CHAPTER-1

DAY TO DAY ACTIVITIES

Basavarajeshwari Group of Institutions
BALLARI INSTITUTE OF TECHNOLOGY & MANAGEMENT
Autonomous Institute under VTU - Belagavi
"Jnana Gangotri" Campus, Bellary-Hospet Road, Near Allipura Village,
BALLARI - 583 104 (Karnataka)
Ph: 08392-237167/237153 Fax: 237197, e-mail: bitmbly@gmail.com
Website: www.bitm.edu.in

Internship Program on Python for BE-3rd Sem students
From 9th April to 28th May 2024 (During 3rd semester vacations).

Student Name: Kenchugundu pavani USN No: 3BR23CD045 Branch: CSE-DS

INDEX PAGE			
Day	Date	Content Covered	Signature of the faculty in-charge
1	9.09.24	Introduction to Python, Setup & Installation, First Python Program, Variables, Data Types, and Basic I/O	
2	10.09.24	Control Structures: If-else, Loops, Functions and Modules	
3	11.09.24	Lists, Tuples, and Dictionaries, File Handling	
4	12.04.24	Exception Handling, Practice exercises on Python basics	
5	13.09.24	Introduction to OOP, Classes, and Objects	
6	14.09.24	Inheritance, Polymorphism, and Encapsulation	
7	15.09.24	Abstract Classes and Interfaces	
8	17.09.24	Practice exercises on OOP concepts	
9	18.09.24	Introduction to DSA, Arrays, and Linked Lists	
10	19.09.24	Stacks and Queues	
11	20.09.24	Trees and Graphs	
12	21.09.24	Searching and Sorting Algorithms	
13	23.09.24	Project Building & Presentations	

14	24.09.24	Project Building & Presentations	
15	25.09.24	Project Building & Presentations	
16	26.09.24	Project Building & Presentations	
17	27.09.24	Project Building & Presentations	
18	28.09.24	Project Building & Presentations	

CHAPTER-2

COMPANY PROFILE

Company Name: EZ Trainings and Technologies Pvt. Ltd.

Introduction:

EZ Trainings and Technologies Pvt. Ltd. is a dynamic and innovative organization dedicated to providing comprehensive training solutions and expert development services. Established with a vision to bridge the gap between academic learning and industry requirements, we specialize in college trainings for students, focusing on preparing them for successful placements. Additionally, we excel in undertaking development projects, leveraging cutting-edge technologies to bring ideas to life.

Mission:

Our mission is to empower the next generation of professionals by imparting relevant skills and knowledge through specialized training programs. We strive to be a catalyst in the career growth of students and contribute to the technological advancement of businesses through our development projects.

Services:

College Trainings:

- Tailored training programs designed to enhance the employability of students.
- Industry-aligned curriculum covering technical and soft skills.
- Placement assistance and career guidance.

Development Projects:

- End-to-end development services, from ideation to execution.
- Expertise in diverse technologies and frameworks.
- Custom solutions to meet specific business needs.

Locations: Hyderabad | Delhi NCR

At EZ Trainings and Technologies Pvt. Ltd., we believe in transforming potential into excellence

CHAPTER-3

ABSTRACT

The system proposed for the Custom Workout Planner is designed to streamline and personalize workout management for athletes, allowing both athletes and coaches to create, monitor, and track workout progress. It would consist of five primary components or pages, each fulfilling a specific role in the overall process of managing athletic training.

The first component of the system would revolve around Athlete Profile Management. This component serves as the foundational layer of the application. It enables the creation and management of athlete profiles, where each athlete's details, such as personal information (age, gender, fitness level, and training goals), are stored. This data is crucial because it provides the necessary input for generating tailored workout plans. Athletes and coaches alike can update this information as needed, ensuring that workout plans remain relevant and adjusted to the athlete's evolving fitness journey.

