GYMTECH Report

“Get fit at your fingertips"

Seif El-Tahtawy 6825

Ali Hassan Mekky 6850

Mohamed Alaa El-Zeftawy 6886

Instructor: Dr. Youssry Taha

Team Number: 17

Project URL: https://github.com/SWE-SP23/term-project-g17

12.05.2023

Glossary of Terms

Administrator: Someone who observes the website and is responsible for data addition and manipulation.

Application: The main program that the user will be interacting with.

Database: Server that will contain user data and workouts.

Developer: Someone who is responsible for creating the website's front-end and back-end.

Graphical User Interface (GUI): A type of interface that allows the user to interact with the graphical components.

Gym tracker app: A mobile application that allows users to track their fitness activities and progress, set goals, and receive recommendations for workouts.

User: A person who downloads and uses the gym tracker app to track their fitness journey.

Workout plan: A set of exercises designed to help users achieve specific fitness goals, such as weight loss, muscle gain, or endurance training.

Progress tracker: A feature of the gym tracker app that allows users to monitor their fitness progress, including weight, body measurements, and workout performance.

Fitness goals: Specific objectives that users aim to achieve through regular exercise and healthy nutrition, such as building muscle, losing weight, or improving cardiovascular health.

Feedback mechanism: A system within the gym tracker app that allows users to provide feedback and suggestions for improving the app's functionality and user experience.

Exercise Library: A collection of workout videos, instructions, and demonstrations available on the gym tracker app for users to access and follow.

Workout logs: A record of users' completed workouts, including the exercises performed, sets, and repetitions, and the duration of the workout.

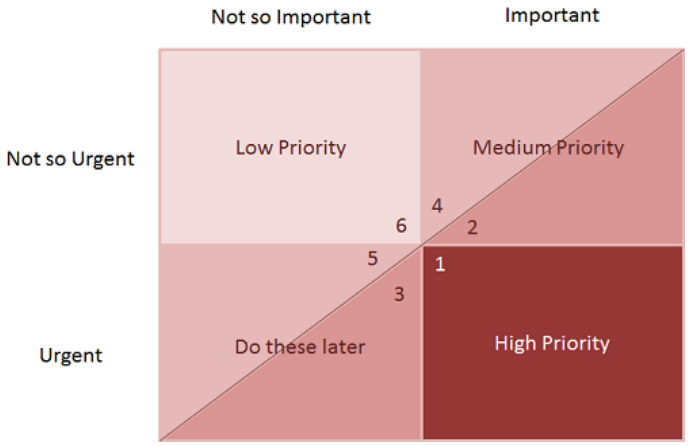
Gamification: The process of turning fitness goals and activities into a game-like experience, using rewards, challenges, and points systems to motivate users and enhance their engagement with the app.

User Requirements

Users Can:

- create an account and log in securely.
- log their workouts, including exercise type, duration, and sets/reps.
- View and track their workout progress over time.
- View the recommended body weight and calories per day.
- Earn badges depending on their progress.
- Create customized workouts.
- View videos for exercises to avoid injuries.
- View nearby gyms.
- Can track the number of burnt calories per day.
- Track the frequency of training per month.
- Track the volumes of the last sessions.
- Post feedback to participate in improvements.

System Requirements



Functional Requirements:

Identifier	PW	Requirement
REQ1	2	The app must have a user-friendly and intuitive interface for ease of use.
REQ2	2	The app should have a feature for users to save their favorite workouts and fitness plans for future reference.
REQ3	1	The app must have a secure database to store user information, workout data, and payment information (if applicable).
REQ4	4	The app should have a feature for users to give feedback and rate workout sessions or fitness plans.
REQ5	4	The app should be able to generate reports for the admin to view the feedbacks and gyms.
REQ6	2	The app should provide charts to analyze the user's progress.
REQ7	5	The app should have forms to allow the admin to add gyms and exercises.
REQ8	6	The app should have a tracker to attach a badge to a user when he accomplishes a goal.

Non-Functional Requirements:

Identifier	PW	Requirement
REQ9	2	The app must be fast and responsive to user inputs.
REQ10	1	The app should be secure and protect user data from unauthorized access.
REQ11	4	The app should be compliant with data protection laws and regulations in the user's country.
REQ12	4	The app should be regularly updated to fix bugs, improve performance, and add new features.
REQ13	2	The app should be tested thoroughly to ensure it meets all user and system requirements.

Software Process

Software Process Type

An Agile software development process would be most suitable for the development of such an app for the following reasons:

1. Flexibility: The agile method allows for flexibility in the development process, as requirements and priorities can change over time. This is particularly useful in the development of a gym tracking app, where user needs and feedback may change as they use the app.
2. Iterative approach: The agile method uses an iterative approach to development, where small parts of the app are developed and tested in short time periods. This allows for quick feedback and adjustments to be made, which is important in the development of a gym-tracking app where users may want to see immediate results.
3. Collaboration: The agile method emphasizes collaboration between the development team and the stakeholders, including the users. This is important in the development of a gym tracking app, as user input and feedback are crucial to ensure the app meets their needs and is user-friendly.
4. Continuous improvement: The agile method promotes continuous improvement

throughout the development process, with frequent reviews and adjustments made based on feedback. This is particularly important for a gym tracking app, as user needs and preferences may change over time, and the app needs to evolve to meet those needs.

Division of phases

The development process will be divided into the following phases:

Phase 1: Requirements Gathering and Analysis

The Requirements Gathering and Analysis phase of the gym tracking app involved gathering user requirements using methods like interviews and surveys, and analyzing the gathered information to identify common themes. The team categorized the requirements into functional and non-functional, and defined the system requirements based on them. This phase set the foundation for the rest of the development process.

Phase 2: Design

In the design phase of the previous gym tracking app, the system architecture and application architecture were determined. The application architecture followed the Model-View-Controller (MVC) design pattern, which separates the application into three interconnected components: the model, which represents the data and business logic; the view, which handles the presentation layer; and the controller, which manages the communication between the model and view. The system architecture was designed as a client-server architecture, where the client (the web app) interacts with the server (the backend) via APIs. The database schema was also designed during this phase, outlining the tables, relationships, and constraints necessary for storing and retrieving user data. Overall, the design phase focused on creating a robust and scalable system that could efficiently handle the demands of a gym-tracking app.

Phase 3: Implementation

The implementation phase of the gym tracking app involved developing and coding the software based on the requirements identified in the previous phases. The team followed an Agile development approach, breaking down the development into sprints, and continuously testing and reviewing the software as it was being developed. They utilized Vue.js, a front-end JavaScript framework, and Node.js, a back-end JavaScript framework, to create a scalable and responsive web application. The implementation phase was a crucial part of the development process, as it brought the gym tracking app to life and

allowed the team to refine and polish the software before release.

Phase 4: Test

In this stage, the app is tested for errors and problems to make sure it complies with all functional and non-functional criteria. To make sure the software functions as intended.

Phase 5: Deployment and maintenance

The deployment and maintenance phase of the previous gym tracking app involves the release and distribution of the app to end-users and the ongoing management and upkeep of the app post-release. This includes ensuring that the app is deployed on appropriate hardware and software platforms, and that any necessary configurations or updates are made to the app and its associated systems. Additionally, the app must be monitored to ensure that it is performing well, and any bugs or issues must be promptly addressed through ongoing maintenance and support. Finally, the app should be regularly updated with new features and improvements to keep it competitive and relevant in the marketplace.

Backlog and Sprints

GYM TECH BACKOG					
As a	I want to be abe to ...	So that …	Priority	sprint	status
user	create a customized workout plans	I can focus on the trainings that best suit my body	must	sprint 5	done
user	start a workout and specify weights and reps	I can track volumes of workouts	should	sprint 5	done
user	track my fitness progress over time	I can review my progress	must	sprint 2	done
user	learn about the technique of new workouts	I can avoid injuries	should	sprint 5	done
user	browse gyms as the best suits me	I can find an appropriate and affordable gym	may	sprint 4	done
user	track my badges and achievments	I can be motivated	may	sprint 4	done
user	Create a new account	I can see my customized workouts	must	sprint 1	done
user	login	I can access my workouts	must	sprint 1	done
user	recommend an improvement	I can help for better user experience	should	sprint 2	done
adminstrator	add new workouts	I can provide various exercises to users	must	sprint 3	done
adminstrator	list customers	I can summarize their interaction	must	sprint 3	done
adminstrator	add new gyms	I can provide multiple gyms to the user so that he can choose from them	must	sprint 3	done
adminstator	view messages in the database and delete them	I can track user suggestions and recommendations	must	sprint 3	done

Architectural Design

System Design

The client-server architecture used in the gym tracking app is a common architecture for web-based applications. The architecture consists of two main components: the client, which is usually a web browser, and the server, which hosts the application and its data. In this architecture, the client sends requests to the server, and the server responds with the requested data or performs the requested action. In the gym tracking app, the client

component was the user's device, such as a smartphone or computer, and the server component hosted the app and its data. The client-server architecture provided a scalable and secure solution for the gym tracking app, allowing users to access their data from multiple devices and ensuring that their information was preserved on the server.

The communication between the client-side and server-side of the application would be through API calls. The web app would use REST APIs to communicate with the server, which would allow for easy integration with different platforms and technologies.

Application Design

Based on the requirements of the gym tracking app, a suitable application architecture would be the Model-View-Controller (MVC) architecture. This architecture separates the application into three interconnected components: the Model, the View, and the Controller. The Model represents the data and the business logic of the application. In the gym tracking app, this would include information about users, workouts, and progress tracking. The View represents the user interface of the application and how the user interacts with the app. In the gym tracking app, this would include screens for logging in, viewing workout history, and setting goals. The Controller acts as the intermediary between the Model and the View, processing user input and managing the application logic. In the gym tracking app, this would include handling user requests to add workouts, update progress, and track goals. Using the MVC architecture design for the gym tracking app would allow for better organization of the code, making it more modular and easier to maintain. Additionally, it would facilitate better collaboration among team members working on different components of the application.

DATA TABLES

User

id:	fname	lname	email	password	gender	age	weight	height	target weight	daily calories burnt	suggested calories per day
int	varchar[20]	varchar[20]	varchar[100]	varchar[100]	char(1)	int	float(5,2)	float(5,2)	float(5,2)	float(6,2)	float(6,2)

Exercise

id	name	image	category	target muscle	video link	calories per set
int	varchar[20]	varchar[200]	varchar[20]	varchar[20]	varchar[200]	float(6,2)

Badges

id	name	image	condition
int	varchar[20]	varchar[200]	varchar[100]

Workout

user id	exercise id	workout_name	number of sets	total calories
int	int	varchar[20]	int	float(6,2)

Dailly_Workout

workout_name	user_id	ex_id	set_number,	date	weight	reps
varchar[20]	int	int	int	Date	int	int

Progress

user id	date	workout_name	trained
int	Date	varchar[20]	Tinyint

Badges Earned

user id	badge id
int	int

GYM

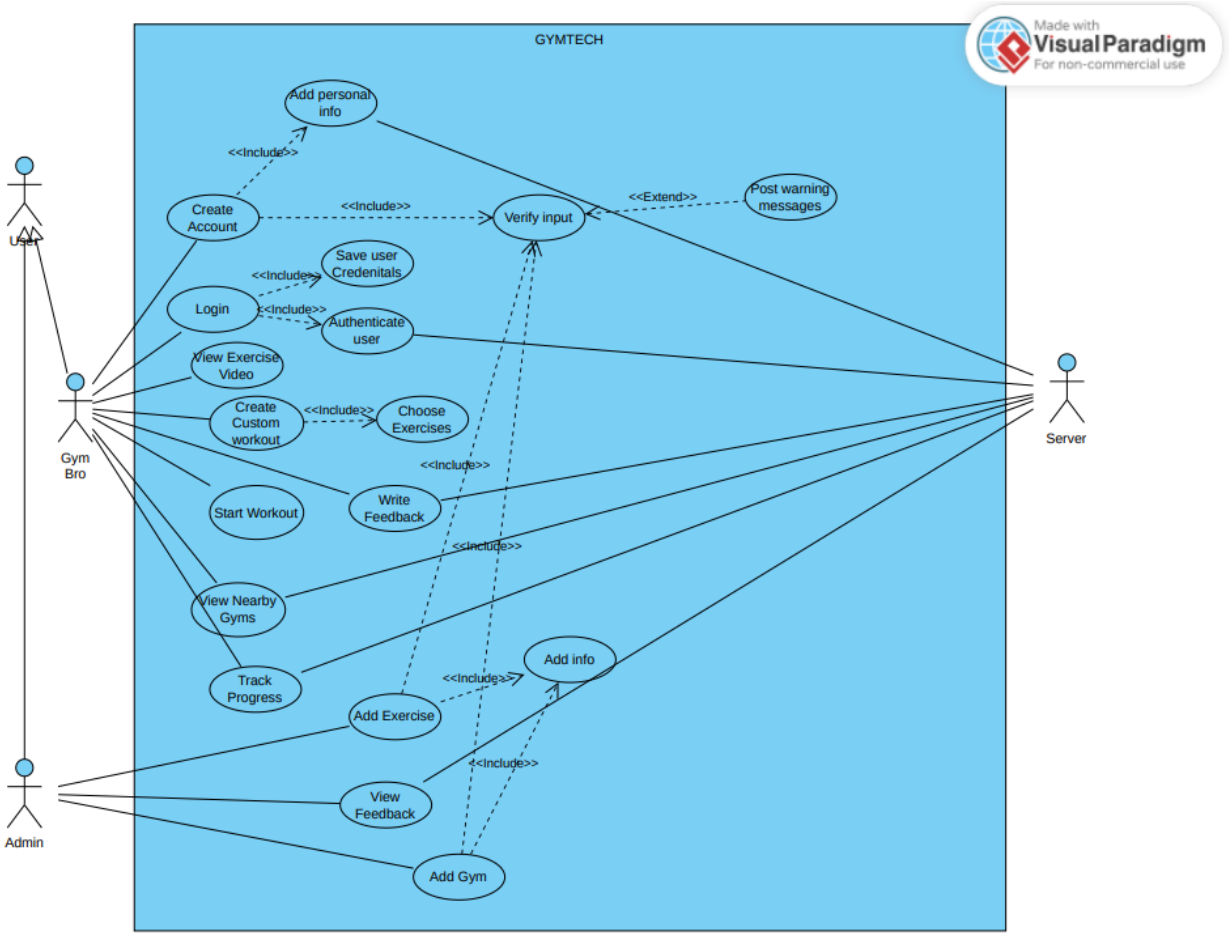
name	location	price/month	working hours	rating	link
varchar[20]	varchar[20]	int	varchar[20]	float(3,1)	varchar[100]

Recommendations

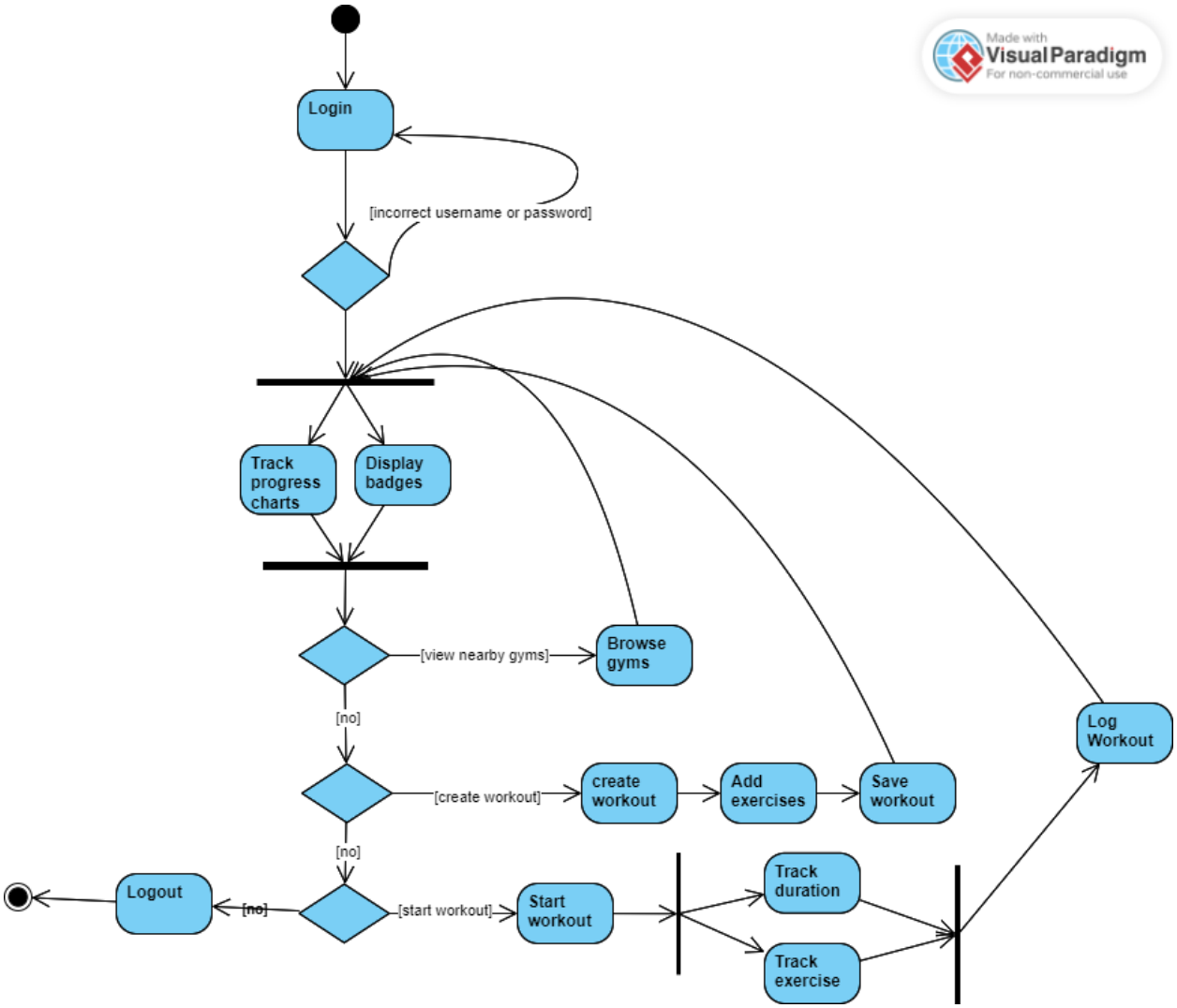
recommendation_id	user id	date	recommendation
int	int	Date	varchar[200]

