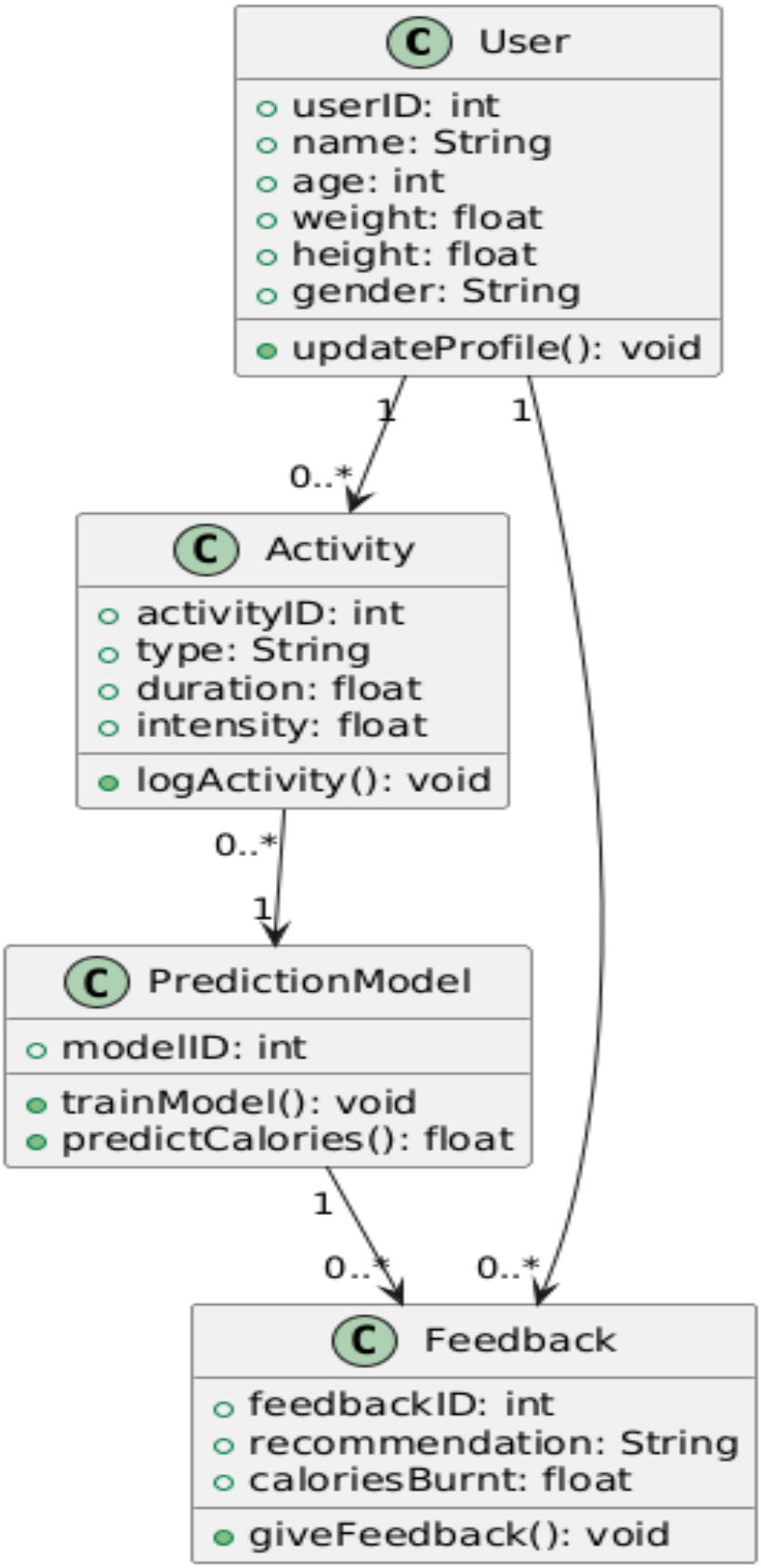
3.4 UML Diagrams

UML (Unified Modeling Language) diagrams help visualize the system's design and the relationships between different components. Below are the essential UML diagrams for the project:

3.4.1 Class Diagram:

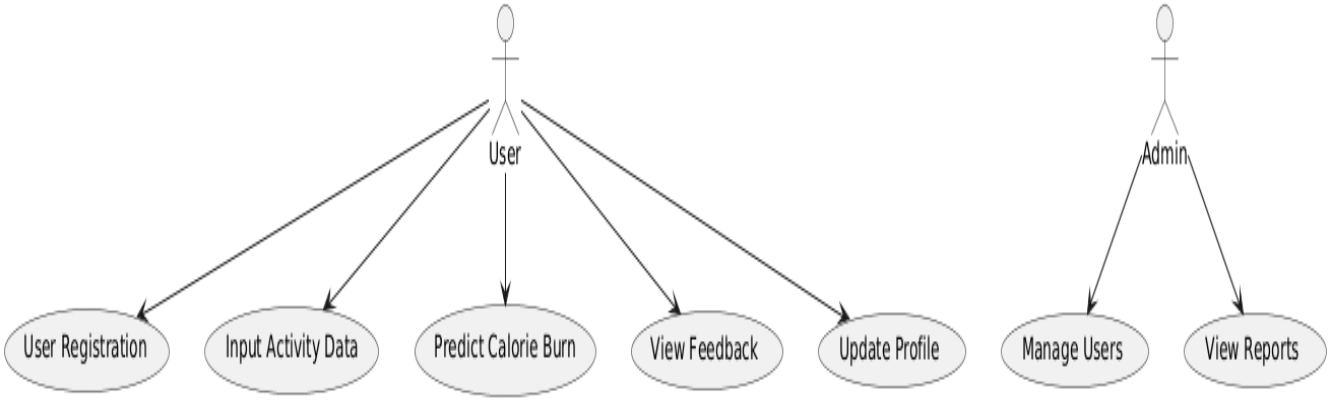
The Class Diagram illustrates the static structure of the system by depicting the system’s classes, their attributes, methods, and the relationships between them. In this diagram, classes such as User, Activity, PredictionModel, and Feedback are represented along with their key attributes and methods.

The relationships between these classes, such as the associations between User and Activity, or PredictionModel and Feedback, help in understanding how data flows and interacts within the system. This diagram serves as the foundation for the system’s object-oriented design.


User
userID: int
name: String
age: int
weight: float
height: float
gender: String
updateProfile(): void
1
0..*
1
Activity
activityID: int
type: String
duration: float
intensity: float
logActivity(): void
0..*
1
PredictionModel
modelID: int
trainModel(): void
predictCalories(): float
1
0..*
0..*
Feedback
feedbackID: int
recommendation: String
caloriesBurnt: float
giveFeedback(): void