The second component of the system focuses on Workout Plan Management, encompassing the essential operations for workout plan creation, updating, viewing, and deletion (CRUD operations). This component allows users to input specific workout details such as exercises, duration, intensity, and repetitions. It gives coaches the flexibility to design unique workout routines that meet general training standards or specific needs. This page will also allow users to browse through existing workout plans, apply filters based on workout type, and update plans as necessary, thus offering a comprehensive tool for workout management.

The third component is where the system's ability to personalize workouts truly shines through. The Custom Workout Generation page generates workout plans that are specific to the needs of individual athletes. By drawing from the information in the athlete's profile—such as their goals, fitness level, and any specific requirements—the system automatically crafts workout routines designed to maximize performance. Coaches and athletes can review these plans and make any adjustments before approving them for execution, thus ensuring that each workout plan is as tailored as possible.

The fourth component focuses on Workout Assignment, which plays a key role in organizing when and how the workouts are carried out. Once a workout plan is created or generated, it needs to be assigned to an athlete. This page allows coaches to schedule workout sessions for athletes on a daily, weekly, or monthly basis. It also provides a visual calendar view, allowing users to see how workouts are distributed over time. By assigning specific workouts to athletes and setting deadlines for their completion, this page ensures that athletes are held accountable for their training routines.

Finally, the fifth component, Workout Completion and Progress Monitoring, provides critical insights into how well an athlete is adhering to and benefiting from their training regimen. This page offers a dashboard that tracks workout completion, giving both coaches and athletes access to detailed statistics about the number of workouts completed, the time taken, and any areas where improvement might be needed. This page would feature graphical representations of workout data—such as progress charts and performance

trends—enabling a clear and visual understanding of an athlete’s progress over time. Moreover, this page providing valuable feedback to both athletes and coaches for future planning.

In essence, the Custom Workout Planner system provides an integrated solution for managing athletic training, starting from athlete profiling and workout plan creation to monitoring workout execution and progress. Each page of the system complements the others, creating a seamless workflow that enables personalized, data-driven workout planning and tracking. This comprehensive approach ensures that athletes can train effectively while coaches can easily oversee and adapt their training strategies based on real-time data.

CHAPTER-4

INTRODUCTION OF THE PROJECT

The project titled "Custom Workout Planner" is focused on building a system that generates personalized workout plans for athletes and monitors their completion. The primary goal is to create a tool that automates the creation of workout routines tailored to individual athlete needs while providing a way to track and oversee workout progress.

The core functionality includes:

CRUD operations: This feature enables users to Create, Read, Update, and Delete workout plans.

Custom workout generation: The system will dynamically generate workout plans based on the athlete's ID, adapting routines to specific fitness goals or requirements.

Workout completion monitoring: The system will track athletes' progress, logging when workouts are completed and providing feedback on their adherence to the workout plan.

To implement this, the solution will contain various classes representing different entities, such as WorkoutPlan and Athlete. The class WorkoutPlanner will serve as the managing component responsible for handling all the operations, from generating custom workouts to tracking progress and completion.

In essence, this project aims to simplify and enhance the workout planning process for athletes and coaches by offering both customization and real-time monitoring capabilities.

CHAPTER-5

MODULE DESCRIPTION

The project’s module descriptions in the Custom Workout Planner would likely focus on various key components responsible for the creation, management, and tracking of workout plans for athletes. Here's a breakdown of the possible modules based on the image:

1. WorkoutPlan Module

Description: This module represents the core workout structure. It includes attributes like exercise names, sets, reps, durations, and intensity. The WorkoutPlan class will be used to create, update, view, or delete individual workout plans, handling all CRUD operations for the workouts.

Functions:

Create new workout plans.

Update or modify existing plans.

Delete outdated or irrelevant plans.

Retrieve and display details of specific workout plans.

2. Athlete Module

Description: This module manages athlete-specific information, such as their personal data (age, fitness goals, skill level, etc.). The Athlete class stores the relevant information that is used to generate customized workout plans for each individual.

