Staria – Capstone Project Documentation (2025)

Team: *Jorge Pereira, Jose Menendez, Humberto Monguia, Alejandro Rodriguez*
Course: Computer Science Capstone II – KFSCIS, FIU
Instructor: Masoud Sadjadi
Semester: Fall 2025

1. Executive Summary

Staria is an iOS-based fitness App that allows each user to design their own workout routine while tracking all their exercise activities with ease and engagement. Many individuals struggle to maintain their commitment toward fitness due to how many current fitness applications are set up. Some applications are difficult to navigate, while others do not allow users to complete a workout in a particular way or use different techniques depending on the exercise being performed.

Staria offers an intuitive platform that fosters mobile engagement by allowing users to build their own workout routines, have guided workouts, and track their progress anytime, anywhere. Staria utilizes React Native as its front-end framework and Appwrite for authentication and data storage, thus, providing the functionality for users to create personalized workout routines by adding exercises to those routines, documenting the sets and/or reps performed during workouts, and receiving performance summaries.

Staria was built with beginner fitness enthusiasts that are looking for an easy introduction to the world of fitness in mind but will also serve intermediate and advanced fitness enthusiasts to organize their training. The following sections will provide details on Staria's problem space and requirements, architecture, development processes, testing and deployment, future improvements, and most importantly the potential uses for future Staria users.

2. Problem and Requirements

2.1 Context and Stakeholders

People who are looking for flexible workout plans will find the most value in Staria. This includes beginner fitness enthusiasts who value straightforward instruction and experienced exercisers who are knowledgeable about standard fitness routines. Generally, trainers and coaches may benefit from being able to easily create workout routines for their clients.

There are three primary user personas:

1. A beginner looking for step-by-step instructions for a workout routine.

2. An intermediate user seeking to create a routine that is tailored to his/her needs.

3. An advanced user focused on tracking long-term progress.

2.2 Problem Summary

Developing a successful workout plan and tracking it for consistency is a challenge for many users. Without consistency, they lack motivation and struggle to see improvements. Our mission at Staria is to develop a user-friendly, straightforward tool that enables users to effectively build and execute their workouts with a minimal amount of confusion.

2.3 Functional Requirements

Users will have the ability to:

- Create workout routines.

- Add exercises to their routines with the following: Sets, Reps, Time

- Track workout sessions and marked off exercises.

- Create a user account and login.

- Access a summary of previously completed workout sessions.

- Navigate seamlessly on the iPhone.

- Keep the interface readable and accessible.

2.4 Non-Functional Requirements

- When switching screens, the application must be responsive to the user's touch.
- All user data must be protected if it is stored or retrieved.
- The layout of the application must be easily understandable.
- The system must manage common error scenarios smoothly.

2.5 Constraints

- The app will only work on iOS devices.
- Appwrite is the service being used to power the app.
- The project duration and number of people working on it, along with the potential issues between app updates, could impact the development of the project.

2.6 Success Criteria

- Users can create, build and store a custom workout.
- A user will be able to complete a workout from beginning to end.
- All workout data should be accurately stored and retrieved.
- The user interface should be simple and user-friendly while doing physical activities.

2.7 Risks

- There are some issues that could arise due to mobile network latency.
- While running a workout, if state updates aren't properly accounted for, the user's workout data could not match what was on their device.
- Network latency may also negatively impact mobile app performance.

2.8 Backlog Overview

- User login
- User registration
- Create workout routine
- Add exercises to a routine
- Edit routines
- Delete routines
- Start guided workout session
- Track progress during a session
- View progress page
- Profile page
- Exercise library
- Routines saved in Appwrite
- Basic offline behavior
- Onboarding screen

3. System Overview and Architecture

3.1 Architecture Summary

Frontend:

- React Native
- Screens (Home, Create Workout, Workout Session, Profile)
- Context/state management

Backend:

- Appwrite for database, authentication, storage, functions

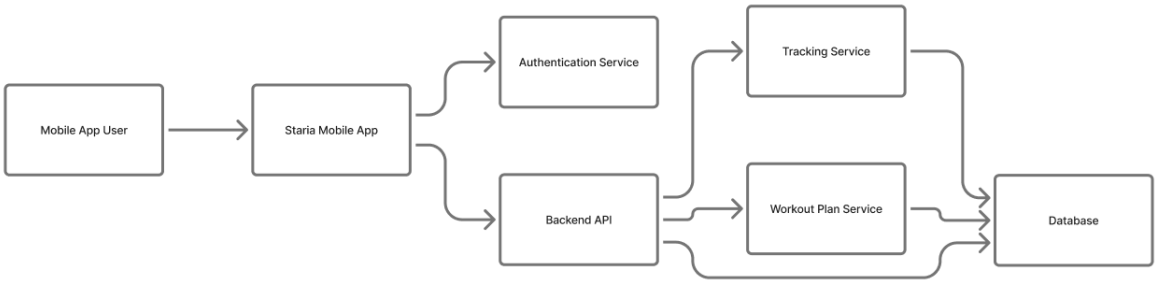
APIs:

- Appwrite REST SDK

Data Storage:

- Collections: Users, Workouts, Exercises, Sessions

3.2 Architecture Diagram



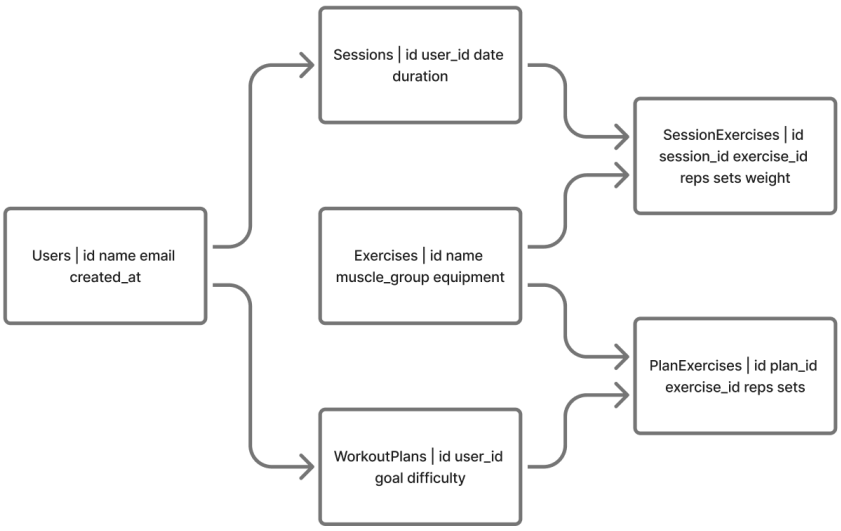
3.3 Key Technologies

- **React Native** (iOS app UI)

- **Appwrite** (auth, storage, database)

- **GitHub** (version control, CI/CD)

3.4 Data Model Overview



4. Discipline-Specific Depth

4.1 Algorithm & Data Structure Choices

- Efficient use of React Native state hooks and lists for rendering workouts.
- Simple normalized structure of data about routines and exercises.
- Lookup operations are optimized by Appwrite indexes.

4.2 Architecture & Patterns

- Component-based UI structure.
- Clean separation of screens and state logic.
- Appwrite was used as BaaS to reduce backend overhead.