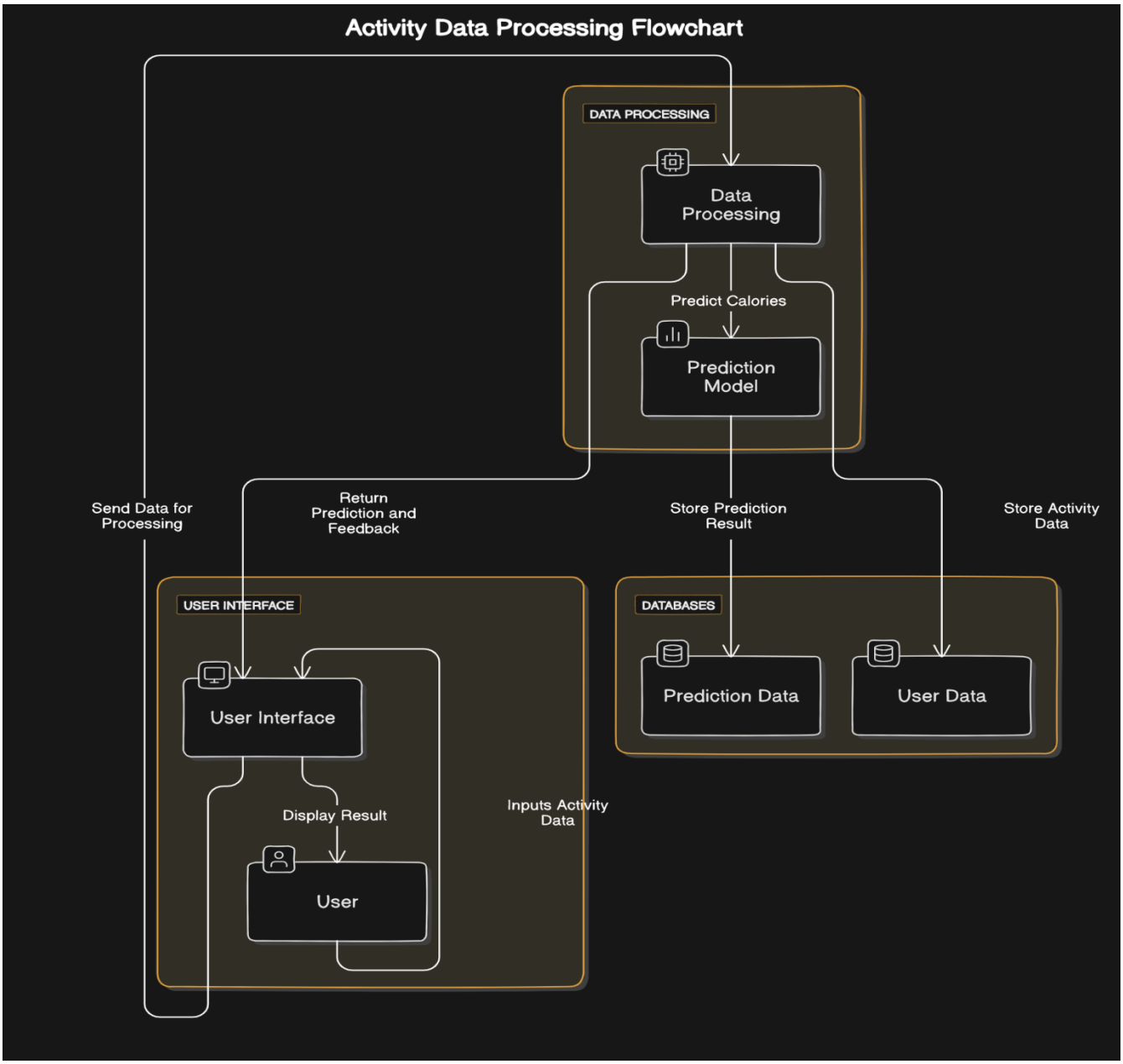
3.4.2 Use Case Diagram

The Use Case Diagram outlines the interactions between the system's users (actors) and the system itself. It identifies the various functionalities (use cases) that users, such as User and Admin, can perform within the system. For example, a User can register, input activity data, predict calorie burn, view feedback, and update their profile. This diagram is useful for capturing the high-level requirements of the system and understanding the scope of what the system will support.



3.4.3 Data Flow Diagram (DFD) Description:

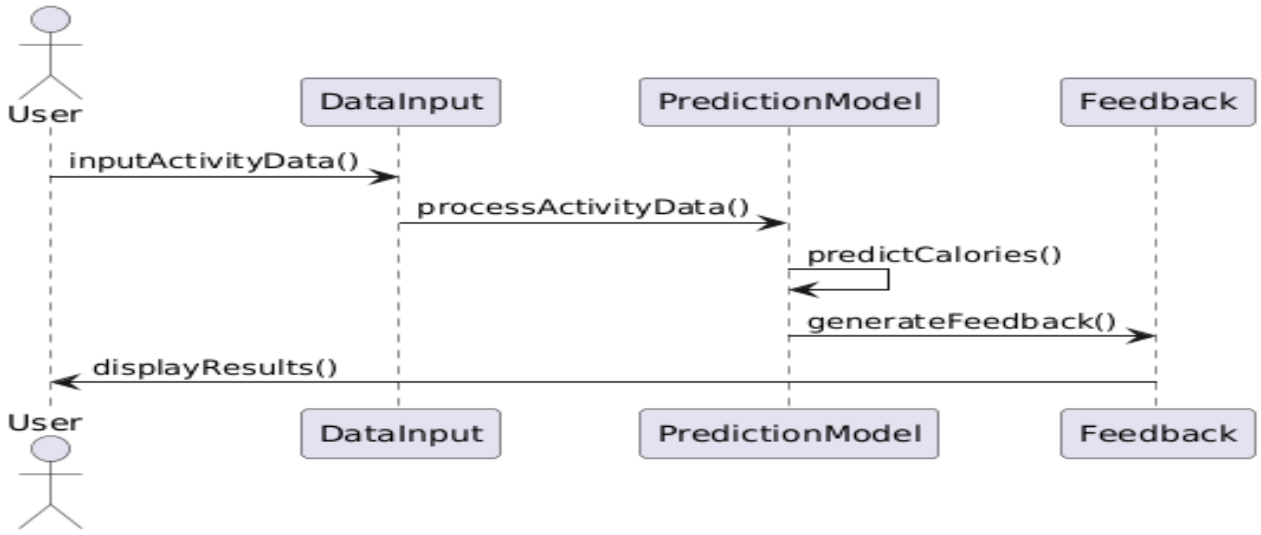
A Data Flow Diagram (DFD) is a graphical representation used to depict the flow of data within a system. It shows how data moves through various processes, how it is stored in data stores, and how it interacts with external entities. The DFD provides a clear and structured overview of the system’s data processing, making it easier to understand and analyze.



3.4.4. Sequence Diagram

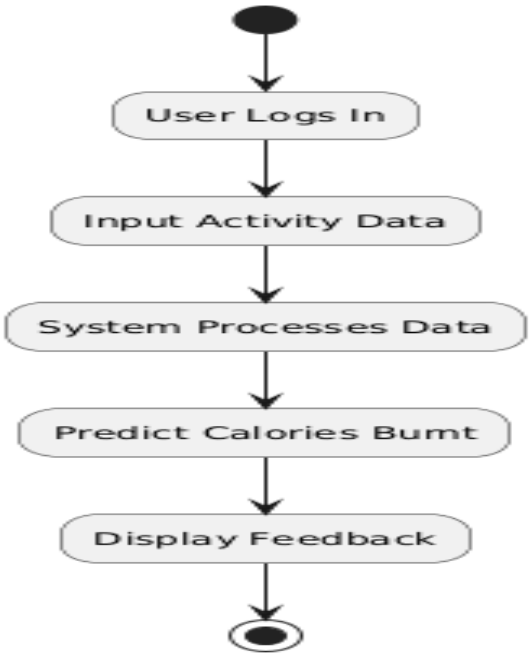
The Sequence Diagram visualizes the order of interactions between different components of the system for a specific use case. It shows how a User interacts with the system to input activity data, how the DataInput module processes this data, and how the PredictionModel generates a calorie prediction. Finally, it shows how feedback is provided to the user. This diagram is crucial for understanding the flow of operations and ensuring that

each step in the process is executed in the correct sequence.



