1. Team info

Provide a concise summary of the project team and project artifacts. Specifically:

- List each team member and their role in the project.

  Ayush Baruah, Backend Engineer / Tester

  Duncan Everson, Full-stack Engineer

  Benjamin Kono, Engineer in charge of UI design / Frontend Engineer

  Aiden McCoy, Engineer in charge of UI design / Frontend Engineer

  Daniel Molina, Backend Engineer

  Nicholas Shininger, Tester / Backend UI Engineer

- Link to each project relevant artifact such as your git repo (this can be empty for now).

  https://github.com/shiningn-osu/software-dev-ii-project

- List communication channels/tools and establish the rules for communication.

  Primary Asynchronous: Discord

  Primary Synchronous (outside of regular meetings): Teams

  Rules for Communication:

  To encourage positive/creative discussion, group members will refrain from interrupting others' ideas and from putting down any ideas during the discussion. It is expected that all group members speak respectfully of each other and allow others to voice their opinions without belittlement. To resolve disagreements the group members will seek to form a compromise between the two perspectives that led to the disagreement. In the case where compromise cannot be reached between the group members, then the group will seek third party input from a source outside of the group. The perspective that the third party selects will be the option that the group members use.

  For project management, we will use a GitHub Project Kanban board. We will manage the tasks' status between ready, in progress, and done to reflect the current status of tasks. We will also assign tasks to individuals to reflect who is responsible for the task. Additionally, we will also put a size and time estimate on tasks to better divide up the workload between us. We will also note the dates that these tasks are due so that we can better manage our time.

  If a team member gets confused about what to do, they should communicate that confusion in the Discord. If the confusion relates to something time sensitive, then they can send an email as well. Synchronous Microsoft Teams meetings may also

be scheduled if the group members deem such a meeting necessary to resolve the confusion.

If a message sender sends a discord message, and they do not hear back within 12 hours (24 hours on the weekend), then they can send an email. This message may either reiterate the points of the discord message, or direct the recipient to the discord message. If there is no response to the email within 12 hours, then follow up text messages may be sent.

If a team member falls behind in their work, then they can post a message on Discord relaying that they are behind on their work. This message may request for either guidance or assistance from the other group member. In the event where the issue is time-sensitive or there is a large amount of work to catch up, an in-person meeting will be scheduled to dedicate time toward the unfinished work.

If a group member believes that they will fall behind due to upcoming exigent circumstances, then they are expected to relay that in the group Discord. From there, the group will redistribute the work such that the person who has the exigent circumstances has a workload that they can accomplish, and the other work has been divided among the group. The group's work redistribution allotments will be such that the redistribution is agreed as reasonable by all group members. For the next workload allotment, the individual that had exigent circumstances will take on an additional workload to make up for the previous reallotment such that the new allotment is agreed as reasonable by all group members.

If a group member fails to contribute sufficiently to the project during a week, then they should do an additional portion or task for the next week's project work. The distribution of the portions between group members should be agreed upon by the group members such that they both deem it fair. In the event where a group member has continuously failed to contribute sufficiently to an assignment, then the group may discuss the issue collectively and agree on an action plan to address the lack of contribution and the method for the non-contributing group member to make up the work that they've failed to complete.

2. Product description

- Start with a short and catchy project title (*not* "CS 362 project pitch") and your full names.

  Meal Match

  Ayush Baruah, Duncan Everson, Benjamin Kono, Aiden McCoy, Daniel Molina, Nicholas Shininger

- **Abstract:** The first paragraph of your document must be an abstract (or executive summary, or TL;DR) that explains your project at a high level. If someone read nothing but that one paragraph, what do you want them to know?

  The services will allow users to be able to get meal plans, and grocery lists to match those plans. Along with these grocery lists, they will be able to see nearby stores which can sell them the required foods, be taught how to make those foods with provided recipes, and get nutrient information as to why those plans are best for them. The service will track their nutrient consumption into a long term tracker, to show their progress.

- **Goal:** What are you trying to do? For example, what task or problem will your system help users with?

  Make meal planning, prep, and long-term tracking easier. We will provide to users a single location where they can assemble their meal plans and get a list of ingredient location, price, and nutrient information for those meal plans. The meal and nutrient information will then be stored and retrievable by the user across usage sessions.

- **Current practice:** How is it done today, and what are the limits of current practice?

  Meal planning can be done through a service called Mealime, which offers custom meal planning, the ability to write a grocery list, and meal options based on dietary preferences. Mealime doesn't integrate with local stores though. Yummly provides many of the same features as Mealime, including meal recommendations and nutrient information. It does not have long term tracking or local ingredient information though. Nutrient and calorie tracking can be done through a service called MyFitnessPal, where the nutrient information that is shown is only for pre-entered meals. It does not have an ingredient list that it works off of, only previously entered nutrient information provided by other people.

- **Novelty:** What is new in your approach and why do you think it will be more successful than other approaches? Do not reinvent the wheel or reimplement something that already exists, unless your approach is different.

  The novelty of this project would be the combination of several services, rolled into one. While some of these approaches have been done before in separate applications/ services, none have been rolling them into one use for the consumer. The novelty is convenience, and an easy logical solution. Additionally, we would provide long-term meal information tracking, which Mealime, Yummly, and MyFitnessPal don't provide. We would also provide the local store interface to give users an actionable plan for gathering these ingredients, which Mealime and Yummly don't provide.

- **Effects:** Who cares? If you are successful, what difference will it make?

  People that utilize fitness tracking apps that are annoyed by the burden of using so many different apps. Also, people that want an actionable plan for gathering ingredients, and that want longer term meal plan tracking.

- **Technical approach:** Briefly describe your proposed technical approach. This may include what system architecture, technologies, and tools you may use.

  We will use Node.JS, specifically EJS templates and Express routing, to serve EJS pages to users. We will use API calls to retrieve storefront information about available products from Kroger. We will use server-based infrastructure on local files.

- **Risks:** What is the most serious challenge or risk you foresee when developing your project on time? How will you minimize or mitigate the risk? Don't state generic risks equally applicable to any project, like "we might run out of time".

  Our biggest risk is having limited access to the necessary APIs. This could include needing paid APIs, or being unable to find an API that gives the information we need correctly. A way of mitigating incorrect information would be to have a report system for the user to prompt the development team with inaccurate information given from an API.

- 4+ major features you will implement.
  - Grocery Price Check for nearby stores
  - Search for grocery
  - Meal plan creator
  - Nutrient tracking system
    - Long term tracking
  - Recipe maker
- 2+ stretch goals you hope to implement.
  - Make the price check system a framework, not solely for grocery, but also for general shopping
  - Create a workout creation system to piece together workouts given the user has input what type of workout they want to do (ex. HIIT, Calisthenic, Weight Lifting, etc.).
    - Workout tracking