CS 121 25wi Final Project Proposal

Due: Friday Feb. 14th 2024, 11:30PM

- *Note: Proposal deadline is exception to weekly Monday deadlines for sets, between A3 and A4. Try to submit by Thursday, but we will accept by Friday.* ***A4 is due on Tuesday Feb. 19th*** *due to Monday holiday but will be shorter with proposal overlap.*

Notable 25wi updates:

- app.py with function stubs is **required** in proposal this year
  - [02/10] El went through some student project ideas and will provide another version of template since we're simplifying client/admin breakdown this year
  - A summary of minimum requirements is added to this document (show_options() and starts to function stubs)
- Setup-proposal.sql (DDL) **required** in proposal
- Procedural SQL and DB Performance **both removed** from proposal but kept for reference
  - Triggers will be **optional** for the Final Project this year
- Other requirements may be reduced a bit to account for lost week during wildfires

For the final project, you may choose to work with up to one other person. For partner-finding, use the #partner-search Discord channel to share your interests, any ideas, and preferences for working with a partner. There will be optional work time in lecture to find partners and ask about any questions with El around; we encourage you to attend to aim for finalizing a project idea, a partner, and any dataset/scoping questions you have. You are welcome to remove any suggestions/notes from this proposal in your proposal, as long as your required components are clearly in the order specified (we've highlighted text for you to answer in blue for ease).

Note: If working with another student, only one of you needs to submit the required files to CodePost, but ***both of you should have equal contributions; i.e. you should ideally work through this document together.*** *For ease of grading please have the other student submit a simple* **collaboration.txt** *file to their CodePost submission in the following format (failure to do so may result in deductions):*

```
Names: <student1>, <student2>

CodePost emails: <email1>, <email2>
```

For full credit, we are looking for a robust proposal demonstrating careful consideration of the motivation, design process, and implementation plan for your application; you should expect to spend 3-4 hours minimum on your proposal (finding a dataset may take the longest). For each part of this proposal that requires your response, you should have 2-3 sentences (bullet points are encouraged as well) justifying your answers. We strongly encourage you to include as much detail as you can in this proposal (this is your only assignment this week) and you can include any diagrams and SQL brainstorming (e.g. a start to your DDL) with your submission on CodePost. If you want to add any additional planning, you can include a diagram as a PDF/PNG/etc. and/or .sql files, though not required. You can also include a starter Python program with function stubs for a command-line implementation you will complete for the Final Project (without SQL integration).

Summary of Files to Submit (max 3MB, reach out to El if you have trouble with this limit, but usually compressing images and/or the PDF file using your system's viewer or an online compressor will work):

- **proposal.pdf (a copy of this document filled out)**
- ***additional diagrams/sketches/brainstorming not otherwise required; you can include these at the bottom of your proposal document or as additional files in your submission.***
- **setup_proposal.sql (start to your DDL)** - your final project submission will rename this appropriately
- **app.py (or app_client.py/app_admin.py) function stubs**

The advantage of adding these in your proposal is to get you started in advance, and also to get feedback from the staff on your design and implementation so far.

Student name(s): Brendan Flaherty and Sujit Iyer

Student email(s): bflahert@caltech.edu and siyer2@caltech.edu

Database/Application Overview

In this proposal, you will be "pitching" your project, in which you have some freedom in choosing the domain of with a structured set of requirements that bring together the course material in a single project, from design to implementation to tuning. Keep in mind that unlike CS 121 assignments, you are free to publish/share your project, which can be useful for internship or job applications. In terms of scope, you should shoot for 8-12 hours on this project, though you are free to go above and beyond if you choose.

First, what type of database application are you planning on designing? Remember that the focus of this project is the database DDL and DML, but there will be **a small Python command-line application** component to motivate its use and give you practice applying everything this term in an interactive program; you can find a template from last year here, which may be adjusted slightly before the Final Project is released, but gives you an idea of the breakdown. Don't worry too much about implementation at this step, and you can jot down a few ideas if you have more than one. Just think about something you would like to build "if you had the data" which could be simulated with a command-line program in Python. Your command-line program will start with a main menu with usage options for different features in your application. **For students who took CS 132, you may alternatively use a web-based application which El can help with.** Staff are here to help you with scoping!

Here's a list of some application ideas to get you started. These encompass most of the applications within the scope of this project we anticipate students might be interested in, but if there's a different application that meets the requirements for your DB schema and implementation, you are welcome to ask!