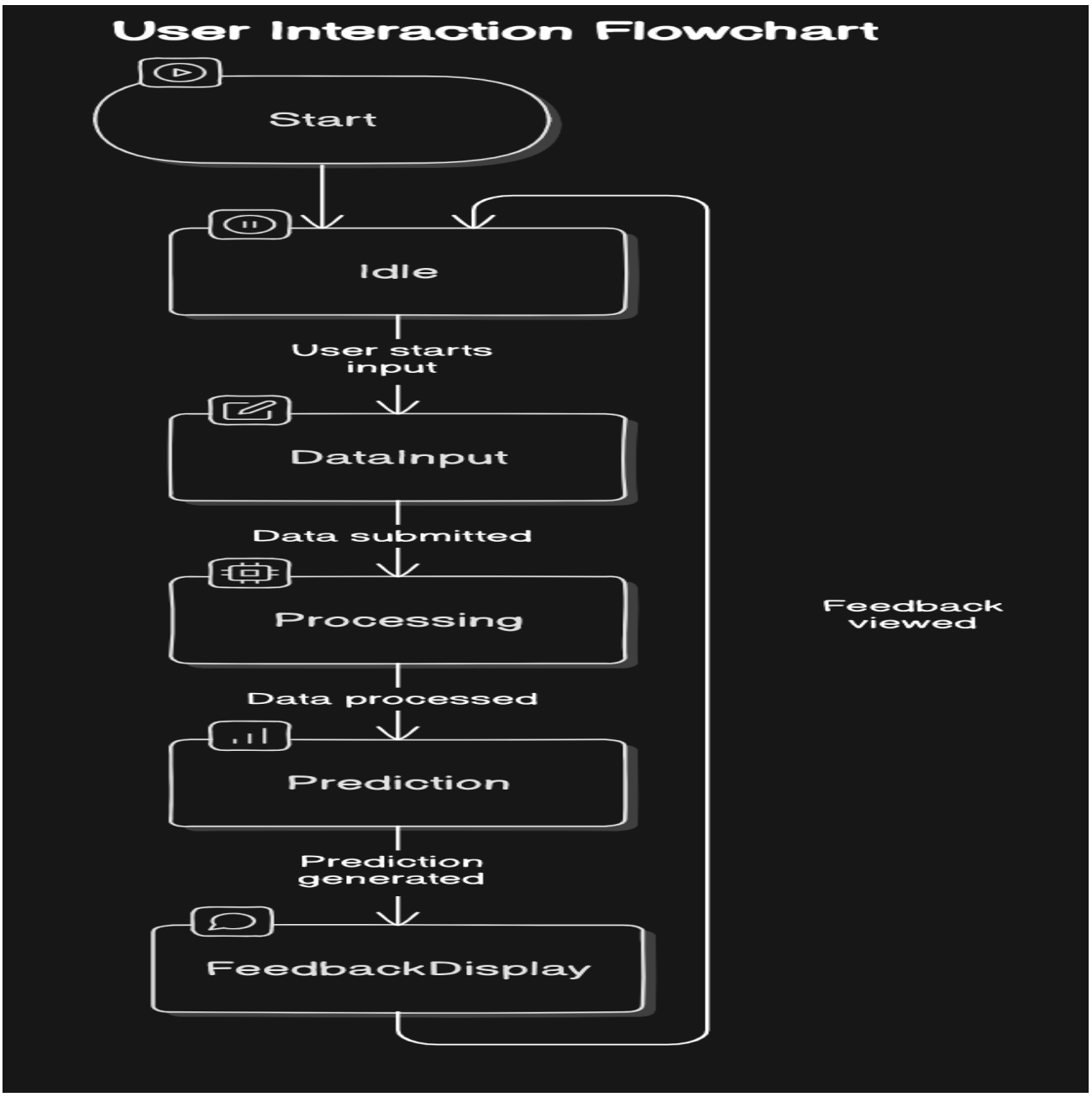
3.4.5. Activity Diagram

The Activity Diagram captures the dynamic flow of activities within the system. It starts from when the user logs in, continues through the input of activity data, processing by the system, calorie prediction, and ends with the display of feedback. This diagram helps in visualizing the workflow and the order in which tasks are performed. It's particularly useful for identifying potential bottlenecks and ensuring that the system supports a logical progression of tasks.



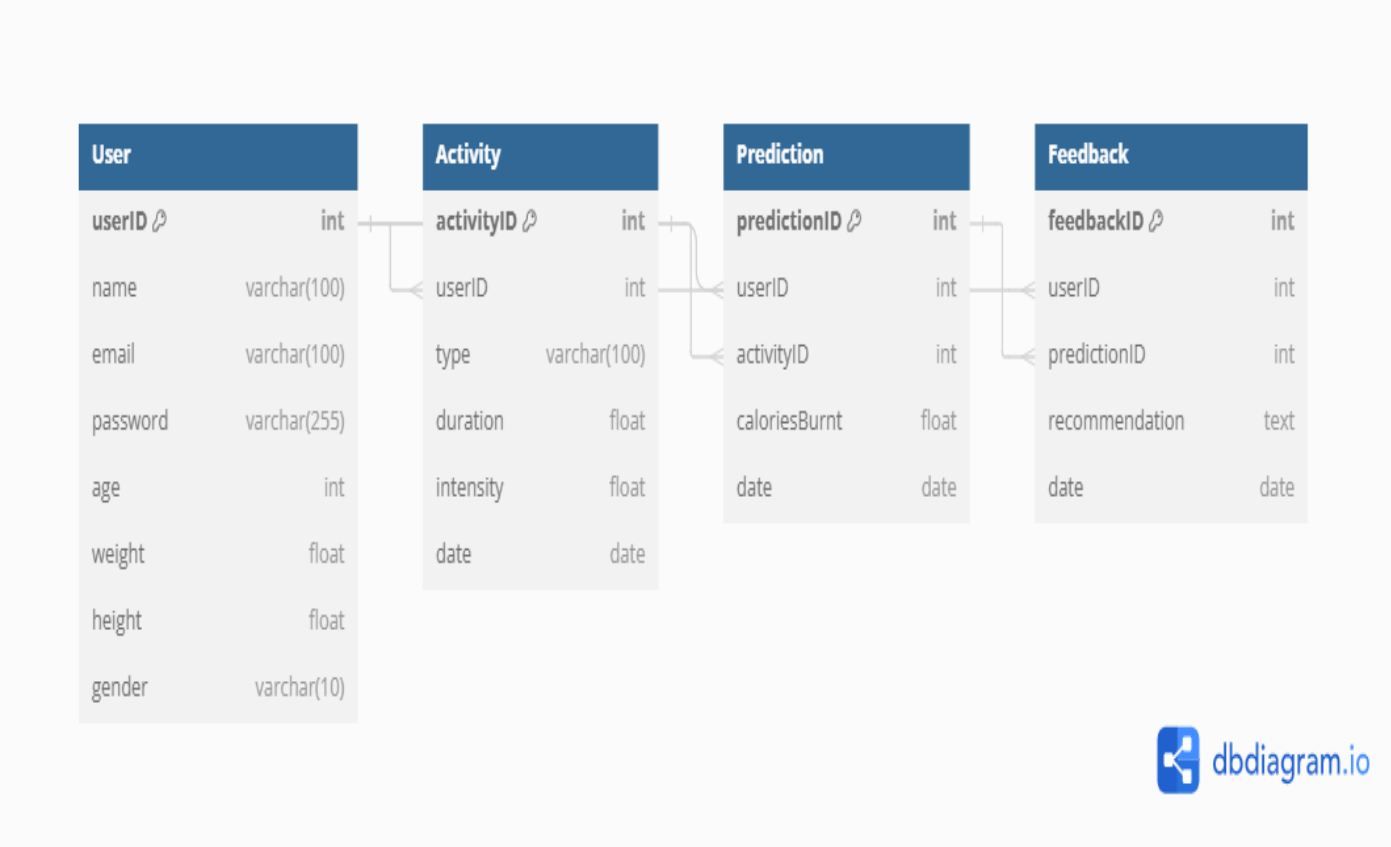
3.4.6. State Chart Diagram

The State Chart Diagram represents the different states the system can be in and the transitions between these states based on user interactions or system processes. For example, the system transitions from an Idle state to DataInput when the user starts entering activity data. After processing, it moves to Prediction and then to FeedbackDisplay before returning to Idle. This diagram is valuable for modeling the system's behavior over time and ensuring that it responds correctly to various events.



3.5 Database Tables

The Database Tables Diagram (ER Diagram) illustrates the logical structure of the system's database by showing the entities (User, Activity, Prediction, Feedback) and the relationships between them. It includes key attributes for each entity and the connections that represent how these entities interact with each other. This diagram is essential for designing the database schema, ensuring data integrity, and understanding how data is stored and retrieved within the system.



SYSTEM IMPLEMENTATION

4.1 Description of Platform, Database, Technologies, Methods, Applications Used, and Involved in Development

1. Platform:

Operating System: The system will be implemented on platforms such as Windows, macOS, or Linux, depending on the development environment and user preferences.

Development Environment: Tools like Visual Studio Code, IntelliJ IDEA, or Eclipse will be used for coding, debugging, and testing. Integrated development environments (IDEs) facilitate efficient development through features like code auto-completion, debugging tools, and version control integrations.

2. Database:

Database Management System (DBMS): The system will use a relational database management system (RDBMS) such as MySQL, PostgreSQL, or SQLite to store user data, activity logs, predictions, and feedback. These DBMSs support structured query language (SQL) for managing and querying the database.

Schema Design: The database schema will include tables for User, Activity, Prediction, and Feedback, designed to efficiently store and retrieve data while maintaining referential integrity.

3. Technologies:

Programming Languages:

Frontend: HTML, CSS, and PyScript will be used to create a responsive and user-friendly interface. Frameworks like React.js or Vue.js may be employed to enhance interactivity.

Backend: Flask (Spring Boot) will be used to handle server-side logic, manage database interactions, and implement the business logic of the application.

Machine Learning: Python will be utilized for developing and training machine learning models using libraries such as TensorFlow, scikit-learn, or PyTorch.

APIs: RESTful APIs will be designed to enable communication between the frontend and backend, facilitating data exchange and interactions with the machine learning model.

4. Methods:

Agile Development: The project will follow Agile methodologies, enabling

iterative development and frequent feedback from stakeholders. This approach allows for flexible adjustments and continuous improvement throughout the development lifecycle.

Model Training: Machine learning models will be trained using historical data to predict calorie expenditure based on user activity. Techniques like supervised learning with regression or classification algorithms will be applied.

Testing: Unit testing, integration testing, and user acceptance testing (UAT) will be conducted to ensure the functionality, performance, and reliability of the system. Tools such as Jest (for JavaScript) or JUnit (for Java) may be used for automated testing.

