Achieve360

Team 16:

- Anish Bhowmik
- Srinjoy Dutta
- Ishaan Pathania
- Mehrzad Gandhi

Index

PURPOSE

There are numerous elements that go into a journey in fitness such as exercise regimens, diet, unique goals, genetic factors, etc. Oftentimes, it is difficult for people to formulate all of these factors into a working plan that can help them achieve their goals. Our project aims to provide adaptable software that can compile each of these elements for any given user and create a feasible workout plan that can help users achieve their specific fitness goals.

Improving personal health is a lifestyle that can be adopted by anyone. However, different people are going to have different health goals, and will therefore require their unique workout and diet plans suited to their aspirations. Therefore, our project will be able to specialize a fitness plan for young athletes who are looking to perfect their craft or go to professional leagues.

When a person begins their journey into fitness, it may be hard to keep track of their progress and stay disciplined. Our website aims to help set goals and keep users disciplined in order to achieve them. Contrary to other products in the market such as Strongr Fastr and Pumpd, our website aims to have a personalized characteristic to it by analyzing user statistics over time to create improved plans for the user. Along with that, we aim to allow users and coaches to communicate with each other to help users achieve their goals with professional help.

Functional Requirements

Client Account Creation/Management

1. As a client, I would like to be able to register for a Achieve360 account
2. As a client, I would like to be able to log in to my pre existing account
3. As a client, I would like to be able to keep track of my workouts

Initial Plan Generation

4. As a client, I would like to be able to input my current metrics (age, weight, height, gender, etc) and goals

5. As a client, I would like to be able to view information about my sport and position to help me decide what my goals should be
6. As a client, I would like to get a workout plan based on my current metrics and goals
7. As a client, I would like to get a diet plan based on my current metrics and goals

Plan Personalization

8. As a client, I would like to customize my workout plan to my personal goals
9. As a client, I would like to customize my diet plan to my personal goals
10. As a client, I would like my workout plans to change dynamically based on current progress (if time allows)
11. As a client, I would like my diet plans to change dynamically based on current progress (if time allows)
12. As a client, I would like my workout/diet plans to be optimized using previous user’s data (if time allows)

Tracking Current Plan

13. As a client, I would like to keep track of my workout progress
14. As a client, I would like to keep track of my diet progress
15. As a client, I would like visualizations of my workout progress
16. As a client, I would like visualizations of my diet progress

Client-Client/Client-Coach Communication

17. As a client, I would like to be able to upload images/videos for coaches to view and provide feedback on
18. As a client, I would like to be able to communicate with other clients with text
19. As a client, I would like to be able to communicate with other clients with videos and images
20. As a client, I would like informational videos and instructions on specific exercises
21. As a client, I would like to be able to compare my metrics and skills to various professional athletes in my sport and position (if time allows)

Coach Account Creation/Management

22. As a coach, I would like to be able to register for a Achieve360 coach account
23. As a coach, I would like to be able to log in to my pre existing coach account

Coach-Client Responsibilities

24. As a coach, I would like to be able to communicate with clients through text
25. As a coach, I would like to be able to view client’s images and videos to critique and help them
26. As a coach, I would like to be able to send images and videos to my clients to show them examples of what their videos should look like
27. As a coach, I would like to keep track of a client’s workout progress
28. As a coach, I would like to keep track of a client’s diet progress
29. As a coach, I would like to customize a client’s workout plan

30. As a coach, I would like to customize a client's diet plan
31. As a coach, I would like to make changes to a client's diet plan at my discretion.
32. As a coach, I would like to make changes to a client's workout plan at my discretion.
33. As a coach, I would like visualizations of a client's workout progress
34. As a coach, I would like visualizations of a client's diet progress

Coach Progress Check Feature

35. As a coach, I would like to hold regular progress checks so that I can ensure that the client's diet/workout plan is effective.
36. As a coach, I would like to hold regular progress checks so that I can ensure that the client's skills are improving.

Non-Functional Requirements

Performance

Our performance requirements are categorized into two groups: the site performance and data handling. In regards to the website performance, our requirements are as follows: the website should be available 24 hours a day, be able to handle 10,000 simultaneous requests, and be able to handle 1,000 different accounts. The rest of our performance requirements deal with data handling: messages that are sent should be received and visible within 500 ms, videos of at most 5 minutes can be supported, up to 10 GB per video can be stored, new data should be visible to users with 10 seconds of its release, and after a user inputs metrics and target data, a plan should be generated within 10 seconds.

Security

The website will authenticate user's credentials using OAuth 2.0. To secure user passwords we will implement strong password hashing using bcrypt and we will enforce password policies such as minimum length and including numbers. To prevent SQL injections we will include input validations and prepared statements with parameterized queries to make sure that the application code doesn't run user inputs directly.

User Interface

The user interface should be easy to navigate and use for someone using the software for the first time. Coaches and clients will have similar interfaces but with some small differences that are specific to each of them. We will make sure that the look and feel of each of the pages look and feel consistent. Since this is a website, it is important that it look right regardless of browser and version. We will regularly test the UI across different browsers and versions to ensure consistent performance.

Accessibility

It is important that our software is accessible to everyone that is trying to reach their fitness goals. We will make sure that our website will meet WCAG standards to make it accessible to users with disabilities. There will be support for both mouse and keyboard navigation to maintain functionality regardless of input.

Developer

As we are adding more features and updating old ones, it is important that our code is structured in a way that will make it efficient to do so. We will write clean and modular code to make it easy to maintain and update. We will also provide extensive documentation for developers to facilitate future updates and maintenance.

Design Outline

Our product is a system that will provide its users with a personalized health plan suited to their specific fitness goals. This output plan will consist of a workout schedule, a customized diet, and any other necessary details to keep users disciplined. Our implementation will take on a server-client format. We will also incorporate different types of clients (general people achieving fitness goals, young athletes looking to perfect their game to a professional level, coaches to help the athlete clients). There will be two different areas to store our data. Standard user data, such as biological information, target fitness goals, diet macros, etc., can be stored in a MySQL database. Furthermore, the backend algorithm will calculate the personalized plan based on the input data and write it to the database for the server and coach clients to access. However, since the athlete clients are able to upload videos for coach clients to give feedback on and coaches can give instructional videos back to athlete clients, there will be a cloud container (Docker in AWS) that stores the video data. The coach and athlete clients will have access to this data.

Client

- **General User Interface:** Provides an interactive platform for users to input their fitness goals, track progress, and receive personalized health plans.
- **Athlete Interface:** Allows athletes to upload videos showcasing their performance, receive a customized workout and diet plans, and access feedback from coaches.
- **Coach Interface:** Enables coaches to review athlete submissions, provide feedback, and upload instructional videos.
- **Communication Module:** Handles http requests to the server for data submission, updates, and retrieval, ensuring a responsive user experience.