Proposed Database and Application Program Answer (3-4 sentences) :

We plan to design an intense database application and service to manage all functionality of an Animal Shelter. The system will be able to track all adopters, all animals currently in a shelter, all past adoptions at a shelter, the medical records of all animals that have visited the shelter, and all staff at each shelter. The user will be able to use this database to keep track of their histories/records and also be able to find new potential adoptions at various shelters.

Next, where will you be getting your data? We will post a list of datasets to get you started, but you can find many here and here (look for ones in CSV file(s), which you will break into schemas similar to the Spotify assignment; we can also help students convert JSON to CSV if needed). You are also welcome to auto-generate your own datasets (e.g. with a Python script or using ChatGPT similar to A2 Part D) and staff are more than happy to help you with the dataset-finding process, which can take longer than the rest of the starting design process.

Data set (general or specific) Answer: We will auto-generate our own dataset for all the tables. We plan on using AI, but if that does not succeed we will create a Python script to accomplish the task.

Who is the intended user base and what is their role? Identify at least one client usertype, and one potential admin. These different users may have different permissions granted for tables or procedural SQL. Consider clients as having the ability to search ("query") data from the command line, submit requests to insert/update data, etc. In past years, some students had a separate client .py file and admin .py file, with admins having admin-related features, such as approving requests, inserting/updating/deleting information, querying, etc. Including app_client.py and app_admin.py is optional for the proposal this year, but would be a recommended exercise for proposing two different applications to interact with your database. app_admin.py would have a separate username/password in get_conn() and an admin-specific set of show_options() related to admin functionality (e.g. managing inventory or user accounts for an e-commerce store).

***Client user(s) Answer*: The primary client users are potential adopters. They will be able to search for available animals across shelters and can filter their searches based on species, breed, age, and other attributes. Other potential users could be visitors or volunteers who don't necessarily want to adopt but would like to browse or check-up on certain animals.**

Admin user(s) Answer: Admin users can be the shelter manager who has access to all medical records, staff records, and current animals. These admin will be the ones who are responsible for processing adoptions in a timely manner and keeping the database organized. Other admin will be a veterinarian who is not necessarily the manager. This person can have access to update the medical records and set a status of each animal as healthy or unhealthy.

Required Features

The full specification of the project will outline the requirements of the project, but you remember you should aim to spend 8-12 hours (it replaces the final exam and A8), depending on how far you want to go with it. The time spent on finding or creating a dataset will vary the most. For the proposal, you will need to brainstorm the following (you can change your decisions later if needed).

DDL: What are possible tables you would have for your database? Include at least 4 tables in your response. Remember to think about your application (e.g. command-line functions for user prompts to select, add, update, and delete data, either as a client or an admin user). To make this step easier, thinking about the menu options your application provides, as well as the queries you might want to support, is helpful to narrow down the information you'll want to model.

1. Provide a general breakdown of your database schema and tables below, as well as any design decisions you may run into in your schema breakdown. You can provide any notes/questions that you would like us to be aware of for your proposed project.
2. Start your DDL with CREATE TABLE statements in the **`setup_proposal.sql`** file attached in your submission. We will not grade strictly on DDL, but your final submission should have documentation, etc. and we will prioritize evaluating your table breakdown, choice of attribute domains, and keys/relationships.

Answer:

I will describe some of the tables that we plan on implementing:

Animals - Stores information about the animals in a shelter.

Animal_id (PK), name, species, age, sex, health_status, shelter_id (FK)

Adopters - Stores information about potential adopters.

Adopter_id (PK), address, first_name, last_name, phone_number, date_added

Adoptions - Stores information about all past adoptions.

Adoption_id (PK), animal_id (FK), shelter_id (FK), date

Shelters - Stores information about the currently operating shelters.

Shelter_id (PK), location, staff_id (FK)

Staff - Stores information about current staff at a shelter.

Staff_id (PK), First_name, last_name, role, phone_number

Medical_Records - Stores information about the medical records of animals

Record_id (PK), animal_id (FK), exam_date, doctor_name, diagnosis, treatment

We will run into decisions when it comes to foreign key constraints and making sure our tables are all properly communicating with one another. With proper organization and clear thinking, we will be able to properly manage the various keys.

Queries: Part of the application will involve querying data in a way that would be useful for a client (e.g. searching for "Puzzle" games made in the last 5 years, ordered by year and price in a Video Game Store database, or finding all applicants who are pending for an adoption agency).