5. Applications Used:

Development Tools: Integrated Development Environments (IDEs) such as Visual Studio Code or IntelliJ IDEA, code version control systems like Git, and project management tools like Jira or Trello will be utilized to streamline the development process.

Database Tools: Tools like MySQL Workbench, pgAdmin, or SQLite Browser will be used for designing, managing, and querying the database.

Machine Learning Platforms: Jupyter Notebooks or Google Colab will be used for developing and experimenting with machine learning models.

6. Involvement in Development:

Frontend Development: Focuses on creating the user interface, ensuring usability, and integrating frontend components with backend services.

Backend Development: Involves setting up server-side logic, API development, and database integration to handle data processing and business logic.

Machine Learning: Includes data preparation, model training, evaluation, and integration with the backend to provide predictive analytics for calorie expenditure.

SYSTEM TESTING

5.1. Test Plan

The test plan outlines the approach to verify the functionality, reliability, and performance of the web application and machine learning model. The primary goals are to ensure that the application correctly predicts calorie burn and handles user inputs effectively.

Objective: Validate the accuracy and reliability of the XGBoost Regressor model and the Flask web application.
Scope: Includes testing data preprocessing, model training, model evaluation, and web application functionality.
Resources: Test datasets, Flask application environment, user input simulations.
Tools: Python's testing frameworks (e.g., unittest or pytest), browser for UI testing, and performance monitoring tools.

5.2. Scenarios

Data Input Validation:

Input: Valid and invalid data (e.g., non-numeric values, missing fields).
Expected Outcome: The application should handle invalid inputs gracefully and prompt the user for correct information.
Model Prediction Accuracy:

Input: Test dataset with known values.
Expected Outcome: The model should provide predictions with an MAE close to 1, as indicated by previous evaluations.
User Interface Testing:

Input: User interactions (e.g., submitting forms, receiving predictions).
Expected Outcome: The interface should be intuitive and responsive, displaying predictions correctly based on user inputs.
Performance Testing:

Input: High volume of concurrent user interactions.
Expected Outcome: The application should handle multiple requests without

significant performance degradation.

Error Handling:

Input: Simulated errors (e.g., server issues, model failures).
Expected Outcome: The application should display appropriate error messages and handle failures without crashing.

5.3. Output Screens

Home Screen:

Features: Input fields for age, height, weight, exercise duration, heart rate, body temperature, and gender.
Expected Output: User-friendly form layout, clear instructions, and input validation messages.
Prediction Results Screen:

Features: Display of predicted calories burned based on user inputs.
Expected Output: Accurate prediction results, formatted clearly with brief recommendations or diet suggestions.
Error Screen:

Features: Error messages for invalid inputs or system issues.
Expected Output: Informative and user-friendly messages guiding users on how to correct their inputs or report issues.

CONCLUSION
&
FUTURE SCOPE

6.1. Conclusion

The development of our calorie prediction web application marks a significant achievement in applying machine learning to practical fitness solutions. By leveraging the XGBoost Regressor, we have created a robust model that effectively predicts the number of calories burned based on user inputs. With a dataset of 15,000 records and achieving a Mean Absolute Error (MAE) of approximately 1, our model demonstrates high accuracy and reliability in making predictions.

The project began with thorough data collection, focusing on crucial features such as age, height, weight, exercise duration, heart rate, body temperature, and gender. Data preprocessing ensured the dataset was clean and formatted correctly, addressing missing values and encoding categorical variables. This meticulous preparation laid a solid foundation for training the XGBoost Regressor.

Model training was conducted with careful attention to optimizing performance, followed by rigorous evaluation to validate its generalizability. The resulting web application, built using Flask, presents a user-friendly interface that allows users to input their data effortlessly and receive instant calorie predictions. This feature empowers users to make informed decisions about their fitness routines and dietary choices.

The application is not only functional but also demonstrates a practical approach to integrating machine learning with web technologies. By providing accurate predictions and an intuitive user experience, it addresses real-world needs and offers valuable insights for fitness enthusiasts.

Looking forward, the project's future scope includes enhancing the application with personalized recommendations, integrating additional data sources for improved accuracy, and expanding its reach through mobile and wearable device integrations. These advancements will further elevate the application's value and usability, ensuring it remains a relevant and powerful tool for users seeking to optimize their health and fitness.

In conclusion, this project exemplifies the potential of combining advanced machine learning techniques with accessible web applications, offering a practical solution for calorie prediction and paving the way for future

innovations in the field.

6.2. Future Scope

Personalized Recommendations:

Integrate advanced algorithms to provide personalized exercise and diet suggestions based on individual user profiles and activity history.
Model Updates:

Continuously update the model with new data to improve prediction accuracy and adapt to changes in fitness trends and user behavior.
Enhanced User Experience:

Develop additional features like data visualization (graphs and charts) and user progress tracking to enhance the overall user experience.
Mobile Application:

Expand the project to include a mobile application for greater accessibility and convenience.
Integration with Wearable Devices:

Incorporate data from wearable fitness devices to provide real-time calorie burn predictions and personalized insights.

REFERENCES

7.1 Bibliography

Chen, T., & Guestrin, C. (2016). XGBoost:
A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785-794). ACM.
This paper introduces XGBoost, a scalable and efficient implementation of gradient boosting for supervised learning problems. It details the algorithm's efficiency and performance improvements over existing methods.
Grinberg, M. (2018). Flask Web Development:
Developing Web Applications with Python. O'Reilly Media.
This book provides a comprehensive guide to Flask, a micro web framework for Python. It covers the essentials of Flask for building web applications, including routing, templates, and web forms.
McKinney, W. (2010):
Data Analysis with Pandas. Retrieved from https://pandas.pydata.org/
McKinney's work on Pandas introduces this powerful data analysis library, explaining its functionalities for data manipulation and analysis in Python.
Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn:
Machine Learning in Python. Journal of Machine Learning Research, 12, 2825-2830.
This paper describes Scikit-learn, a Python library for machine learning, emphasizing its wide range of algorithms and tools for data analysis and model building.
The Python Software Foundation. (2021):
pickle-mixin. Retrieved from https://pypi.org/project/pickle-mixin/
The pickle-mixin library extends Python's pickle module to support a wider range of object serialization, useful for saving and loading machine learning models.
The Pallets Projects. (2021):
Werkzeug Documentation. Retrieved from https://werkzeug.palletsprojects.com/
Werkzeug is a comprehensive WSGI web application library. The documentation provides details on its utilities for creating web applications in Python.

Iglewicz, B., & Hoaglin, D. C. (1993):
How to Detect and Handle Outliers. Wiley.
This book offers techniques for identifying and addressing outliers in data analysis, which is crucial for effective data preprocessing.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An Introduction to Statistical Learning:
With Applications in R. Springer.
The text covers fundamental concepts of statistical learning and machine learning techniques, including model evaluation and practical applications.

Real Python. (2021):
Python Web Development with Flask. Retrieved from https://realpython.com/tutorials/flask/
This online resource offers tutorials and practical guides on Flask for building Python web applications, suitable for both beginners and advanced users.

Towards Data Science. (2020):
Practical Guide to XGBoost in Python. Retrieved from https://towardsdatascience.com/xgboost-for-python-a-practical-guide-9e636e500474

This guide provides practical advice on using XGBoost for regression and classification problems in Python, including implementation details and performance tips.

Catenacci, V. A., & Pan, X. (2017).
The Role of Caloric Expenditure in Weight Management. Journal of Obesity, 2017.
The article explores the relationship between caloric expenditure and weight management, offering insights relevant to fitness and dietary recommendations.