Server

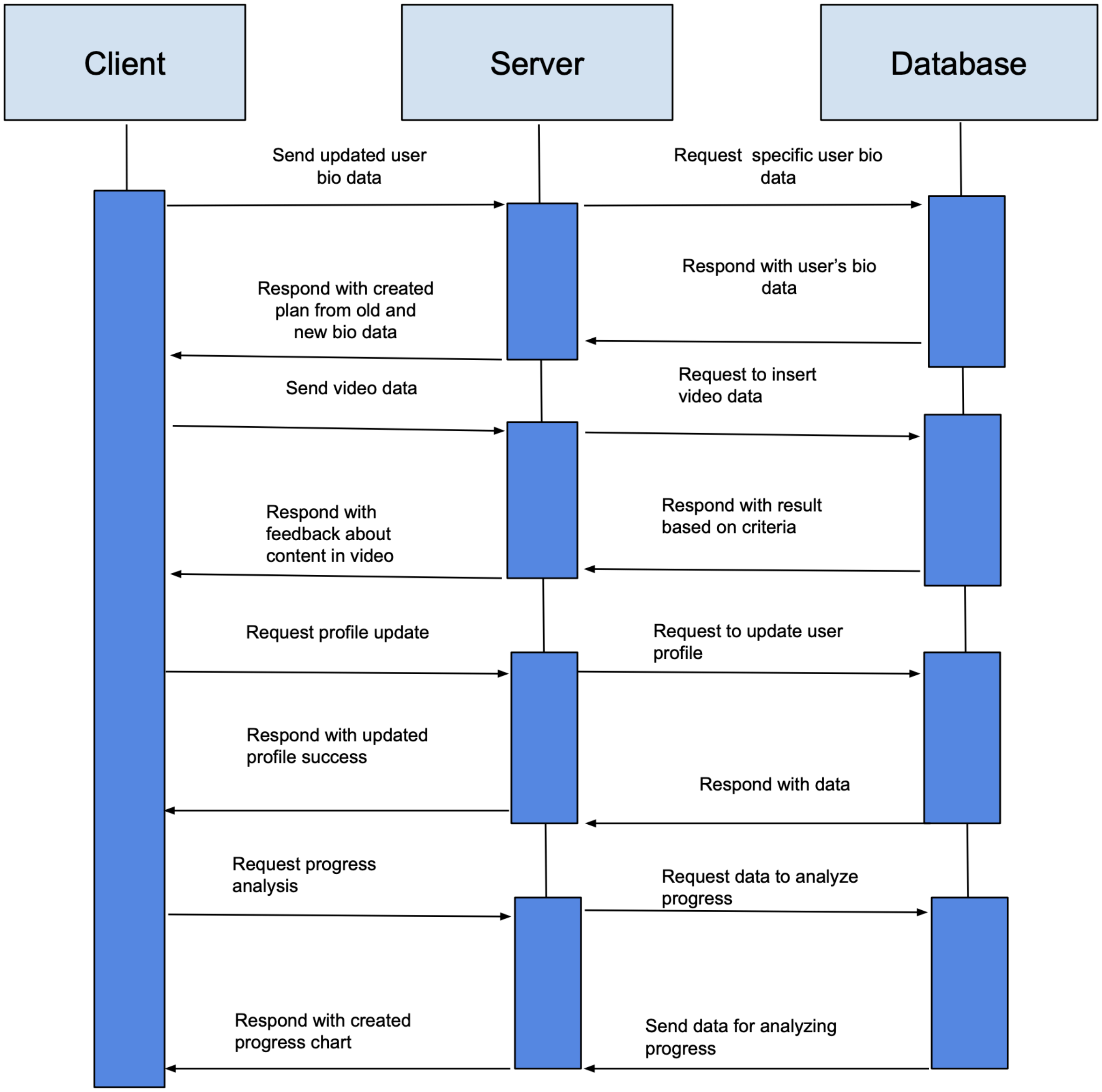
- **Request Handler:** Manages incoming http requests from clients, including data submissions, updates, and requests for information.
- **Authentication and Authorization Module:** Ensures secure access to the system, verifying user identities and granting permissions based on roles (general users, athletes, coaches).

- **Data Processing Engine:** Analyzes user inputs to generate personalized health plans, incorporating algorithms for diet and workout schedules tailored to individual goals.
- **Multimedia Manager:** Coordinates the storage and retrieval of video content between the clients and cloud storage, optimizing for performance and accessibility.

Database

- **Relational Database (MySQL):** Stores standard user data, including biological information, fitness goals, and diet plans, ensuring data integrity and efficient querying. Also stores the personalized health plans of which the user is receiving and can access
- **Cloud Storage (Docker in AWS):** Dedicated to hosting video content uploaded by athletes and instructional videos from coaches, providing scalable storage solutions and facilitating easy access for authorized users.

UML Diagram


Client
Server
Database
Send updated user bio data
Request specific user bio data
Respond with user's bio data
Respond with created plan from old and new bio data
Send video data
Request to insert video data
Respond with feedback about content in video
Respond with result based on criteria
Request profile update
Request to update user profile
Respond with updated profile success
Respond with data
Request progress analysis
Request data to analyze progress
Respond with created progress chart
Send data for analyzing progress

Design Issues

Functional Issues

What information should the user provide when signing up for a new account?

- Option 1: Username, password, account type
- Option 2: Username, password, email, account type
- Option 3: Username, password, phone number, account type
- Option 4: Username, password, phone number, email, account type
- Option 5: Email, password, account type
- Option 6: Email, password, phone number, account type

Choice: Option 1: Username, password

Justification: A password is always required for securing an account. We will also always require what type of account is being created when signing up for a new account, whether it is a Client or a Coach account. The phone number wouldn't be necessary for us as we don't plan on adding any two-factor authentication with a phone number. Same with email, as we don't plan on adding any functionality with the user's email. The best choice would be option 1 where only a username, password, and account type are needed.

How would workout plans be generated?

- Option 1: Algorithmic generation based on the user's goals and biodata
- Option 2: Integration with existing workout databases and libraries

Solution: Option 1: Algorithmic generation based on the user's goals

Justification: This option allows for workout plans to be personalized to each user's needs and goals, ensuring maximum effectiveness. Existing workout plans aren't designed with a specific person in mind, making them less effective.

How would meal plans be generated?