4.3 Engineering Evidence

- Performed basic performance checks, observing the behavior of a system.
- Utilized profiling to identify bottlenecks and other areas of improvement
- Wrote simple scripts to repeat scalability experiments

5. Implementation Notes

5.1 Repository Structure

/staria

  /src

  /.vscode

  .gitignore

  README.md

  app.json

  babel.config.js

  eslint.config.js

  jsconfig.json

  package.json

  package-lock.json

5.2 Coding Conventions

- Follows JavaScript conventions using Expo and React Native.
- Consistent use of functional components.
- Hooks (useState, useEffect, etc.) used for state and lifecycle management.

- Screens and components separated inside the /src folder.

- Linting managed by ESLint to maintain uniform formatting.

5.3 Notable Tradeoffs

- Component-Based Architecture: Screens and UI elements are reusable and modular.

- State Management: Managed using React hooks for simplicity.

- Backend-as-a-Service (Appwrite): Chosen to avoid building a custom backend and reduce development complexity.

- Expo: Selected to streamline mobile development and testing, at the cost of reduced native-level customization.

6. Testing & Evaluation

6.1 Testing Strategy

- Unit Tests: Component logic and validation.

- Integration Tests: Appwrite interactions.

- E2E Tests: Workout flow (create → track → summary).

- Manual Testing: Device testing on iOS.

6.2 KPIs

- Successful workout creation rate.

- App load time (under 2 seconds).

- No crashes during workout sessions.

6.3 Example Test Cases

1. Authentication
2. Creating workouts
3. Editing workouts
4. Tracking sessions
5. Progress display
6. Offline behavior
7. Error handling (network loss)

7. Security, Privacy, Accessibility & Compliance

7.1 Security

- Appwrite handles secure auth tokens.
- HTTPS communication.
- Input validation on forms.

7.2 Privacy

- User workout data stored in private collections.
- No sensitive health data beyond exercise logs.

7.3 Accessibility

- Large tap targets, high contrast UI.

- Clear typography.

7.4 Ethical Considerations

- Avoids misleading health claims.
- Provides balanced and customizable workout suggestions.

8. Deployment & Operations

8.1 Deployment

- iOS build via Xcode / EAS
- Appwrite cloud instance or self-hosted
- GitHub for version control

8.2 Operations

- Monitor Appwrite dashboard.
- Maintain backups and logs.

9. Limitations & Future Work

9.1 Current Limitations

- Limited exercise library
- Minimal social features

- No Android version yet

- Offline mode is basic

9.2 Future Improvements

- Workout recommendations based on goals

- Social/community features

- Enhanced analytics & statistics

- Cross-platform Android support

- Personalized reminders and scheduling

10. References

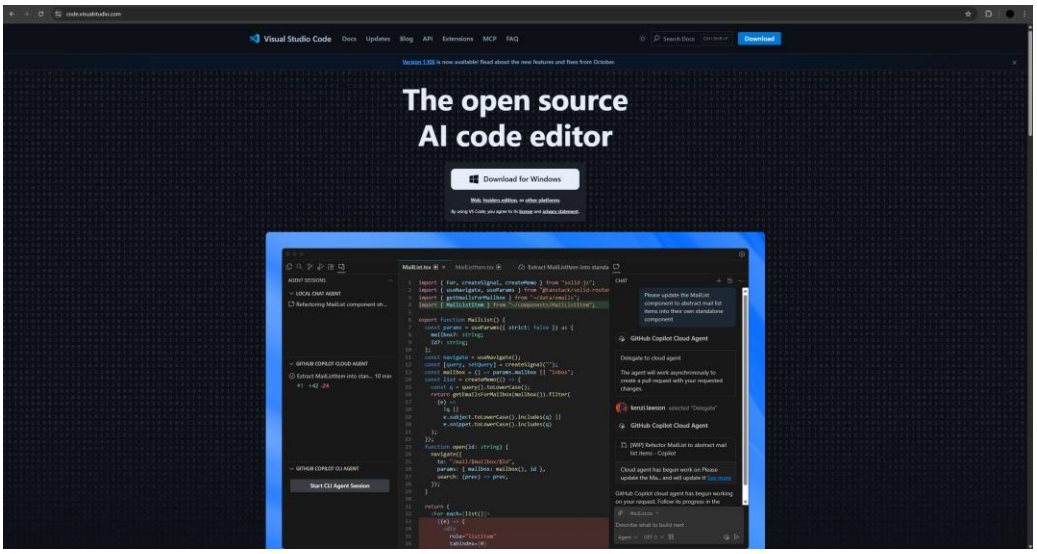
1. Appwrite. 2025. *Appwrite: Open-source backend server for web, mobile, and flutter applications*. https://appwrite.io. Accessed Nov. 12, 2025.
2. Appwrite Documentation. 2025. *Appwrite SDKs and API reference for integrating authentication, databases, and cloud functions*. https://appwrite.io/docs. Accessed Nov. 13, 2025.
3. Expo. 2025. *Expo – Tools and services for building React Native apps*. https://expo.dev/. Accessed Sep. 24, 2025.
4. GitHub, Inc. 2025. *GitHub – Version control and collaboration platform*. https://github.com. Accessed Sep. 4, 2025.
5. React. 2025. *React – A JavaScript library for building user interfaces*. https://reactjs.org/. Accessed Sep. 7, 2025.
6. React Native. 2025. *React Native – A framework for building native apps using React*. https://reactnative.dev. Accessed Sep. 8, 2025.