Identify at least 3 queries that would make sense in a simple command-line application. In your answers, provide a brief description of the query or pseudocode, as well as the purpose in your application. These will likely be implemented as SQL queries within Python functions, wrapping your SQL queries (the methods of which will be taught in class). You are welcome (and encouraged) to add more, though not required.

Answers:

1. Search for animals based on species and age and returning their name and shelter_id. This allows the user to be able to find an animal that matches their specifications making them more likely to adopt. The pseudocode involves selecting from the Animals table and matching the specifications.

2.Another query would be finding the animals that are closest to your own location. This query would be helpful for the user so that the adopter can find animals that are the easiest to adopt. The pseudocode would involve calculating the distance between shelter and current location and returning the animals in ascending distance order.

3.Look through all past adoptions and select based on breed or age. This could help a manager determine which animals are flying off the shelves at certain time periods. This again would involve manipulation of the animal and adoption tables.

4. Another query could be finding all animals of certain characteristics that are healthy. This would involve manipulation of the medical_records table and returning the animal_ids of healthy ones.

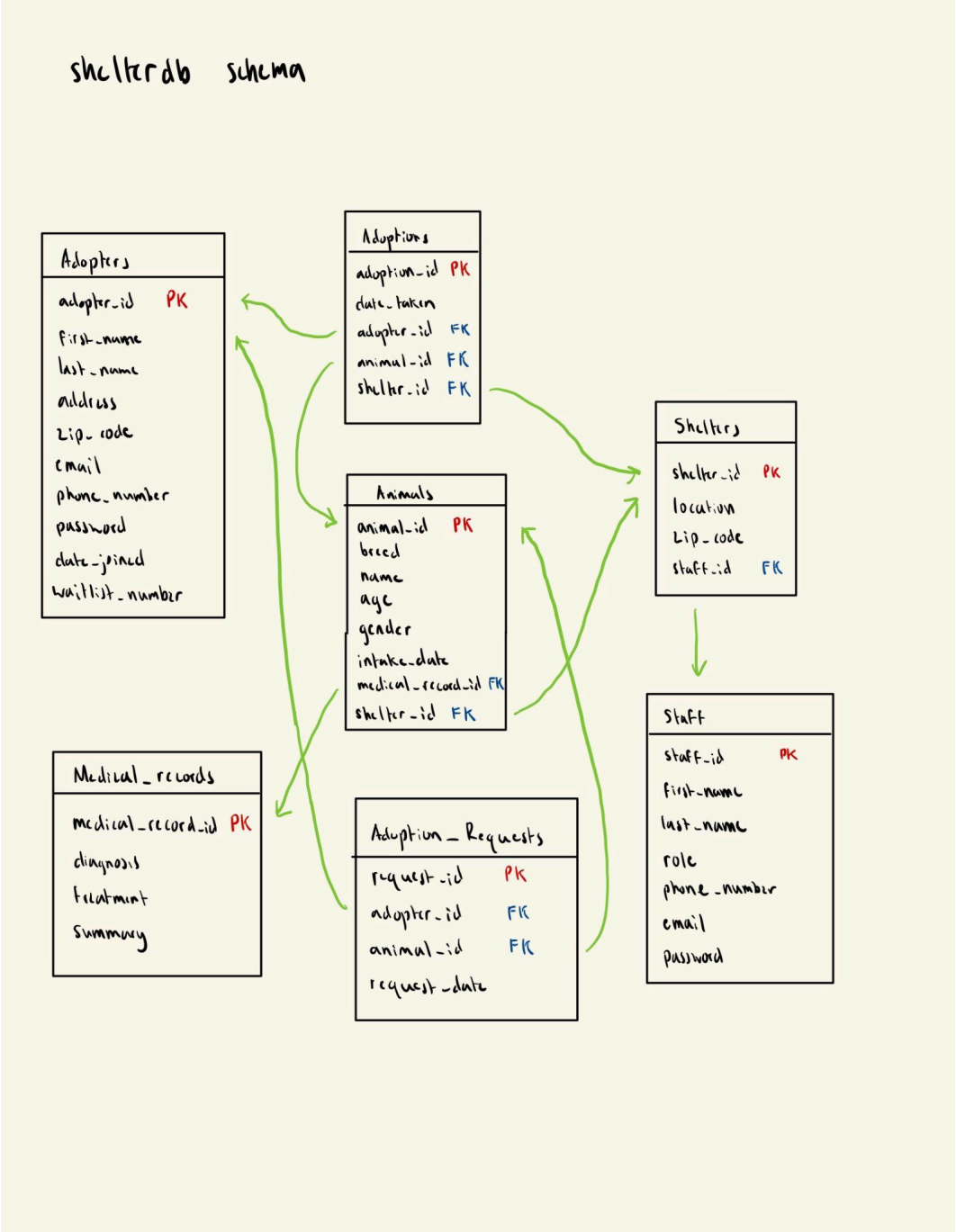
5. An admin may also want to check which medications are needed for animals in the shelter to order more. We can create a query for the admin that adds up all the required medications for all animals in their specific shelter.