7.2 Web References

XGBoost Documentation:

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785-794). ACM. Retrieved from https://arxiv.org/abs/1603.02754

Flask Documentation:

Grinberg, M. (2018). Flask Web Development: Developing Web Applications with Python. O'Reilly Media. Retrieved from https://flask.palletsprojects.com/en/2.0.x/

Pandas Documentation:

McKinney, W. (2010). Data Analysis with Pandas. Retrieved from https://pandas.pydata.org/

Scikit-Learn Documentation:

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825-2830. Retrieved from http://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf

Pickle-Mixin Documentation:

The Python Software Foundation. (2021). pickle-mixin. Retrieved from https://pypi.org/project/pickle-mixin/

Werkzeug Documentation:

The Pallets Projects. (2021). Werkzeug Documentation. Retrieved from https://werkzeug.palletsprojects.com/

Data Preprocessing Techniques

Iglewicz, B., & Hoaglin, D. C. (1993). How to Detect and Handle Outliers. Wiley. Retrieved from https://www.wiley.com/en-us/How+to+Detect+and+Handle+Outliers-p-9780471002532

Machine Learning Model Evaluation

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An Introduction to Statistical Learning: With Applications in R. Springer. Retrieved from https://www.springer.com/gp/book/9781461471370

Web References and Tutorials

Real Python. (2021). Python Web Development with Flask. Retrieved from https://realpython.com/tutorials/flask/
Towards Data Science. (2020). Practical Guide to XGBoost in Python. Retrieved from https://towardsdatascience.com/xgboost-for-python-a-practical-guide-9e636e500474

Fitness and Caloric Expenditure

Catenacci, V. A., & Pan, X. (2017). The Role of Caloric Expenditure in Weight Management. Journal of Obesity, 2017. Retrieved from https://www.hindawi.com/journals/jobe/2017/4253684/

APPENDIX

Annexure-1: Sample Coding

Code:

```
from flask import Flask, render_template,
request
import pandas as pd
import pickle
from werkzeug.utils import quote

app = Flask(__name__)

with open('model.pkl', 'rb') as f:
    model = pickle.load(f)


@app.route('/')
def index():
    return render_template('index.html')


@app.route('/predict', methods=['POST'])
def predict():
    try:
        if request.method == 'POST':
            age =
float(request.form['age'])
            height =
float(request.form['height'])
            weight =
float(request.form['weight'])
            duration =
float(request.form['duration'])
            heart_rate =
float(request.form['heart_rate'])
            body_temp =
float(request.form['body_temp'])
            gender = request.form['gen']

            gender_encoded =
encode_gender(gender)
            if gender_encoded is None:
                return "Invalid gender
value", 400

            user_data = pd.DataFrame({
                'Age': [age],
                'Height': [height],
                'Weight': [weight],
                'Duration': [duration],
                'Heart_Rate':
[heart_rate],
                'Body_Temp': [body_temp],
                'Gender': [gender_encoded]
            })

            user_data =
user_data[['Gender', 'Age', 'Height',
'Weight', 'Duration', 'Heart_Rate',
'Body_Temp']]

            calories =
model.predict(user_data)[0]
```

```
            return
render_template('result.html',
calories=calories)
    except Exception as e:
        return str(e), 500


@app.route('/recommendation',
methods=['POST'])
def recommendation():
    try:
        calories =
float(request.form.get('calories', 0))
        recommendations =
get_diet_recommendations(calories)
        videos = get_videos()
        articles = get_articles()
        return
render_template('Recommendation.html',
recommendations=recommendations,
videos=videos, articles=articles)
    except Exception as e:
        return str(e), 500


def encode_gender(gender):
    if gender == 'male':
        return 0
    elif gender == 'female':
        return 1
    else:
        return None
```

```
def get_diet_recommendations(calories):
    if calories < 200:
        return "Consider light snacks like
a piece of fruit or a small handful of
nuts. ..."
    elif 200 <= calories < 500:
        return "You could have a balanced
snack, such as yogurt with granola or a
smoothie. ..."
    elif 500 <= calories < 800:
        return "A small meal with lean
protein, whole grains, and vegetables
would be beneficial. ..."
    else:
        return "A hearty meal with a good
balance of protein, carbs, and fats is
recommended. ..."


def get_videos():
    return [
        {
            "title": "Top 5 Benefits of
Exercise | Why Exercise is Important",
            "thumbnail":
"https://img.youtube.com/vi/dQw4w9WgXcQ/mq
default.jpg",
            "link":
"https://www.youtube.com/user/FitnessBlend
er"
        },
        {
```

```
            "title": "Exercise and Weight
Loss | The Ultimate Guide",
            "thumbnail":
"https://img.youtube.com/vi/3C2WThErFZ0/mq
default.jpg",
            "link":
"https://www.youtube.com/user/yogawithadri
ene"
        }
    ]


def get_articles():
    return [
        {
            "title": "10 Benefits of
Regular Exercise",
            "link":
"https://www.healthline.com/nutrition/10-
benefits-of-exercise"
        },
        {
            "title": "How Much Exercise Do
You Really Need?",
            "link":
"https://www.mayoclinic.org/healthy-
lifestyle/fitness/in-depth/exercise/art-
20048389"
        }
    ]


if __name__ == '__main__':
    app.run(debug=True, host='0.0.0.0',
port=5000)
```

Index:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
content="width=device-width, initial-
scale=1.0">
    <title>Calories Burned
Predictor</title>
    <link
href="https://fonts.googleapis.com/css2?fa
mily=Roboto:wght@400;500&display=swap"
rel="stylesheet">
    <style>
        body {
            background-color: #585fa3;
            font-family: 'Roboto', sans-
serif;
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
            margin: 0;
            flex-direction: column;
        }
        h1 {
            text-align: center;
            color: #050e0d;
        }
        .container {
            background-color: #ffffff;
            padding: 20px;
            border-radius: 10px;
            box-shadow: 0 0 15px rgba(0,
0, 0, 0.2);
            width: 90%;
            max-width: 600px;
            margin-bottom: 20px;
        }
        label {
```

```
            display: block;
            margin-bottom: 5px;
            color: #0c0c0c;
            font-weight: 500;
        }
        input[type="text"], textarea {
            width: 100%;
            padding: 10px;
            margin-bottom: 15px;
            box-sizing: border-box;
            border: 1px solid #010303;
            border-radius: 5px;
            font-family: 'Roboto', sans-
serif;
        }
        input[type="submit"] {
            width: 100%;
            padding: 12px;
            background-color: #00796b;
            color: white;
            border: none;
            border-radius: 5px;
            cursor: pointer;
            font-family: 'Roboto', sans-
serif;
            font-size: 16px;
            font-weight: 500;
        }
        input[type="submit"]:hover {
            background-color: #004d40;
        }
        textarea {
            height: 100px;
            resize: none;
        }
        .error {
            color: red;
            font-weight: 500;
            margin-bottom: 15px;
        }
    </style>
</head>
<body>
    <div class="container">
        <h1>Metabometer - Calorie
Predictor</h1>
```

```
        <form id="calorieForm"
action="/predict" method="post"
onsubmit="return validateForm()">
            <div id="error-message"
class="error" style="display:
none;"></div>
            <label
for="gen">Gender:</label>
            <input type="text" id="gen"
name="gen">
            <label for="age">Age:</label>
            <input type="text" id="age"
name="age">
            <label for="height">Height
(cm):</label>
            <input type="text" id="height"
name="height">
            <label for="weight">Weight
(kg):</label>
            <input type="text" id="weight"
name="weight">
            <label for="duration">Duration
(minutes):</label>
            <input type="text"
id="duration" name="duration">
            <label for="heart_rate">Heart
Rate:</label>
            <input type="text"
id="heart_rate" name="heart_rate">
            <label for="body_temp">Body
Temperature (°C):</label>
            <input type="text"
id="body_temp" name="body_temp">
            <input type="submit"
value="Predict">
        </form>
    </div>

    <script>
        function validateForm() {
            var gender =
document.getElementById('gen').value.trim(
);
            var age =
document.getElementById('age').value.trim(
);
```

```
            var height =
document.getElementById('height').value.tr
im();
            var weight =
document.getElementById('weight').value.tr
im();
            var duration =
document.getElementById('duration').value.
trim();
            var heart_rate =
document.getElementById('heart_rate').valu
e.trim();
            var body_temp =
document.getElementById('body_temp').value
.trim();
            var errorMessage =
document.getElementById('error-message');

            var errors = [];

            if (!gender) {
                errors.push('Gender is
required.');
            }
            if (age <=10  || isNaN(age)) {
                errors.push('Age must be a
number greater than 0.');
            }
            if (height <= 0 ||
isNaN(height)) {
                errors.push('Height must
be a number greater than 0.');
            }
            if (weight <= 15 ||
isNaN(weight)) {
                errors.push('Weight must
be a number greater than 0.');
            }
            if (duration <= 0 ||
isNaN(duration)) {
                errors.push('Duration must
be a number greater than 0.');
            }
            if (heart_rate <= 0 ||
isNaN(heart_rate)) {
                errors.push('Heart Rate
must be a number greater than 0.');
            }
            if (body_temp <= 0 ||
isNaN(body_temp)) {
                errors.push('Body
Temperature must be a number greater than
0.');
            }

