Workout Split Manager

CS 1103 Project Presentation

Vihaan Dumont – 3761518

Darin Thomson - 3776723

Abdelrahman Abdelsadek - 3764220

Dwyane Luy - 3775996

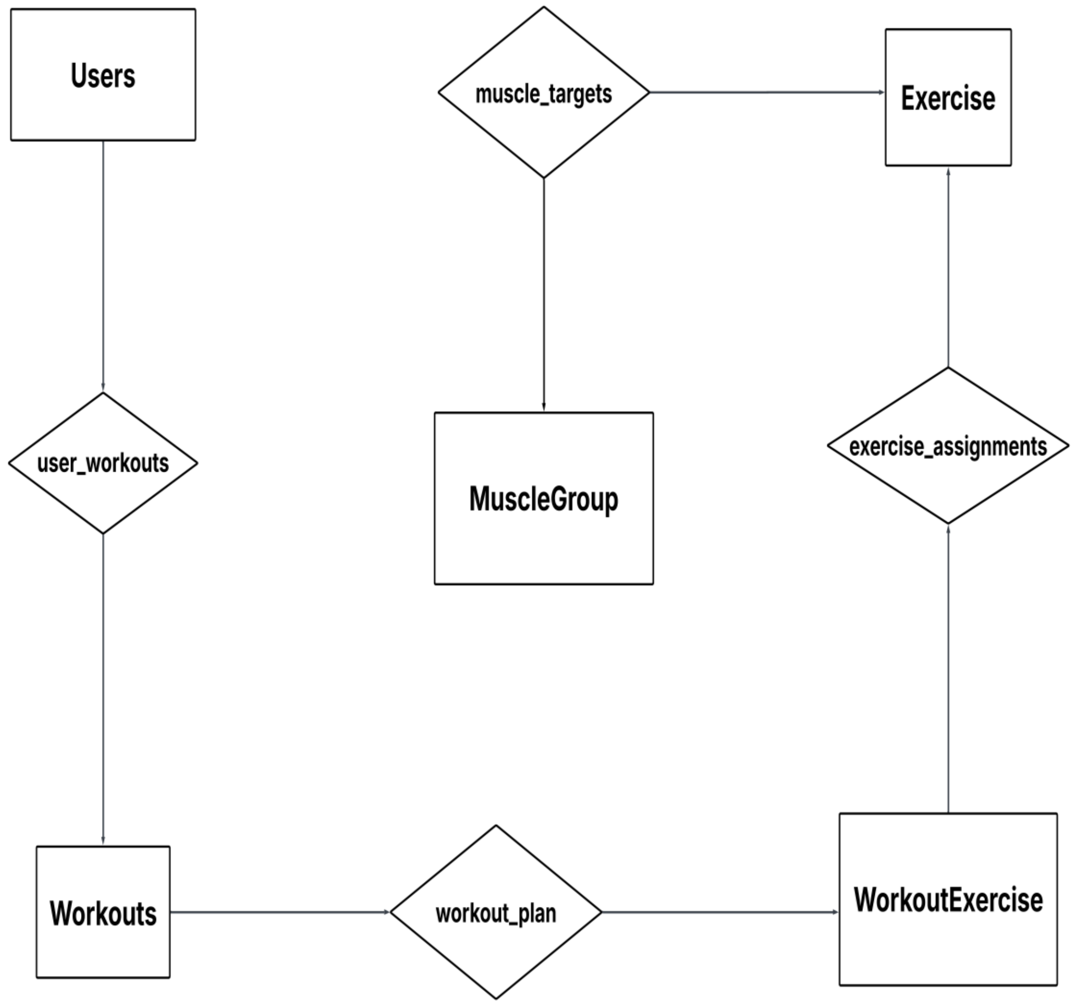
Introduction

Workout Split Manager is a Java desktop program that schedules and coordinates personalized fitness routines for multiple users. It simulates a live gym environment where each user follows a systematic 5-day workout routine based on his/her goal, membership status, and available days.