E-R Model: There will be an ER model component, which we will cover in a few weeks. ER models are an essential part of database projects, and can vary in conventions. For the proposal, you will not need to complete formal ER diagrams, but you will need to provide 1) a visual representation of your database (a very useful strategy for the design and planning phase, especially when proposing a database model to collaborators with varying technical backgrounds) and 2) some visualization of one possible program flow for a client user from start to finish. For 2), you are encouraged to also include an example program flow for an admin user, though are not required for the proposal. You have flexibility in how you represent your database application here and both diagrams do not necessarily need to be separate, but you must:

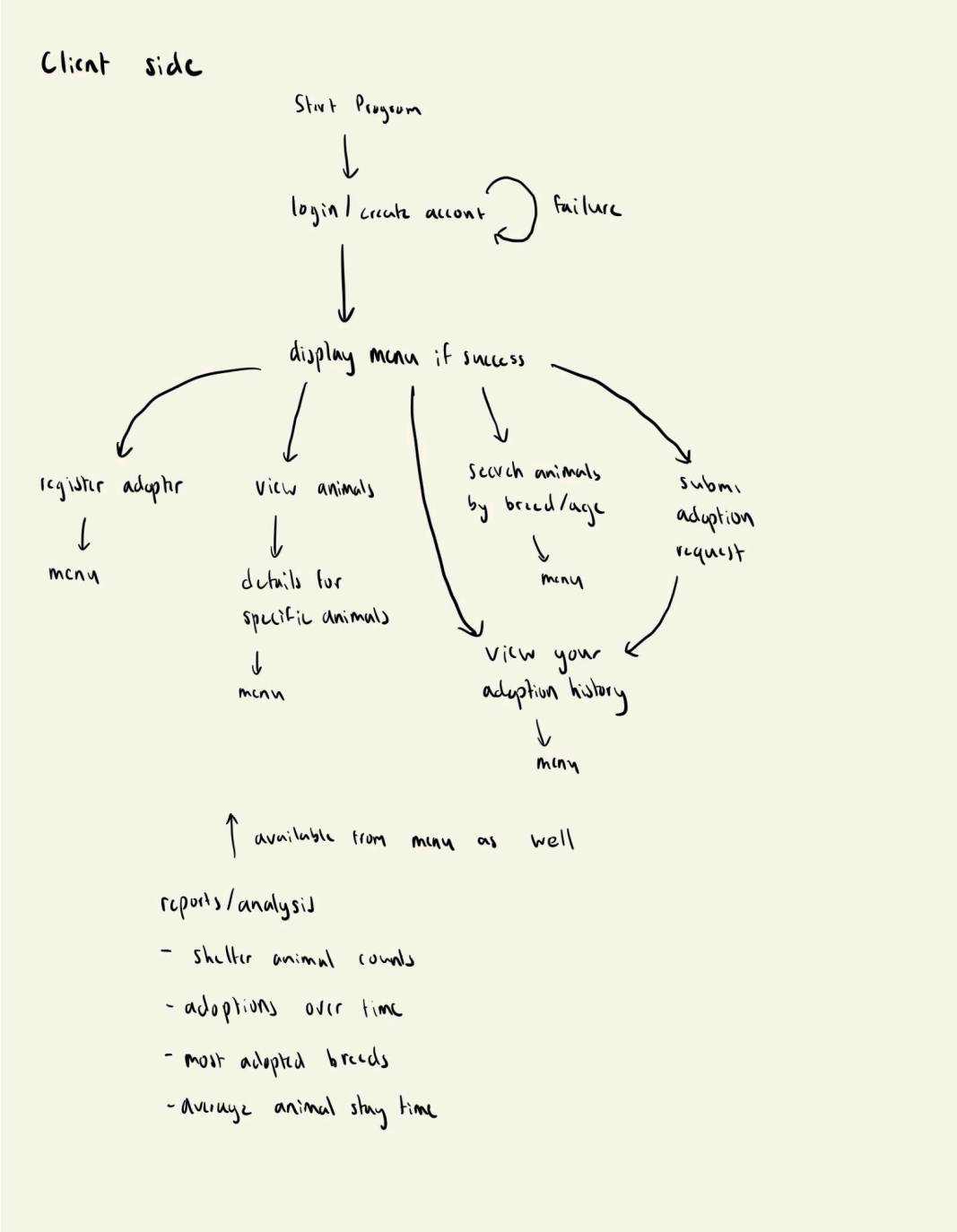
1) Have some representation of (all) proposed schemas and relationships from your DDL (with arrows for foreign keys between tables) that is interpretable.
    - Use annotations (or equivalent) for your proposed queries/client interaction such that someone (e.g. a reviewer) could identify which tables/attributes would be relevant to clients/users. Consider color-coding of keys, etc. to improve your proposed schema organization. Students have also previously added questions for design/implementation decisions they are uncertain about (e.g. "Should this *menu* table be broken down into *drinks* and *foods* or kept as *menu* with the category attribute?")
2) Include at least one visualization (e.g. flow-chart) of a runthrough of your client program from start to finish (it may help to think of your command-line client program as a simulation of a website that interacts with your database). The following is an example (a text-based version, though a visual flow chart is an improvement at the proposal stage to show the different possible edge cases):
    - User starts program (or visits website) -> creates an account after being prompted to login/create account (inserts into loyal customers table with inputted data, such as email) -> logs into created account if success (validation is done on database side) -> is presented a list of options -> inputs ('menu') for show all cafe products listed in '<n>: <product name> <price>' format, where <n> starts with 1 -> inputs 3 to add a Black Coffee to their cart -> confirms 'y' to finish adding to cart and submit order -> order is added to orders table with customer's id and product id -> gets a success message and estimated wait time ('simulated') and is directed back to the main menu -> chooses to log out and finish the order