Appendices

A. Installation Guide

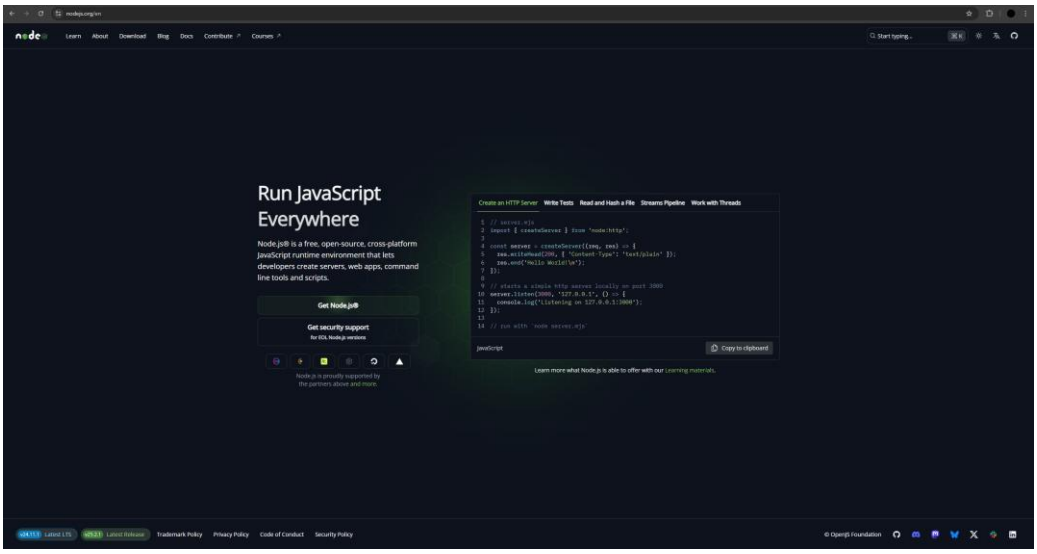
1. Download VSCode or Another IDE

VSCode Link: VSCode



2. Download Node.js

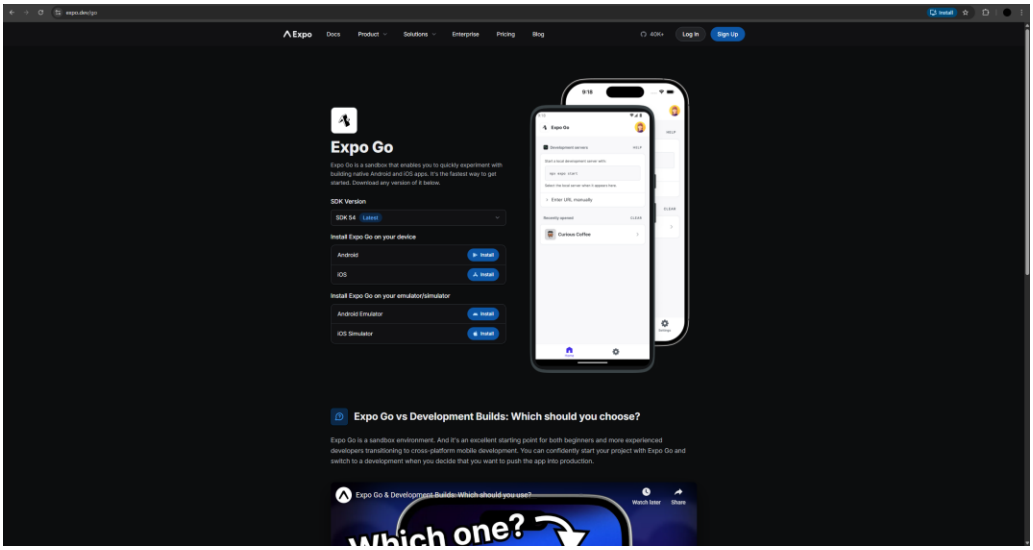
Node.js Link: Node.js



3. Download The Expo Go Application in Your Mobile Device

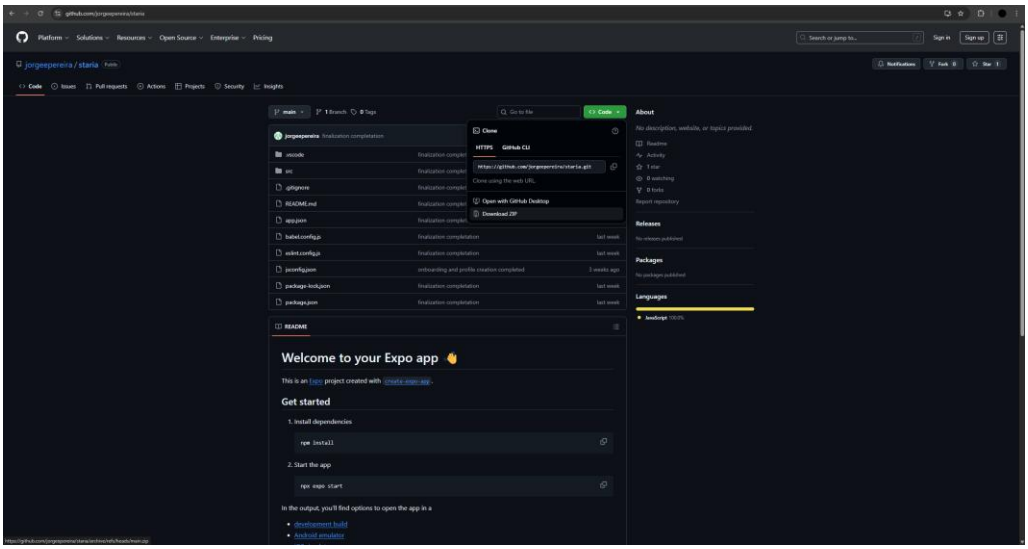
Expo Go Link: Expo Go

Note: Use The App Store or Play Store to Download the App Depending on the Device