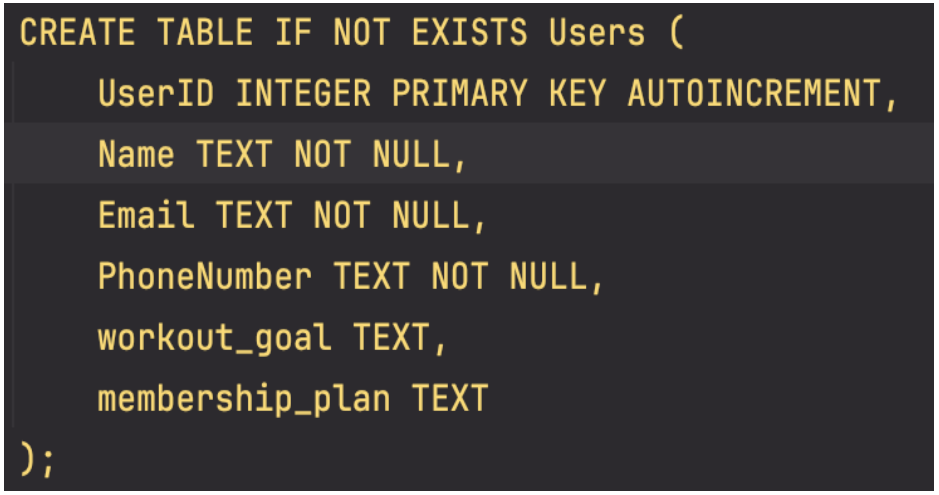
With Java, SQLite, and JavaFX, the program demonstrates the integration of a relational database and a graphical user interface. In the backend, there are well-normalized tables such as Users, Workouts, Exercises, MuscleGroups, and WorkoutExercise, and the frontend provides interactive data visualisation through styled table views and buttons.

One of the strengths of this project is its randomization mechanism for data, wherein exercises are dynamically created and assigned to each user with an eye towards variety and customization. This project focuses on the practicality of database design, ER modeling, and Java programming skills in a real-world fitness management context.

ER Diagram

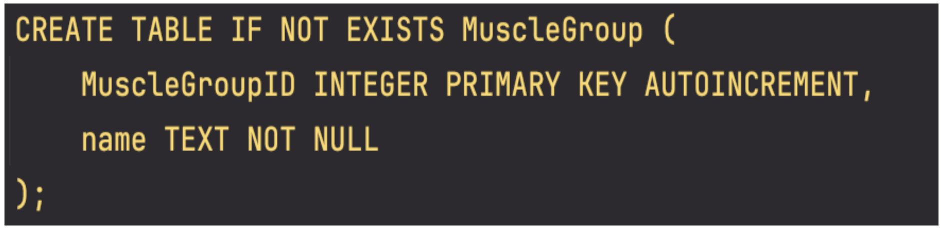

Users
muscle_targets
Exercise
user_workouts
MuscleGroup
exercise_assignments
Workouts
workout_plan
WorkoutExercise

SQL DDL: Users



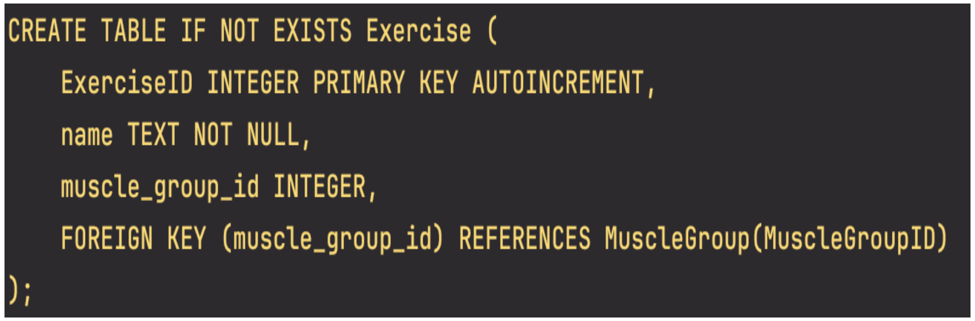
```
CREATE TABLE IF NOT EXISTS Users (
    UserID INTEGER PRIMARY KEY AUTOINCREMENT,
    Name TEXT NOT NULL,
    Email TEXT NOT NULL,
    PhoneNumber TEXT NOT NULL,
    workout_goal TEXT,
    membership_plan TEXT
);
```

