12/07/2024

*Digital Object Identifier N/A*

# JavaTrainer: A Java Framework For Exercise Tracking (Final Report)

**JUSTIN EUGENE[1], TONY NGUYEN[2], YAMEN ABOGAMIZA[3], AND JAKE SALTER.[4]**
[1]Computer Science Department, Middle Tennessee State University, Murfreesboro, TN 37129 USA

**ABSTRACT** This report details the progress of developing a desktop-based lifting tracker application with a graphical user interface (GUI). The project incorporates frontend GUI design and backend database support for storing user data. The app is designed to help users track their weightlifting progress, create personalized workout plans, and manage their fitness routines. By providing an intuitive interface for planning and modifying workouts, it aims to streamline the user's fitness journey. The target audience includes both beginner and advanced gym-goers who want a convenient way to log their lifting activities and receive tailored workout recommendations.

**INDEX TERMS** JavaFX, SQLite, Personal Fitness Goals, User Authentication

## I. INTRODUCTION

WITH this project we have used the Java programming language to create a GUI that can allow users to take control of their fitness journey. We want users of different ages to be able to achieve their fitness goals whether they are too skinny and want to put on weight or wish to lose a few pounds so that they may feel better about themselves. Our project Java Trainer can make it so that users have a unique and easy experience as they navigate through their fitness journey.

### A. PROJECT OVERVIEW

1) *Purpose and Goals:* JavaTrainer aims to empower individuals in their fitness journeys by providing a comprehensive and user-friendly tool for tracking workouts. Many individuals face challenges in maintaining a consistent workout routine. This application aims to streamline the process of logging workouts, and achieving personal fitness goals, ultimately enhancing user engagement and motivation in their fitness routines.
2) *Target Audience:* The Application is designed for a diverse group of users who are invested in their fitness and health journeys
    a) *Demographic Information:* The primary target audience includes individuals aged 18-40, comprising both males and females who are health-conscious and technologically adept. This demographic is likely to utilize mobile and desktop applications to enhance their fitness routines.
    b) *Fitness Levels:* The Application caters to a wide spectrum of fitness levels. Beginners new to weightlifting or fitness will be set up with simpler exercises. Intermediate users can use the application to refine their routines, track progress, and explore new training techniques.
3) *Key Features:* JavaTrainer provides a tailored experience to users of all skill levels, allowing customization, personalization, and personally curated recommendations.
4) *Technology Stack:* Java GUI is utilized for front-end interactions, a JDBC driver is used for connections with a SQLite database.

## II. SYSTEM DESIGN

### A. FRONTEND: GUI FRAMEWORK

For the GUI Framework JavaFX was used for this project. It has a vast library with many helpful features allowing for a simple and clean construction of this project.

### B. BACKEND: DATABASE AND AUTHENTICATION

A database is necessary for several key components of the application. For a user, a database stores both the information of the user's login - the username and password, and also their current progress so that new exercise recommendations can be made daily. From an algorithmic standpoint, a view of all our exercises needs to be stored and organized based on their difficulty, body parts, or equipment needed.

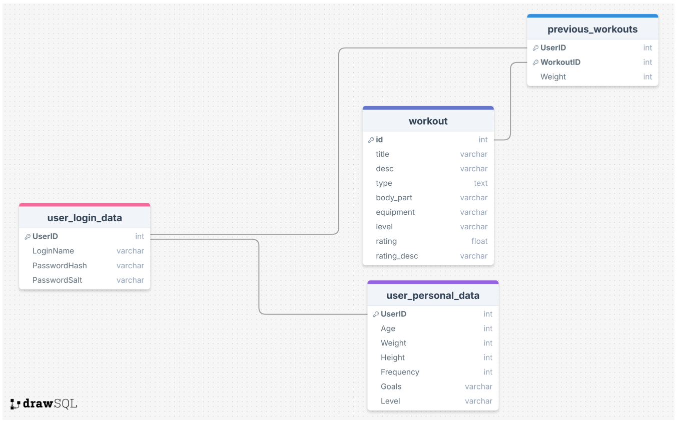


FIGURE 1. An outline for the JavaTrainer database schema.

## III. TECHNOLOGY OVERVIEW

A brief outline of the implemented components.

### A. GUI DEVELOPMENT

For the development of the GUI, we decided to go for a blue-and-white color scheme to keep a friendly and simple appearance. The login and welcome screens have been completed and functioned rather well. They welcome the user and provide them with a survey asking things such as how many times a week they want to work out or what their goal is for working out rather than weight loss or strength gain (etc). The GUI also provides a login and register screen for them so that they can either sign up or log back in as they please. After the user completes their questionnaire successfully they will then be given their workout plans over a 7 day period. The user still has full ability to go back and forth between the days and login and register screen as well as a button to return back to the home page on the 7th workout day.