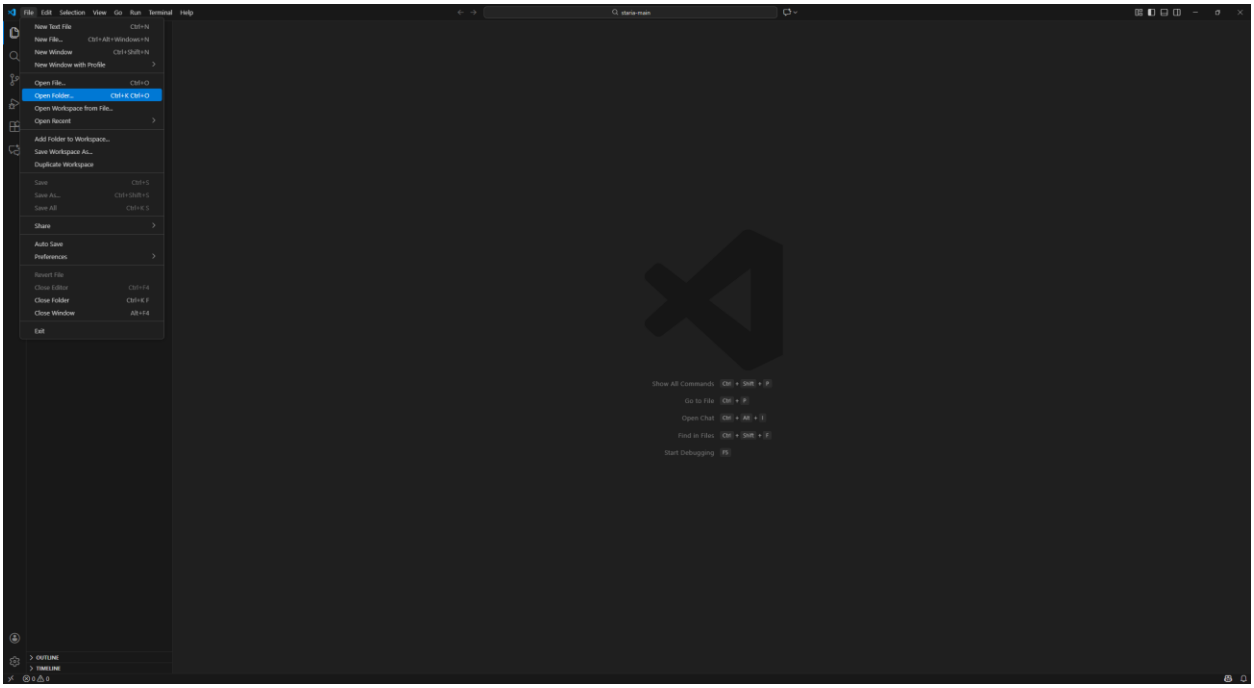


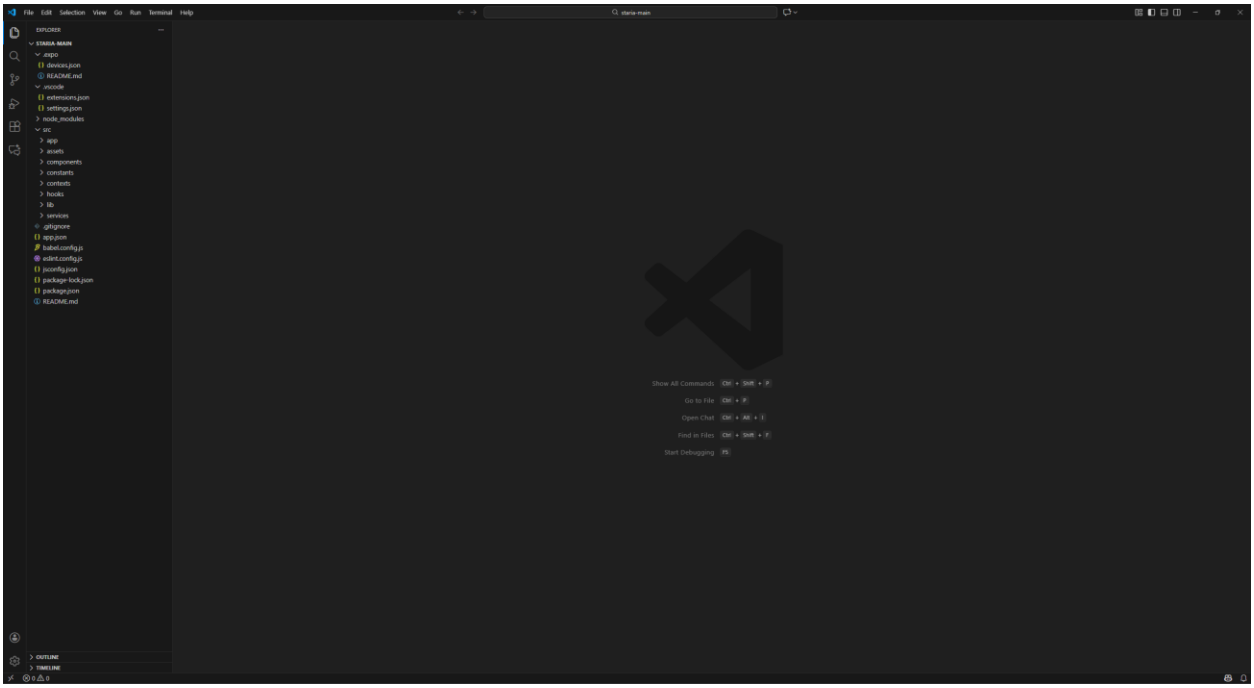
4. Download Staria GitHub Repository

Staria GitHub Repository Link: Staria GitHub Repository

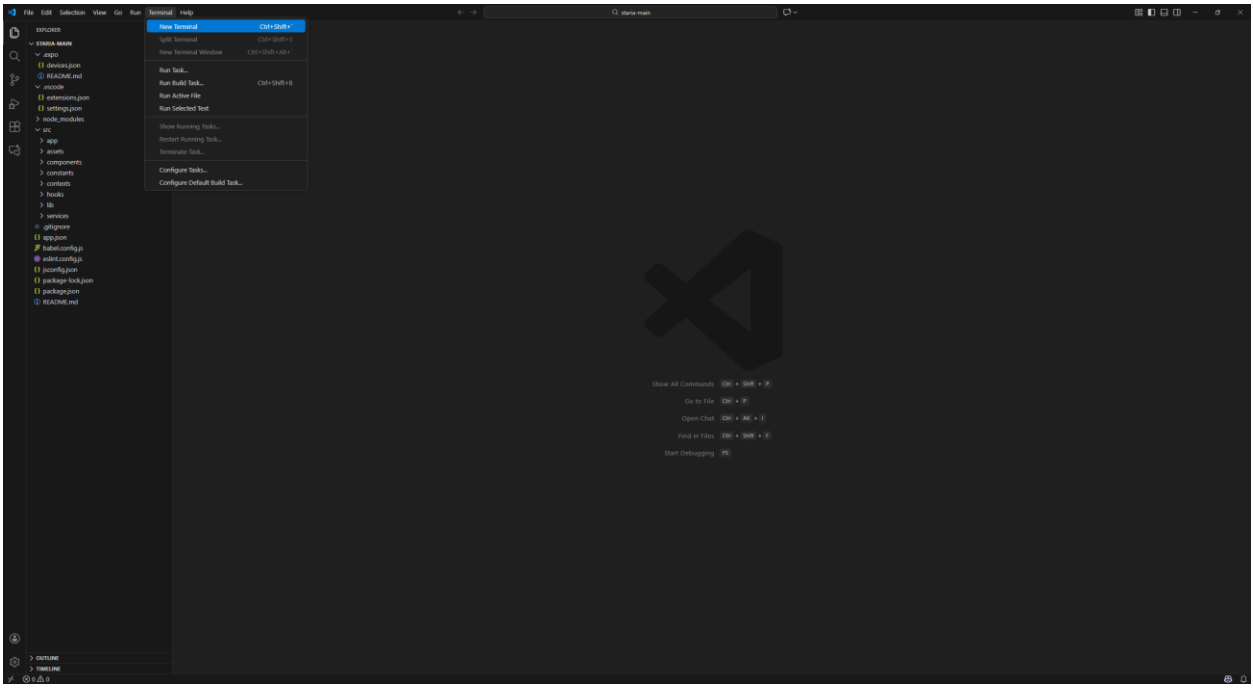


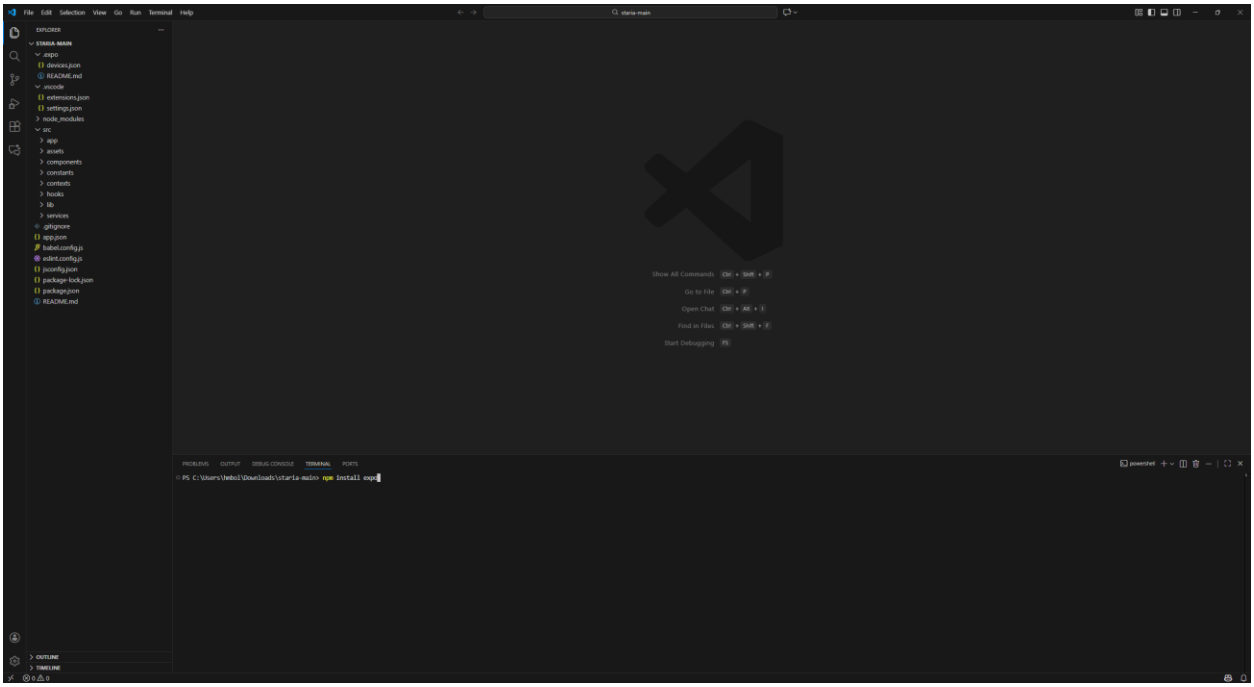
5. Extract The Zip File and Open the Staria Folder in VSCode