Functions:

Store athlete profiles, including fitness level and goals.

Retrieve and update athlete data.

Use the athlete’s ID to generate custom workout routines tailored to their needs.

3. CustomWorkout Generation Module

Description: The generate_custom_workouts(athlete_id) function resides here, generating workout plans based on an athlete’s profile. This module will customize workout plans according to the athlete’s fitness goals, body type, and required training intensity.

Functions:

Generate personalized workouts based on athlete profile inputs.

Adjust workout parameters dynamically as per specific goals or needs.

Enable manual modifications for customization by coaches or users.

4. WorkoutCompletion Monitoring Module

Description: This module focuses on tracking athlete progress. The monitor_workout_completion(completion_data) function keeps track of which workouts have been completed, how well the athlete adhered to the schedule, and provides feedback on their progress.

Functions:

Log completed workouts and track progress.

Generate completion reports or summaries.

Provide insights into performance improvements or areas needing more focus.

5. WorkoutPlanner Management Module

Description: This is the central module, handling the main operations of the system. The WorkoutPlanner class coordinates all CRUD operations, custom workout generation, and completion tracking. It serves as the system manager, overseeing all interactions between workout plans and athletes.

Functions:

Manage CRUD operations across the system.

Interface with other modules to generate and assign workout plans.

Ensure consistency in workout tracking and feedback mechanisms.

These modules, when implemented together, will create a robust and flexible system that simplifies the management of custom workout routines while providing insightful tracking and monitoring of athlete progress.

CHAPTER-6

ALGORITHM

The algorithm for this workout planner code follows a logical flow that allows users to create, view, update, delete, and mark workout plans as complete for different athletes. Here's the step-by-step algorithm breakdown:

1. Class Definitions

Athlete Class:

Attributes:

name: The name of the athlete.

age: The age of the athlete.

goal: The fitness goal of the athlete.

Methods:

init: Constructor to initialize an athlete object with name, age, and goal.

WorkoutPlan Class:

Attributes:

athlete: The associated athlete object.

exercises: A list of exercises in the workout plan.

completed: A boolean to track whether the workout plan is completed (initially False).

Methods:

init: Constructor to initialize the workout plan with an athlete and exercises.

display_plan: Displays the workout plan.

Mark complete: Marks the workout as completed and prints a message.

Custom Workout Planner Class:

Attributes:

workout_plans: A list to store multiple workout plans.

Methods:

init: Initializes the planner with an empty list of workout plans.

create_workout_plan: Prompts the user to create a workout plan by inputting exercises, then adds the plan to the list.

view_workout_plans: Displays all stored workout plans.

update_workout_plan: Allows the user to select a workout plan and update its exercises.

delete_workout_plan: Allows the user to select and delete a workout plan.

mark_completion: Marks a selected workout plan as complete.

2. Main Program Flow:

1. Initialize the Workout Planner:

An instance of the CustomWorkoutPlanner class is created.

2. Main Loop:

Display a menu of options.

Depending on the user’s choice, execute one of the following operations:

1. Create Workout Plan:

Collect details about the athlete (name, age, goal).

Call create_workout_plan to add exercises and store the plan.

2. View Workout Plans:

Call view_workout_plans to display all available workout plans.

3. Update Workout Plan:

Call view_workout_plans to list the plans.

Select the plan to update and modify its exercises.

4. Delete Workout Plan:

Call view_workout_plans to list the plans.

Select a plan to delete from the list.

5. Mark Workout Completion

Call view_workout_plans to list the plans.

Mark a selected plan as completed.

6. Exit Program:

Break out of the loop to exit the application.

3. Input Validation and Error Handling:

Each operation checks if a valid index or input is provided by the user.

If invalid input is given, the program displays an error message and prompts the user to try again.