- Option 1: Integration with nutrition databases to suggest meals based on dietary requirements
- Option 2: Algorithmic generation based on user's goals and biodata

Solution: Option 2: Algorithmic generation based on the user's goals and biodata

Justification: An algorithm to generate meal plans based on the user's fitness goals and their data would be able to fine-tune the meal plan to their needs, making them more effective rather than following a pre-existing meal plan.

What fitness goals would the user be able to select?

- Option 1: Weight gain, weight loss, maintaining weight
- Option 2: Strength gain, cardiovascular, mobility, sport-specific
- Option 3: Weight gain, weight loss, maintaining weight, Strength gain, cardiovascular, mobility, sport-specific

Choice: Option 3: Weight gain, weight loss, maintaining weight, Strength gain, cardiovascular, mobility, sport-specific

Justification: Only looking at weight when generating workouts wouldn't be able to create workouts according to the user's taste. Same with only looking at what kind of training the user wants to do. The user being able to select both what type of training and their weight goals would allow a more personalized workout and meal plan to be generated.

Non-Functional Issues

1. What web service should we use?

- Option 1: AWS
- Option 2: AZURE
- Option 3: GOOGLE CLOUD
- Option 4: ORACLE CLOUD

Choice: Option 1: AWS

Justification: AWS's cloud infrastructure can easily scale to accommodate our project's growing video storage needs. AWS also provides a suite of media services for processing and delivering video content, such as Amazon Elastic Transcoder and AWS Elemental MediaConvert which we can use. Along with the security features to protect the video content such as encryption and detailed access logs, these benefits are why we chose AWS.

2. What frontend language/framework should we use?

- Option 1: HTML + JavaScript
- Option 2: React.js
- Option 3: Angular.js

Choice: Option 2: React

Justification: React's component-based architecture would be highly beneficial as it allows for the modular development of user interfaces such as the client and coach dashboards. It enhances the user experience by ensuring smooth interactions, especially important for dynamic content like workout progress visualizations and video feedback. The widespread community support and abundance of resources make React a practical choice for rapid development and ease of learning for new team members.

3. What database should we use?

- Option 1: MySQL
- Option 2: MongoDB

Choice: Option 1: MySQL

Justification: MySQL is one of the most reliable database management

system It is also free and open source and comes with many guides/tutorials which can help in our implementation.

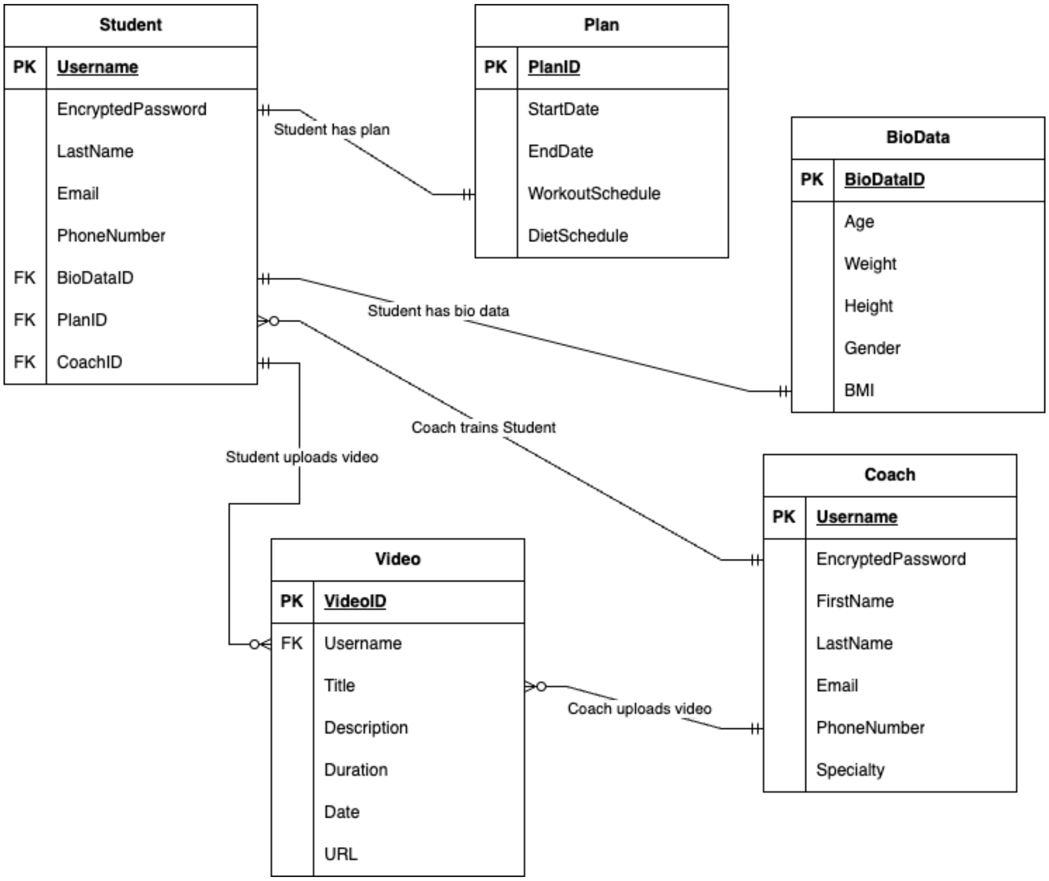
4. Which backend framework/language should we use?

- Option 1: Spring Boot (Java)
- Option 2: PHP
- Option 3: Node.js (JavaScript)
- Option 4: Python

Choice: Option 1: Spring Boot (Java)

Justification: Considering the possibility that our product can scale up to a seriously complex level, we decided that Spring Boot would be the best because Java's Just-In-Time (JIT) compilation and extensive use of threads allow Spring Boot applications to handle high traffic loads at a more efficient rate. Furthermore, our group has the most combined programming experience in Java, so as a collective, we would best be able to contribute equality with this language.