SQL DDL: MuscleGroup



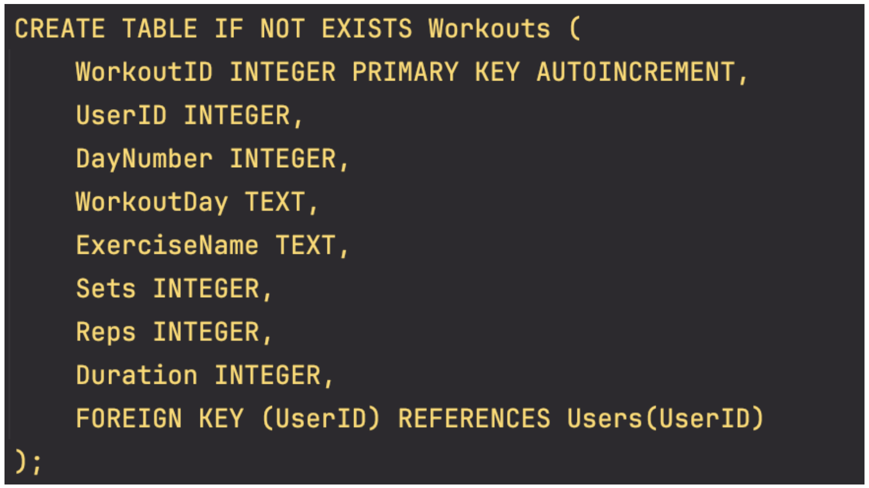
```sql
CREATE TABLE IF NOT EXISTS MuscleGroup (
    MuscleGroupID INTEGER PRIMARY KEY AUTOINCREMENT,
    name TEXT NOT NULL
);
```

SQL DDL: Exercise



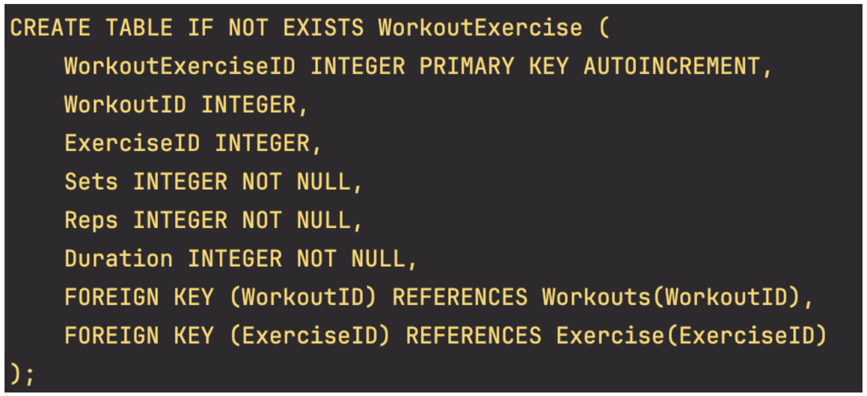
```sql
CREATE TABLE IF NOT EXISTS Exercise (
    ExerciseID INTEGER PRIMARY KEY AUTOINCREMENT,
    name TEXT NOT NULL,
    muscle_group_id INTEGER,
    FOREIGN KEY (muscle_group_id) REFERENCES MuscleGroup(MuscleGroupID)
);
```

SQL DDL: Workouts



```
CREATE TABLE IF NOT EXISTS Workouts (
    WorkoutID INTEGER PRIMARY KEY AUTOINCREMENT,
    UserID INTEGER,
    DayNumber INTEGER,
    WorkoutDay TEXT,
    ExerciseName TEXT,
    Sets INTEGER,
    Reps INTEGER,
    Duration INTEGER,
    FOREIGN KEY (UserID) REFERENCES Users(UserID)
);
```

SQL DDL: WorkoutExercise



```sql
CREATE TABLE IF NOT EXISTS WorkoutExercise (
    WorkoutExerciseID INTEGER PRIMARY KEY AUTOINCREMENT,
    WorkoutID INTEGER,
    ExerciseID INTEGER,
    Sets INTEGER NOT NULL,
    Reps INTEGER NOT NULL,
    Duration INTEGER NOT NULL,
    FOREIGN KEY (WorkoutID) REFERENCES Workouts(WorkoutID),
    FOREIGN KEY (ExerciseID) REFERENCES Exercise(ExerciseID)
);
```

DDL to ERD

Foreign keys enforce integrity across linked tables

- Example: `UserID` in `Workouts` references `Users(UserID)`
- Ensures each workout is connected to a valid user

WorkoutExercise handles many-to-many relationships

- Between `Workouts` and `Exercise`
- Allows multiple exercises per workout and vice versa

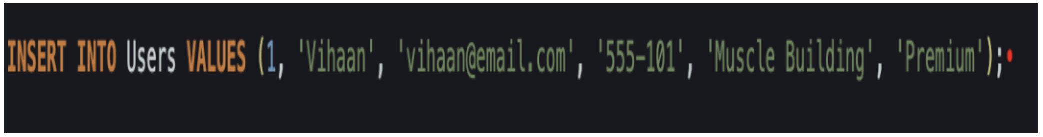
MuscleGroup organizes exercise categories

- Each exercise can target one specific muscle group
- Enables structured categorization of exercises