            if (errors.length > 0) {
                errorMessage.style.display
= 'block';
                errorMessage.innerHTML =
errors.join('<br>');
                return false;
            } else {
                errorMessage.style.display
= 'none';
                return true;
            }
        }
    </script>
</body>
</html>
```

Recommendation:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
content="width=device-width, initial-
scale=1.0">
    <title>Diet Recommendations</title>
    <style>
        body {
            font-family: Arial, sans-
serif;
            background-color: #f0f8ff;
            color: #333;
```

```
            display: flex;
            flex-direction: column;
            align-items: center;
            justify-content: center;
            height: 100vh;
            margin: 0;
        }
        .container {
            text-align: center;
            background-color: #fff;
            padding: 20px;
            border-radius: 10px;
            box-shadow: 0 0 10px rgba(0,
0, 0, 0.1);
            width: 80%;
            max-width: 600px;
        }
        h1 {
            color: #2e8b57;
        }
        p {
            font-size: 1.2em;
        }
        .videos, .articles {
            margin-top: 20px;
        }
        .videos {
            display: flex;
            flex-wrap: wrap;
            justify-content: center;
        }
        .video {
            margin: 10px;
            text-align: center;
        }
        .video img {
            width: 100%;
            height: auto;
        }
        .video a, .article a {
            display: block;
            margin-top: 10px;
            color: #00796b;
            font-weight: bold;
            text-decoration: none;
        }
        .video a:hover, .article a:hover {
            text-decoration: underline;
        }
        .articles ul {
            list-style-type: none;
            padding: 0;
        }
        .articles li {
            margin: 10px 0;
        }
        .articles a {
            color: #00796b;
            text-decoration: none;
        }
        .articles a:hover {
            text-decoration: underline;
        }
    </style>
</head>
<body>
```

```
    <div class="container">
        <h1>Diet Recommendations</h1>
        <p>{{ recommendations }}</p>
        <div class="videos">
            {% for video in videos %}
                <div class="video">

                    <a href="{{ video.link
}}" target="_blank">{{ video.title }}</a>
                </div>
            {% endfor %}
        </div>
        <div class="articles"
id="articles">
            <h2>Recommended Articles</h2>
            <ul>
                {% for article in articles
%}

                    <li class="article">
                        <a href="{{
article.link }}" target="_blank">{{
article.title }}</a>
                    </li>
                {% endfor %}
            </ul>
        </div>
    </div>
</body>
</html>
```

Results:

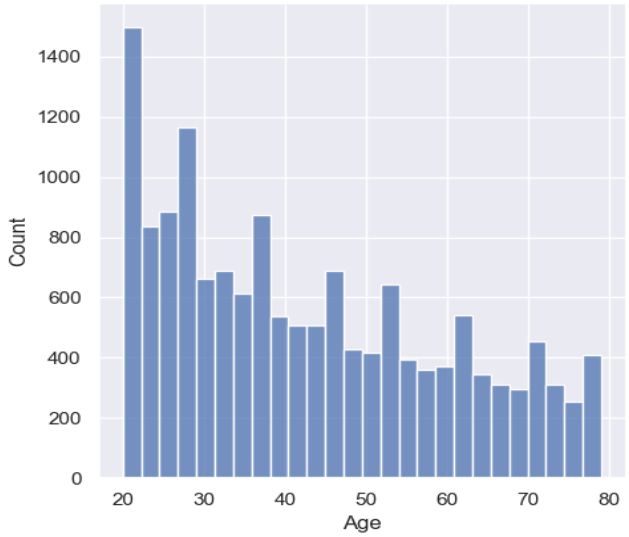
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
content="width=device-width, initial-
scale=1.0">
    <title>Calories Burned Result</title>
    <style>
        body {
            font-family: Arial, sans-
serif;
            background-color: #f0f8ff;
            color: #333;
            display: flex;
            flex-direction: column;
            align-items: center;
            justify-content: center;
            height: 100vh;
            margin: 0;
        }
        .container {
            text-align: center;
            background-color: #fff;
            padding: 20px;
            border-radius: 10px;
            box-shadow: 0 0 10px rgba(0,
0, 0, 0.1);
            width: 80%;
            max-width: 600px;
        }
        h1 {
            color: #2e8b57;
        }
        p {
            font-size: 1.2em;
```

```
        }
        .recommendations {
            margin-top: 20px;
        }
        #chart-container {
            margin-top: 20px;
            width: 100%;
        }
        .recommendation-button {
            margin-top: 20px;
            padding: 10px 20px;
            background-color: #00796b;
            color: white;
            border: none;
            border-radius: 5px;
            cursor: pointer;
            font-size: 16px;
            font-weight: bold;
            text-decoration: none;
            transition: background-color
0.3s ease;
        }
        .recommendation-button:hover {
            background-color: #004d40;
        }
        .videos {
            margin-top: 20px;
            display: flex;
            flex-wrap: wrap;
            justify-content: center;
        }
        .video {
            margin: 10px;
            text-align: center;
        }
        .video img {
            width: 100%;
            height: auto;
        }
        .video a {
            display: block;
            margin-top: 10px;
            color: #00796b;
            font-weight: bold;
            text-decoration: none;
        }
        .video a:hover {
            text-decoration: underline;
        }
    </style>
</head>
<body>
    <div class="container">
        <h1>Calories Burned Result</h1>
        <p>Total Calories Burned: <span
id="calories">{{ calories }}</span></p>
        <div class="recommendations"
id="recommendations"></div>
        <div id="chart-container">
            <canvas
id="caloriesChart"></canvas>
        </div>
        <form action="/recommendation"
method="post">
            <input type="hidden"
name="calories" value="{{ calories }}">
```

```
            <button type="submit"
class="recommendation-button">Get
Recommendations</button>
        </form>
        <div class="videos"
id="videoList"></div>
    </div>

    <!-- Include Chart.js from CDN -->
    <script
src="https://cdn.jsdelivr.net/npm/chart.js
"></script>
    <script>

document.addEventListener("DOMContentLoade
d", function() {
            var calories =
parseInt(document.getElementById("calories
").textContent);

            // Example chart data (replace
with actual data if needed)
            var actualCalories =
[calories];
            var predictedCalories =
[calories + 100, calories + 200, calories
+ 300, calories + 400, calories + 500];

            var allCalories =
actualCalories.concat(predictedCalories);
            var labels = ["Now", "1 day",
"2 days", "3 days", "4 days", "5 days"];
```

```
            var data = {
                labels: labels,
                datasets: [{
                    label: 'Calories
Burned',
                    data: allCalories,
                    backgroundColor:
'rgba(46, 139, 87, 0.2)',
                    borderColor: 'rgba(46,
139, 87, 1)',
                    borderWidth: 1,
                    fill: false,
                    tension: 0.1
                }]
            };