Schema Diagram (may also alternatively as diagrams.png or diagrams.pdf)

POST DIAGRAM



Example User Flowchart(s):



app.py Client Program

Now that you have brainstormed your DDL and UI flowchart, you have a good start to draft your client application for the project. For the scope of the proposal, we want students to get started early with app.py, prioritizing "proof of concept" and modeling the possible UI flow. app.py will be the client program interfacing with your MySQL database, and we have provided a template to get you started.

For the proposal, your app.py should at minimum have:

- A list of 3 or more possible options for your program in show_options()
    - These will likely correspond to your diagrams and queries above
- Function stubs for each option/feature in your app.py
    - Use docstrings/comments to specify any arguments/input, and returns/output
    - You can use comments to show your current ideas for implementing the functionality
    - For input prompts, you are welcome to "mock" examples; this will make it easier to understand the intended flow for your application, and also easily translate to SQL when implementing the FInal Project requirements
    - Think about possible helper fucntions as you work through these
- If you choose to include both a client and admin application for your proposal, you can submit app_client.py and app_admin.py (there may be overlap with get_conn() and show_options(), with get_conn() having a separate username/password for admin privileges, covered in Wed. 02/12 slides)

You aren't expected to spend more than 30 minutes or so on this part when starting with the template, but should use to make as much progress early on before the Final Project later. Students can see some past demos in lectures/OH this week as well. Include a comment at the top of the program with any questions you have about your proposed client program and feedback you would like us to prioritize (we won't give in-depth feedback to this otherwise, knowing it's a draft subject to change).

Procedural SQL and Database Performance answers are not required for 25wi proposal, but left in for a preview of the corresponding Final Project component)

Procedural SQL *(complete requirements are left in for a preview of the corresponding Final Project component with slides posted in advance):* You will need to implement at least 1 each of a UDF (user-defined function), procedure, and optionally, a trigger in your project. Identify at least one UDF and/or one procedure here. For each:

1. What is the name of the function/procedure?

Get_most_common_breed is an example of a user-defined function that will be helpful for users who are interested in adopting.

A procedure would be adopt_animal is an example that could be implemented. It would fulfill the entire process and updating all the necessary tables.

2. Why is it motivated in your database? Consider the differences discussed in lecture for UDFs, procedures, queries, temporary tables, and views. In your Final Project, we'll be looking for you to demonstrate an understanding of appropriate design decisions here (and we're happy to help discuss any trade-offs)

   The adopt_animal process is motivated to streamline the entire process in the database. Adoption involves updating many different tables so the process would make updating all of these very easy and not user intensive. It makes the process more reusable and efficient.