System Analysis and UML Diagrams

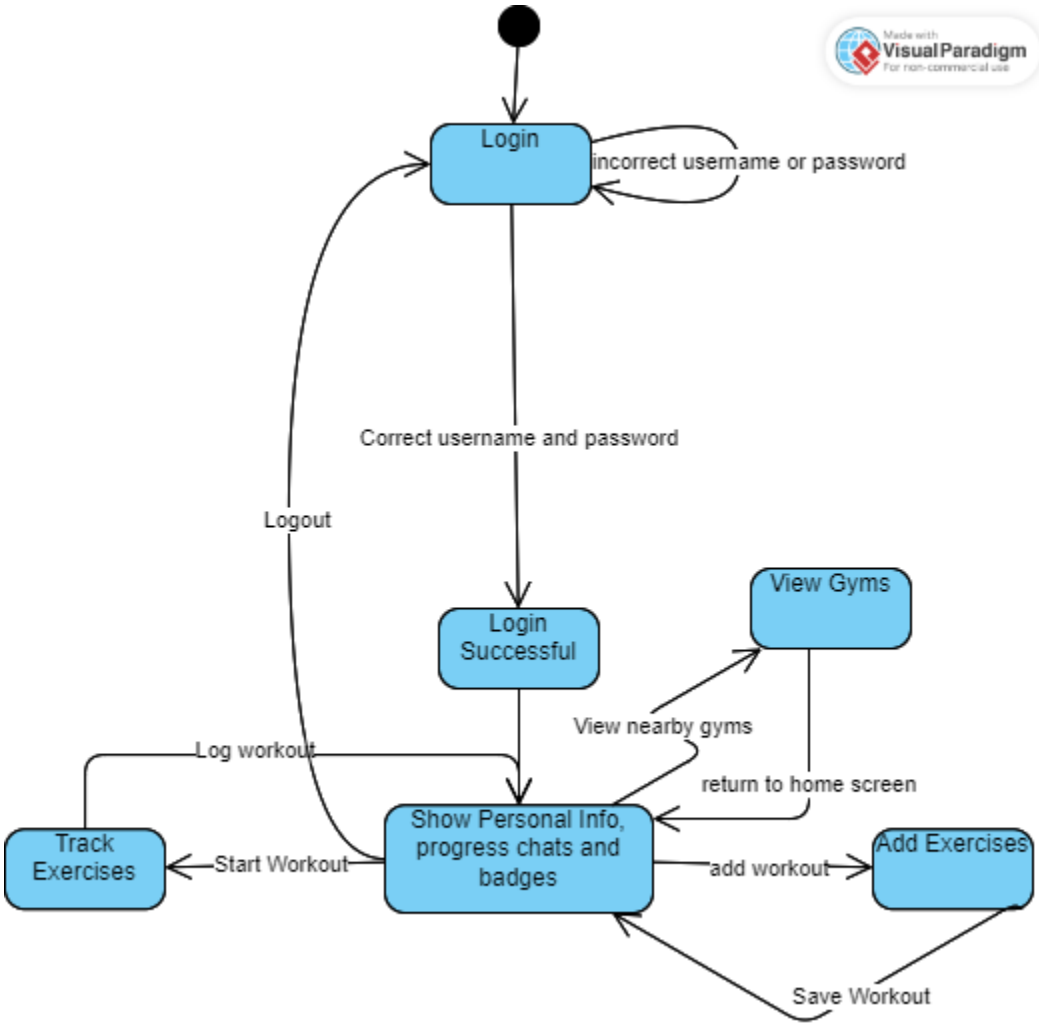
Use Case Diagram



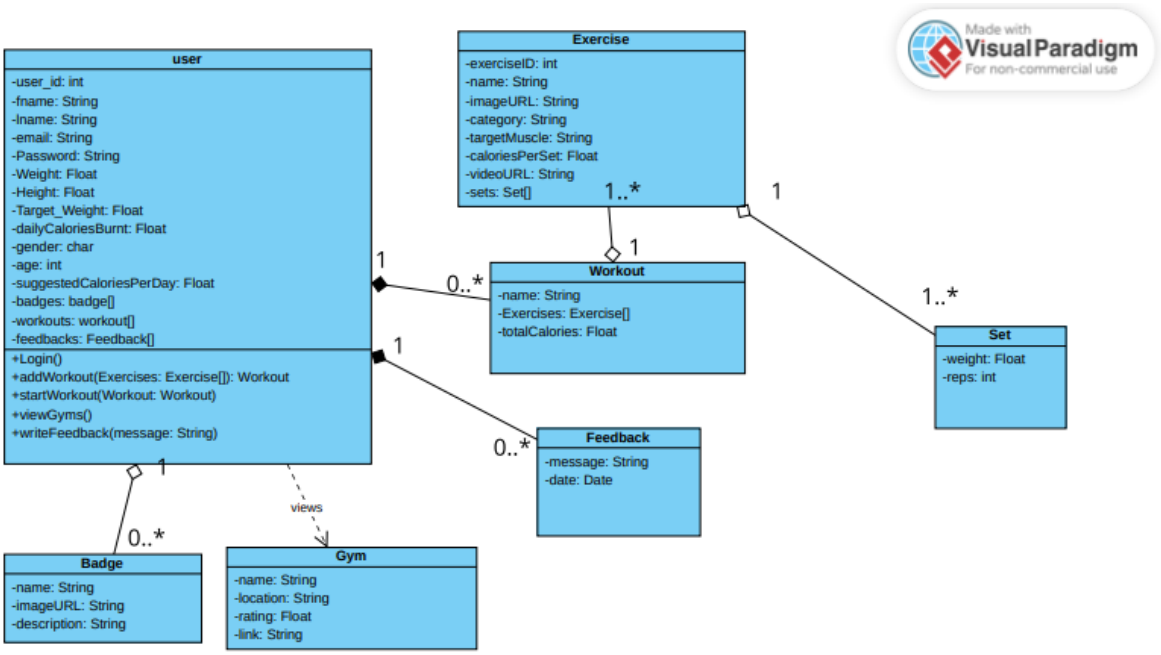
Activity Diagram



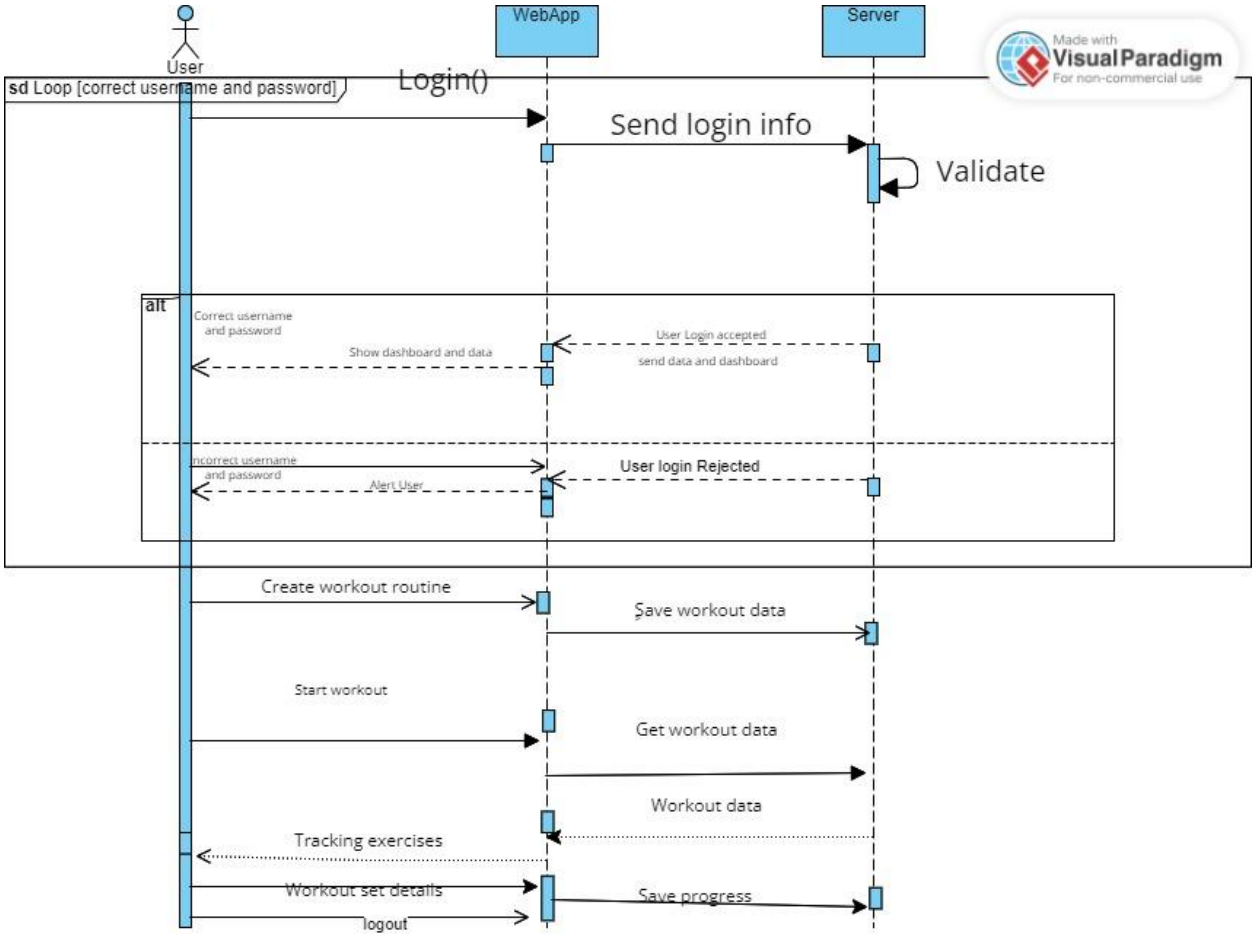
State Machine Diagram



Class Diagram

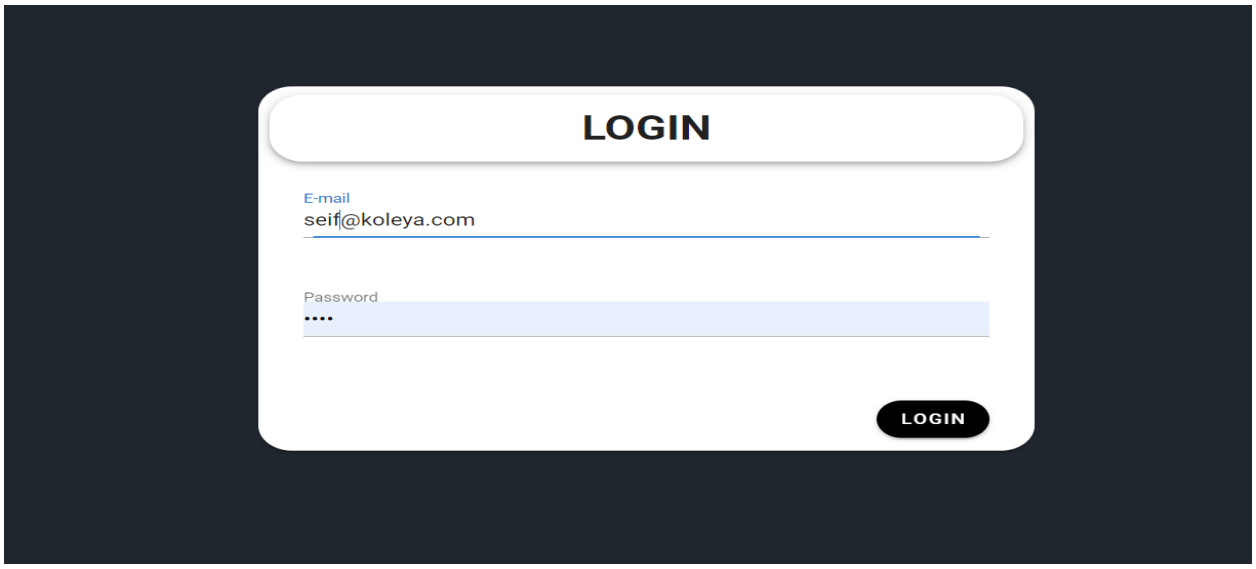


Interaction Diagram (Sequence Diagram)

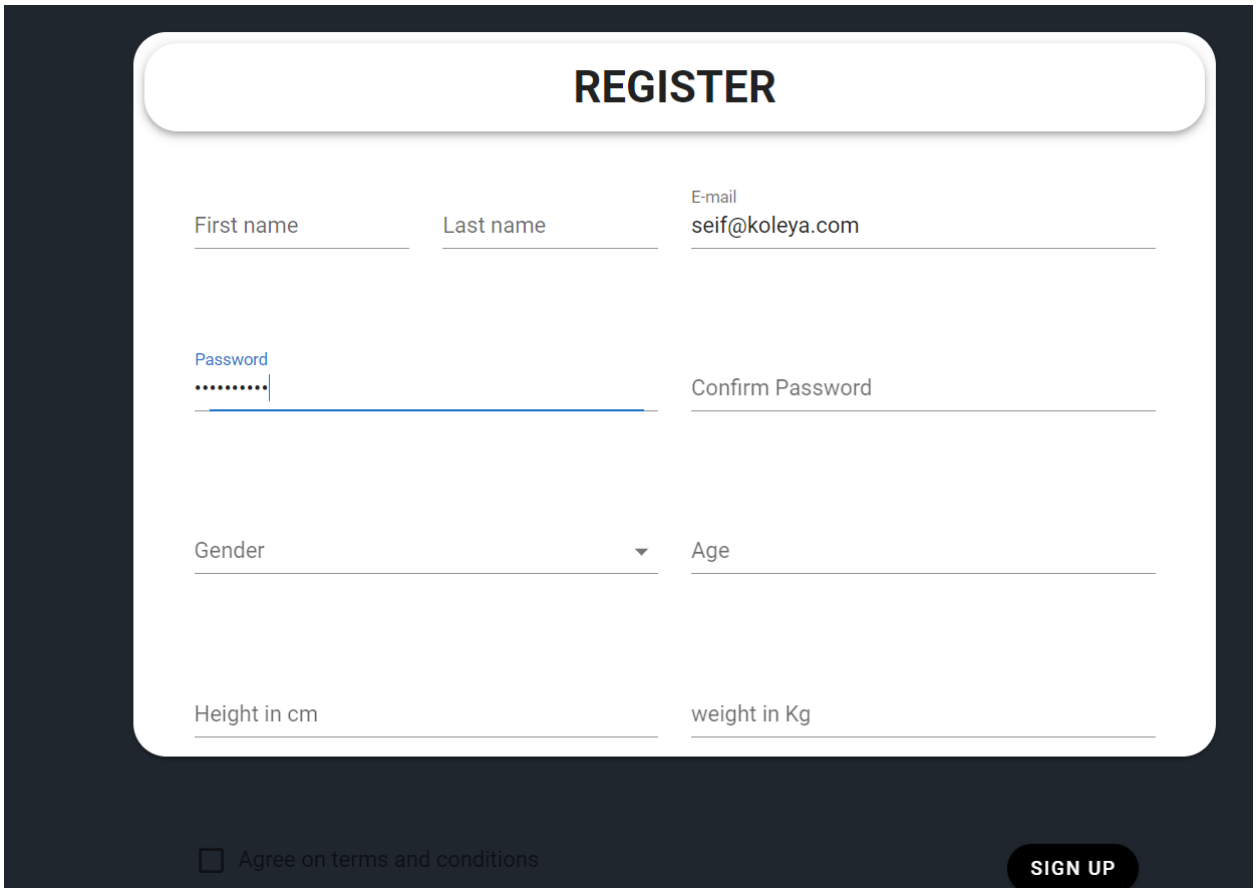


User Interface Specification

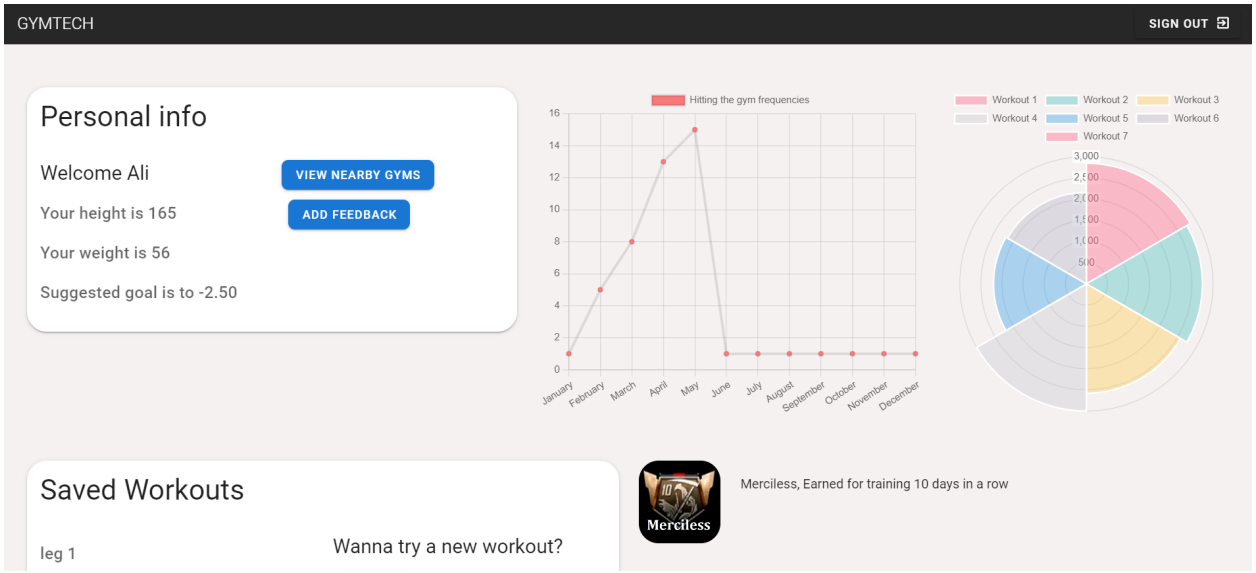
On the starting page, the user is allowed to sign-in, sign-up or sign-in as admin.



