FITNESS TRACKER APP

Group6

Elena Murgia, Paula Calvo, Deborah Kamanya, Nora Rodriguez, Amber Mahmood, Ekta Shah

Table of Contents

Introduction *2*

Aims and objectives of the project *2*

Background *2*

Core functionality *3*

User management **3**

Main Menu Options **4**

Workout Menu **4**

Exercise search and storage *5*

Specifications and Design *5*

Functional VS Non-functional Requirements **5**

Design and Architecture *6*

Implementation and Execution *7*

- **Development approaches and team member roles** **7**
- **Implementation process** **8**

Testing and Evaluation *9*

Testing strategy **9**

Functional and user testing **9**

System limitations **10**

Conclusion *10*

Introduction

This report is organised into several key sections. We begin with an Introduction outlining the goals and objectives of the Fitness Tracker App. The Background section describes the core functionality, including user flows and the application's main features. Specifications and Design cover the functional and non-functional requirements, followed by a breakdown of the application's architecture and design decisions. The Implementation and Execution section discusses the development process, the main challenges encountered, and the tools and technologies used. In Testing and Evaluation, we explain the testing strategy, methods used and system limitations. Finally, the report concludes with a summary of the outcomes and reflections on the development experience.

Aims and objectives of the project

The project aims to develop a user-friendly fitness tracking application that efficiently supports planning, tracking and maintaining a workout routine. The application provides exercise suggestions using an external API, personalisation of the exercises (via user input), and a workout log to track the user's workouts.
The main purposes of the project are:

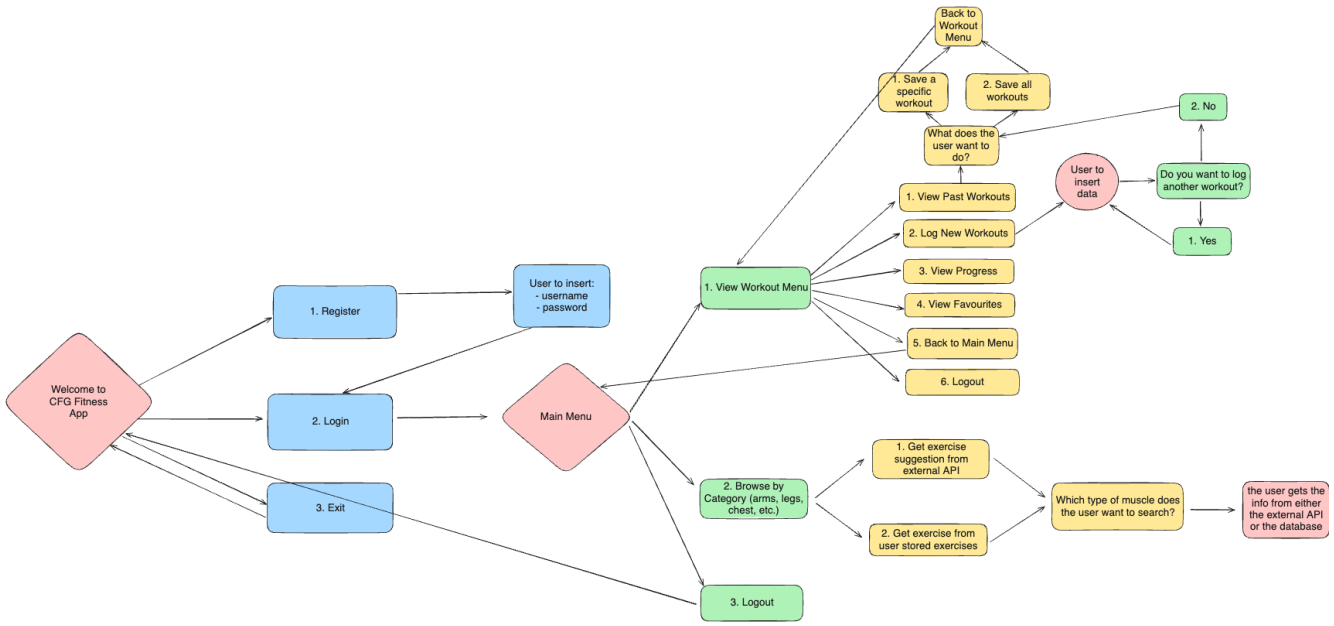
- Integrate an external exercise API (API-Ninjas Exercises) for automated suggestions.
- To allow users to log workouts (MySQL database), including the option to log the exercise sets and view their workout history.
- To allow users to save their workouts to a separate file.
- To enable users to save their chosen exercises from the API and/or create their own.