            var ctx =
document.getElementById('caloriesChart').g
etContext('2d');
            var caloriesChart = new
Chart(ctx, {
                type: 'line',
                data: data,
                options: {
                    scales: {
                        y: {
                            beginAtZero:
true
                        }
                    },
                    plugins: {
                        legend: {
                            display: true
```

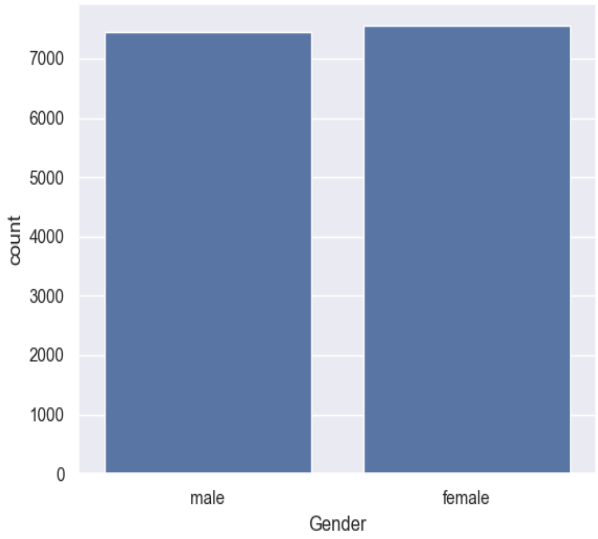
```
                    }
                }
            }
        });
    });
    </script>
</body>
</html>
```

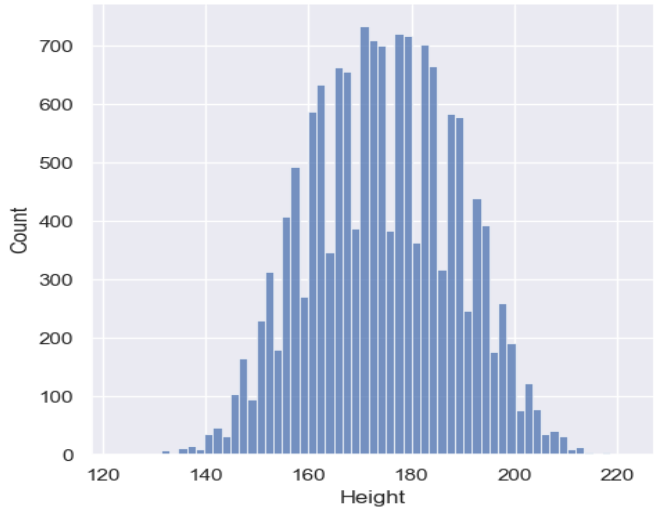


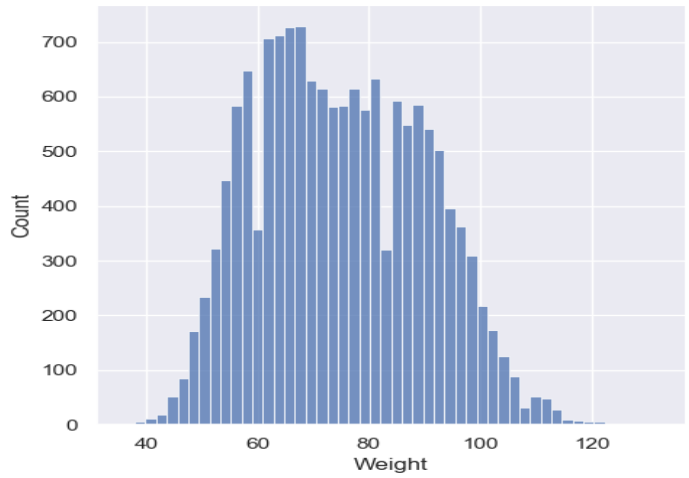
Annexure-2:

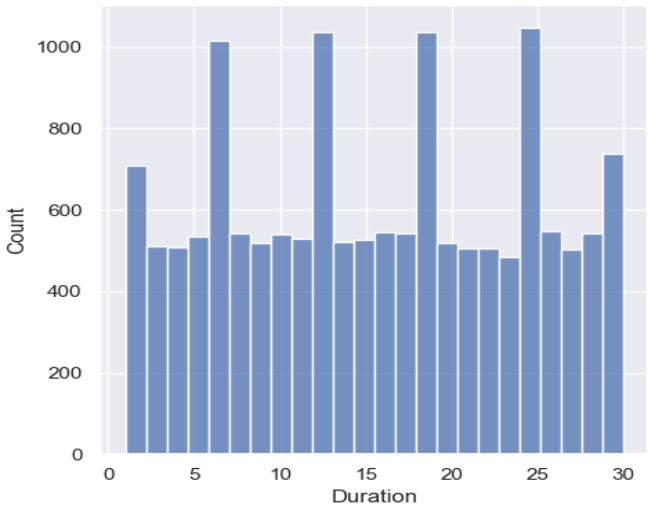
List of Figures

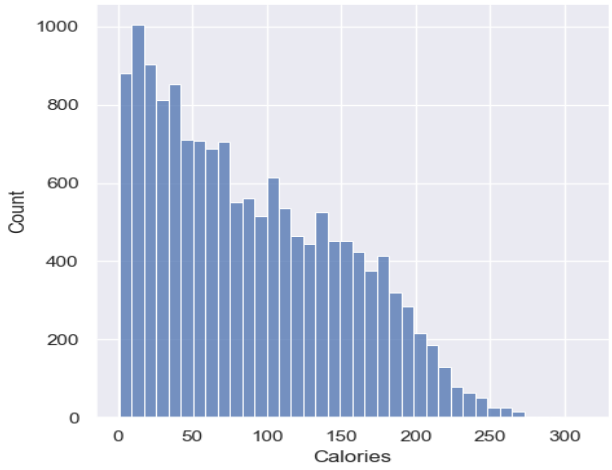
Statistical Measures of Data Sets In Graphical Representation.

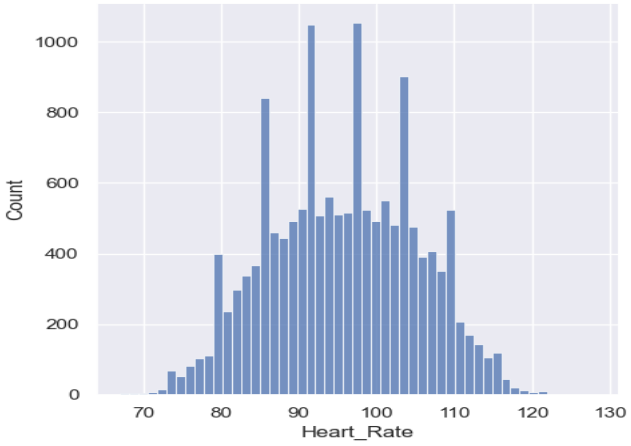


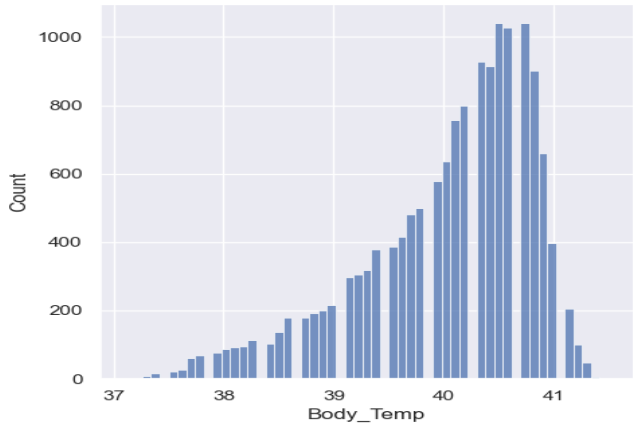


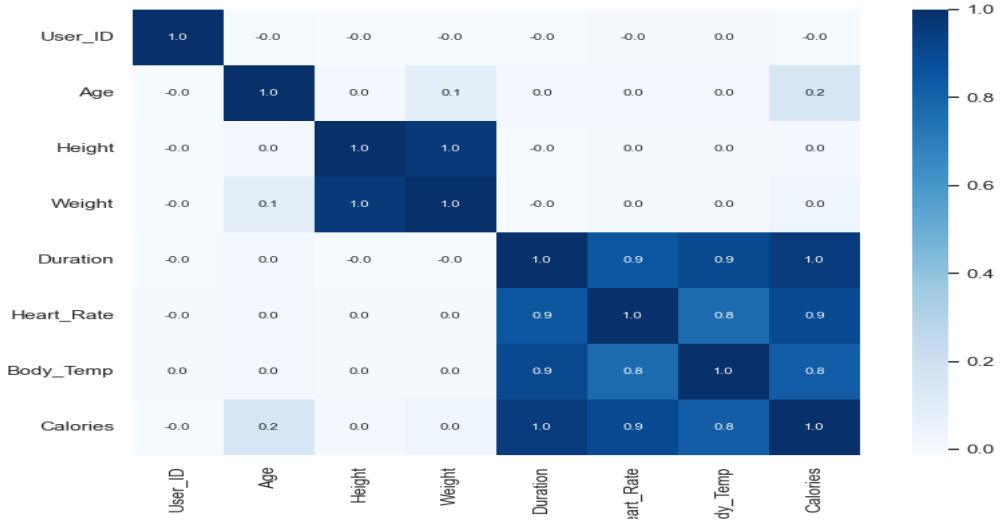



Count
Duration
0
200
400
600
800
1000
0
5
10
15
20
25
30


Count
Calories
0
200
400
600
800
1000
0
50
100
150
200
250
300

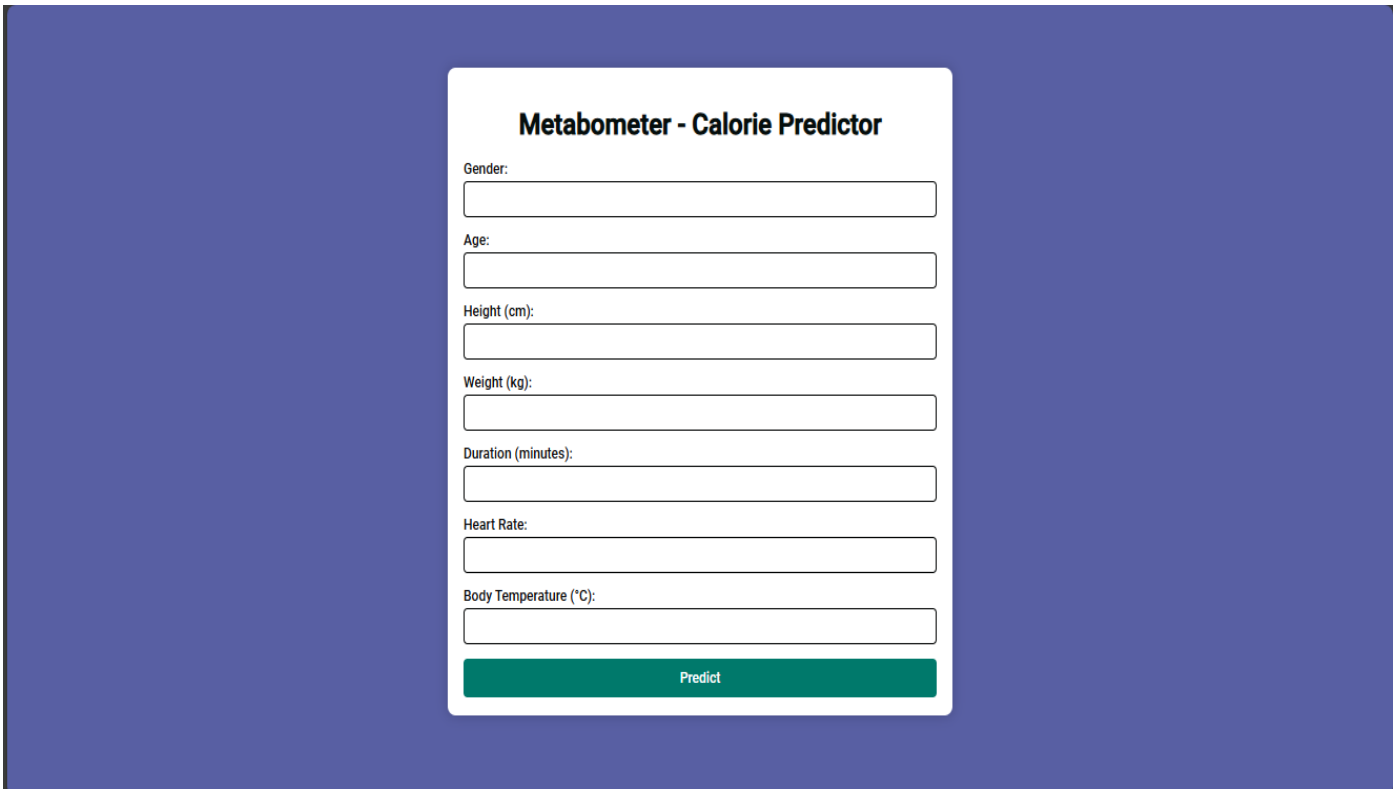

Count
Heart_Rate
0
200
400
600
800
1000
70
80
90
100
110
120
130


Count
Body_Temp
0
200
400
600
800
1000
37
38
39
40
41


User_ID
Age
Height
Weight
Duration
Heart_Rate
Body_Temp
Calories
1.0
0.8
0.6
0.4
0.2
0.0