On the login page, the user is allowed to sign in into his account.

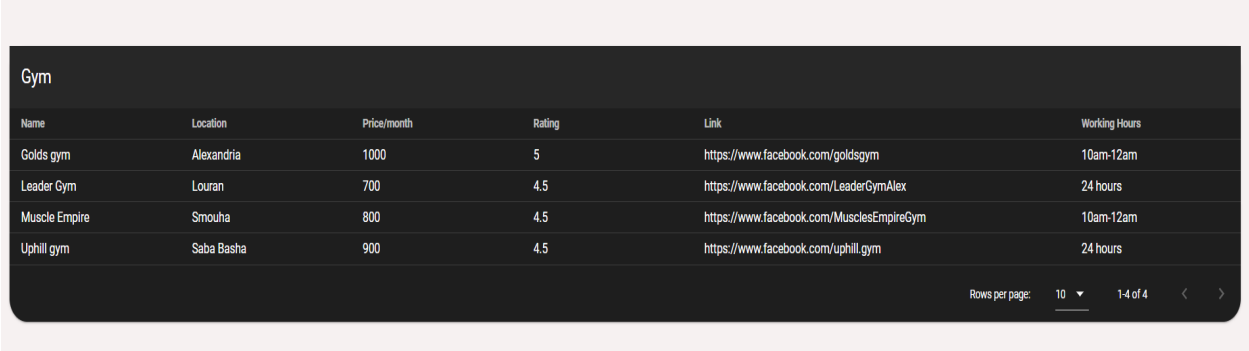


On the sign-up, the user is allowed to create his account.

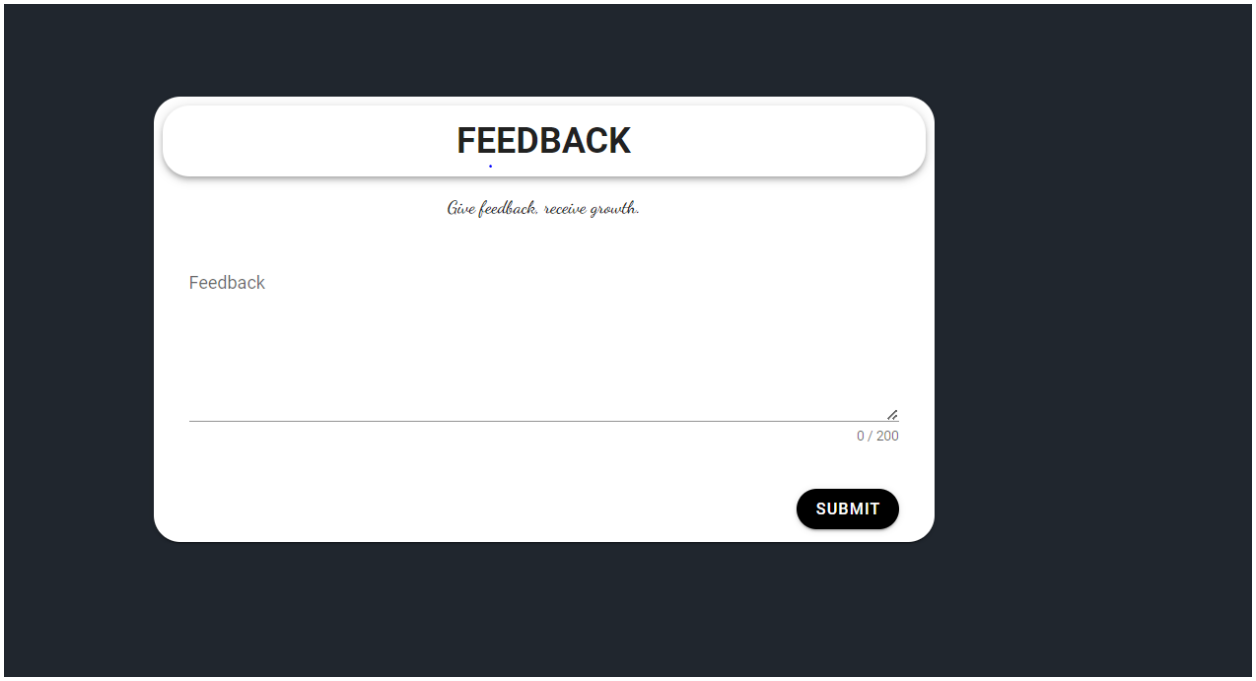


On the dashboard page, the user is allowed to: view nearby gyms, add feedback, start a

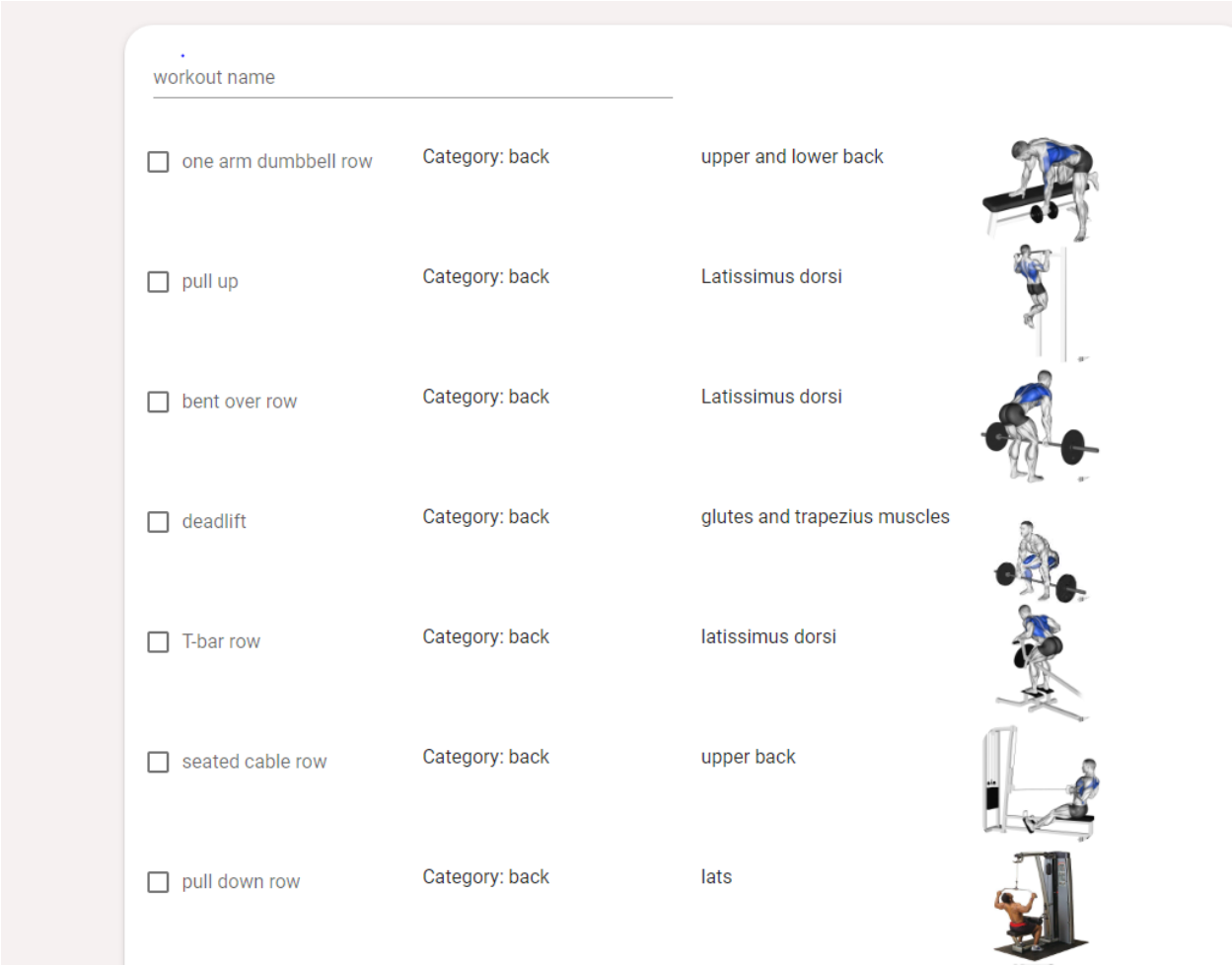
workout session, add a new workout routine, view frequency of training each month and view the volume of each workout in the last 7 days.



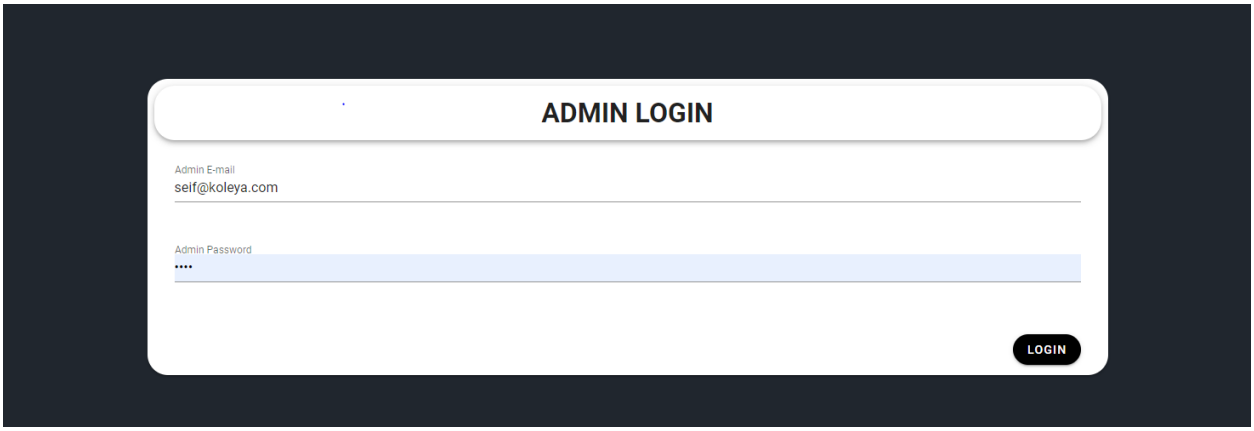
Example of the view nearby gyms page.



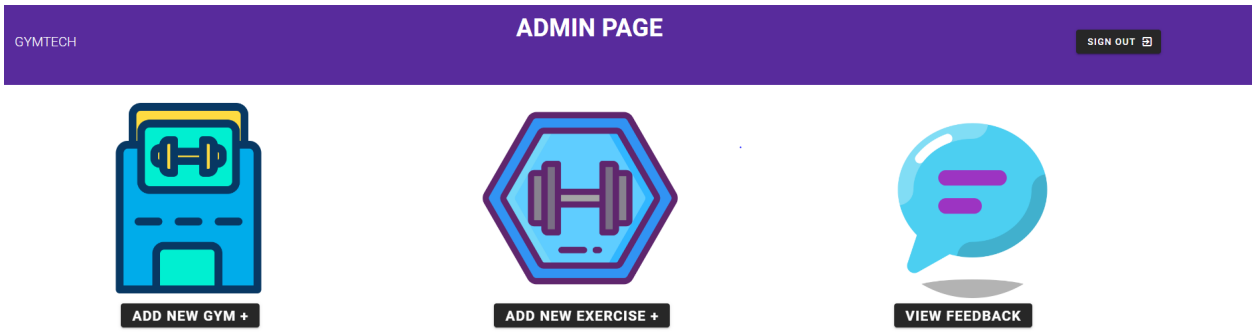
Example of the add feedback page.



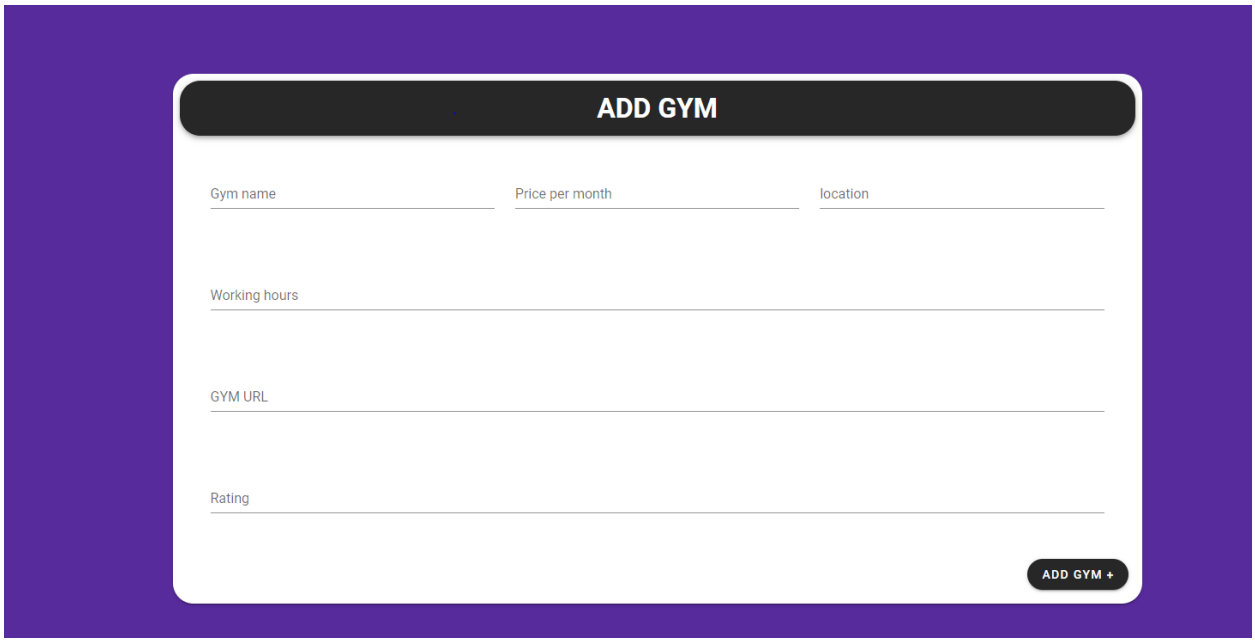
Example of the add a workout page.



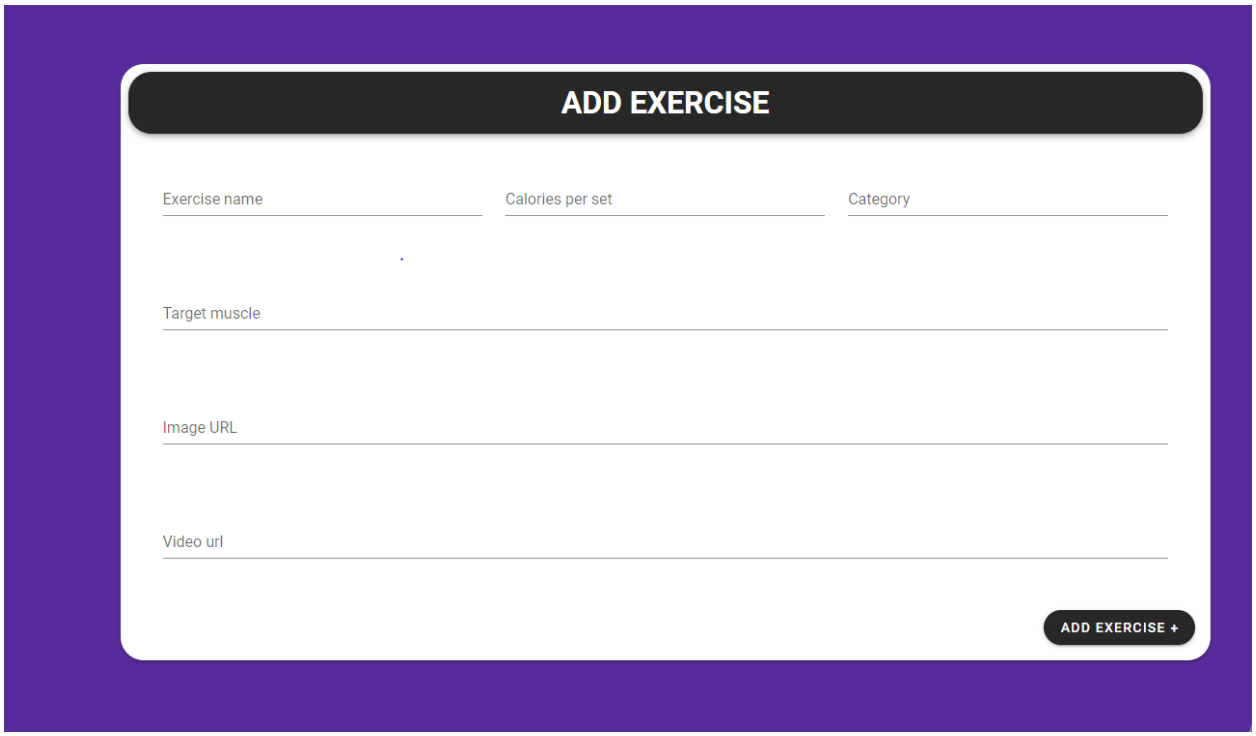
On the admin login page, the admin is allowed to login into his account.



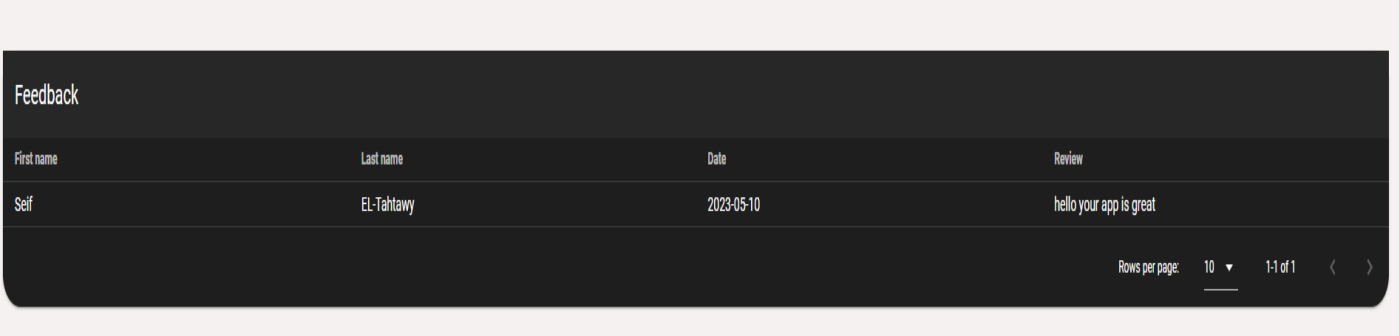
On the admin page, the admin is allowed to add a new gym, add a new exercise and view user feedback.