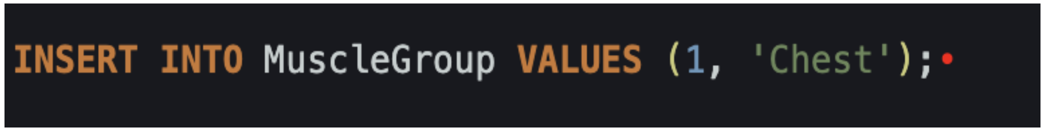
Workouts are linked to users via foreign key

- Each user can have multiple workouts
- Personalized plans are assigned using `UserID`

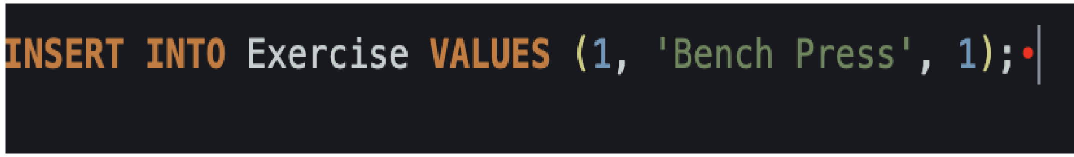
Sample INSERT Statements



```sql
INSERT INTO Users VALUES (1, 'Vihaan', 'vihaan@email.com', '555-101', 'Muscle Building', 'Premium');
```



```sql
INSERT INTO MuscleGroup VALUES (1, 'Chest');
```

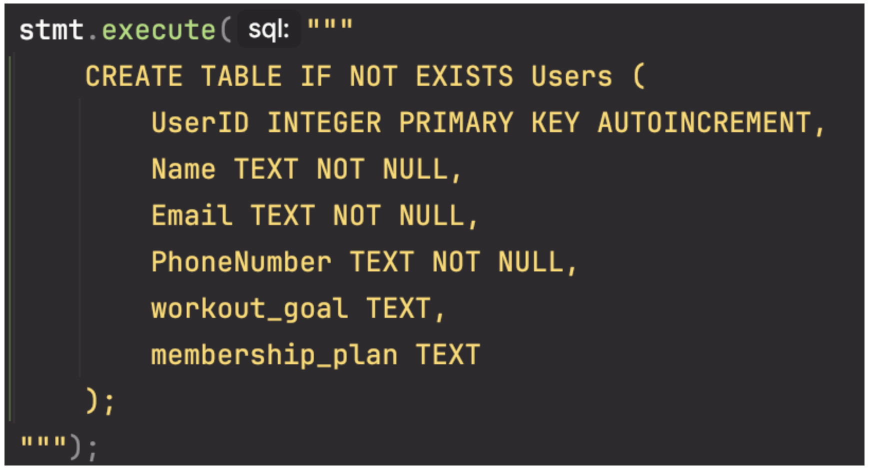


```sql
INSERT INTO Exercise VALUES (1, 'Bench Press', 1);
```

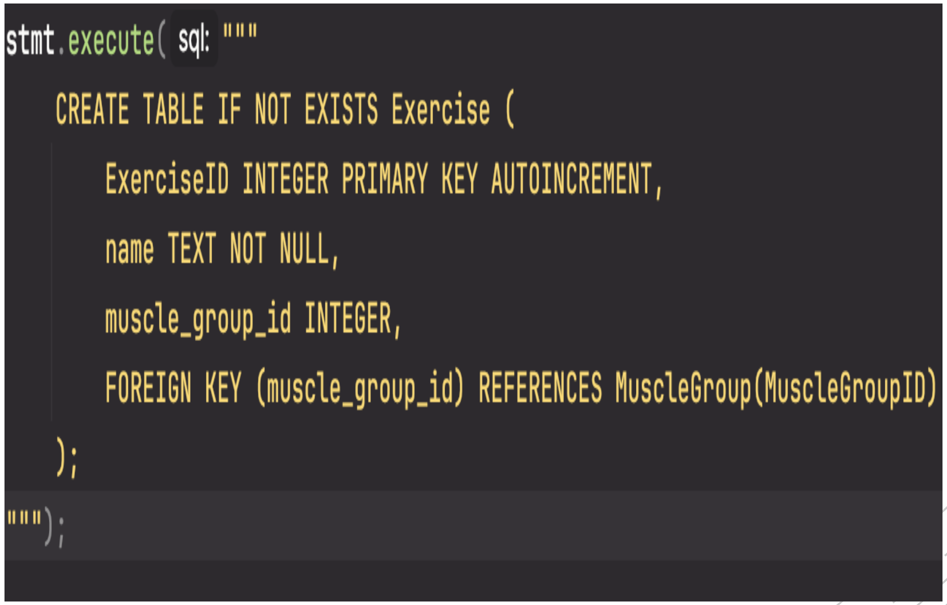
JDBC Connection

- Connection conn = DriverManager.getConnection("jdbc:sqlite:workout_split.db");