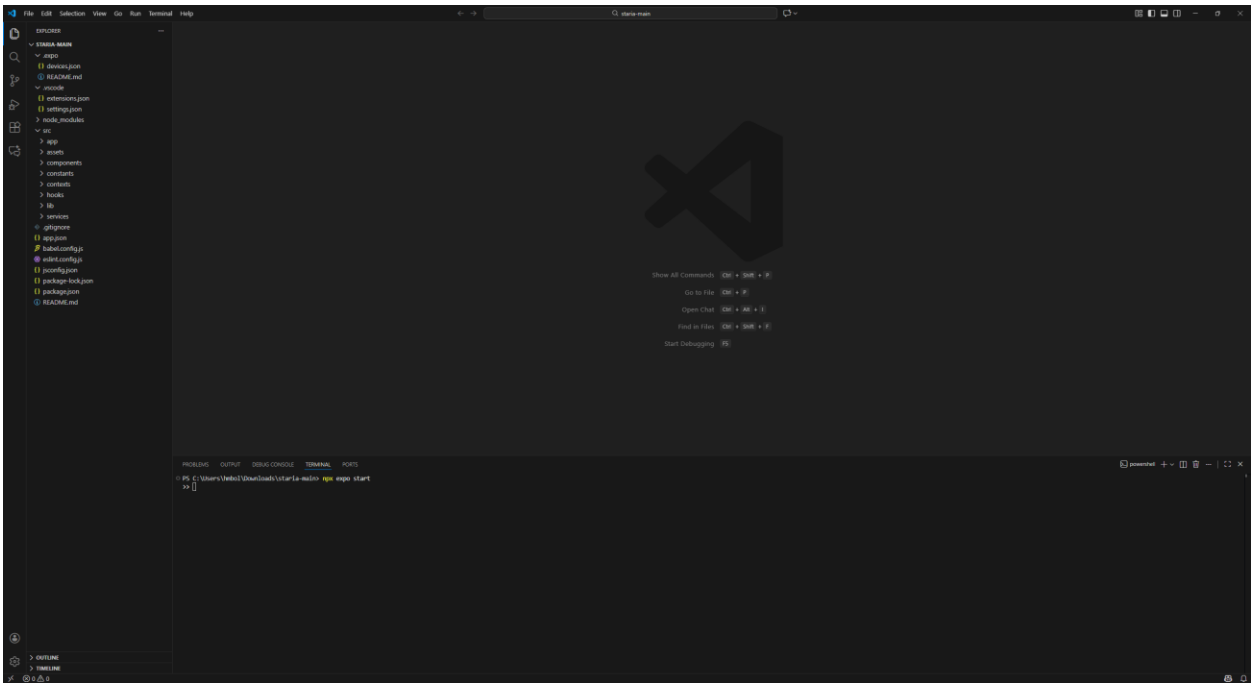


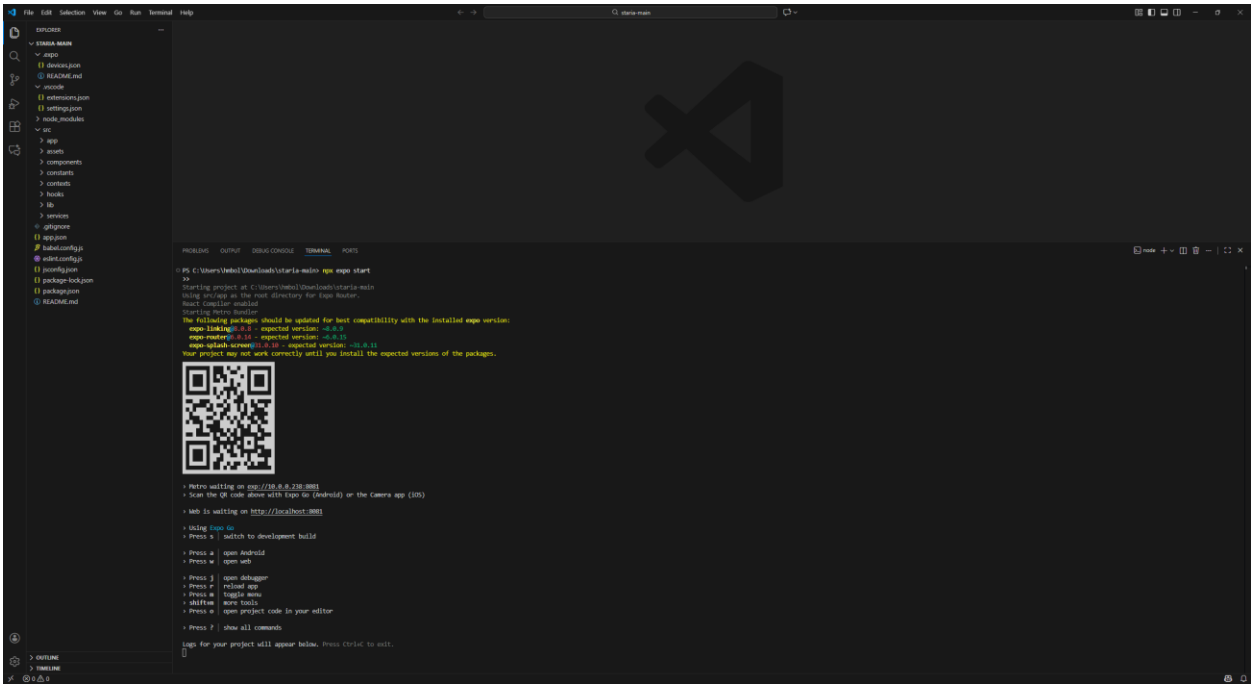
6. Open the Terminal in VSCode and Run the Command “npm install expo”