DESIGN DETAILS



ERD

Student Class
username: String
studentId: Integer
EncryptedPassword: Integer
firstName: String
lastName: String
email: String
phoneNumber: String
BioDataID: Integer
PlanID: Integer
CoachID: Integer
Functions: - set<attribute>(), get<attribute>(), uploadVideo()

Plan Class
PlanID: String
StartDate: Integer
EndDate: Integer
WorkoutSchedule: String
DietSchedule: String
Functions: - set<attribute>(), get<attribute>(), generatePlan()

Video Class
VideoID: Integer
Username: String
Title: Integer
Description: String
Duration: String

Date: String
URL: String
Functions: - set<attribute>(), get<attribute>(), uploadVideo(), deleteVideo()

BioData
BioDataID: Integer
Age: Integer
Weight: Integer
Height: Integer
Gender: String
BMI: Integer
Functions: - set<attribute>(), get<attribute>(), sendInfo()

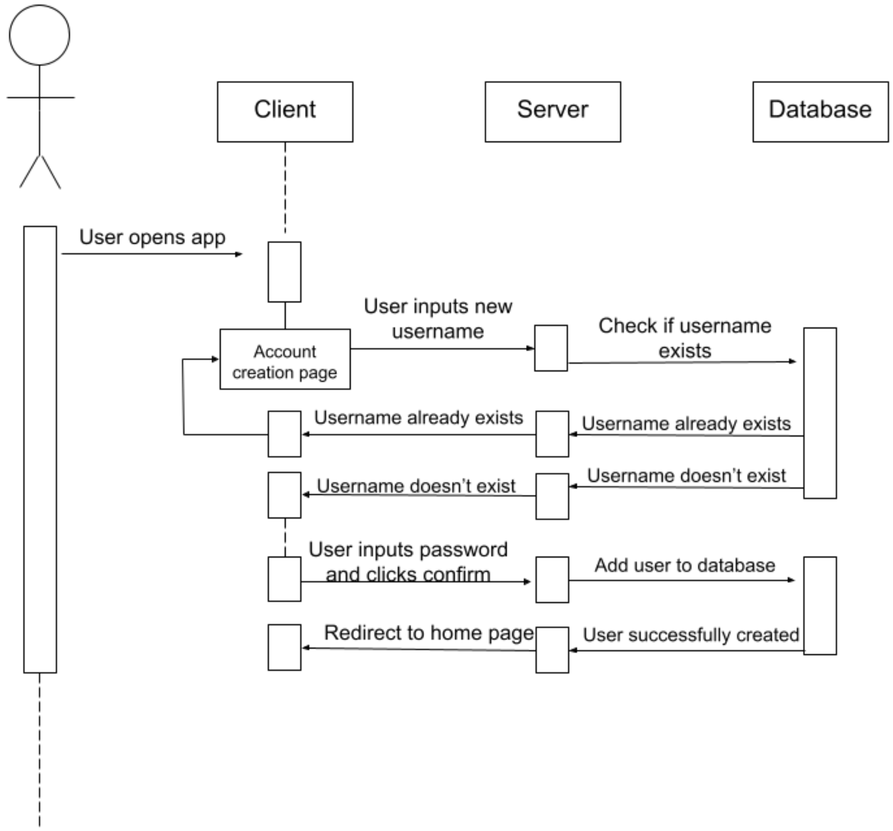
Coach Class
Username: String

EncryptedPassword: String
FirstName: Integer
LastName: String
Email : String PhoneNumber: Integer
Speciality: String
Functions: - set<attribute>(), get<attribute>(), uploadVideo(), requestBioData(), deleteVideo()

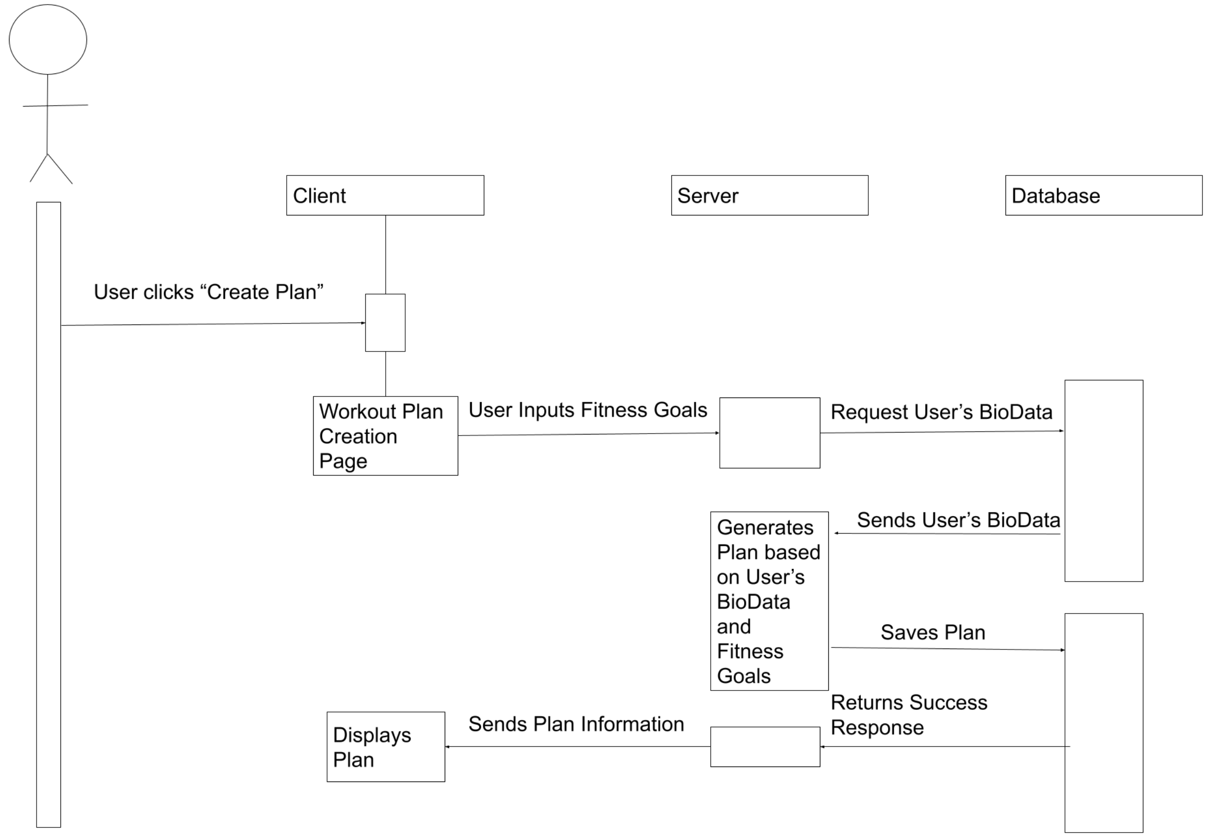
- Student Class:
    - Attributes: The student class includes information about the student's account, personal information, and relationships with other entities such as bio data, fitness plans, and the assigned coach.
    - Functions: Methods to set and get attribute values, and a function to upload videos (perhaps to a personal profile or as part of a fitness challenge).
    - Interactions:
        - With BioData: A Student has a BioDataID, indicating a one-to-one association. This means each Student has a single set of bio data associated with them.
        - With Plan: A Student has a PlanID, suggesting a one-to-one association with a fitness plan.
        - With Coach: A Student has a CoachID, indicating a one-to-one association, meaning each student is assigned to a single coach.
- Plan Class:
    - Attributes: The Plan class contains details about a fitness plan, such as the start and end dates, workout schedule, and diet schedule.
    - Functions: Methods to set and get attribute values, and a function to generate a new fitness plan, on student bio data, and goals.
- Video Class:

  - Attributes: The Video class has a video with a unique identifier, associated username, title, description, duration, upload date, and URL.
  - Functions: Methods to set and get attribute values, upload videos (which could be used by Students or Coaches), and delete videos.
  - Interactions:
    - With Student: Videos are linked to Students or Coaches, depending on who uploads the video.
- BioData Class:
  - Attributes: The BioData class stores bio data for a student, including age, weight, height, gender, and BMI.
  - Functions: Methods to set and get attribute values, and a function to send this information to a Coach or a system that generates fitness plans.
  - Interactions:
    - With Student: Each BioData entry is linked to a specific Student through the BioDataID.
- Coach Class:
  - Attributes: The Coach class contains details about a coach, including their encrypted password, personal information, and specialty.
  - Functions: Methods to set and get attribute values, upload videos, request bio data, and delete videos.
  - Interactions:
    - With Student: Coaches are linked to Students via a CoachID, so they manage the fitness plans and oversee the progress of their assigned students.
- General Interactions:
  - Student and Plan: Students have a PlanID attribute that links them to a specific fitness plan. The generatePlan() function in the Plan class uses student bio data to create a personalized fitness plan.
  - Student and Video: Students can upload videos, which associates a video with the student who uploaded it. This is used for progress tracking, demonstrating exercises, or for educational purposes.
  - Coach and Student: Coaches can request bio data from students, to monitor their progress or adjust their fitness plans. They may also upload videos, which students could use as part of their fitness plans.
  - Coach and Video: Coaches have the ability to upload and delete videos, which will be used for providing instructions, feedback, or removing outdated content.

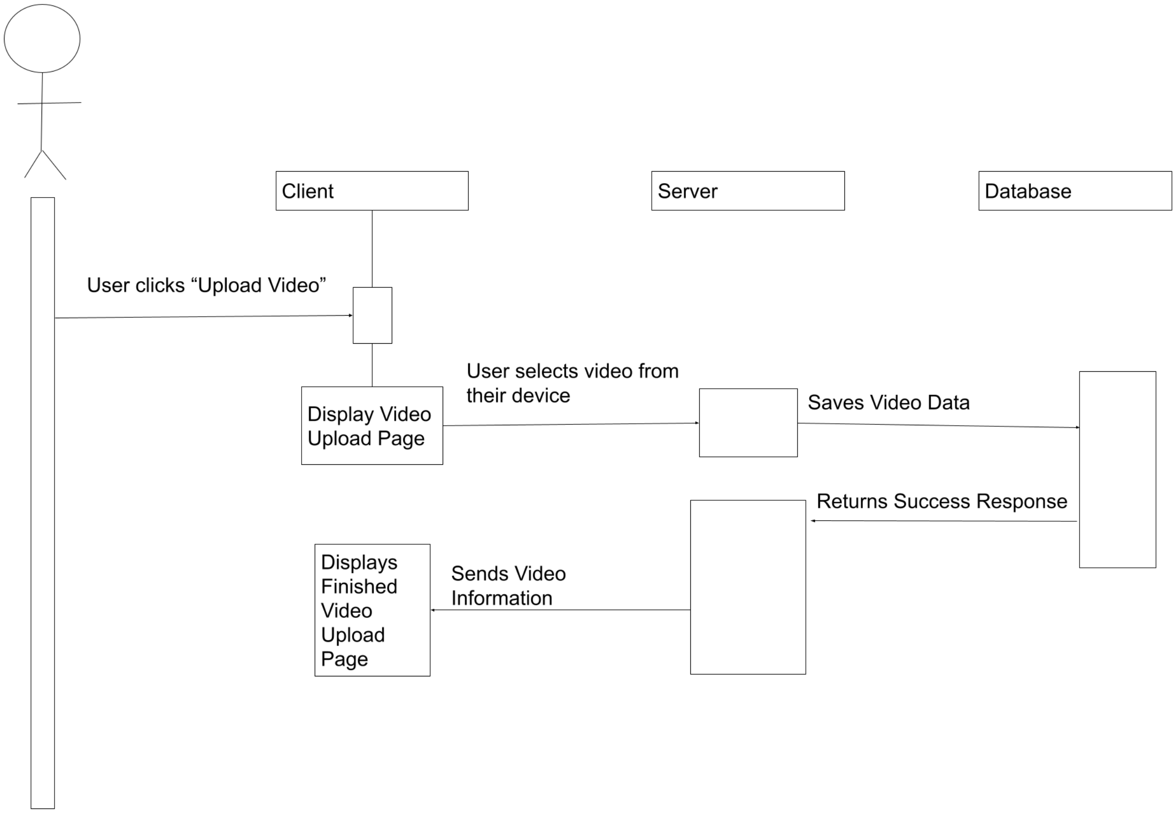
Sequence Diagram when User Creates Account:



Sequence Diagram when Client Creates Plan:

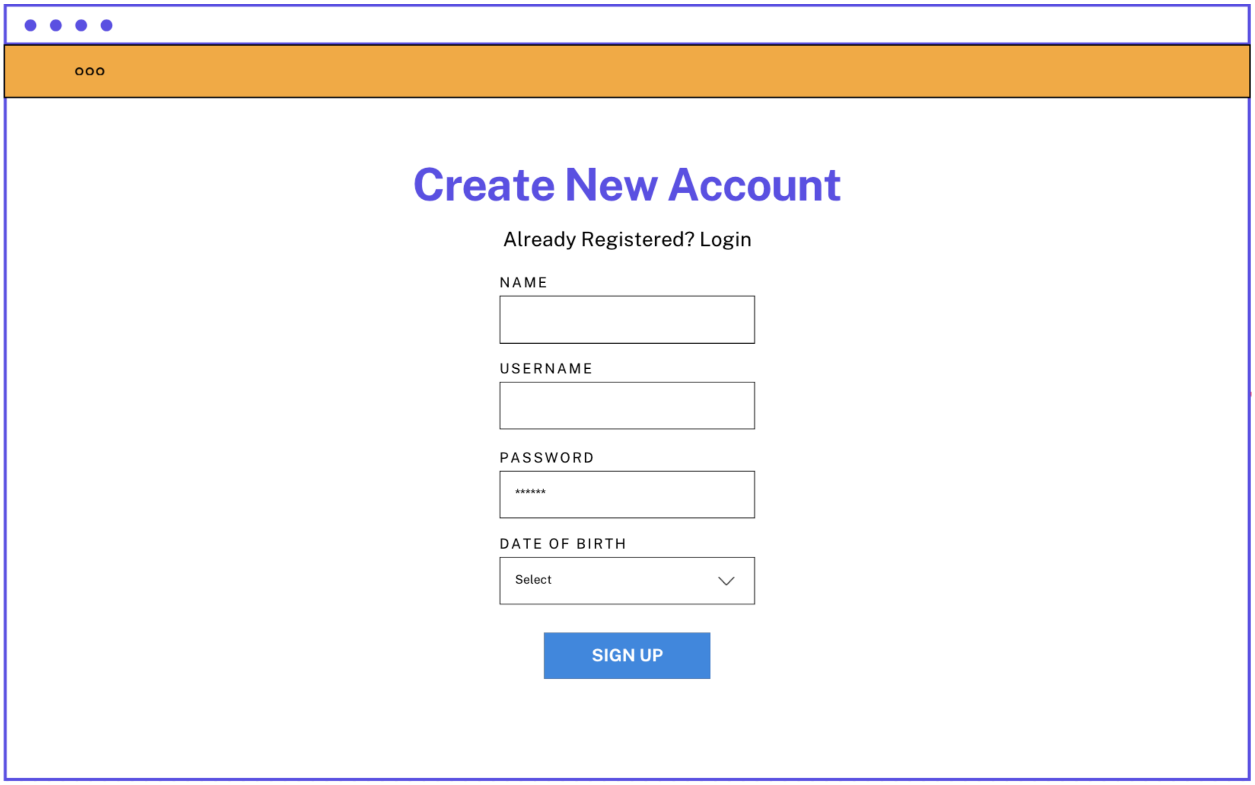

Client
Server
Database
User clicks “Create Plan”
Workout Plan Creation Page
User Inputs Fitness Goals
Request User’s BioData
Sends User’s BioData
Generates Plan based on User’s BioData and Fitness Goals
Saves Plan
Returns Success Response
Sends Plan Information
Displays Plan

Sequence Diagram When User Uploads Video


Client
Server
Database
User clicks "Upload Video"
User selects video from their device
Display Video Upload Page
Saves Video Data
Returns Success Response
Displays Finished Video Upload Page
Sends Video Information

UI Mockup

We have provided a rough, high-level representation of what users will see when using our product. We have organized these basic diagrams in the order that we believe users would follow as they access the different stages of our product. The first diagram is the login/signup page, where student and coach clients can either log in to an existing account, or register for a new account.