Creating Tables via Java



```
stmt.execute( sql: """
    CREATE TABLE IF NOT EXISTS Users (
        UserID INTEGER PRIMARY KEY AUTOINCREMENT,
        Name TEXT NOT NULL,
        Email TEXT NOT NULL,
        PhoneNumber TEXT NOT NULL,
        workout_goal TEXT,
        membership_plan TEXT
    );
""");
```



```
stmt.execute( sql: """
    CREATE TABLE IF NOT EXISTS Exercise (
        ExerciseID INTEGER PRIMARY KEY AUTOINCREMENT,
        name TEXT NOT NULL,
        muscle_group_id INTEGER,
        FOREIGN KEY (muscle_group_id) REFERENCES MuscleGroup(MuscleGroupID)
    );
""");
```

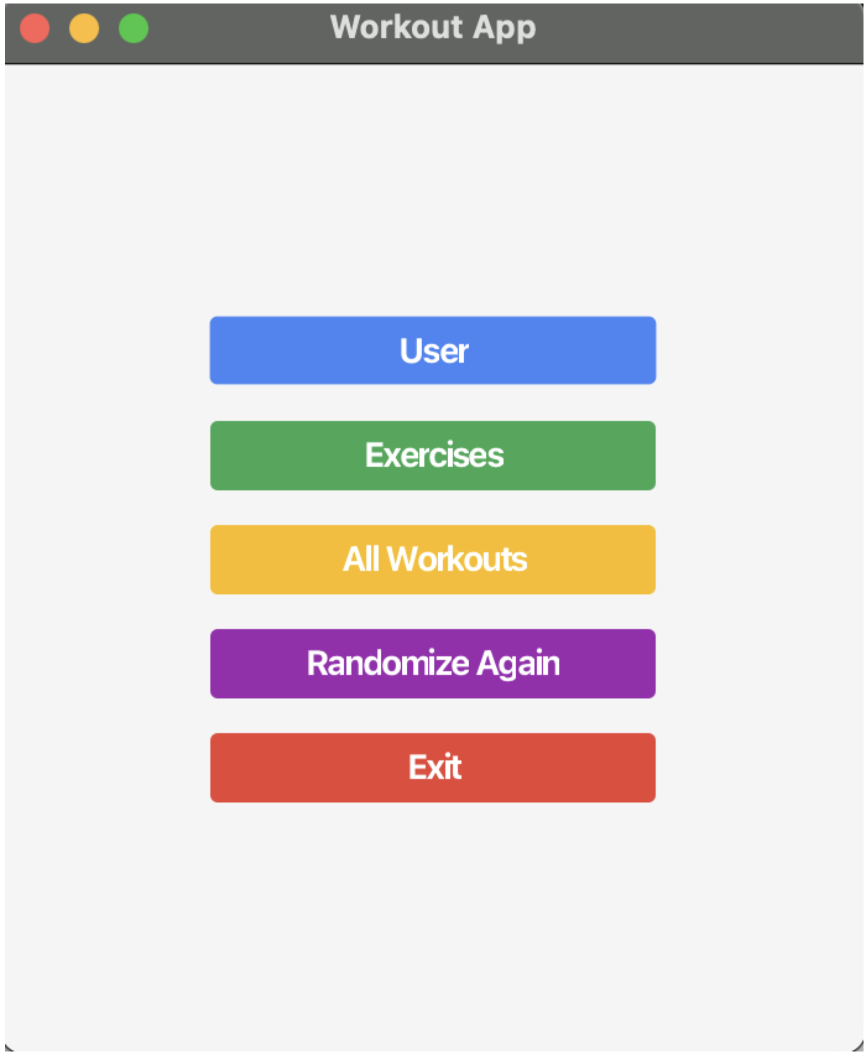
Inserting Randomized Data

- - 15 users
- - Each with 5-day randomized plan
- - Exercises, sets, reps, duration vary