Background

The Fitness tracker App is designed as a system that allows users to register, log in, browse exercises, log their workouts including the exercise sets and the option to save their workouts to a separate file. The app aims to create a personalised experience where users can easily find exercises, log their workouts, and monitor their improvements while minimising external API calls.

Core functionality

As shown in this diagram:



User management

Users can register with a username and password or log in if they already have an account. After authentication, users are directed to the main menu to navigate different options.

```
Welcome to the CFG Fitness App
Select a number to choose an option:
1. Login
2. Register
3. Exit
Choose an option: 1
Username: paula_C
Password:
Login successful. Welcome, paula_C!

Logged in
```

Main Menu Options

Users can choose between View the Workout Menu, Browse exercises by category or Logout.

```
MAIN MENU
Select a number to choose an option:
1. View Workout Menu
2. Browse by Category (arms, legs, chest, etc.)
3. Logout
>> 1
```

Workout Menu

Users can view their past workouts, log new ones, track their progress and log out from the Workout Menu.

```
WORKOUT MENU
1. View Past Workouts
2. Log New Workout
3. View Progress
4. View Favourites
5. Go back to the Main Menu
6. Logout
>>
```

Users can choose to View Past Workouts and Log New workouts. Users can also save their workouts to a file.

```
WORKOUT MENU
1. View Past Workouts
2. Log New Workout
3. View Progress
4. View Favourites
5. Go back to the Main Menu
6. Logout
>> 2

Available Exercises:
1. Push-Up (chest, beginner)
2. Squat (legs, beginner)
3. Bicep Curl (arms, intermediate)
4. Plank (core, intermediate)
5. Deadlift (back, advanced)

Enter the number of the exercise you want to log: 1
Duration (in minutes): 20
Any notes or comments that you would like to add? No
Do you want to add sets for this workout? (y/n): n
Workout logged without sets
```

Also, users can go back to the Main Menu or Logout.

Exercise search and storage

Users who want new exercise suggestions when searching for exercises can manually start a new API search or access an existing exercise previously saved in the database.
Exercises pulled from the API can be saved into an Exercises table.

Specifications and Design

Functional VS Non-functional Requirements

- **Functional requirements:**
    - Users must be able to register and log in using a unique username and password.
    - Authenticated users must be able to log workouts, view workout history
    - The app must provide exercise suggestions using Ninjas API.
    - Users must be able to save favourite exercises or create custom ones.

- Users should be able to search exercises by muscle group.
- The system must store all user and workout data in a MySQL database.
- The terminal interface must provide menu-based navigation for user actions.

- **Non-functional requirements:**

  - Usability: The interface should be intuitive and easy to navigate via the terminal.
  - Reliability: The app should handle errors correctly.
  - Maintainability: Code should be modular and well-documented for future updates or debugging.
  - Scalability: The architecture should allow for future upgrades.

Design and Architecture

- **Terminal interface:**
  Handles user input and the navigation logic.
  Presents the menu to the user(register, login, main menu, workout logging, search, exit).

- **Backend modules:**

1. User management:
   - Register/login functions
   - Handles session management.

2. Exercise management:
   - Search in the Exercises table.
   - API integration with the Ninjas API Exercises for external exercise data.

3. Workout logging:
   - Save the user's workouts to the database.

4. Diary:
   - Retrieve past workouts based on timeframes.

5. Data storage:
   - Connects to the database (MySQL)
   - Tables:
     Users
     Exercises
     Exercise sets
     Workout_Logs

Implementation and Execution

• Development approaches and team member roles

We made early design decisions such as incorporating a terminal interface, which API to use (Ninjas API exercises), and the database used (MySQL).