Example of the add gym page.



Example of the add exercise page.



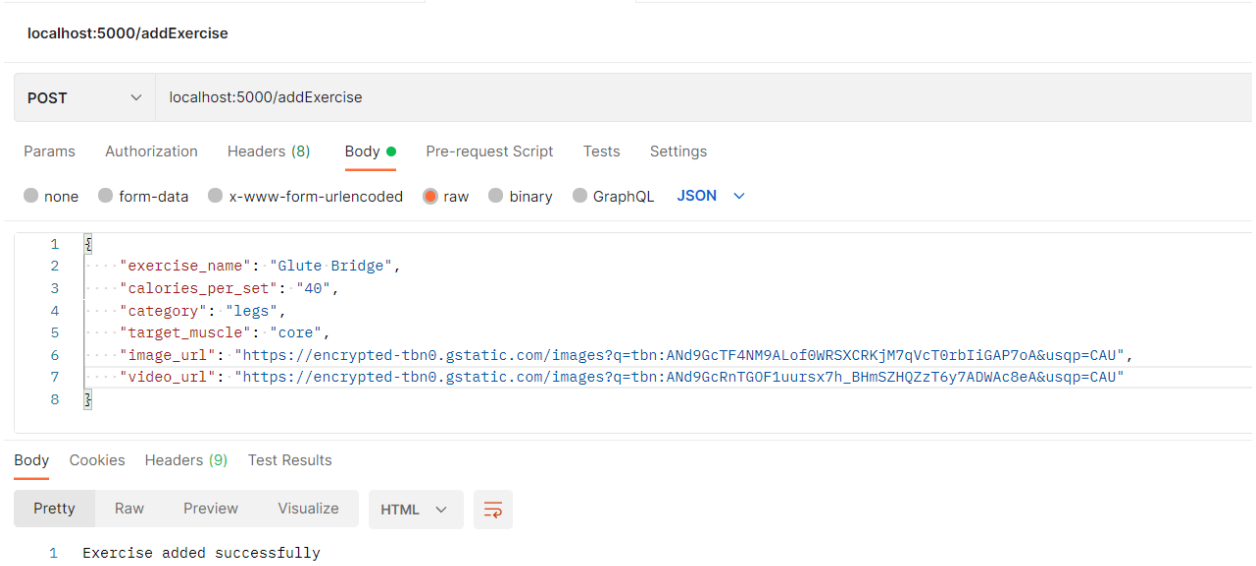
Example of the view feedback page.

System Testing

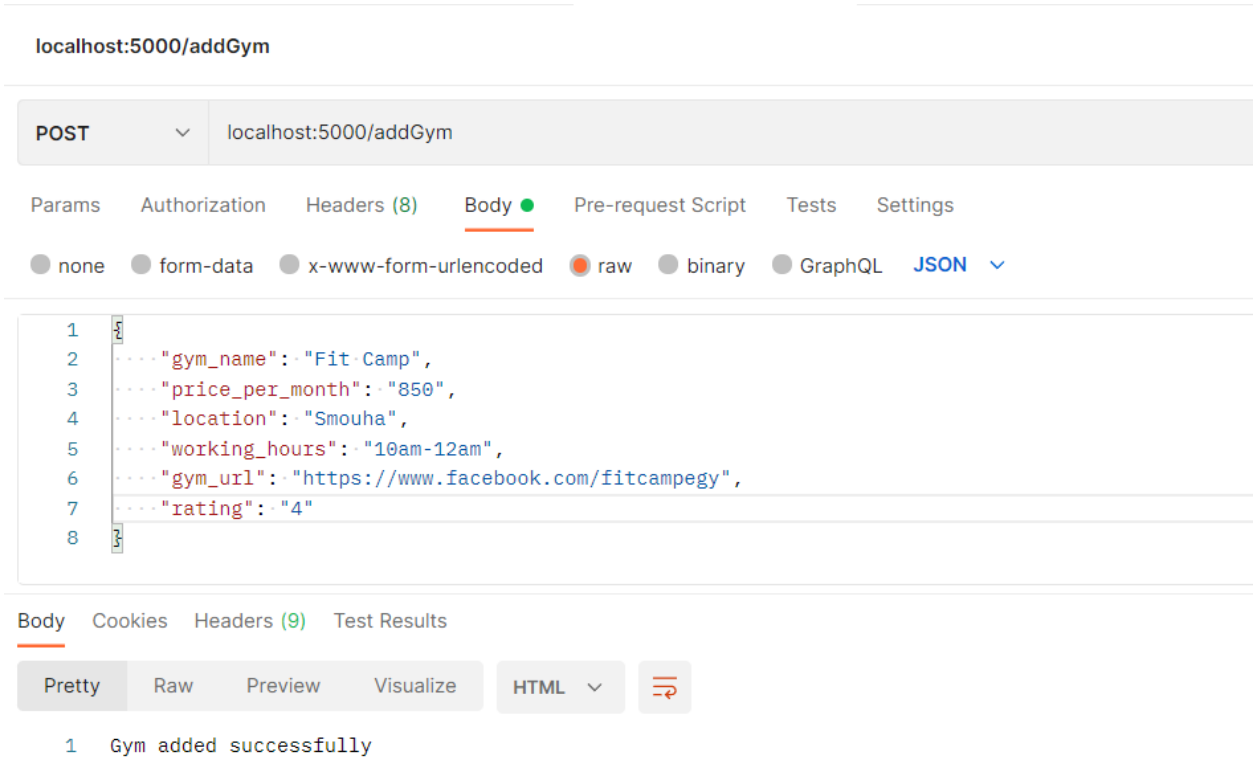
Backend Testing

Backend server will be tested using Postman by sending a request and analyze the correctness of the response received

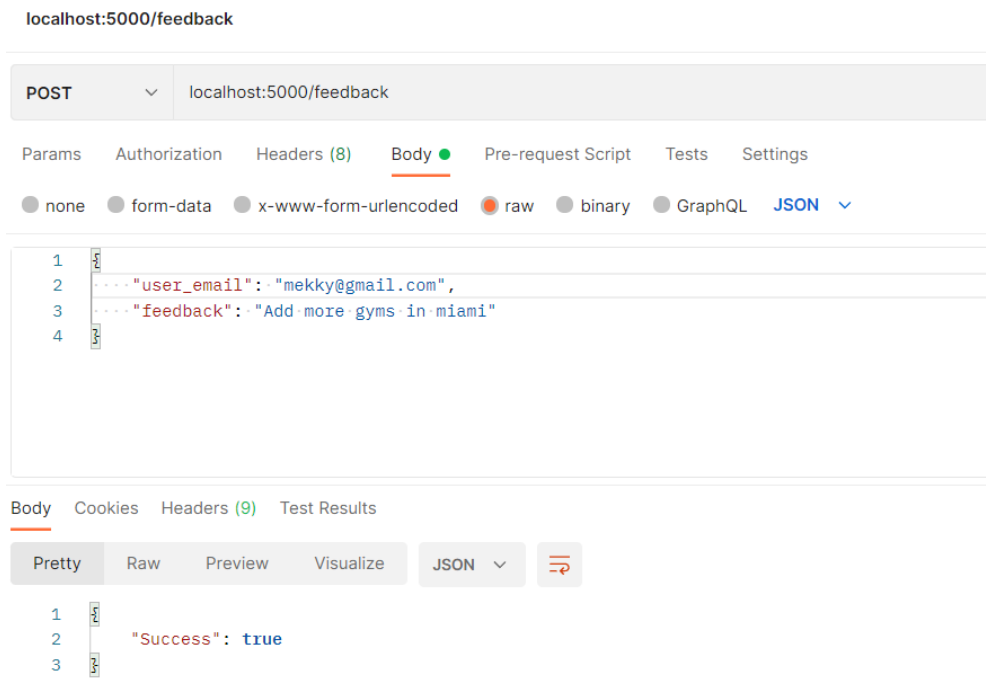
addExerciseTesting:



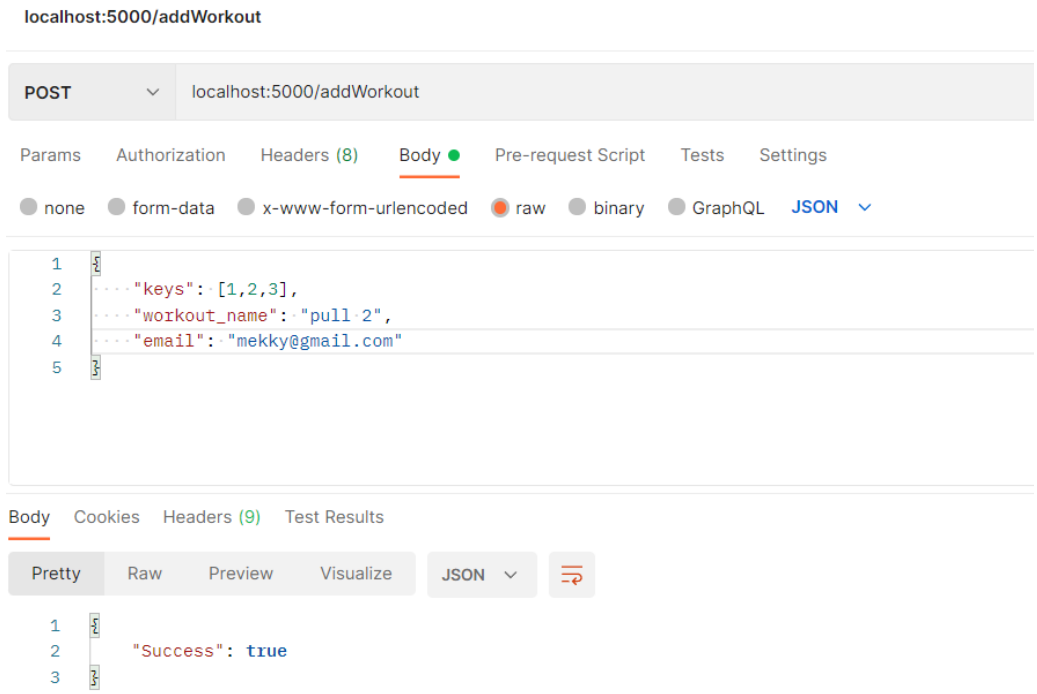
addGymTesting:



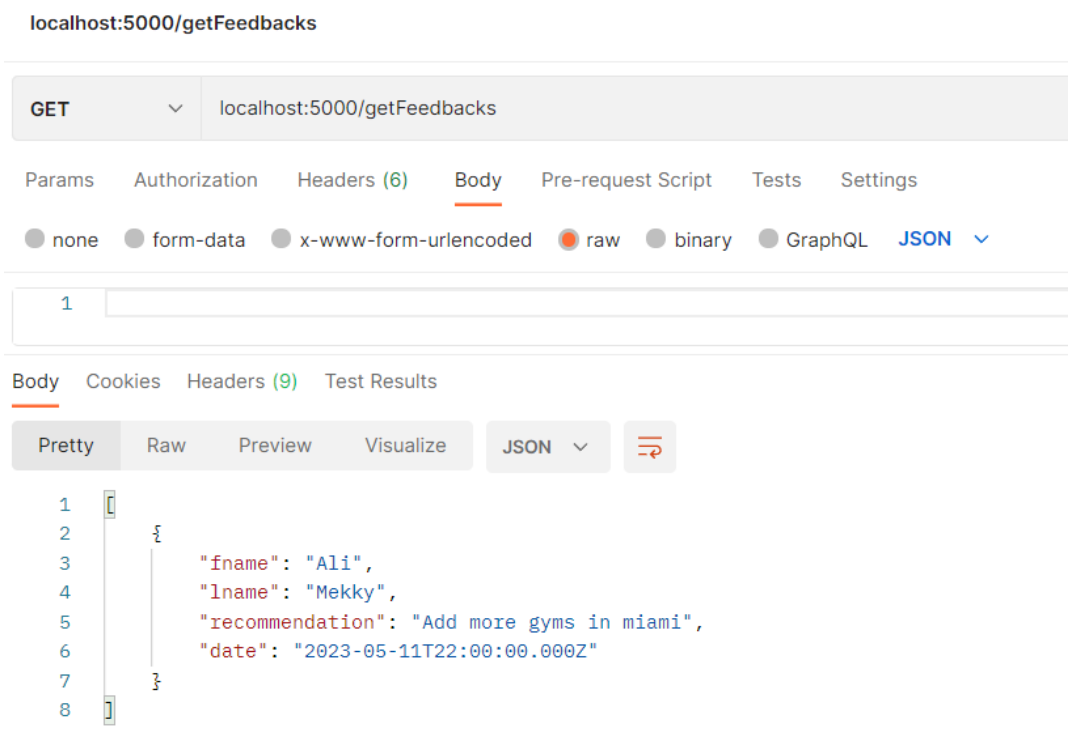
postingFeedbackTesting:



addWorkoutTesting:

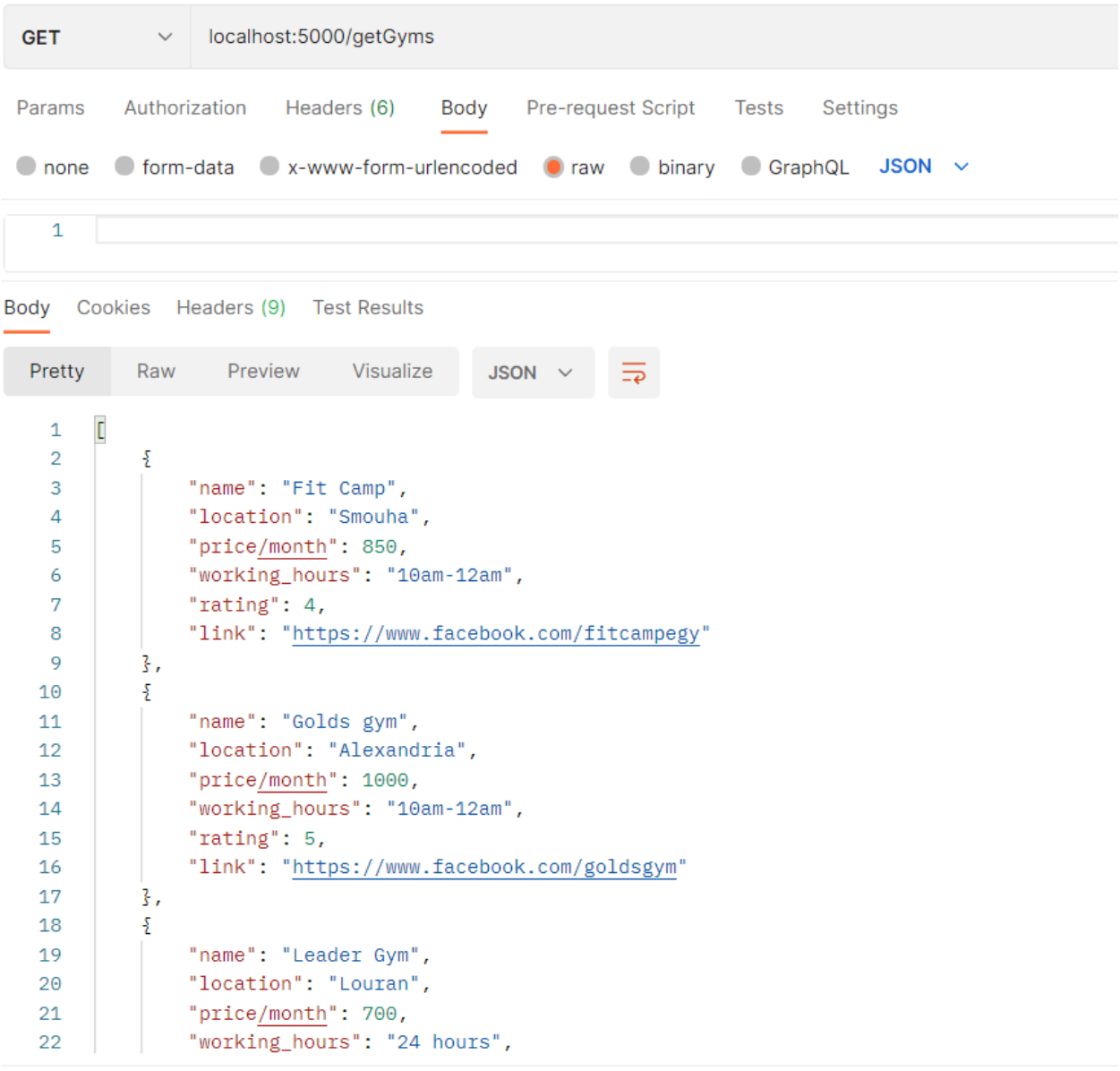


getFeedbacksTesting:



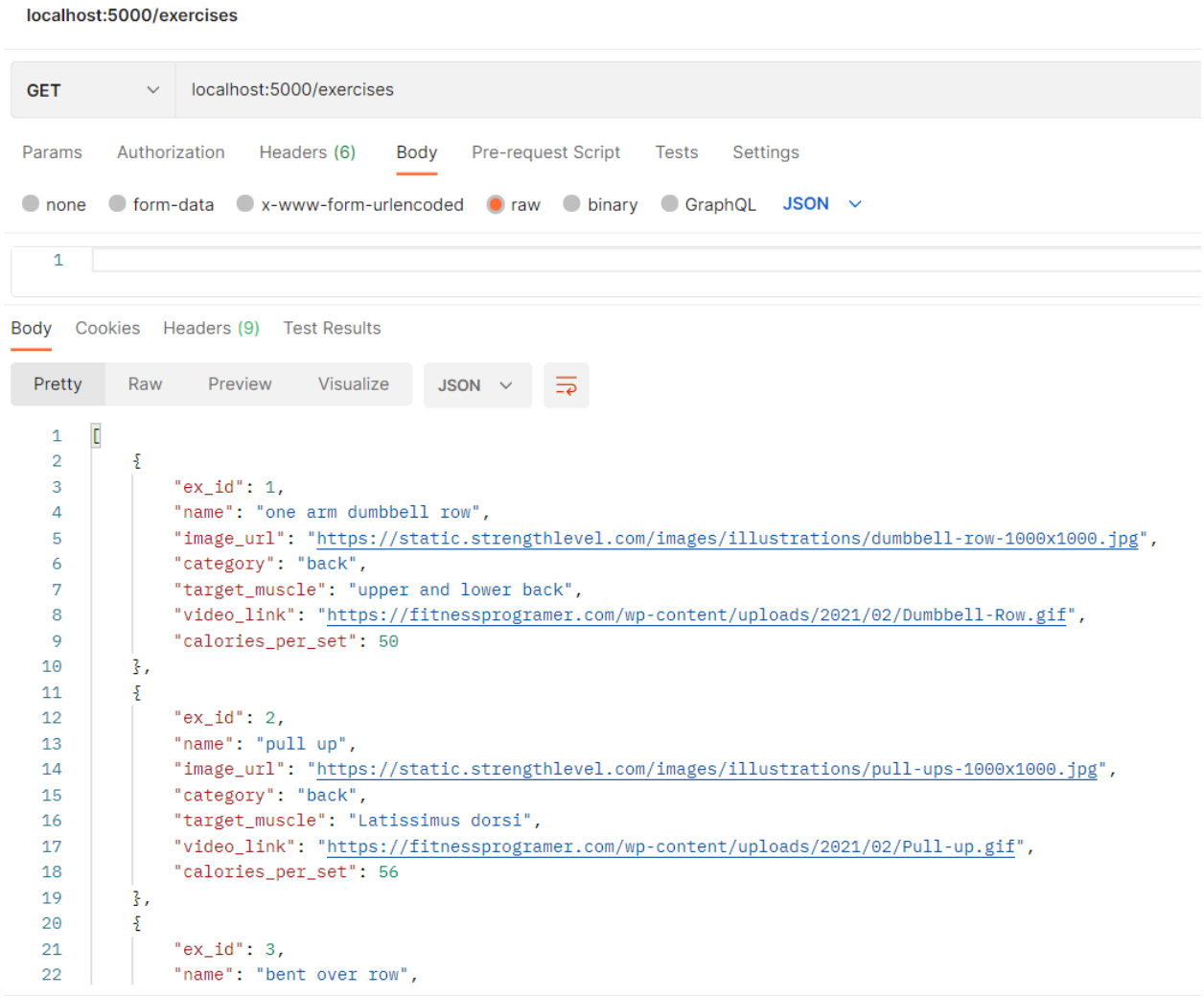
getGymsTesting:

localhost:5000/getGyms




```
[
    {
        "name": "Fit Camp",
        "location": "Smouha",
        "price/month": 850,
        "working_hours": "10am-12am",
        "rating": 4,
        "link": "https://www.facebook.com/fitcampegy"
    },
    {
        "name": "Golds gym",
        "location": "Alexandria",
        "price/month": 1000,
        "working_hours": "10am-12am",
        "rating": 5,
        "link": "https://www.facebook.com/goldsgym"
    },
    {
        "name": "Leader Gym",
        "location": "Louran",
        "price/month": 700,
        "working_hours": "24 hours",
```

getExercisesTesting:



getDashboardDataTesting

localhost:5000/dashboardData

POST localhost:5000/dashboardData

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

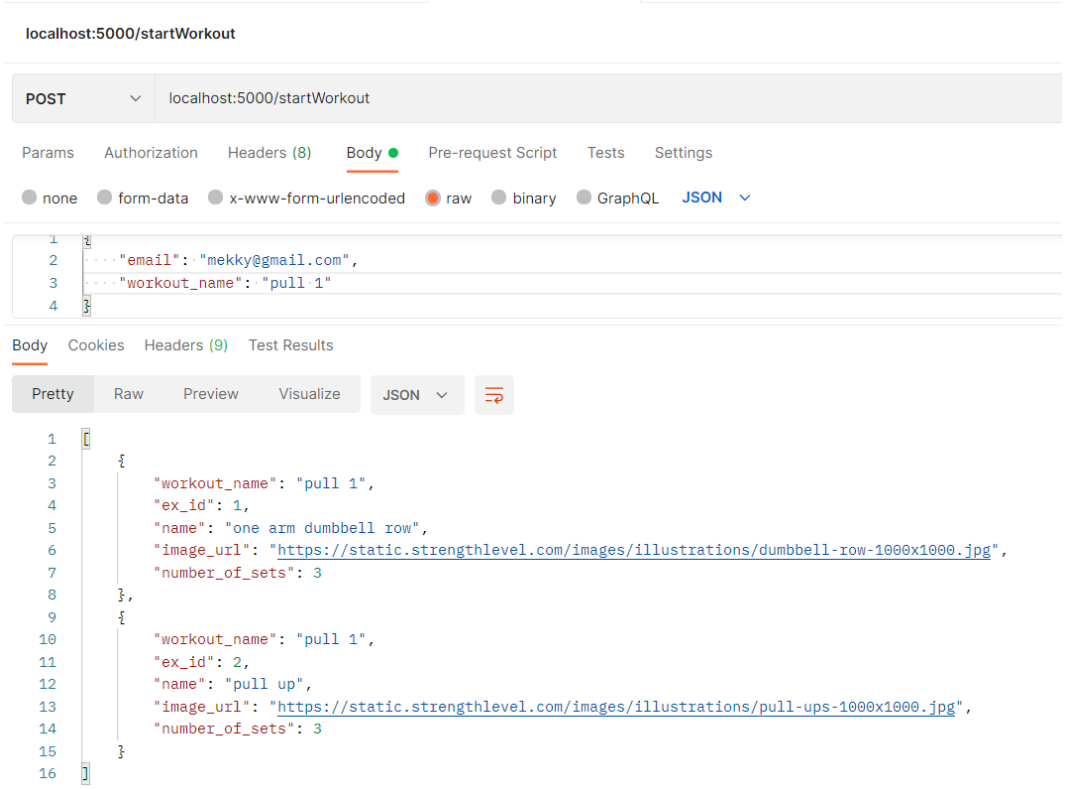
```
{
    "email": "mekky@gmail.com"
}
```

Body Cookies Headers (9) Test Results

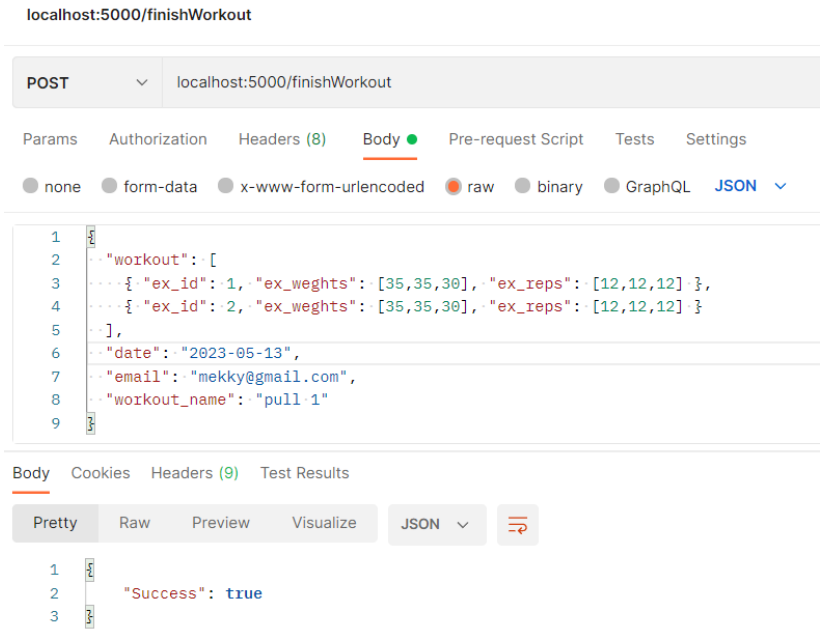
Pretty Raw Preview Visualize JSON


```
{
    "user_info": {
        "user_id": 1,
        "fname": "Ali",
        "lname": "Mekky",
        "email": "mekky@gmail.com",
        "weight": 56,
        "height": 165,
        "target_weight": 58.5,
        "daily_calories_burnt": 0,
        "gender": "M",
        "age": 21,
        "suggested_calories_per_day": 1209.6
    },
    "frequencies": [
        0,
        4,
        7,
        12,
        14,
```

startWorkoutTesting



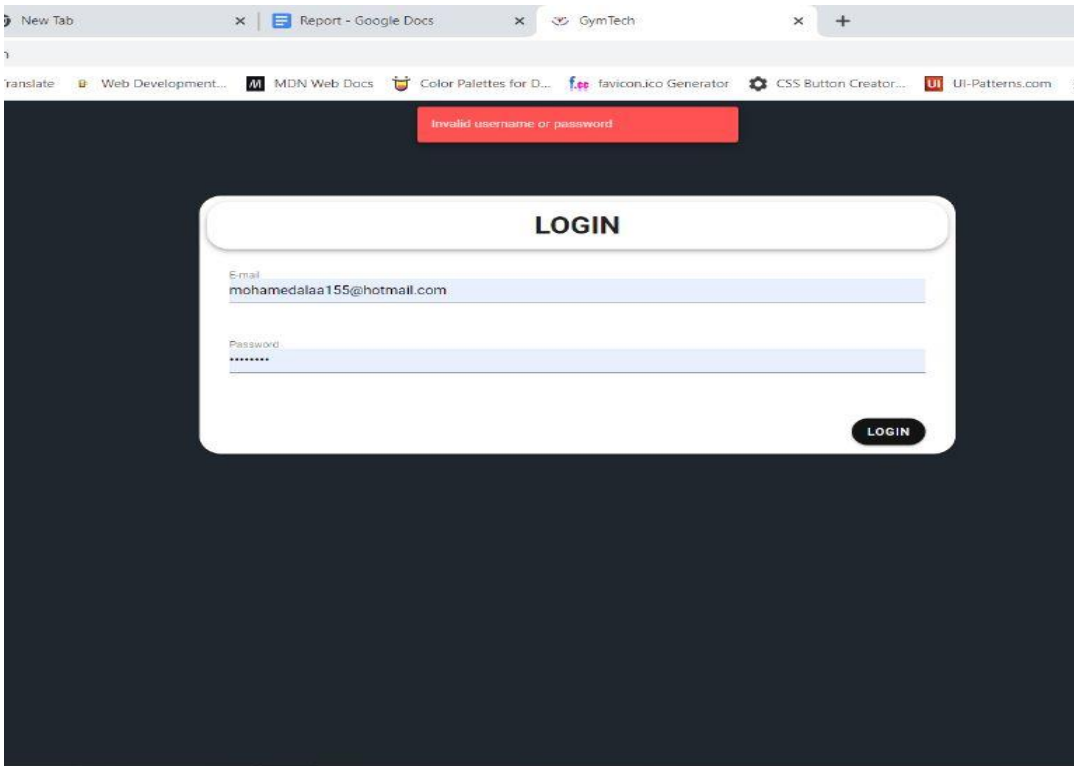
finishWorkoutTesting

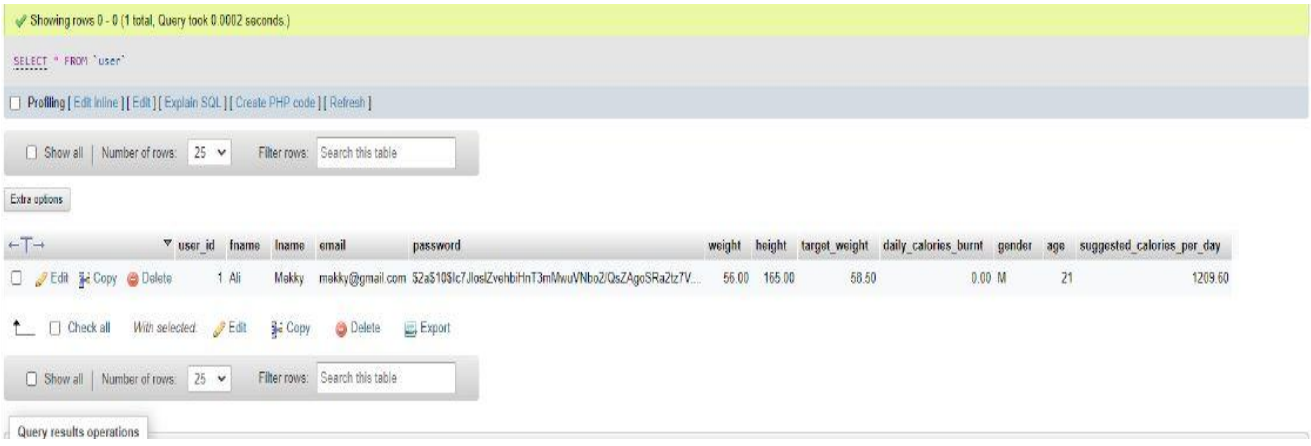


Frontend Testing

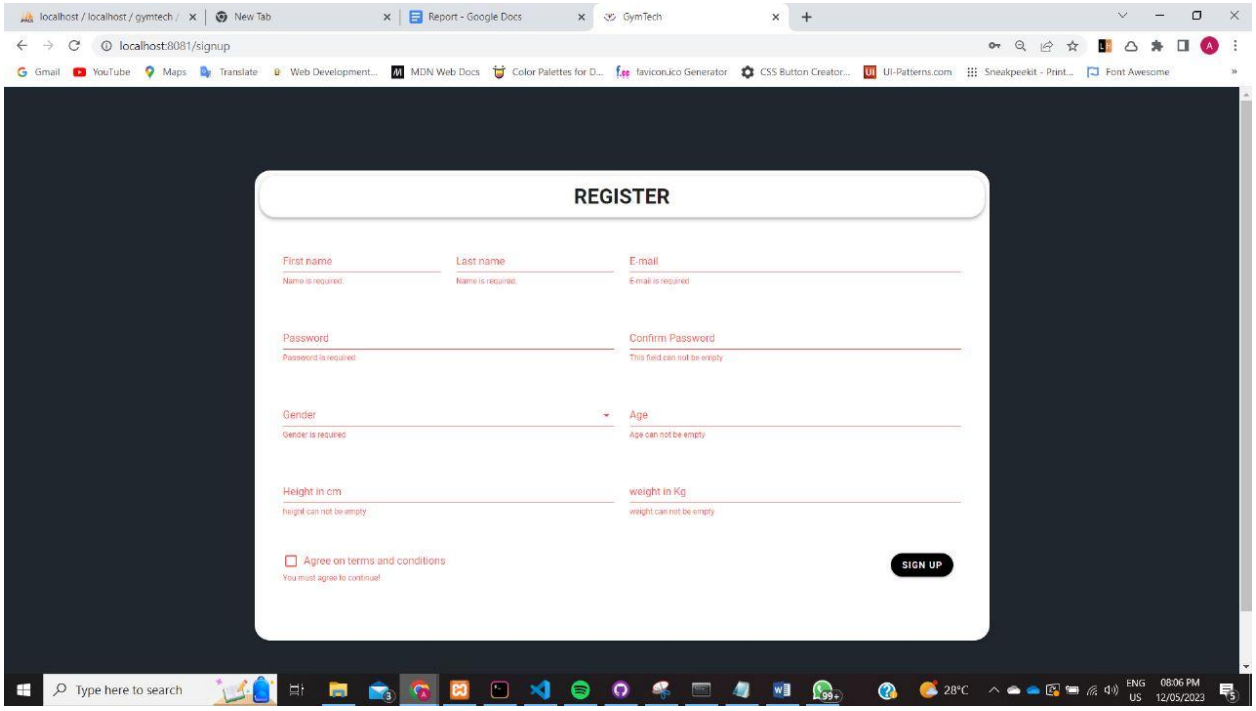
Frontend will be tested using correct and incorrect errors to display error messages or proceed with the normal flow.