High-Level Algorithm:

1. Initialize the CustomWorkoutPlanner.

2. Display a menu of actions (create, view, update, delete, mark completion, exit).

3. Based on user input:

Create Workout Plan: Collect athlete data, exercises, and store the plan.

View Plans: List all workout plans with details.

Update Plan: Modify a workout plan's exercises.

Delete Plan: Remove a selected plan.

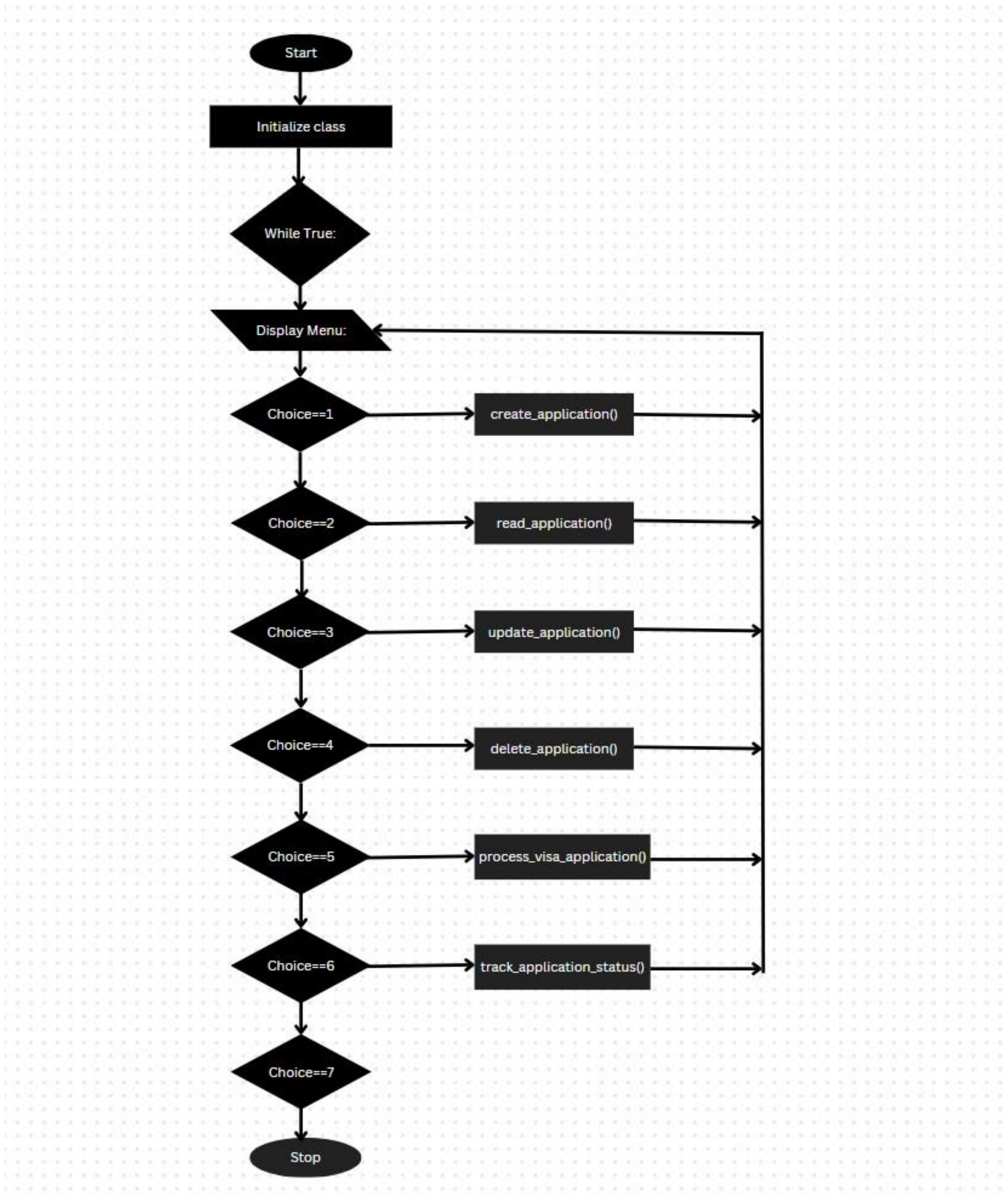
Mark Completion: Mark a selected workout plan as complete.