Create New Account

Already Registered? Login

NAME

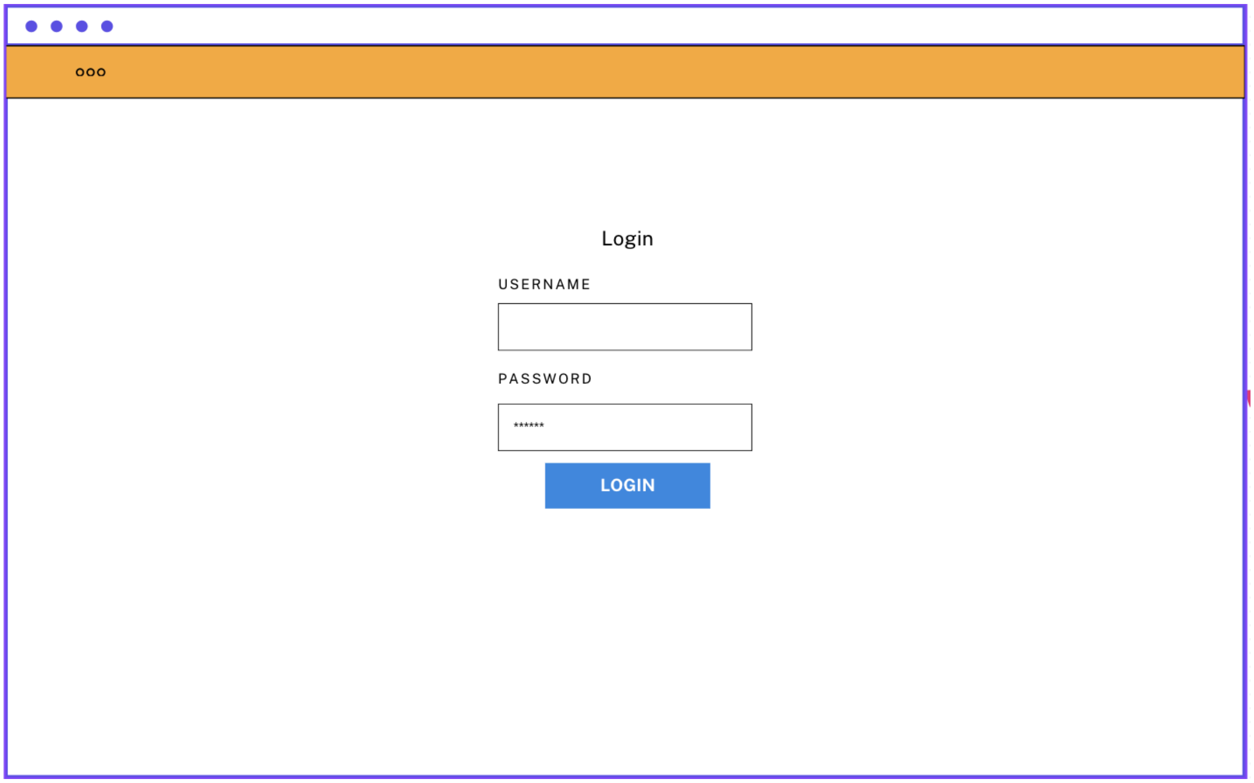
USERNAME

PASSWORD

******

DATE OF BIRTH

Select

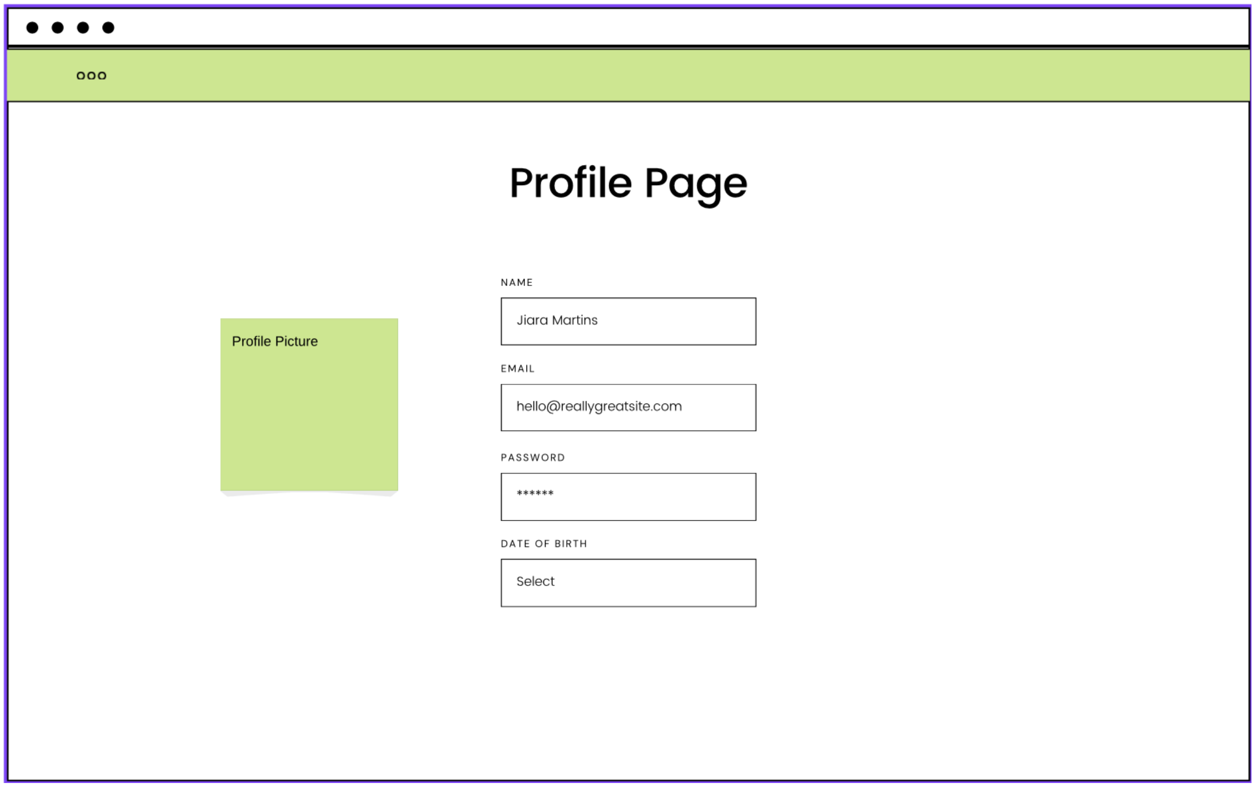


Login

USERNAME

PASSWORD

******



Profile Page

NAME

Jiara Martins

EMAIL

hello@reallygreatsite.com

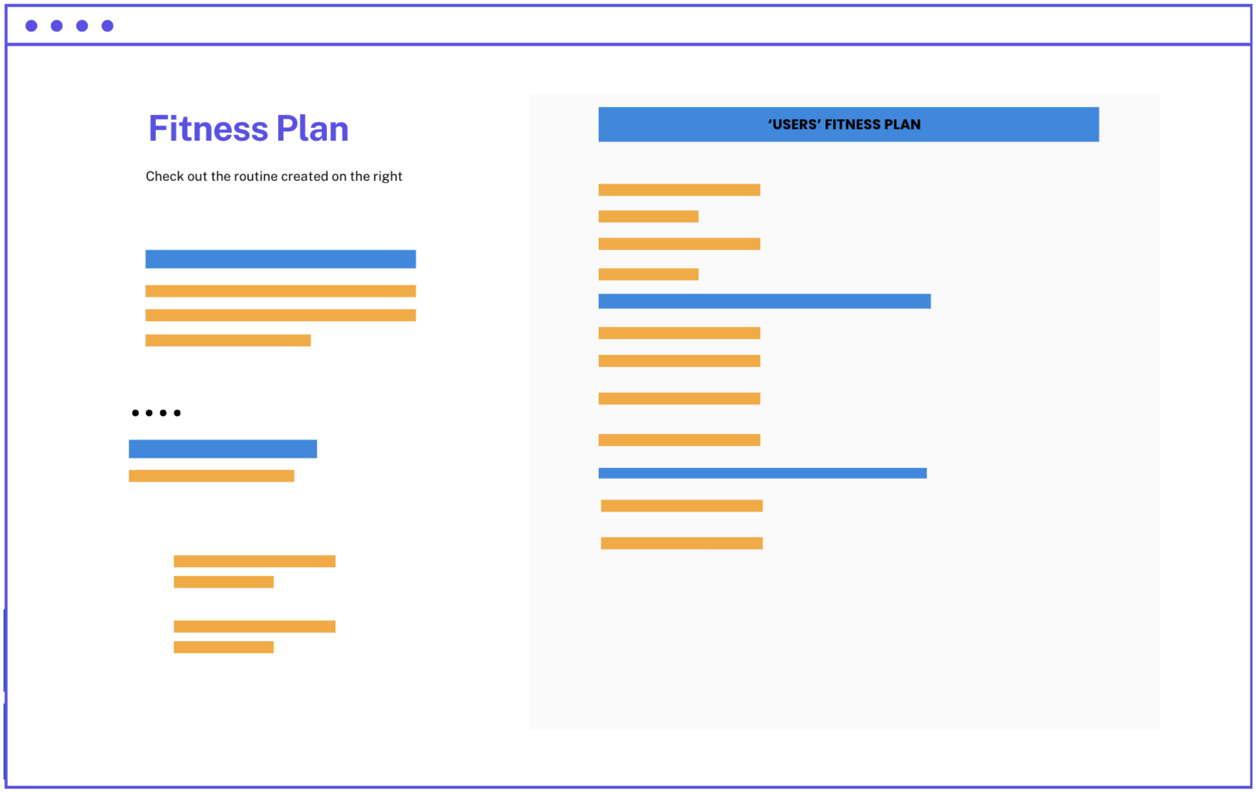
PASSWORD

******

DATE OF BIRTH

Select

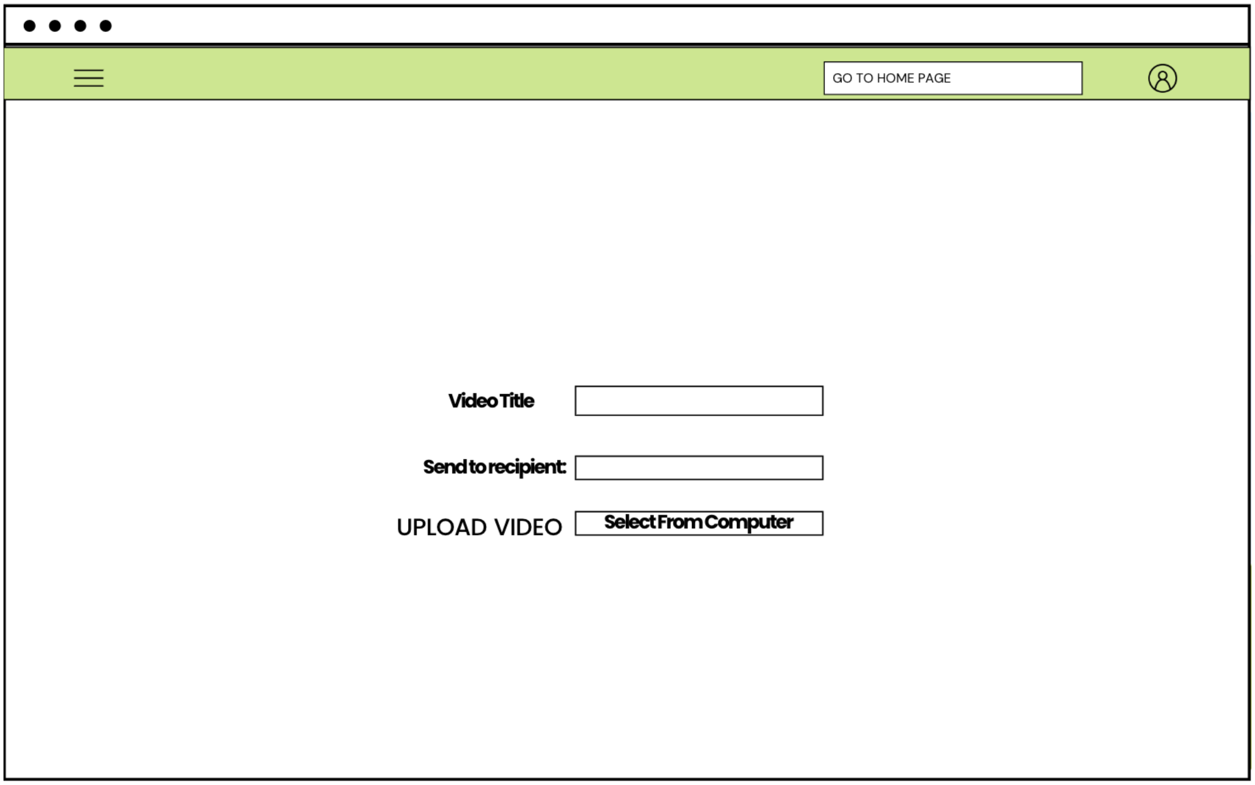
This diagram is the “dashboard” for a user once they are logged in or registered. If