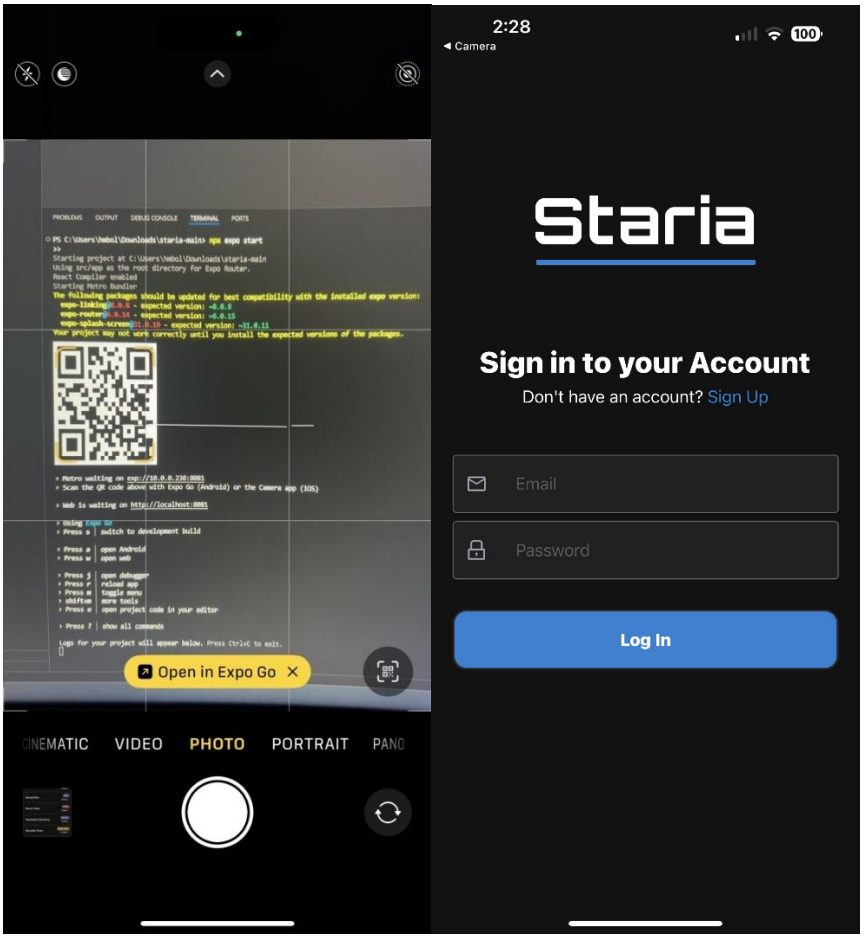


7. After the Installation, Run the Command “npx expo start”





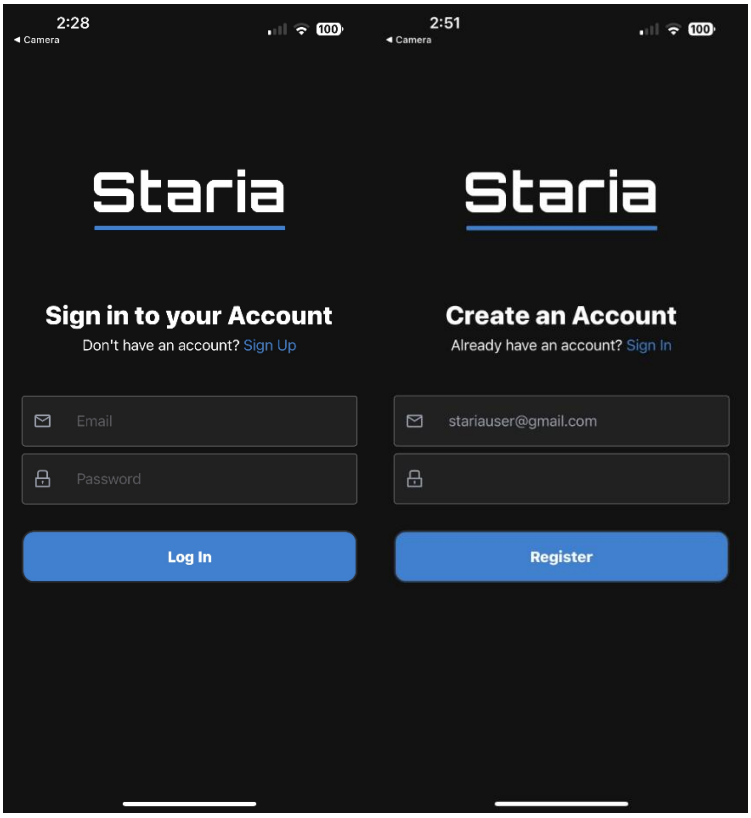
8. Open the Camera Application on Your Mobile Device and Scan the QR Code



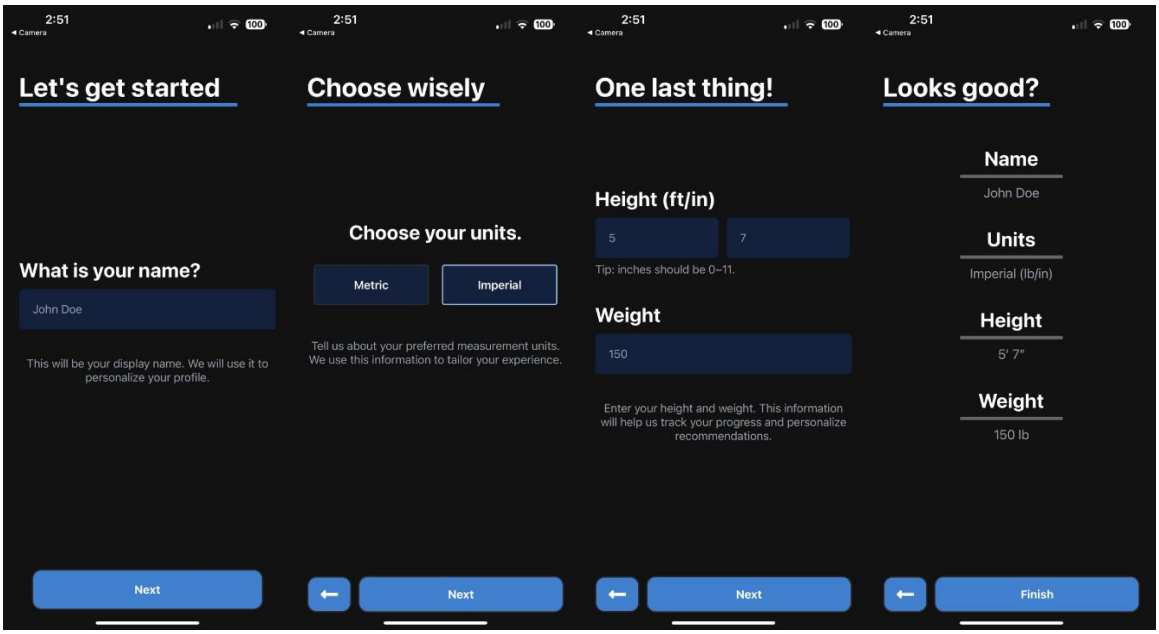
9. You are Now Able to Create an Account and Enjoy Full Access to Staria