Consider some examples in class/HW, such as logging DML queries, adding an extra layer of constraint-handling (e.g. the overdraft example from lecture), etc. For procedures, these can be called in an application program written in a language like Python or Node.js, so these are especially useful to avoid writing queries in such application programs, *especially* SQL that performs **`INSERT`**/**`DELETE`**/**`UPDATE`** which you do not want to leave to an application user. Remember that you can also set permissions for different users to access tables and procedures. In your Python program, you can connect to your database as different users defined in your database (similar to the Node.js program El will demo).

(Optional Answers)

UDF(s):

1.
2.

Procedure(s):

1.
2.

Trigger(s):

1.

Database Performance: *At this point, you should have a rough feel for the shape and use of your queries and schemas. In the final project, you will need to add at least one index and show that it makes a performance benefit for some query(s). You don't need to identify what index(es) you choose right now (that comes with tuning) but you will need to briefly describe how and when you would go about choosing an index(es). Refer to the material on indexes if needed.*

Performance Tuning Brainstorming:

To optimize performance, we will add indices on frequently queried columns. These include animal_id, adopter_id, and health_status. It would greatly speed up searches. We will also analyze the entire query execution pipeline to identify slow queries and apply fixes when necessary. We can implement fast look-ups along with efficient modifications and querying to perfect performance.

"Stretch goals"

If you are particularly eager for a certain application (have your own start-up in mind?), it is easy to over-scope a final project, especially one that isn't a term-long project. You can list any stretch goals you might have "if you had the time" which staff can help give feedback on prioritizing.

Answer: Adding pictures to the user interface of each animal. This would be pretty challenging and would require a lot of other parts in the project.

Potential roadblocks

List any problems (at least one) you think could come up in the design and implementation of your database/application.

Answer: We are not too familiar with varying levels of access to user and admin. This will be challenging to implement as we do not have any experience in doing this. We will attend office hours and go on discord for help in achieving this.

Review

This year, we are adding a small component to get students feedback from their peers and to practice giving feedback on proposed projects. You do not need to spend a lot of time on feedback (but may earn up to 3 additional points for above-and-beyond effort, such as reviewing multiple student posts with thoughtful feedback), but for for full credit you must:

- Post your overview followed by one part of your proposal (e.g. diagrams or DDL) on #final-project-review and include at least one question you have you'd like feedback on
- Review one other posted project and provide 1-3 sentences of feedback that demonstrates at least 5 minutes of reflection of trade-offs and concepts you've learned so far. Feel free to bring in your own user experiences! For example, you may have experience inputting a long last name that a website has a size limit on when creating an account, and could note this when suggesting a modification to another student's DDL for representing names.

Link to your Discord post:
https://discord.com/channels/1325070896913846344/1339359506836230230/1340195862923706419

Link(s) to your Discord feedback:

https://discord.com/channels/1325070896913846344/1339359506836230230/1339837697904214047

Collaboration

For projects with partners, this section is required (if not, you can leave it out, or note that you are decided yet). Both students should provide a brief summary of their planned workload distribution, method(s) of collaboration, and 1-2 points you are most interested to do in the project. You may also clarify any concerns/confidence for your partner work here. Feel free to provide a paragraph or bullet points; we're looking for you to have thought this out and discussed your plan for collaboration at this step.

Brendan Flaherty Answer:

I plan on working in lockstep with Sujit. We both seem very excited to do this project so I do not foresee any problems in our partnership. I am most interested in developing our own tables and developing cool schemas that will accomplish cool tasks for both the user and admin. The distribution of work will be equal and we will do all parts together.

Sujit Iyer Answer:

I plan on working in tandem with Brendan. I know that we are both very committed to delivering a great project, so if we have to split deliverables up later on, we will find an appropriate way to do so. I am most excited about finding a way to augment our current flow chart by providing unique ways for our clients and admin to navigate our selection of schema. I am definitely looking forward to the project and finding ways to leverage AI for our dataset.

Open Questions

Is there something you would like to learn how to implement in lecture? Any other questions or concerns? Is there anything the course staff can do to help accommodate these concerns?

Answer: I would like to learn more about the credentials and verification processes. Another thing would be to learn how to learn how to look at the time spent at each query during a process. Also just learn any other tricks that could be useful for our project.

Have fun!

Optional Brainstorming/Sketches/Other Notes

This is an optional section you can provide any other brainstorming notes here that may help in the design phase of your database project (you may find it helpful to refer to the early design phase in your Final Project Reflection).