4. Loop back to display the menu until the user chooses to exit.

This design is an interactive, menu-driven approach for managing workout plans for athletes.

CHAPTER-7

FLOWCHART



CHAPTER-8

SOURCE CODE

```
# Define classes for Workout Plan and Athlete


class Athlete:

    def _init_(self, name, age, goal):

        self.name = name

        self.age = age

        self.goal = goal


class WorkoutPlan:

    def _init_(self, athlete, exercises):

        self.athlete = athlete

        self.exercises = exercises

        self.completed = False


    def display_plan(self):

        print(f"Workout Plan for {self.athlete.name}:")

        for i, exercise in enumerate(self.exercises, 1):

            print(f"{i}. {exercise}")


    def mark_complete(self):

        self.completed = True

        print(f"{self.athlete.name}'s workout plan marked as complete!")


class CustomWorkoutPlanner:

    def _init_(self):

        self.workout_plans = []
```

```
def create_workout_plan(self, athlete):
    exercises = []
    num_exercises = int(input("How many exercises do you want to add? "))
    for i in range(num_exercises):
        exercise = input(f"Enter exercise {i + 1}: ")
        exercises.append(exercise)

    new_plan = WorkoutPlan(athlete, exercises)
    self.workout_plans.append(new_plan)
    print("Workout plan created successfully!")

def view_workout_plans(self):
    if not self.workout_plans:
        print("No workout plans available.")
    else:
        for i, plan in enumerate(self.workout_plans, 1):
            print(f"\nPlan {i}:")
            plan.display_plan()

def update_workout_plan(self):
    self.view_workout_plans()
    plan_index = int(input("Select a plan to update (by number): ")) - 1
    if 0 <= plan_index < len(self.workout_plans):
        plan = self.workout_plans[plan_index]
        print("Updating workout plan...")
        plan.exercises.clear()
        num_exercises = int(input("How many new exercises do you want to add? "))
        for i in range(num_exercises):
            exercise = input(f"Enter new exercise {i + 1}: ")
```

```
                plan.exercises.append(exercise)
            print("Workout plan updated!")
        else:
            print("Invalid selection!")

    def delete_workout_plan(self):
        self.view_workout_plans()
        plan_index = int(input("Select a plan to delete (by number): ")) - 1
        if 0 <= plan_index < len(self.workout_plans):
            del self.workout_plans[plan_index]
            print("Workout plan deleted.")
        else:
            print("Invalid selection!")

    def mark_completion(self):
        self.view_workout_plans()
        plan_index = int(input("Select a plan to mark as complete (by number): ")) - 1
        if 0 <= plan_index < len(self.workout_plans):
            self.workout_plans[plan_index].mark_complete()
        else:
            print("Invalid selection!")

# Main flow for the workout planner
def main():
    planner = CustomWorkoutPlanner()
    while True:
        print("\nCustom Workout Planner")
        print("1. Create Workout Plan")
        print("2. View Workout Plans")
print("3. Update Workout Plan")
print("4. Delete Workout Plan")
print("5. Mark Workout Completion")
print("6. Exit")

choice = input("Enter your choice: ")

if choice == '1':
    name = input("Enter athlete's name: ")
    age = input("Enter athlete's age: ")
    goal = input("Enter athlete's goal (e.g., strength, endurance): ")
    athlete = Athlete(name, age, goal)
    planner.create_workout_plan(athlete)

elif choice == '2':
    planner.view_workout_plans()

elif choice == '3':
    planner.update_workout_plan()

elif choice == '4':
    planner.delete_workout_plan()

elif choice == '5':
    planner.mark_completion()

elif choice == '6':
    print("Exiting...")
    break
```

```
        else:
            print("Invalid choice. Please try again.")

if _name_ == "_main_":
    main()
```