B. User Manual

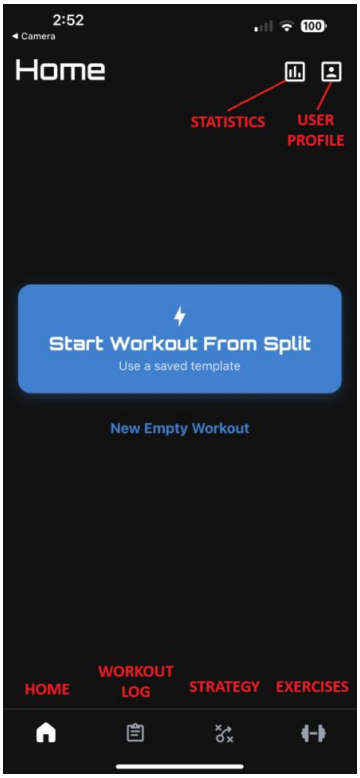
1. Create Your Account



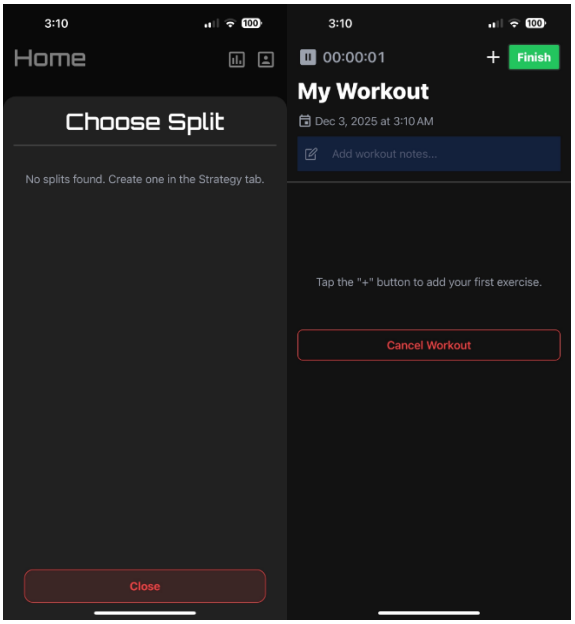
2. Select the Information That Corresponds to You



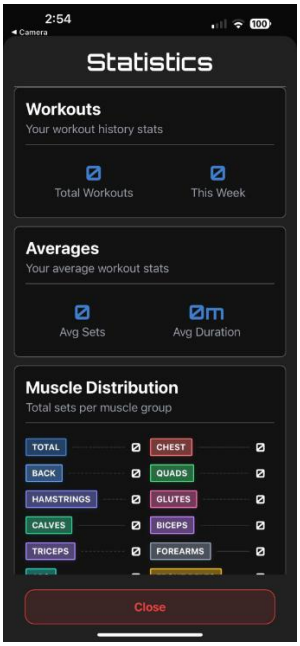
3. You are Now Sent to The Home Page, Where You Have an Array of Options



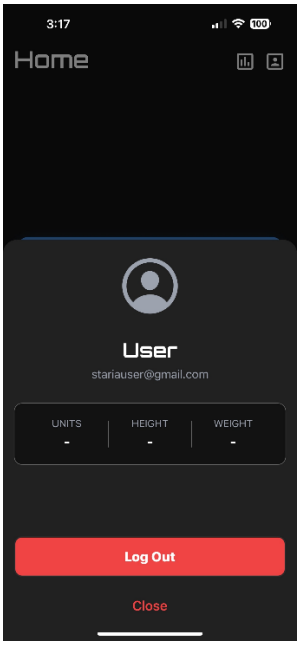
- You can now start a workout from a split you have previously created and saved, or you can start a brand-new workout.



- In the statistics tab, you can see information about your workout history and other pieces of data, such as averages and muscle distributions.

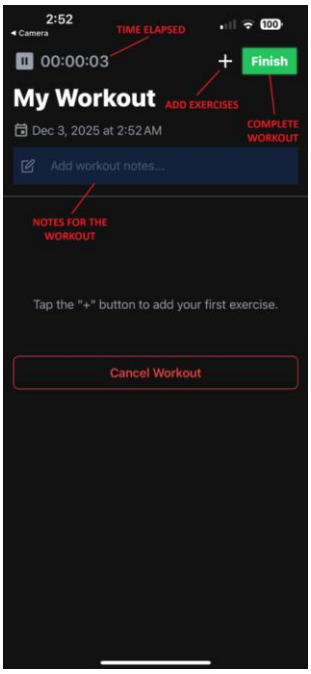


- In the user profile tab, you can see your profile, along with the information you provided while creating your account. You are also able to log out from this tab.

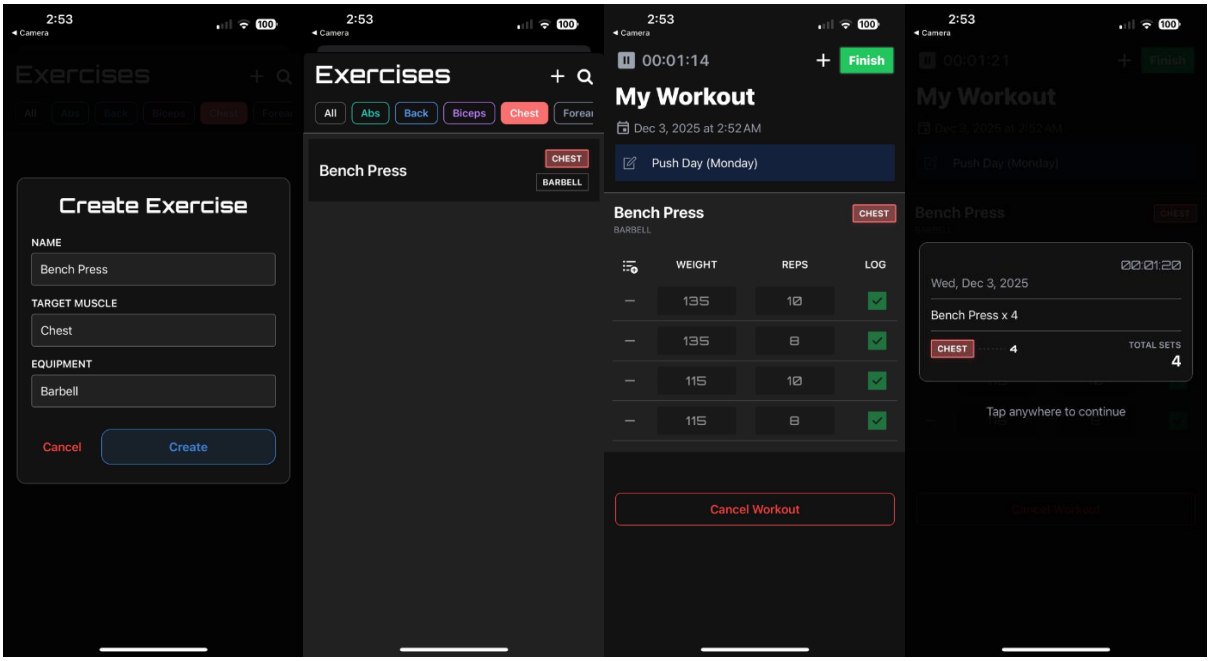


4. When Starting a New Workout from the Home Page, You Will See the Following:

- You can see how much time has passed since you started your workout, you can cancel or complete it, you are able to add notes of any kind, and you can add the exercises you are performing during the entire session.