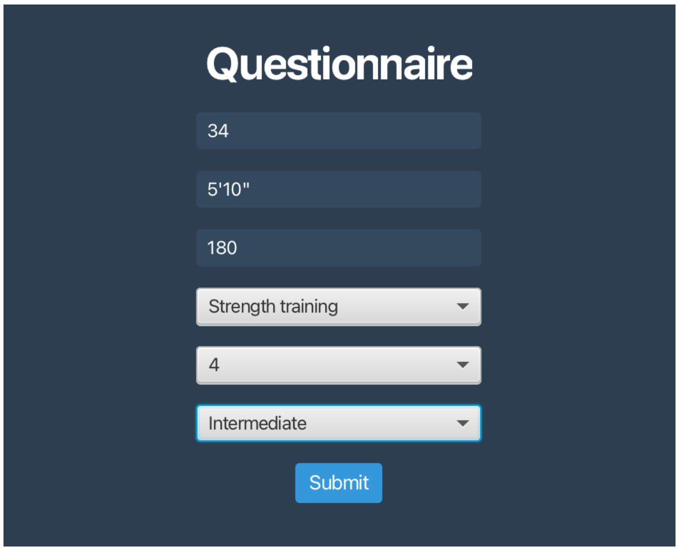
### B. DATABASE SETUP

SQLite is a lightweight database framework that can be utilized by an application locally, without establishing a server for data storage. Working in a development environment, with version control managed on GitHub, this local solution fits the problem design. The utilized schema is outlined in Fig. 1. In Java, an SQL JDBC driver is required to make connections to an SQLite database, available for download as a JAR file from a variety of authors and maintainers.

### C. AUTHENTICATION IMPLEMENTATION

When storing a customer's personal information, such as a password, having security standards is necessary. Hashing a password is the process of abstracting a collection of characters into another collection. The starting string is input for a hashing algorithm function that adheres to the following set of rules:

1) A variable length input produces a fixed length output
2) The algorithm is easily and efficiently computed in one direction but computationally infeasible to reverse


Questionnaire

34

5'10"

180

Strength training

4

Intermediate

FIGURE 2. Questionnaire to obtain user info

3) The output should be collision-free, meaning that no two inputs produce the same or similar outputs
4) Has a level of diffusion - two similar inputs should produce a largely different output

These rules are standards meant to ensure that the hashing algorithm keeps the private information of the users safe - having an algorithm that fails any of these leaves the obscured passwords vulnerable.

To further enhance the diffusion of the hashed passwords, salt is added. Hash salt is a small string of characters added to a password to add a level of randomness to each password stored in the database. Adding even small strings of characters creates a unique password database where users who enter the same password, e.g. 'password123', would still generate unique keys.

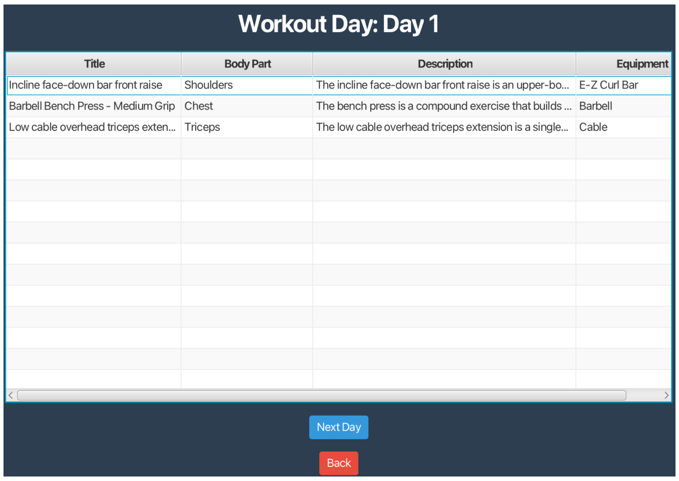
### D. USER SESSIONS

All sections of our application require access to the currently signed-in user's information from the database - this includes their login information, their demographic information, their goals, and the previous exercises we have recommended. This information is best stored in a singleton pattern, a design pattern that limits the number of instantiated objects to just one [1].

```
public class CurrentUserSession {
 private static CurrentUserSession instance;
 private User current_user;

 private CurrentUserSession() {};
```

This class is called in the login method to instantiate the current user if the login credentials provided match what's in the database. This class is also called in the workout recommendation methods to find the user's goals, like the frequency of workouts they want to do, their goals, and experience level.