Implementing the Fitness tracker App followed a collaborative and flexible development process, incorporating several Agile development principles throughout the project:

- Sprint-based planning: We divided our development into weekly sprints.
- Task table ownership: We used a shared table in Notion to track all implementation tasks. Each team member selected the tasks for which they would be responsible and updated their progress through the Slack chat.

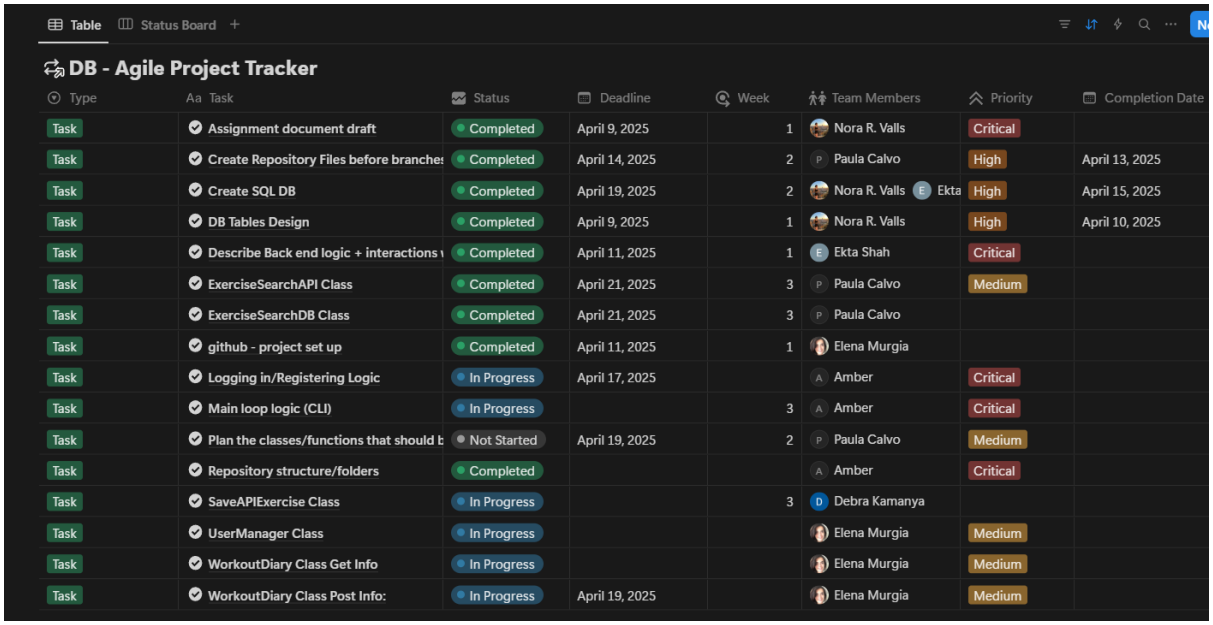


Table | Status Board +

DB - Agile Project Tracker

Type	Task	Status	Deadline	Week	Team Members	Priority	Completion Date
Task	Assignment document draft	Completed	April 9, 2025	1	Nora R. Valls	Critical	
Task	Create Repository Files before branche	Completed	April 14, 2025	2	Paula Calvo	High	April 13, 2025
Task	Create SQL DB	Completed	April 19, 2025	2	Nora R. Valls, Ekta	High	April 15, 2025
Task	DB Tables Design	Completed	April 9, 2025	1	Nora R. Valls	High	April 10, 2025
Task	Describe Back end logic + interactions	Completed	April 11, 2025	1	Ekta Shah	Critical	
Task	ExerciseSearchAPI Class	Completed	April 21, 2025	3	Paula Calvo	Medium	
Task	ExerciseSearchDB Class	Completed	April 21, 2025	3	Paula Calvo		
Task	github - project set up	Completed	April 11, 2025	1	Elena Murgia		
Task	Logging in/Registering Logic	In Progress	April 17, 2025		Amber	Critical	
Task	Main loop logic (CLI)	In Progress		3	Amber	Critical	
Task	Plan the classes/functions that should b	Not Started	April 19, 2025	2	Paula Calvo	Medium	
Task	Repository structure/folders	Completed			Amber	Critical	
Task	SaveAPIExercise Class	In Progress		3	Debra Kamanya		
Task	UserManager Class	In Progress			Elena Murgia	Medium	
Task	WorkoutDiary Class Get Info	In Progress			Elena Murgia	Medium	
Task	WorkoutDiary Class Post Info:	In Progress	April 19, 2025		Elena Murgia	Medium	