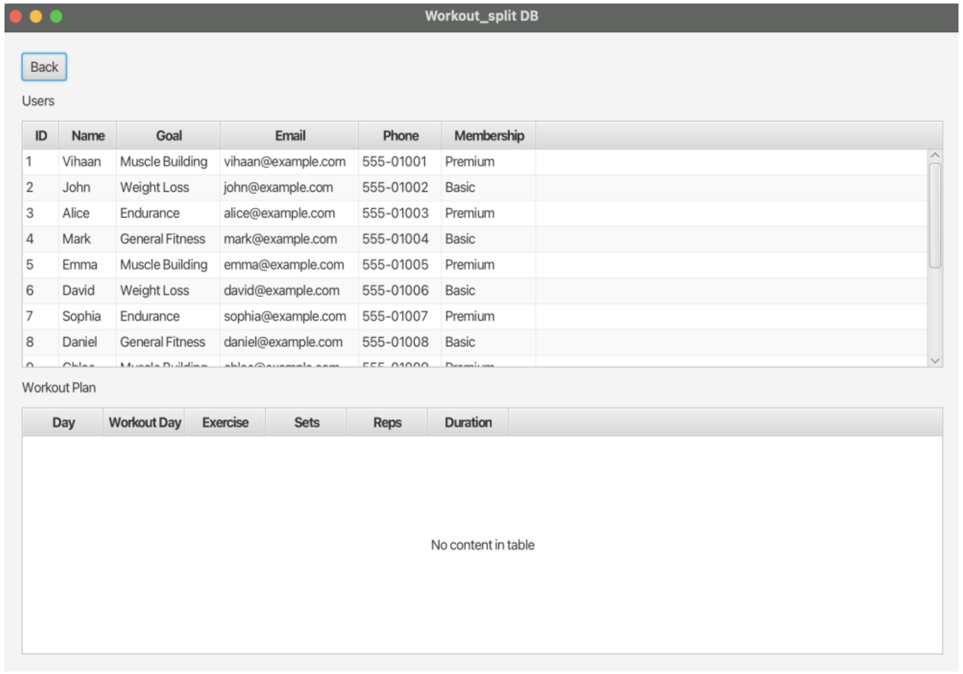
Special Features

- - Personalized 5-day plans
- - Workout days: Push, Pull, Legs, Cardio, Full Body
- - 2–3 exercises/day

JavaFX GUI


Workout App
User
Exercises
All Workouts
Randomize Again
Exit

User View



Users

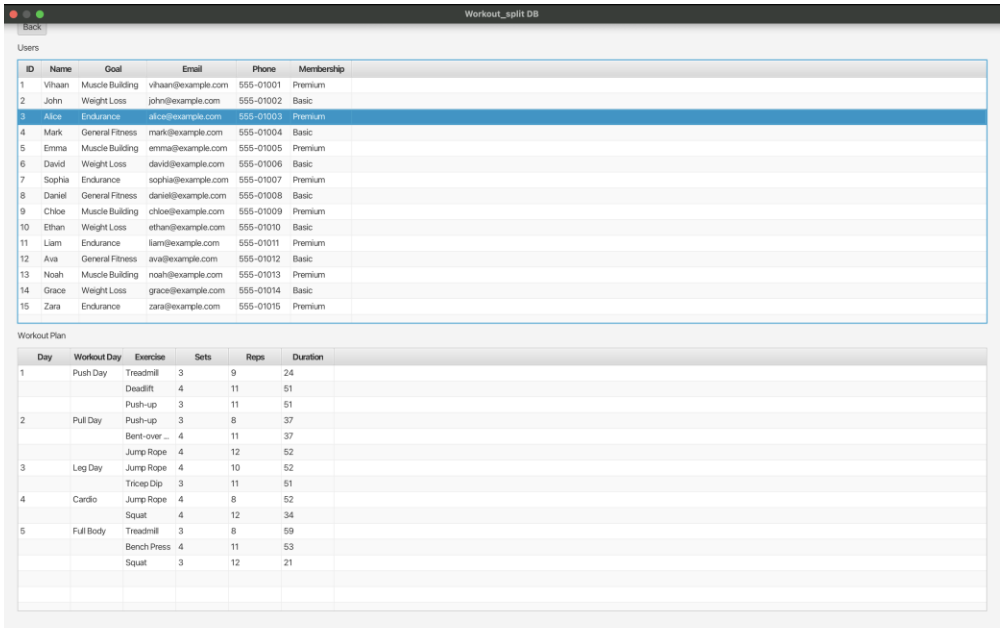
ID	Name	Goal	Email	Phone	Membership
1	Vihaan	Muscle Building	vihaan@example.com	555-01001	Premium
2	John	Weight Loss	john@example.com	555-01002	Basic
3	Alice	Endurance	alice@example.com	555-01003	Premium
4	Mark	General Fitness	mark@example.com	555-01004	Basic
5	Emma	Muscle Building	emma@example.com	555-01005	Premium
6	David	Weight Loss	david@example.com	555-01006	Basic
7	Sophia	Endurance	sophia@example.com	555-01007	Premium
8	Daniel	General Fitness	daniel@example.com	555-01008	Basic
9	Chloe	Muscle Building	chloe@example.com	555-01009	Premium