Workout Day: Day 1

| Title | Body Part | Description | Equipment |
|---|---|---|---|
| Incline face-down bar front raise | Shoulders | The incline face-down bar front raise is an upper-bo... | E-Z Curl Bar |
| Barbell Bench Press - Medium Grip | Chest | The bench press is a compound exercise that builds ... | Barbell |
| Low cable overhead triceps exten... | Triceps | The low cable overhead triceps extension is a single... | Cable |

**FIGURE 3.** Table showing displayed workouts.

### E. WORKOUT RECOMMENDATION

To recommend the workouts, Justin created a database with close to 450 different workouts. The workouts can be classified by the targeted muscle, level of experience, and type. The class, WorkoutPlan, holds the specialized workout plan for each user. The class, WorkoutPlanBuilder, is used to create and customize a workout plan based on user preferences, fitness goals, and physical capacity. The WorkoutPlanBuilder dynamically selects workouts from the database by filtering them according to the user's selected criteria, such as muscle groups they want to focus on, their experience level (beginner, intermediate, or advanced), and the type of workout (strength, hypertrophy, endurance, etc.).

Once the WorkoutPlanBuilder compiles the relevant workouts, it generates a personalized plan, which is then associated with the WorkoutPlan class for that user. This plan can be modified over time, based on progress or changing fitness goals. The WorkoutPlan class ensures that each user's plan is stored in the database and can be retrieved for future use, allowing for easy tracking and adjustments as the user's performance improves.

## IV. INDIVIDUAL CONTRIBUTIONS

### A. FRONTEND DEVELOPMENT CONTRIBUTIONS

- Yamen Abogamiza
  1) Created welcome page
  2) Created login/register page
  3) Created questionnaire
  4) Created screens for all 7 days
  5) Created back-and-forth buttons to traverse back and forth between the screens at the user's discretion.

### B. BACKEND DEVELOPMENT CONTRIBUTIONS

- Justin Eugene
  1) Provided the data for all workouts: Compiled and organized comprehensive workout data, including exercise descriptions, required tools, and difficulty levels for various workout categories. This data was structured for easy integration with the database to ensure accurate retrieval and user-specific recommendations.
  2) Developed the WorkoutPlanBuilder class: Designed and implemented the algorithm within the WorkoutPlanBuilder class to dynamically generate personalized workout plans for each user. The class takes into account user preferences, fitness levels, goals, and available equipment to curate effective, well-balanced workout routines.
  3) Created the WorkoutPlan class: Developed a class to hold and manage the workout plan objects. This class organizes the workout schedule, exercise details, and tracks progress throughout the plan. It integrates seamlessly with the database and the WorkoutPlanBuilder class to ensure user-specific workout plans are updated and accessible.
- Jake Salter
  1) Designed the schema for the database, as well as writing the SQL code for database creation. Also installed the JDBC driver for the app to interact with the database for user authentication, workout recommendation, and progress tracking.
  2) Created a class for password hashing and salting. This allows JavaTrainer to securely store a hash instead of a literal text password.
  3) Created CurrentUserSession class to instantiate a current user with database information in a single class which can be called anywhere.
  4) Created the getExercise method, which returns an ArrayList of exercise objects from the database. Queried based on the level, body part, and type.

### C. SYSTEM INTEGRATION/CONNECTIONS

- Tony Nguyen
  1) Completed Algorithm and Backend integrations
  2) Debugged GUI for Start Up Screen
  3) Debugged Connection Issues arising from GUI with WorkoutRecommendation Class

## V. CHALLENGES AND SOLUTIONS

### A. FRONTEND/BACKEND INTEGRATION ISSUES

Connecting the databases to the GUI and making it all sync up so that it runs was rather difficult as connecting GitHub with Eclipse proved to be rather challenging.

### B. DATABASE STRUCTURING CHALLENGES

The biggest challenge was connecting to a SQLite database. Java has many built-in classes for many different use cases, but SQLite databases are not one of them. To accomplish this connection, a JDBC driver had to be installed manually by downloading a JAR file from a third-party contributor.

### C. VERSION CONTROL

Sharing files and updates across the various group members was challenging, particularly because of Eclipse's integration with GitHub. After adding the src/ files to our GitHub, it was difficult to pull the files into an Eclipse project, so a new project had to be built every time a member pulled the files. This was later resolved by including the .project, .classpath, and .modulepath files which are often added to a .gitignore file to make the project cross-platform to other IDEs. This allowed group members to pull an existing Eclipse project from the GitHub and JavFX was already installed. This was only found after several hours of frustration.

## VI. CONCLUSION

### A. SUMMARY

Our team was able to implement a workout recommendation and tracker that is tailored to a user's individual needs by utilizing JavaFX, Java, and SQLite. JavaTrainer excels in speed and accuracy with meticulously designed recommendation algorithms, a purposeful and stylish fronted GUI, and a mature data-handling back end. All members of this team grew immensely over the semester and worked hard to master our domains.

### B. LEARNING OUTCOMES

Members learned crucial skills in communication, version control, file sharing, GUI development, robust algorithm writing, database management and design, and project delivery through the undertaking of this project.

## REFERENCES

[1] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, 1994.

[2] Oracle Corporation. Java Platform, Standard Edition Documentation. 2024. Available at: https://docs.oracle.com/javase/. Accessed: December 5, 2024.

• • •