- Video call retrospectives: At the end of each sprint, we held a video call using the Gather platform to share updates, demonstrate progress, discuss challenges and agree on the following tasks.

Implementation process

Challenges

- Time management: Due to scheduling conflicts and members' availability, we began coding later than planned.

- External API limitations: Although the Ninjas API was advertised as offering free access, we discovered it was unavailable during certain times of the day unless upgraded to a premium account. This restriction disrupted our testing and development of the exercise search functionality.

- Notion collaboration platform limitations: After two weeks of using Notion to assign tasks and share project documentation, our free access as a team expired. As a result, we transitioned to Google Docs to continue managing and sharing our tasks.

- Testing and debugging: Testing was necessary to ensure no conflicts before merging branches, especially since some code segments were similar. When conflicts did occur, identifying and successfully resolving their sources through debugging required careful attention and collaboration .

- Communication and collaboration: We encountered some collaboration challenges during the development process. Certain team members contributed portions of the codebase at the beginning of the project, making significant changes to the agreed-upon structure, but were later unavailable to explain their implementations. This led to unanticipated structural changes that required time and effort to understand and adapt to. Additionally, a few contributors could not complete their assigned tasks or provide updates, which resulted in the need to revise or remove incomplete code.

Achievements

Despite the challenges faced during development, the team successfully delivered a functional Fitness Tracker App. Some team members demonstrated strong adaptability by restructuring the codebase, resolving merge conflicts, and taking ownership of abandoned tasks.

Collaboration among the remaining members was efficient and solution-focused, allowing the team to overcome setbacks.

The project provided a valuable learning experience in agile teamwork, version control, and problem-solving under pressure.

- **Tools and Libraries**

  - **Libraries**

    - requests
    - flask
    - unittest
    - unittest.mock
    - pytest
    - json
    - Random
    - Datetime
    - mysql.connector
    - load_dotenv

  - **Tools**
    - mySQL database
    - API Ninjas exercise
    - Flask
    - PyCharm

Testing and Evaluation

Testing strategy

Our testing strategy combined automated testing using pytest with mock objects via unittest.mock to simulate user inputs and external dependencies.
Debug messages were temporarily included in key code areas to trace logic and identify faulty behaviour during the development.
Exception handling was also integrated to catch and log errors.

Functional and user testing

Functional testing verified that each feature met its requirements. Input handling was validated manually by simulating user interaction in the terminal.
Exceptions were intentionally triggered to ensure that invalid inputs did not cause crashes and that appropriate messages were shown to the user.

System limitations

While most features were tested, some user input cases may not have been exhaustively covered. Debug messages were helpful during development, but were removed in the final steps. Exception handling covers most predictable errors, but unexpected failures may still occur.

Conclusion

The Fitness-tracker App successfully combines local database features with real-time data from an external Exercise API, creating a flexible and user-friendly workout logging experience. Users can securely register and log in, track their workouts with detailed sets and notes, and view progress over time using customisable date filters. The integration with the API Ninjas Exercise API adds a dynamic layer by allowing users to discover new exercises tailored to specific muscle groups, helping to keep their routines engaging and effective. This blend of internal and external resources showcases the app's scalability and potential to evolve into a comprehensive personal fitness assistant. Future improvements may include adding features to collaborate with other users to create a competitive environment, graphical progress charts, and improved validation for user inputs.