CHAPTER-9

OUTPUT

```
Custom Workout Planner

1. Create Workout Plan

2. View Workout Plans

3. Update Workout Plan

4. Delete Workout Plan

5. Mark Workout Completion

6. Exit

Enter your choice: 1

Enter athlete's name: lahari

Enter athlete's age: 19

Enter athlete's goal (e.g., strength, endurance): strength

How many exercises do you want to add? 3

Enter exercise 1: push-ups

Enter exercise 2: squats

Enter exercise 3: deadlifting

Workout plan created successfully!


Custom Workout Planner

1. Create Workout Plan

2. View Workout Plans

3. Update Workout Plan

4. Delete Workout Plan

5. Mark Workout Completion

6. Exit

Enter your choice: 2


Plan 1:
```

```
Workout Plan for lahari:
1. push-ups
2. squats
3. deadlifting

Custom Workout Planner
1. Create Workout Plan
2. View Workout Plans
3. Update Workout Plan
4. Delete Workout Plan
5. Mark Workout Completion
6. Exit
Enter your choice: 3

Plan 1:
Workout Plan for lahari:
1. push-ups
2. squats
3. deadlifting
Select a plan to update (by number): 1
Updating workout plan...
How many new exercises do you want to add? 1
Enter new exercise 1: push-ups
Workout plan updated!

Custom Workout Planner
1. Create Workout Plan
2. View Workout Plans
3. Update Workout Plan
```

```
4. Delete Workout Plan

5. Mark Workout Completion

6. Exit

Enter your choice: 4


Plan 1:

Workout Plan for lahari:

1. push-ups

Select a plan to delete (by number): 1

Workout plan deleted.


Custom Workout Planner

1. Create Workout Plan

2. View Workout Plans

3. Update Workout Plan

4. Delete Workout Plan

5. Mark Workout Completion

6. Exit

Enter your choice: 5

No workout plans available.

Select a plan to mark as complete (by number): 1

Invalid selection!


Custom Workout Planner

1. Create Workout Plan

2. View Workout Plans

3. Update Workout Plan

4. Delete Workout Plan

5. Mark Workout Completion
```

```
6. Exit
Enter your choice: 6
Exiting...
```

CHAPTER-10

CONCLUSION

Conclusion for Custom Workout Planner Project

The Custom Workout Planner project is a comprehensive and scalable system that addresses the need for personalized workout plans and efficient management of athletes' fitness journeys. Through a well-structured solution incorporating CRUD operations, customizable workout generation, and performance tracking, this project aims to transform how athletes interact with and manage their fitness routines. In this conclusion, we will elaborate on the key strengths of the system, the potential it holds for future development, and its overall impact on the fitness industry.

1. Personalization and Flexibility :
   One of the most significant aspects of the Custom Workout Planner is its focus on personalization. Each athlete is unique in their fitness goals, physical capabilities, and preferences. The system's ability to generate custom workouts tailored to specific needs ensures that users can follow routines that align closely with their fitness objectives. Whether an athlete is training for a marathon or simply working to improve their general health, the planner can adapt workouts accordingly.This flexibility sets the system apart from generic workout apps or templates, which often fail to consider the nuances of individual fitness levels. By focusing on individualized plans, the Custom Workout Planner encourages long-term commitment and motivation, as users are more likely to engage with a program that suits their specific needs.Moreover, the customization extends beyond just the workout plans themselves. Athletes can adjust routines based on changing schedules, recovery needs, or emerging goals, ensuring that the workout plan remains dynamic and responsive to personal circumstances. This adaptability makes the planner a lifelong tool, capable of evolving alongside the athlete