the user already has an active plan, it will be displayed on the right of the page, with account options on the left. If users do not have an active plan, there will be an option for users to request a plan by entering their bio data.

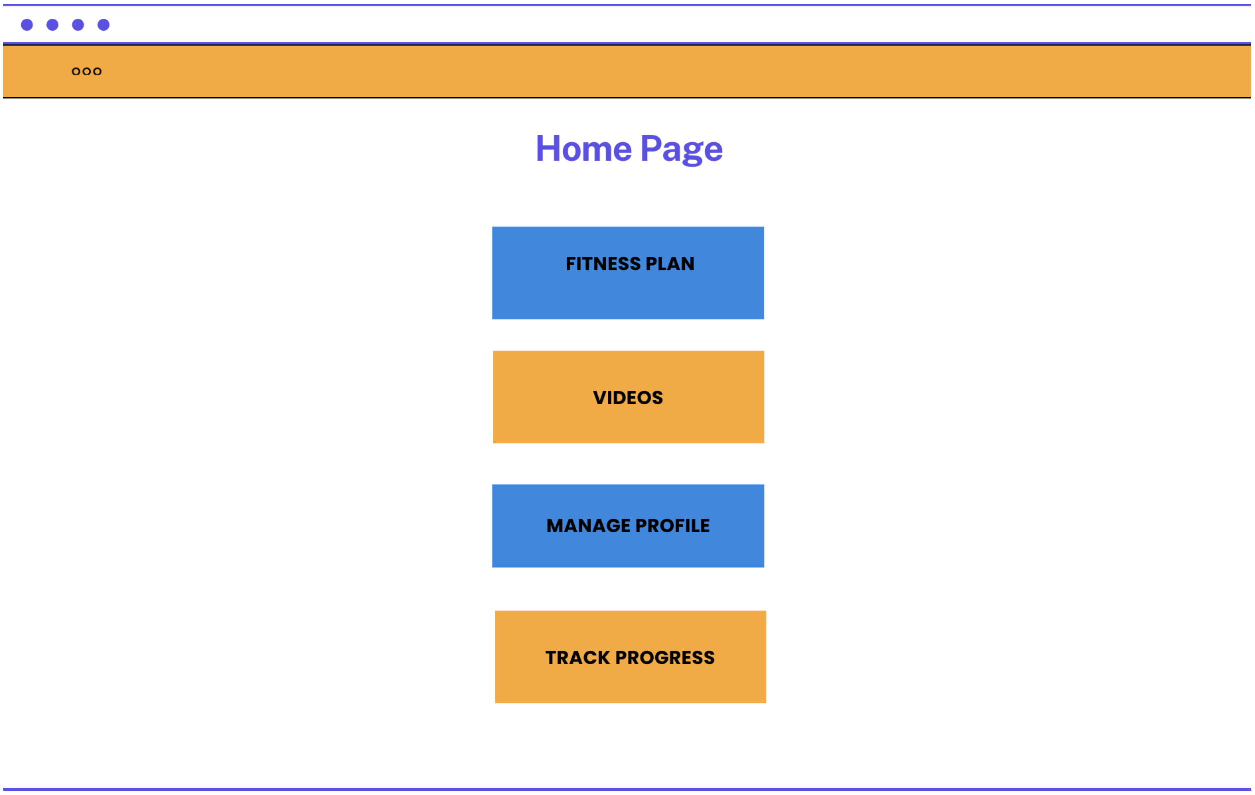


Fitness Plan

Check out the routine created on the right

‘USERS’ FITNESS PLAN


GO TO HOME PAGE
Video Title
Send to recipient
UPLOAD VIDEO
Select From Computer


ooo
Home Page
FITNESS PLAN
VIDEOS
MANAGE PROFILE
TRACK PROGRESS