Workout Plan

Day	Workout Day	Exercise	Sets	Reps	Duration

No content in table

Workout Plan View



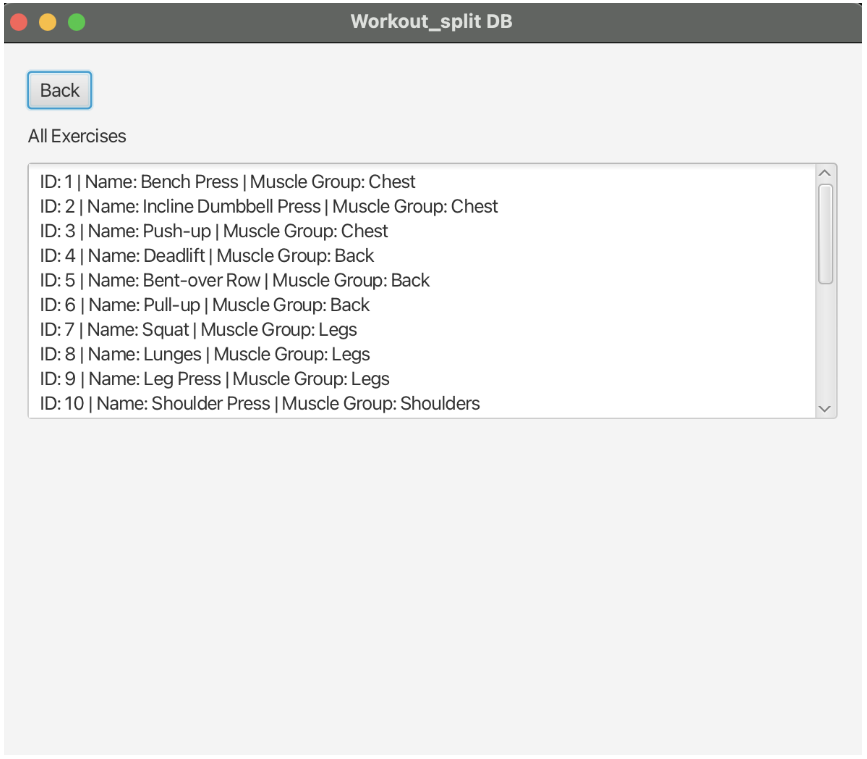
Users

ID	Name	Goal	Email	Phone	Membership
1	Vihaan	Muscle Building	vihaan@example.com	555-01001	Premium
2	John	Weight Loss	john@example.com	555-01002	Basic
3	Alice	Endurance	alice@example.com	555-01003	Premium
4	Mark	General Fitness	mark@example.com	555-01004	Basic
5	Emma	Muscle Building	emma@example.com	555-01005	Premium
6	David	Weight Loss	david@example.com	555-01006	Basic
7	Sophia	Endurance	sophia@example.com	555-01007	Premium
8	Daniel	General Fitness	daniel@example.com	555-01008	Basic
9	Chloe	Muscle Building	chloe@example.com	555-01009	Premium
10	Ethan	Weight Loss	ethan@example.com	555-01010	Basic
11	Liam	Endurance	liam@example.com	555-01011	Premium
12	Ava	General Fitness	ava@example.com	555-01012	Basic
13	Noah	Muscle Building	noah@example.com	555-01013	Premium
14	Grace	Weight Loss	grace@example.com	555-01014	Basic
15	Zara	Endurance	zara@example.com	555-01015	Premium

Workout Plan

Day	Workout Day	Exercise	Sets	Reps	Duration
1	Push Day	Treadmill	3	9	24
		Deadlift	4	11	51
		Push-up	3	11	51
2	Pull Day	Push-up	3	8	37
		Bent-over ...	4	11	37
		Jump Rope	4	12	52
3	Leg Day	Jump Rope	4	10	52
		Tricep Dip	3	11	51
4	Cardio	Jump Rope	4	8	52
		Squat	4	12	34
5	Full Body	Treadmill	3	8	59
		Bench Press	4	11	53
		Squat	3	12	21