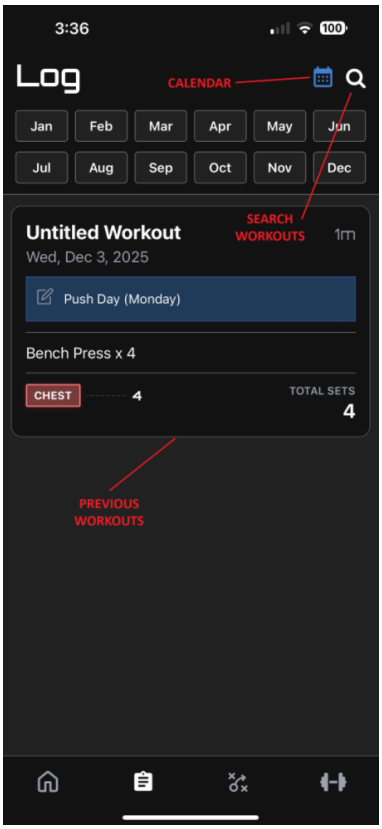


- When you Click on the Add Exercises Tab, you will be able to choose the name of the exercise, the muscle group it targets, and the equipment you used. You are then able to select how many sets and repetitions you did, as well as the weight used.



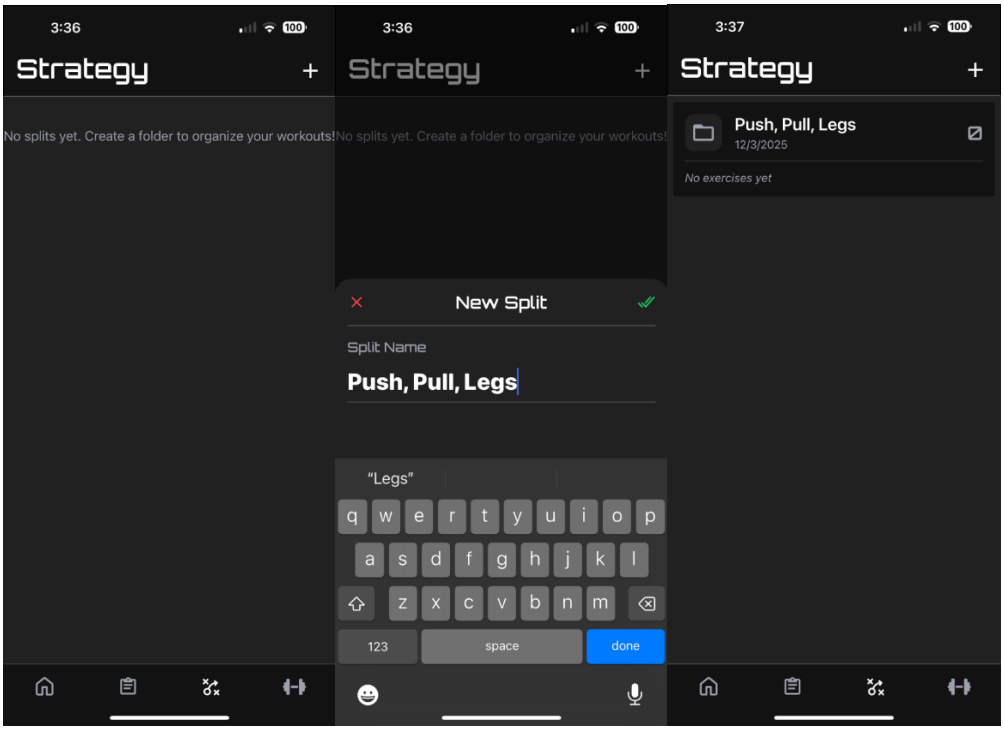
5. In the Workout Log Tab, you Will Have Access to More Options:

- You can see all your past workouts, search for a specific one, and use the calendar to sort them based on date.



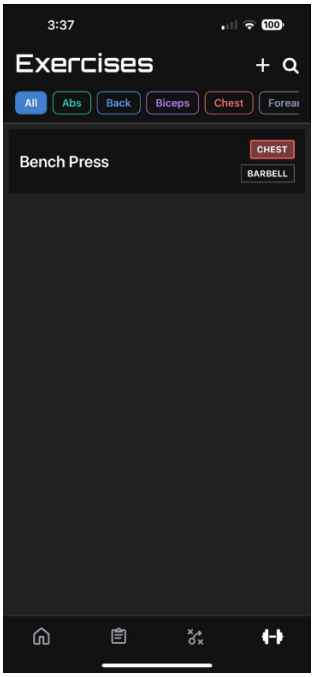
6. In the Strategy Tab, you can Create Workout Splits Using the + Symbol

- Inside these splits, you can select how many days you want to have a workout, the name of each workout, all the exercises you will be doing, and even how many sets, repetitions, and weight you will be doing for each exercise.



7. In the Exercises Tab, you can Add Any Exercise You Plan on Performing

- In this tab, you can create any exercise you wish, and then you can use it on your splits and workouts in a more seamless manner.



8. You Can Now Use All of the Tools Available in Staria to Design and Track Your Workouts in a Much Easier and Smoother Way.

C. Admin/Operations Guide

1. **Admin Access**

- Login to Appwrite console to manage users, collections, and logs

2. **Daily Tasks**

- Check server and database health
- Review cloud function execution
- Monitor new users

3. **Backup & Restore**

- Export collections (Users, Workouts, Exercises, Sessions) to JSON
- Restore collections by importing JSON

4. **Troubleshooting**

- App not syncing → check network or restart server
- Authentication issues → verify API keys
- Data missing → restore from backup

D. API Reference

1. Authentication

- POST /auth/register – Register user
- POST /auth/login – Login user

2. Workouts

- GET /workouts – List workouts
- POST /workouts – Create workout
- PATCH /workouts/{id} – Update workout
- DELETE /workouts/{id} – Delete workout

3. Exercises

- GET /exercises – List exercises
- POST /exercises – Add exercise

4. Workout Sessions

- POST /sessions – Start session
- PATCH /sessions/{id} – Update session
- PATCH /sessions/{id}/end – End session

5. Example Request/Response


```
{
  "id": "12345",
  "name": "Morning Routine",
  "exercises": ["bench press", "barbell row"],
  "status": "completed"
}
```


E. Poster, Slides, Videos

- Poster: Poster Link

- Slides: Slides Link

- Intro Video: Intro Video Link

- Demo Video: Demo Video Link

F. Scrum Evidence Index

- Daily Scrum Meeting Minutes: Daily Scrum Meeting Minutes

- Sprint Planning Meeting Minutes: Sprint Planning Meeting Minutes

- Sprint Backlog Grooming Meeting Minutes: Sprint Backlog Grooming Meeting Minutes

- Sprint Retrospective Meeting Minutes: Sprint Retrospective Meeting Minutes

- Sprint Review Meeting Minutes: Sprint Review Meeting Minutes