Login with an invalid user:

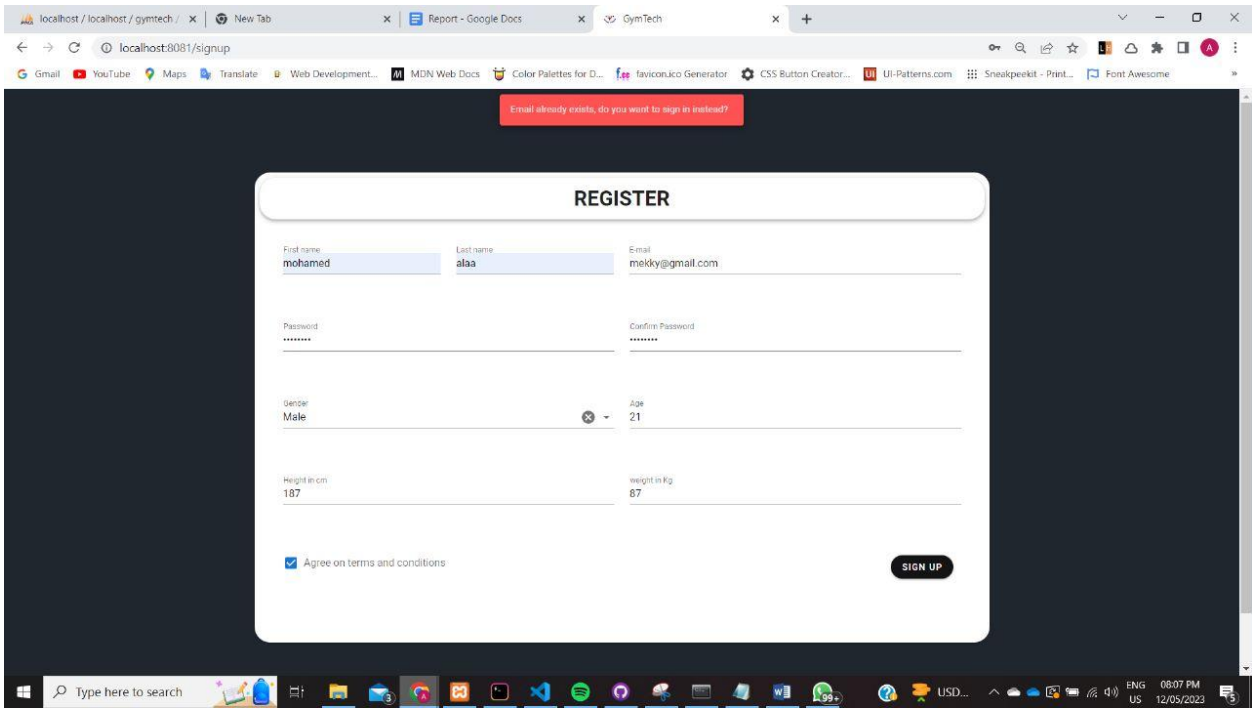




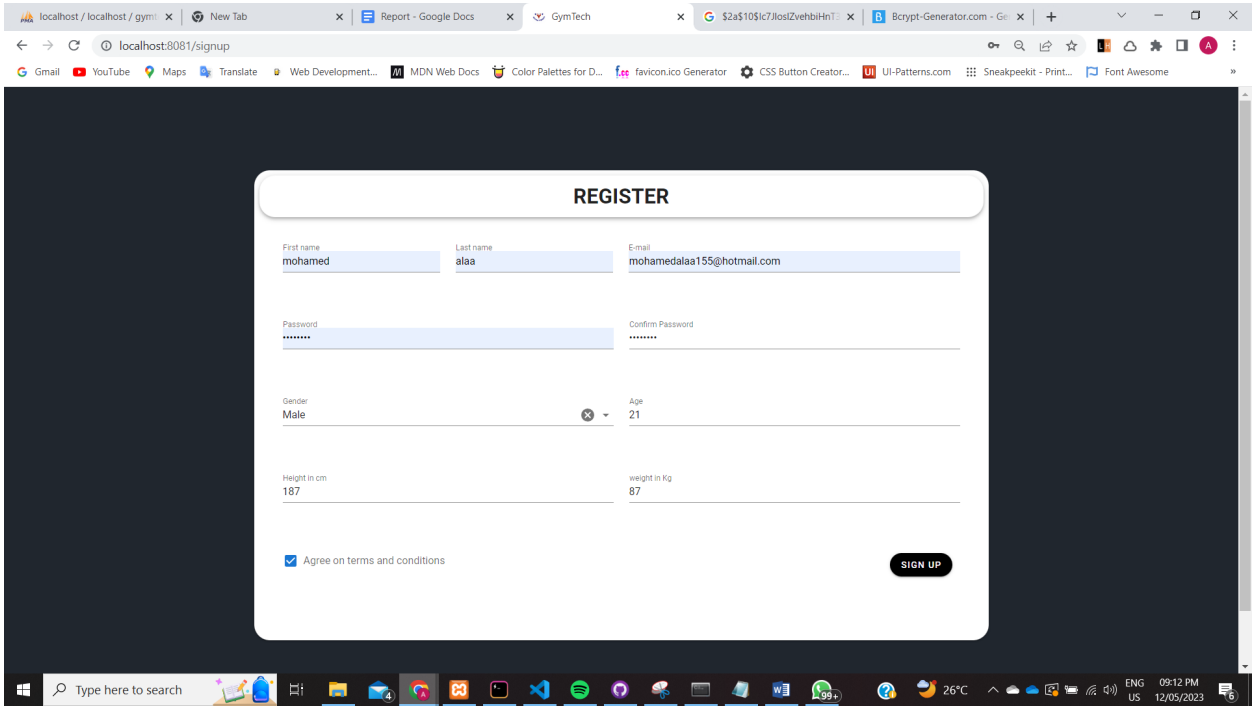
Trying to sign up with empty fields:

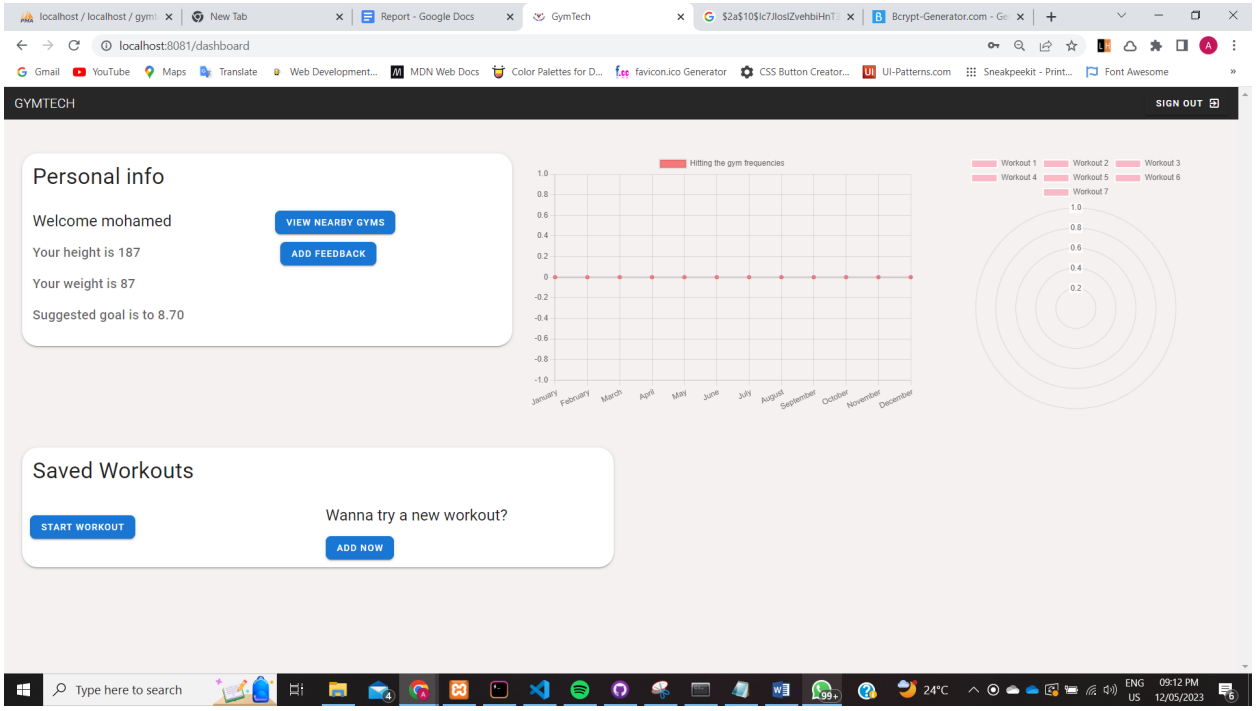


Signing up with an existing email:

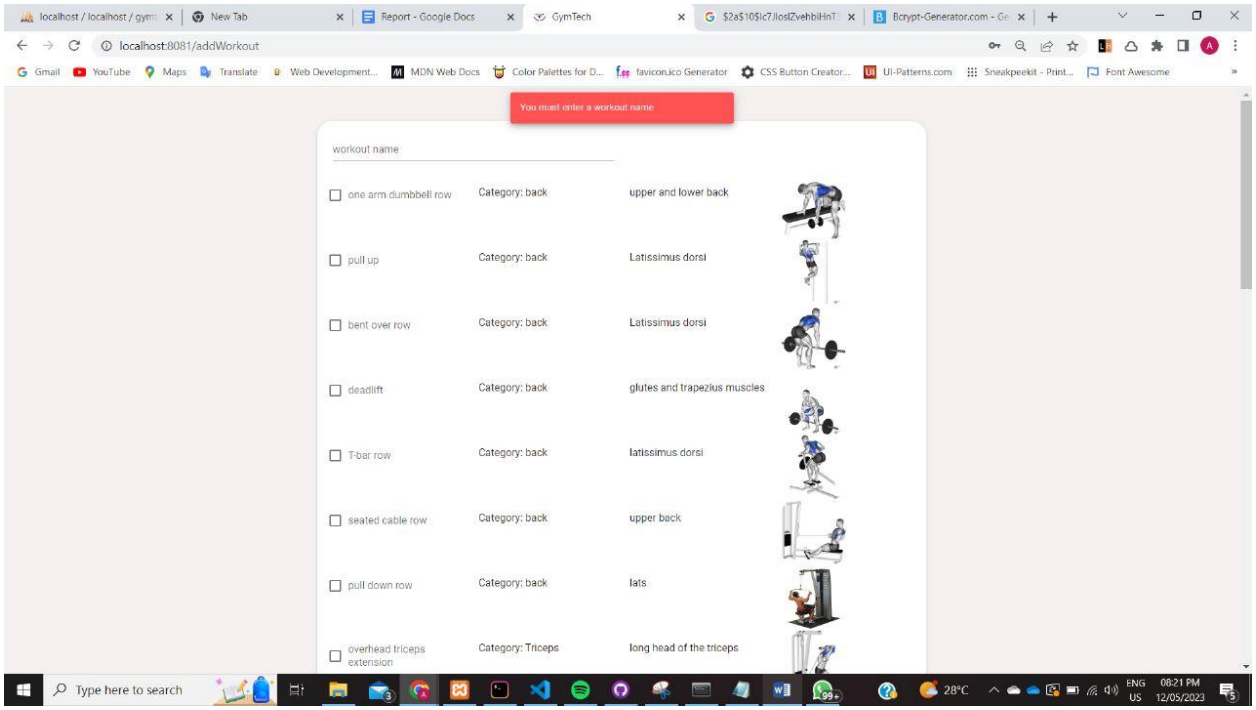


Signing up with a new email:

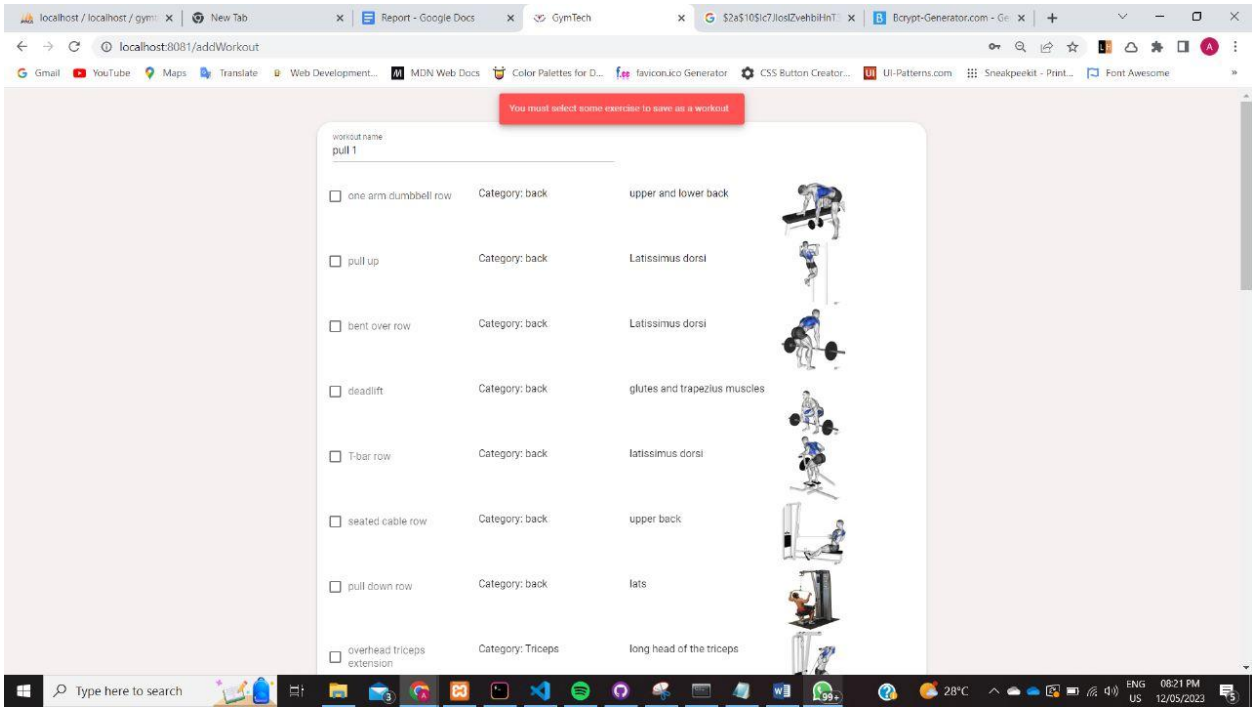




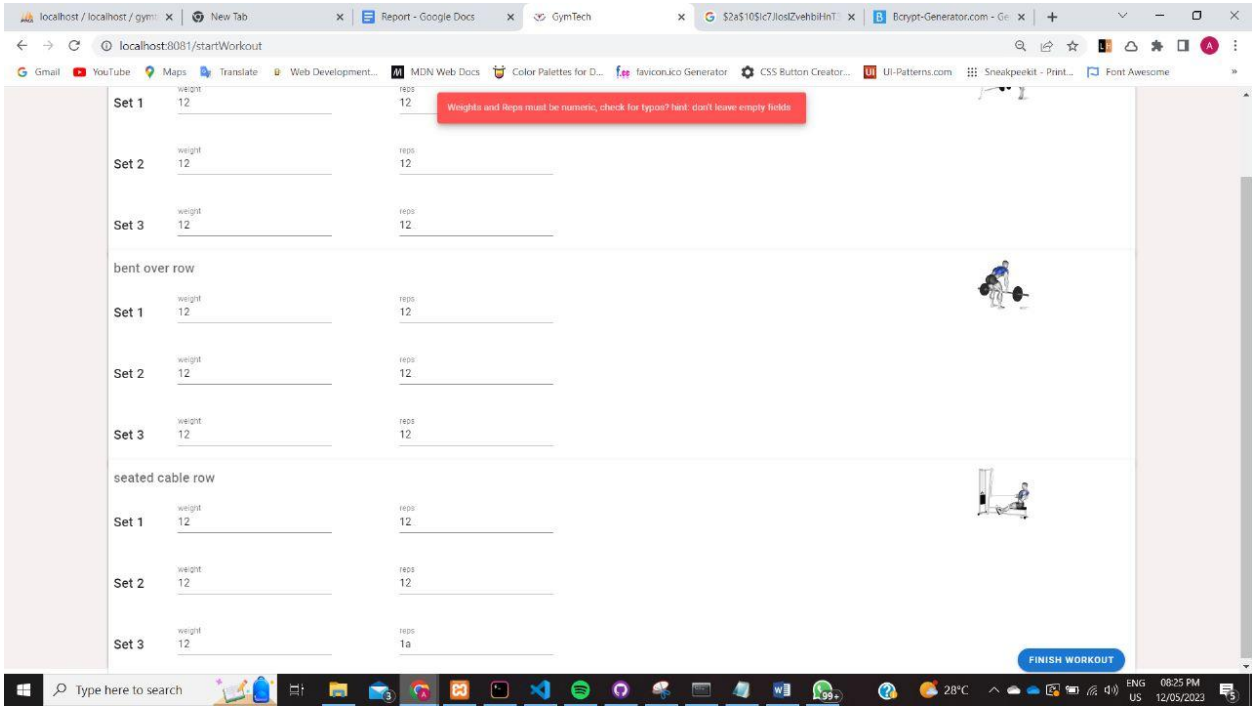
Trying to add a workout without a name:



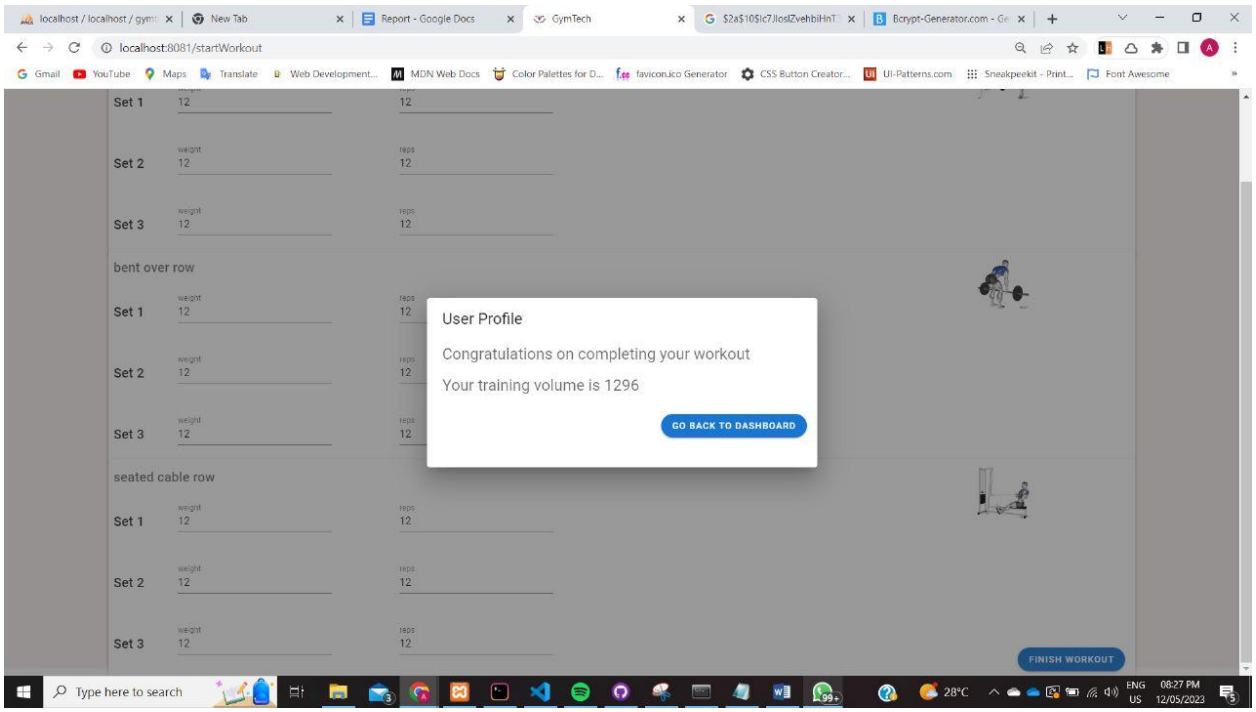
Trying to add a workout without any exercises:

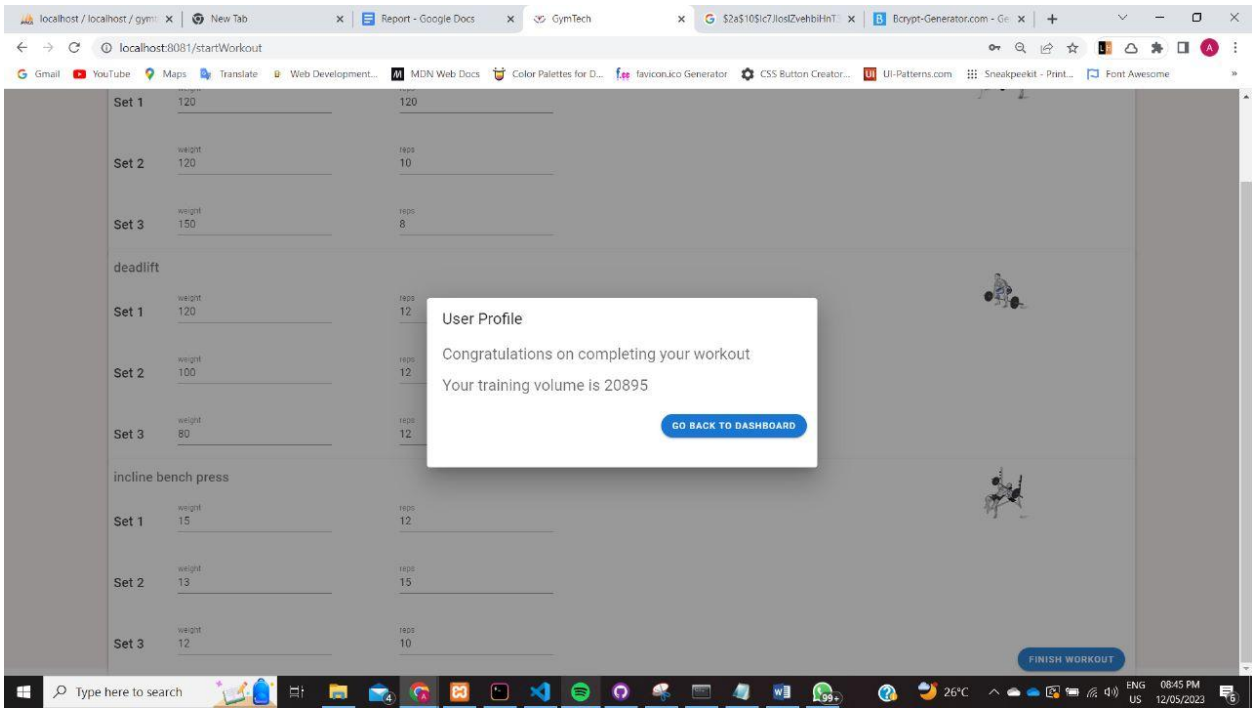


Trying to finish a workout with numeric and alphabetic fields:

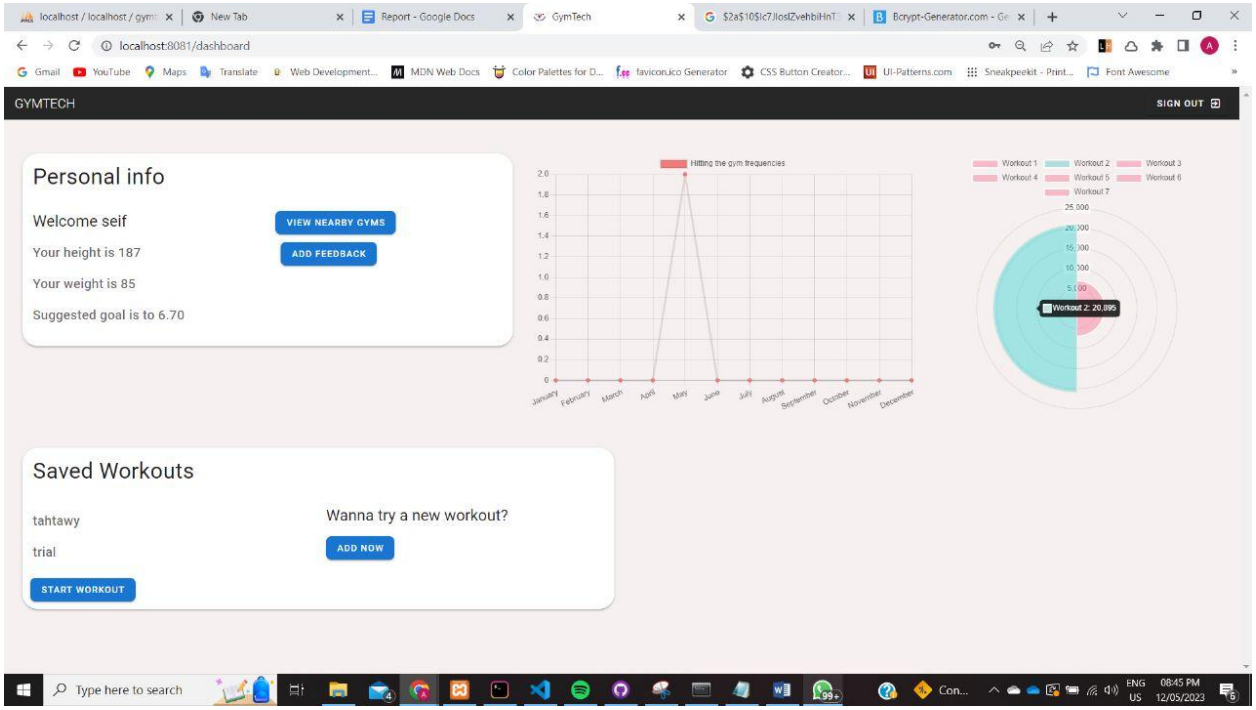


Trying to finish a workout with numeric fields:

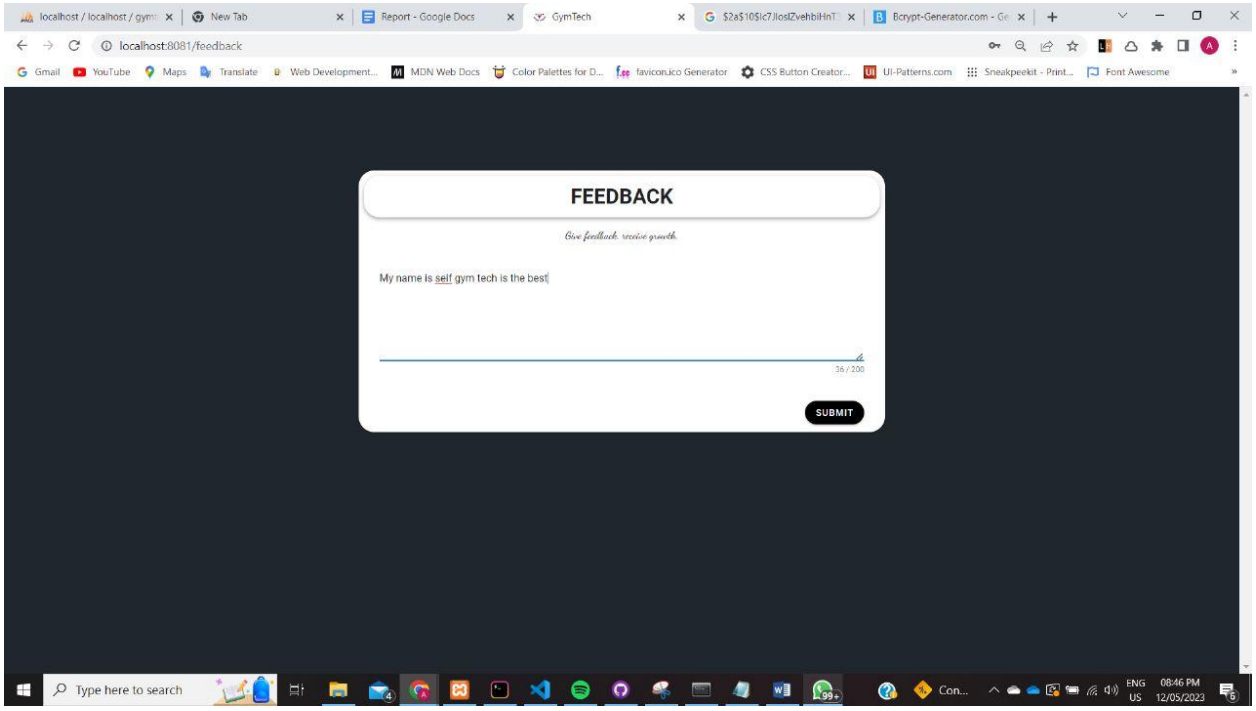


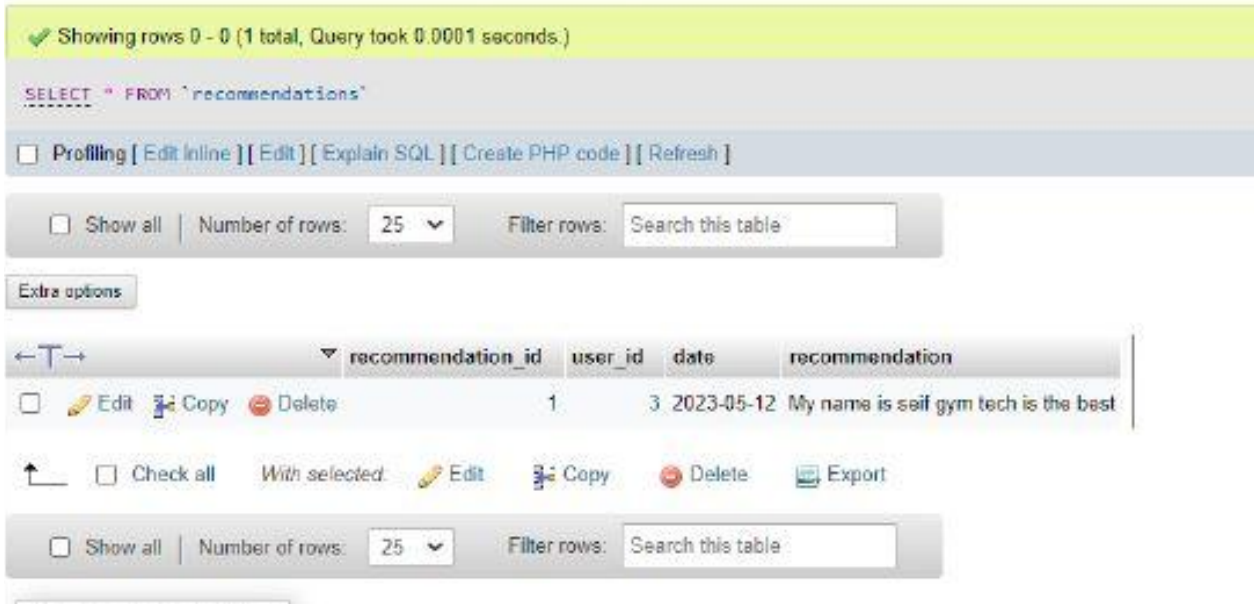


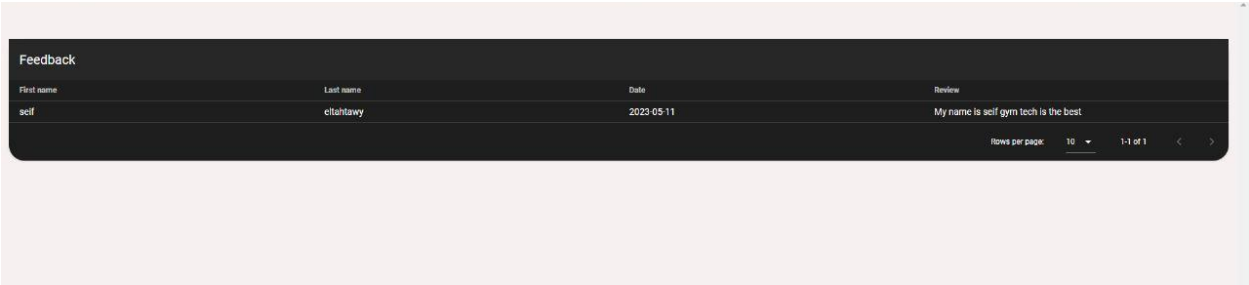
Trying to finish a workout with numeric fields:



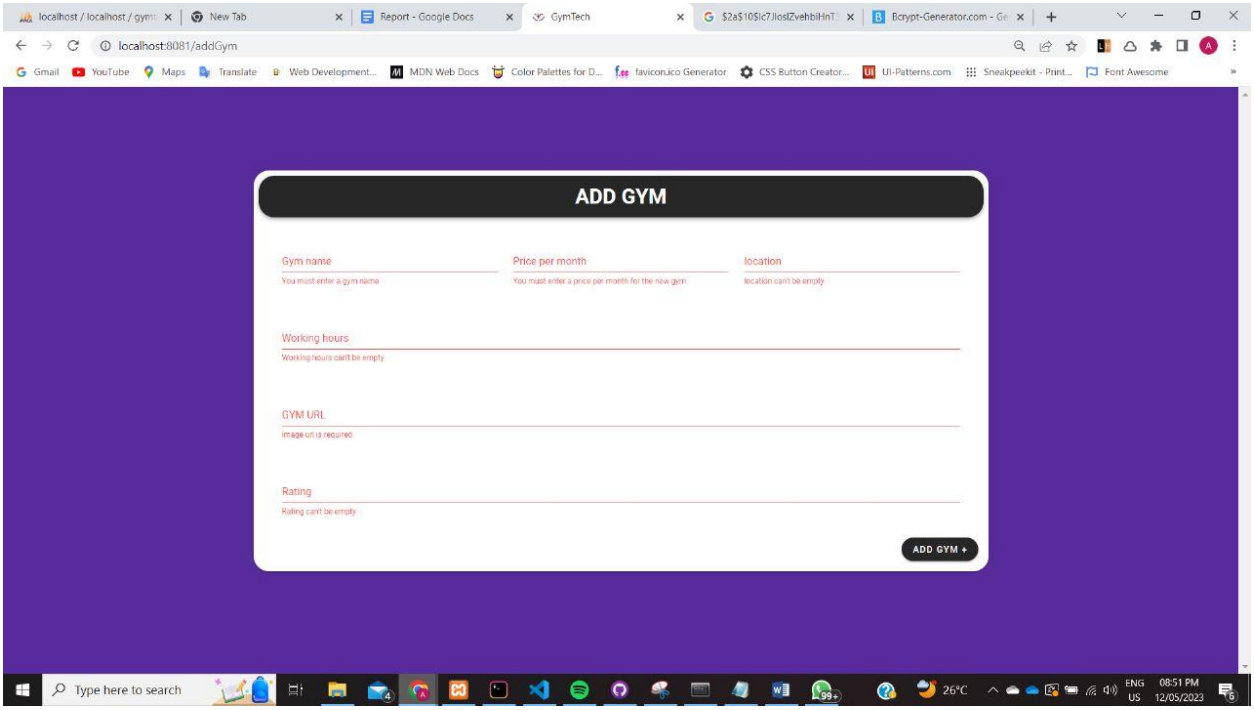
Trying to add a feedback:



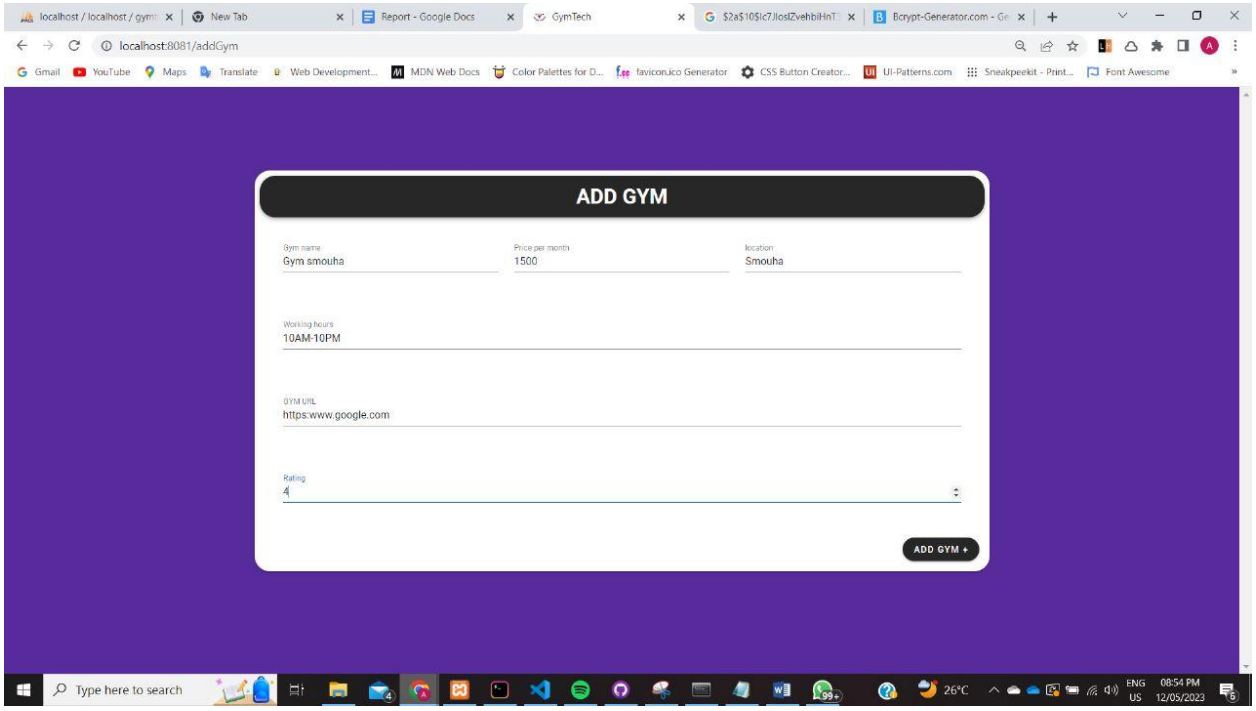


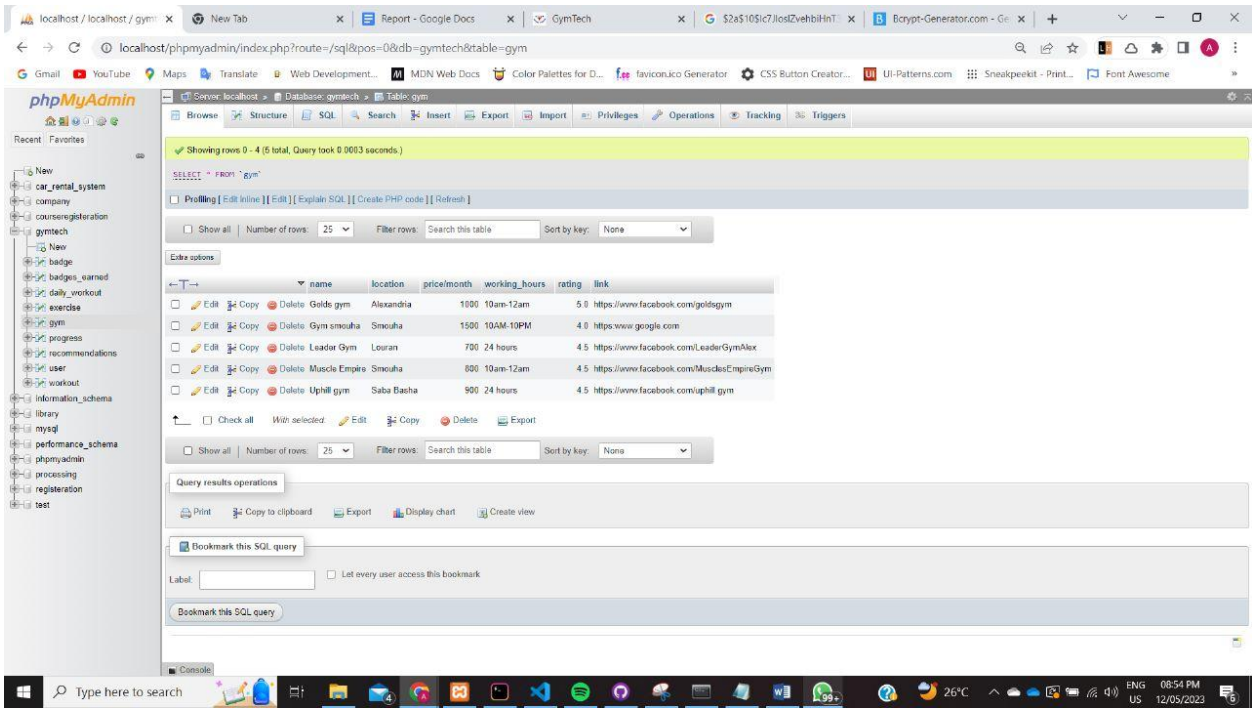


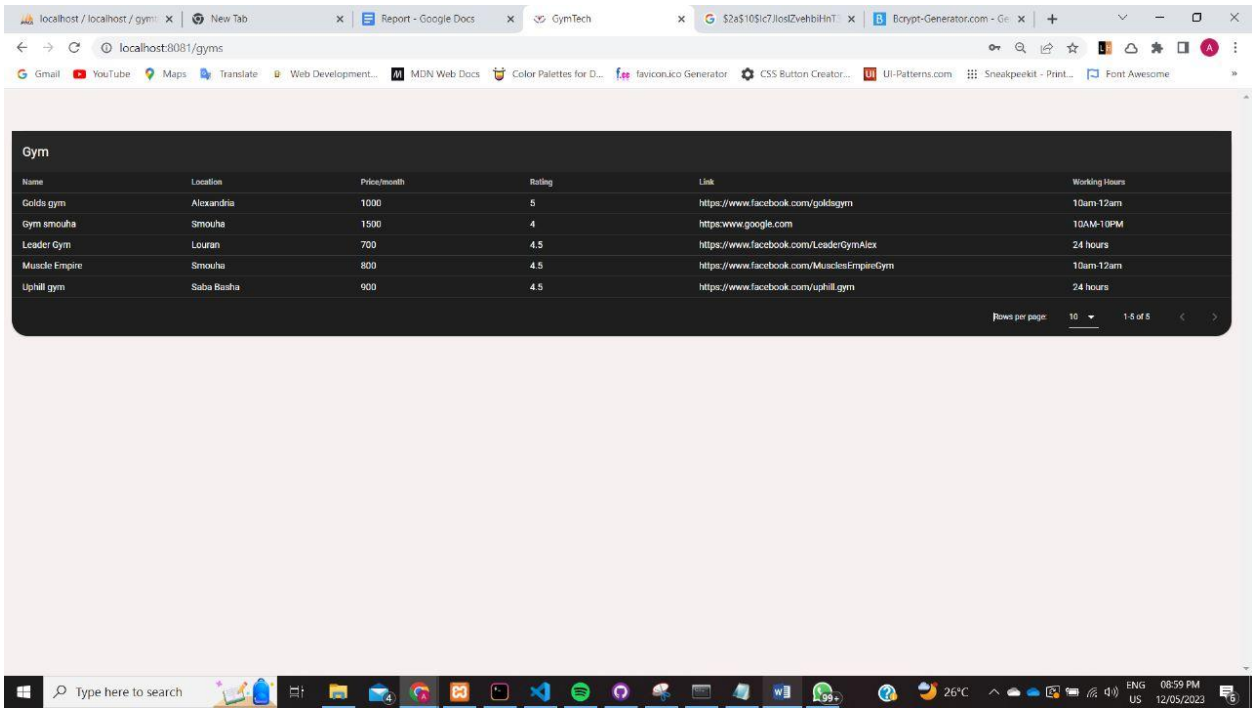
Trying to add a gym as an admin with empty fields:



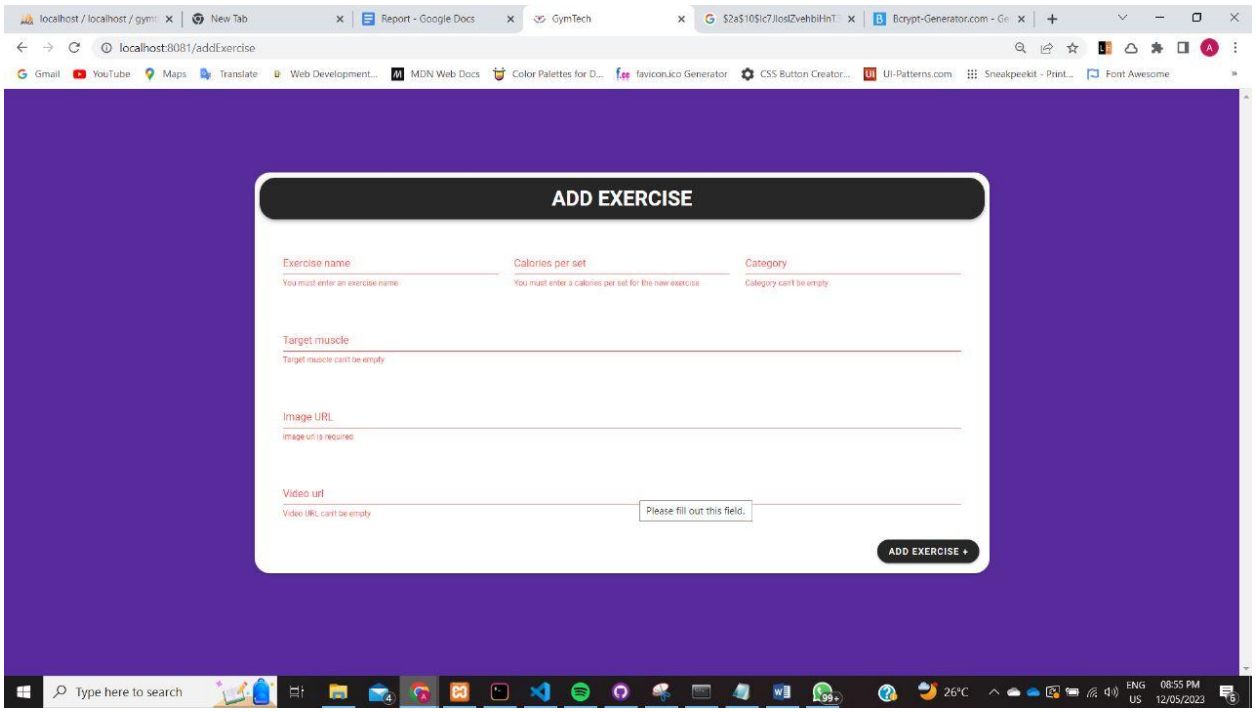
Trying to add a gym as an admin with non empty fields:



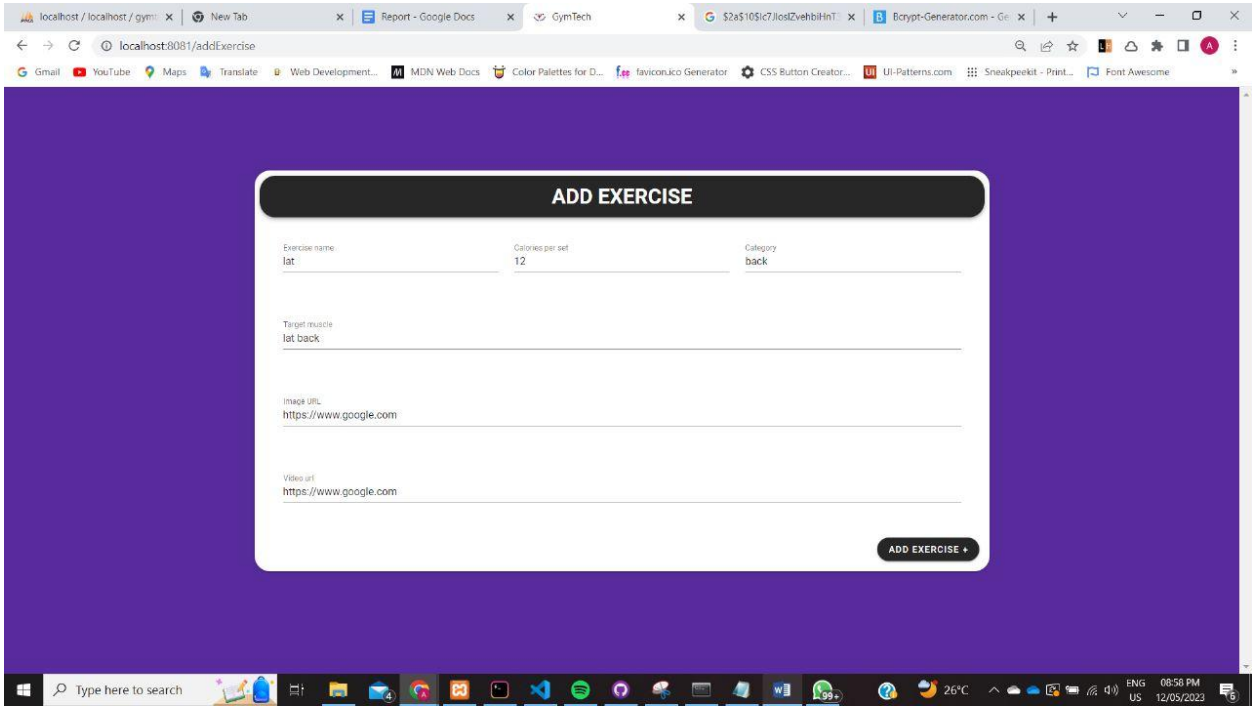


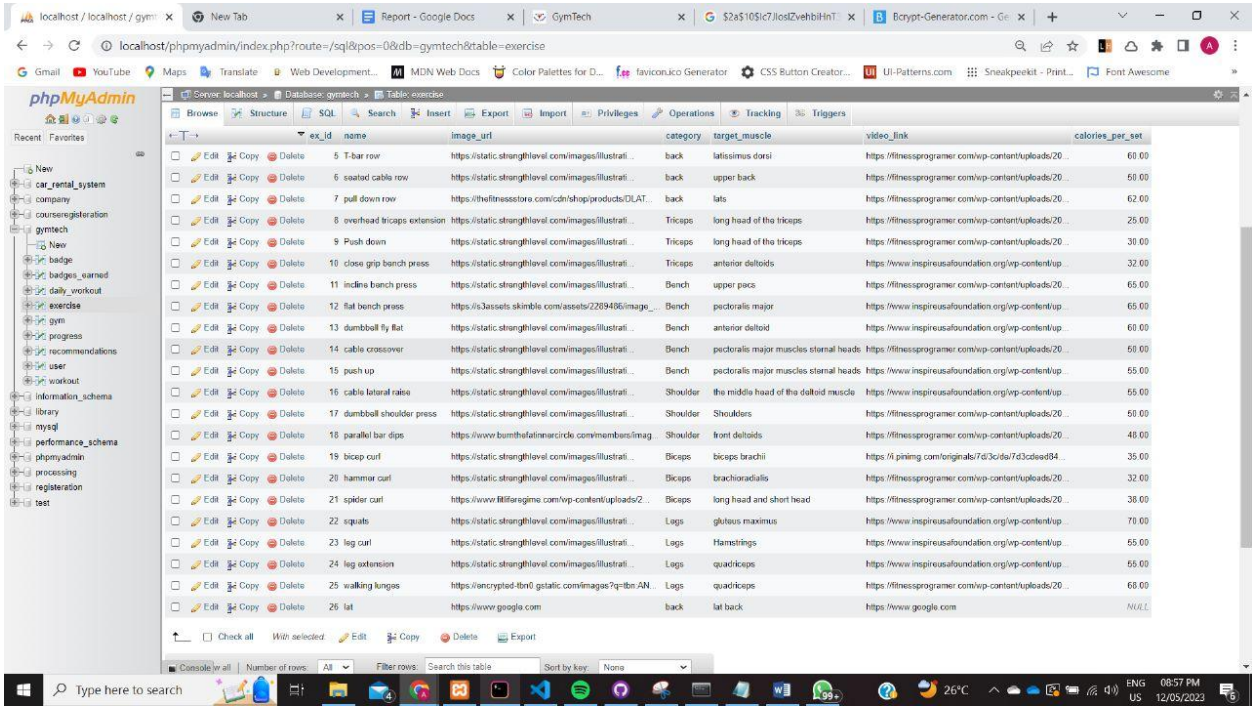


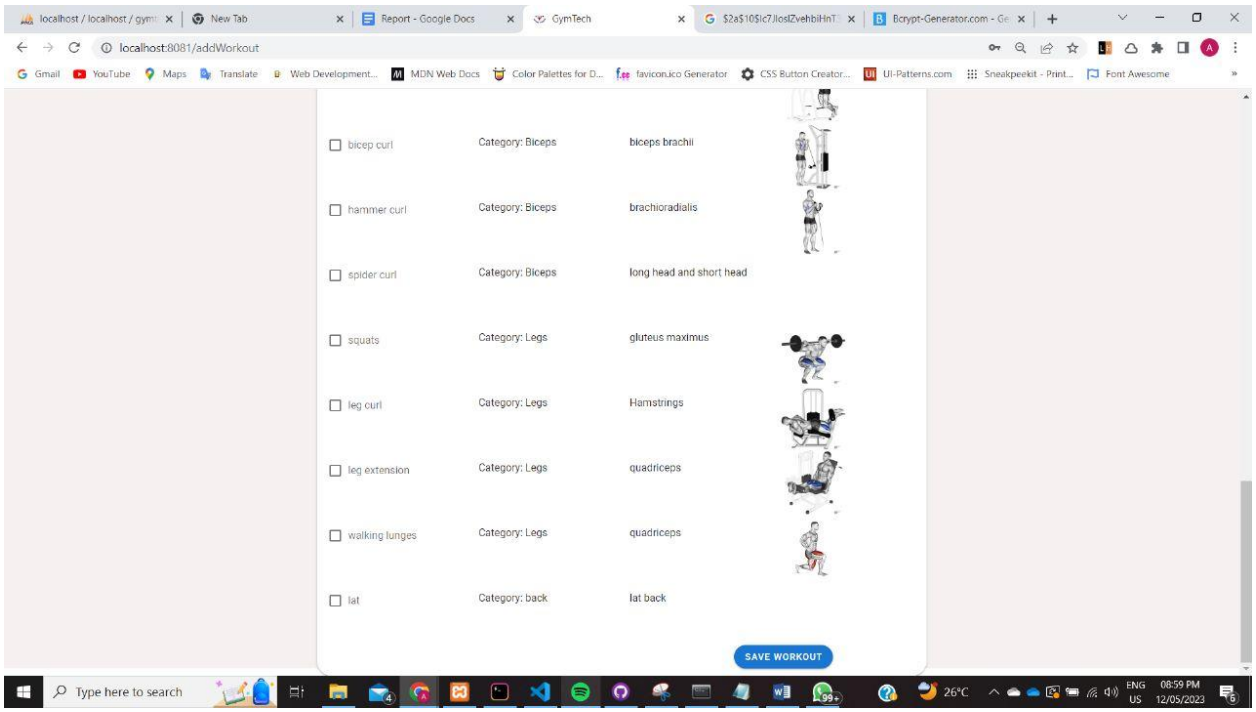
Trying to add an exercise as an admin with empty fields:



Trying to add an exercise as an admin with non empty fields:






localhost / localhost / gym
New Tab
Report - Google Docs
GymTech
Bcrypt-Generator.com
localhost:8081/addWorkout
Gmail
YouTube
Maps
Translate
Web Development...
MDN Web Docs
Color Palettes for D...
faviconico Generator
CSS Button Creator...
UI-Patterns.com
Sneakpeekit - Print...
Font Awesome
bicep curl
Category: Biceps
biceps brachii
hammer curl
Category: Biceps
brachioradialis
spider curl
Category: Biceps
long head and short head
squats
Category: Legs
gluteus maximus
leg curl
Category: Legs
Hamstrings
leg extension
Category: Legs
quadriceps
walking lunges
Category: Legs
quadriceps
lat
Category: back
lat back
SAVE WORKOUT
Type here to search
26°C
ENG
US
08:59 PM
12/05/2023