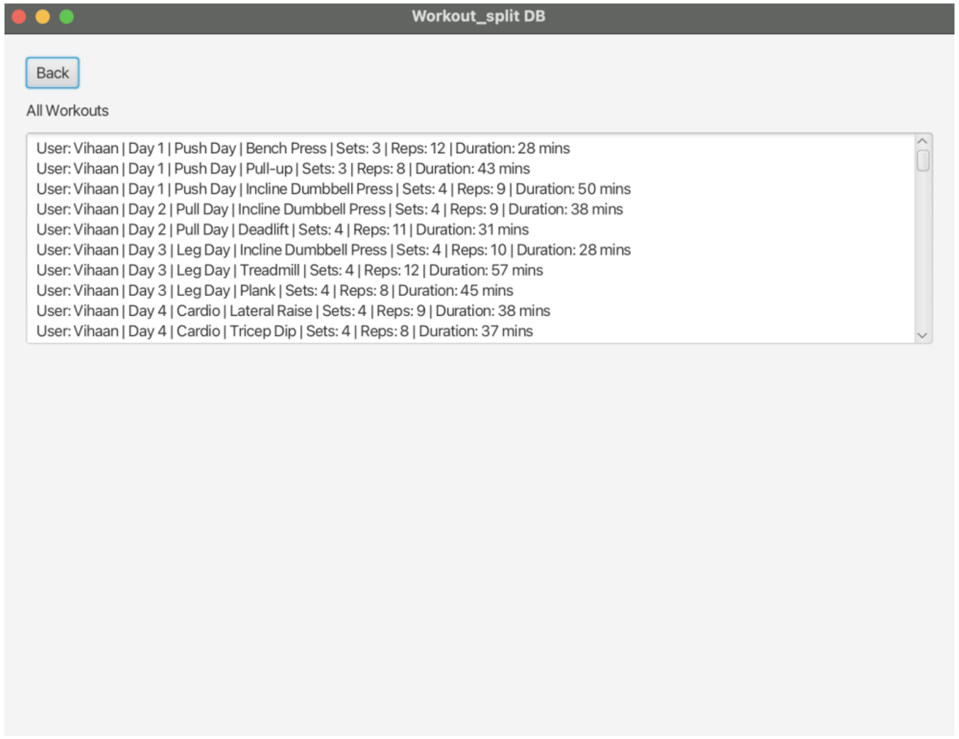
Exercise



All Exercises

```
ID: 1 | Name: Bench Press | Muscle Group: Chest
ID: 2 | Name: Incline Dumbbell Press | Muscle Group: Chest
ID: 3 | Name: Push-up | Muscle Group: Chest
ID: 4 | Name: Deadlift | Muscle Group: Back
ID: 5 | Name: Bent-over Row | Muscle Group: Back
ID: 6 | Name: Pull-up | Muscle Group: Back
ID: 7 | Name: Squat | Muscle Group: Legs
ID: 8 | Name: Lunges | Muscle Group: Legs
ID: 9 | Name: Leg Press | Muscle Group: Legs
ID: 10 | Name: Shoulder Press | Muscle Group: Shoulders
```

All Workouts View



```
User: Vihaan | Day 1 | Push Day | Bench Press | Sets: 3 | Reps: 12 | Duration: 28 mins
User: Vihaan | Day 1 | Push Day | Pull-up | Sets: 3 | Reps: 8 | Duration: 43 mins
User: Vihaan | Day 1 | Push Day | Incline Dumbbell Press | Sets: 4 | Reps: 9 | Duration: 50 mins
User: Vihaan | Day 2 | Pull Day | Incline Dumbbell Press | Sets: 4 | Reps: 9 | Duration: 38 mins
User: Vihaan | Day 2 | Pull Day | Deadlift | Sets: 4 | Reps: 11 | Duration: 31 mins
User: Vihaan | Day 3 | Leg Day | Incline Dumbbell Press | Sets: 4 | Reps: 10 | Duration: 28 mins
User: Vihaan | Day 3 | Leg Day | Treadmill | Sets: 4 | Reps: 12 | Duration: 57 mins
User: Vihaan | Day 3 | Leg Day | Plank | Sets: 4 | Reps: 8 | Duration: 45 mins
User: Vihaan | Day 4 | Cardio | Lateral Raise | Sets: 4 | Reps: 9 | Duration: 38 mins
User: Vihaan | Day 4 | Cardio | Tricep Dip | Sets: 4 | Reps: 8 | Duration: 37 mins
```

Conclusion

- The **Workout Split Manager** successfully integrates database principles with JavaFX to simulate a real-world gym scheduling system.
- It demonstrates the practical use of **ER modeling, SQL DDL/DML, JDBC**, and **Java GUI design** in a cohesive project.
- The system showcases how randomized, personalized workout routines can be dynamically generated and managed for multiple users.
- Through proper table design, foreign key constraints, and normalized relationships, the project ensures both data integrity and flexibility.
- Overall, this project reflects a strong understanding of how theoretical database concepts can be applied to solve real-world problems with intuitive, interactive software.