**Meeting Goals:**

Defines our solution for everything in the project proposal outline

**Project Idea:**

Proposal : A workout website that creates custom workouts based upon various criteria saved in a user profile database. This is specifically for new workouts, with minor tracking, and the user provided constraints (type of workout, machine, time).

**Functionality:**

* Sign up for the site
* Enter personal details (height, weight, workout experience, health experience)
* Enter daily details (mood, energy, length of time you have to work out)
* Some historical data (workouts for past week/month)
* User entered criteria to help with constraints
* Cross platform compatibility (web app with mobile support through responsive web design)
* After workout is provided, ability to refine and choose components (or select all) if we want

**Web App:**

* Login page
* Registration page
* Profile page/Dashboard
  + Stores your personal health data, preferences, settings, etc.
* History page
  + Previous workout data (“conversation history”) or Calendar?
  + Uneditable?

* Main page “Request a workout”
  + Output for workouts is text based
  + Ability to select and edit workouts

Constraints:

**Technology Stack:**

* Gemini LLM
  + Good documentation
* DB:
  + MongoDB (noSQL) vs postgres (with SQLalchemy)
  + Probably use SQLalchemy if using react and django
* FE:
  + React
* BE:
  + Django, django REST framework or fastAPI
    - Comes with some auth stuff out of the box
    - Testing stuff as well

* Hosting: Provided servers vs docker?

**Team Roles:**

Front End Devs (3):

* Rhys, Ryan, Riley, Tom
  + Lead: Ryan

* Web Pages/Modules

* Testing:

* Documentation:
  + All

* Requirements

Back End Devs (4):

* Frank, Riley, Tobin, Jeff

* Lead: Frank

* Database:

* API

* LLM:

* Hosting/Deployment:

* Testing
  + Riley

* Documentation:
  + All

* Requirements

Management Roles

* GitHub
* Timeline/dates/correspondence

**Functional/NonFunctional Requirements:**

* Functional (10min):
  + Login to the site
  + Registration (profile setup)
  + Edit profile details
  + Select daily workout details
  + Confirm/edit workout suggestion (stretch)
  + Save history
  + View history
  + Generate workout
  + Admin panel (admin access) (stretch)
  + Add new daily parameters
  + Workout feedback
  + Delete profile
  + Images/videos of workout (stretch)

* NonFunctional(5min):
  + Responsive on computer/phone
  + Cross platform
  + Saved history duration (#records)
  + Ability to register with email or username (authentication)
  + Storing data in database (method)
  + Ability to work with injury/disability (accessibility)
  + Integration with LLM (interoperability)
  + Easy to navigate userface
  + Fast enough

**To Do:**

* High level Functional/Non Functional
  + In either user stories
  + 10 functional, 5 non functional (minimum)
* Choose team
* Learning your stack
* Starting to assemble documentation
  + Front end: user stories
    - Use Case Diagram: Tom
    - UI sketch: Rhys (look into react router)

* Back end:
  + ER Diagram: Jeff

* Overview
  + Ryan & Frank
  + Project proposal: Riley/Tobin

* Project proposal (500 words look at the project proposal documentation)

* Create github: Frank
* Choose name:
  + Discord poll

Tobin to do:

Create google calendar with deadlines

Book room for next meeting

* TFDL
* Engg building?

Project Proposal: Large Lifting Model

Project Idea:

- A workout website that creates custom workouts based upon various criteria saved in a user profile database. This is specifically for new workouts, with minor tracking, and the user provided constraints (type of workout, machine, time).

- allows for specific user input for each workout based on the way they feel

- Allows for the user to choose which components of the workout they wish to do.

- saves previous workouts

Tech Stack:

- Gemini LLM due to good documentation and cheap price

- Using the react library for the front end (Javascript)

-Using Django for the back end (Python) (Framework)

Database solution:

- Using postgreSQL with SQLalchemy

Our project will be focused on the development of a workout application that is driven by an LLM. The LLM will facilitate in increasing our applications functionality and responsiveness. It will be cross platform and be a web-based application. Our application will enable users to get custom tailored workout plans. Our application helps individuals who are looking to start working out but don’t have the knowledge and or time to come up with an appropriate workout plan. The core feature of our application is its ability to provide users with custom tailored workout plans, specifically designed to meet their individual fitness goals, physical abilities, and preferences regardless of their level of experience. Since our application allows for the selection of what kind of workout, duration, equipment, etc., it’s highly customizable nature and simple user interface makes it a perfect application for anyone trying to start working out. The application name we have selected for our project is Large Lifting Model.

The large language model that we intend to use is Google Gemini. Gemini was selected due to its impeccable documentation enabling quicker development of our application, as well as for its relatively cheap price compared to other LLM’s. The LLM will be used to generate workout plans for the user based on user specified criteria. The LLM will also have access to the user’s previous workouts, enabling it to tailor newer workouts based on what the user has already done before. For the front-end of our application, we plan on using React and the React router framework. React is a powerful JavaScript library that is primarily used to create user interfaces for web applications. Our application will feature several different pages and will work with multiple unique users. All of our page’s user interfaces will be handled through the React library. The user will have a limited view of the database used by our application through our workout history page and the user profile page. We chose this library as members of our team were familiar with its use, reducing our time needed to learn new technologies. For the back-end we will be using Django and the Django Rest framework. Django is a Python framework used to create web applications. Similarly, Django will be used in our application due to members of our team being familiar with its use.

A database will be required in our application in order to store our user’s information. For this purpose, we will be utilizing PostgreSQL as it is a powerful relational database system with ample documentation and is a software that members of our team is already familiar with. Another reason why we will be using PostgreSQL is that it is very commonly used in Django applications, reducing any risk in stability that may arise in our application. In addition to PostgreSQL we will also use SQLalchemy to simplify all operations to the database used by our application. SQLalchemy allows for complex queries and more control over the database compared to Django’s native ORM. SQLalchemy is also more efficient than the ORM provided with Django.