Annexure-3: List of Output Screens

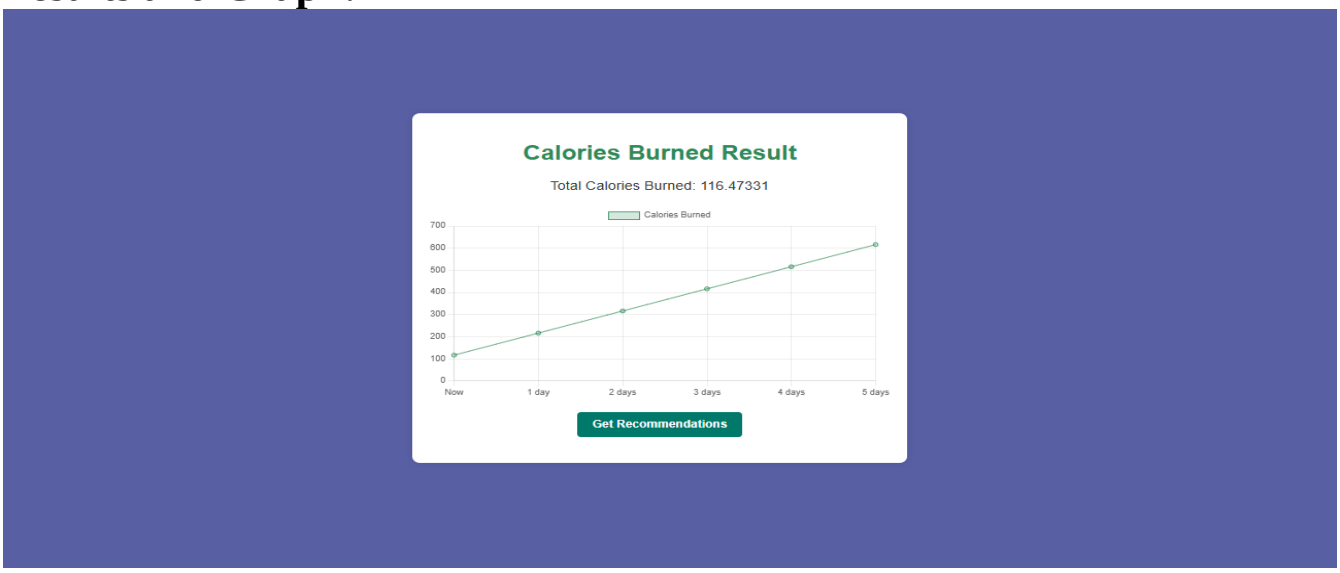
Metabometer - Calorie Predictor:



The front page of the Metabometer - Calorie Predictor is designed to be user-friendly and visually engaging. Users are prompted to enter the following details:

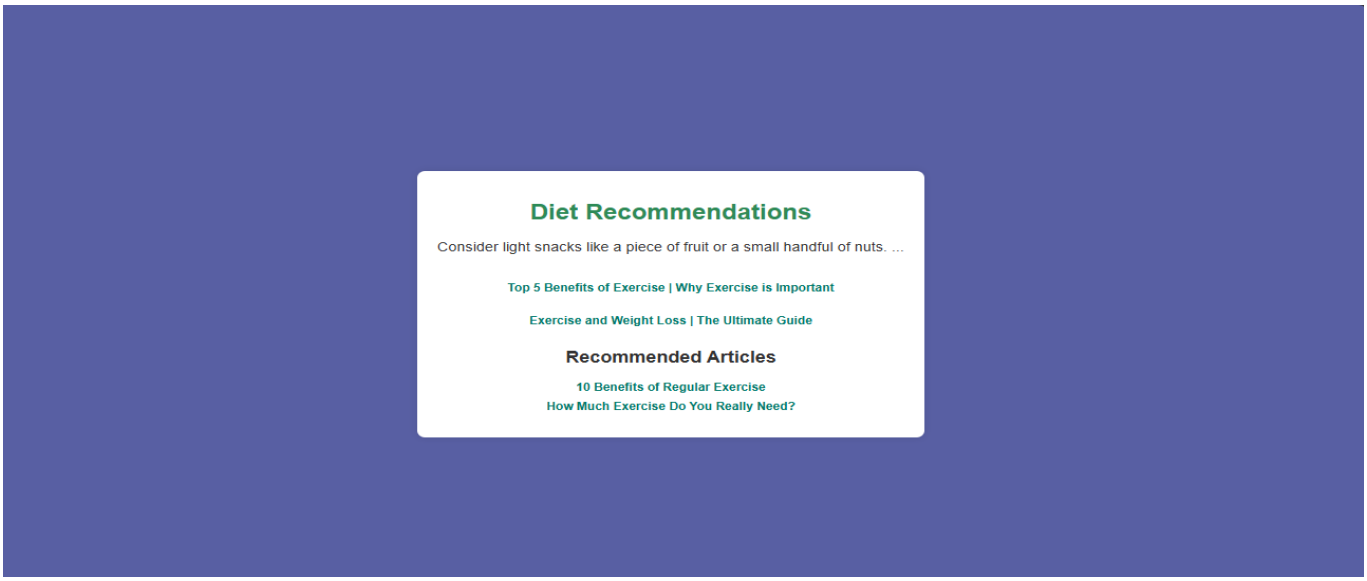
Gender: A dropdown menu where users select their gender.
Age: A text field for the user to enter their age.
Height (cm): A text field for height in centimeters.
Weight (kg): A text field for weight in kilograms.
Duration (minutes): A text field to enter the duration of activity in minutes.
Heart Rate: A text field to input the heart rate during the activity.
Body Temperature (°C): A text field for the current body temperature in degrees Celsius.
After filling in these details, users click on the Predict button.

Results and Graph:



Once the user clicks Predict, the system calculates the predicted calories burned based on the inputs. The results are displayed prominently on the page. Additionally, a graph is generated to visualize the calorie expenditure over the next 5 days, allowing users to track their progress.

Diet Recommendations:



Below the graph, there's a Recommendations button. When clicked, this button provides personalized tips and suggestions to help users optimize their activity, diet, and overall lifestyle to better achieve their health and fitness goals.