2. Comprehensive Workout Management with CRUD Functionality:
   The inclusion of CRUD (Create, Read, Update, Delete) functionality ensures that the system remains robust and easy to use. Fitness routines often need constant adjustment—whether due to progress, injury, or new goals. The ability to create, modify, and delete workouts provides both trainers and athletes the autonomy to manage their plans in real-time.This ensures that athletes are not bound by rigid structures and can experiment with different types of workouts to find what works best for them. The CRUD functionality empowers users to take control of their fitness journey, making changes as needed without starting from scratch. Additionally, the ability to read and access past workouts offers valuable insights into progress and helps in planning future routines. By retaining historical data, the system can provide athletes with a clearer picture of their fitness trajectory, enabling more informed decisions.From a technical perspective, the CRUD system lays the groundwork for a scalable and

maintainable architecture. Future enhancements, such as integrating new workout types or expanding data points for tracking performance, can be incorporated seamlessly without disrupting the core functionality. This scalability ensures that the planner can grow alongside user demand and technological advancements in the fitness industry.

3. Monitoring and Accountability:
   The inclusion of a monitoring system to track workout completion is another essential feature. In any fitness journey, consistency is critical. However, staying consistent can be a challenge without the proper tools to track progress. By allowing users to monitor their workout completion, the planner acts as an accountability partner, ensuring that athletes remain on track with their fitness goals.This monitoring not only boosts motivation but also helps users recognize patterns in their behavior. For instance, athletes may discover that they tend to skip workouts on certain days or after specific routines. Armed with this knowledge, they can adjust their plans or seek guidance to stay motivated. The planner can potentially offer reminders, progress charts, or even milestone celebrations to further encourage adherence to workout plans.Furthermore, the tracking of completion data offers an opportunity for deeper insights. As the system accumulates data over time, trends can be analyzed to provide users with personalized feedback. This could range from recommending rest days based on overtraining signs to suggesting new workout routines that align with the user's progress. As such, the Custom Workout Planner holds the potential to become a data-driven fitness tool that evolves from simple tracking to a smart, adaptive assistant

.4. Usability and User Experience:
The success of any fitness tool hinges on its usability. The Custom Workout Planner project has been designed with simplicity in mind. Whether the user is an experienced athlete or a beginner, the intuitive interface and logical structure of the system ensure that it can be navigated effortlessly. The focus on clean code architecture, with dedicated classes for different components such as WorkoutPlan, Athlete, and WorkoutPlanner, ensures that the system remains organized and easy to extend.In addition, the user-centric design promotes a seamless experience. From creating a workout plan to monitoring its completion, each step of the process is streamlined. Users can easily modify their plans, track their progress, and adapt their workouts without needing technical expertise. This ease of use encourages higher engagement, which is crucial for maintaining long-term fitness routines.As a technical POC (Proof of Concept), the solution demonstrates how a complex system can still remain user-friendly. The code structure is designed for scalability, meaning future features such as mobile app integration, social sharing options, or even AI-driven suggestions can be added without overhauling the existing system. This flexibility makes it a future-proof solution that can evolve with user needs and technological trends

.5. Future Prospects and Potential Enhancements:
The Custom Workout Planner POC provides a strong foundation for future development. Several potential enhancements can further amplify the system's value. For instance, integrating machine learning to analyze workout patterns could help in generating smarter workout suggestions based on the athlete's progress. By leveraging data over time, the system can offer adaptive plans that evolve with the user's fitness level, promoting continual improvement without plateaus.Moreover, adding features such as social integration could foster community engagement, where athletes share their progress, compete in challenges, or simply motivate each other.

CHAPTER-11

REFERENCES

→Google

→https://youtu.be/bSrm9RXwBaI?si=_97bfU2bMSxOd1Pd

→https://www.geeksforgeeks.org/object-oriented-programming-oops-concept-in-python/