1. *Describe what data is stored in the database. (Where is the data from, what attributes and information would be stored?)*
    1. Database for quiz questions
        i. Where is the data from
            1. https://opentdb.com/api_config.php
            2. https://github.com/neotenic/database-dumps
            3. https://www.quizdb.org/?query=&category%5B0%5D=21&search_type%5B0%5D=Question&difficulty%5B0%5D=regular_high_school&subcategory%5B0%5D=25&question_type%5B0%5D=Tossup
            4. https://colab.research.google.com/drive/10lPWqhxmogyfg7Fy5IH8Jx8x_vdGLeSX?usp=sharing
        ii. Attributes/Information stored
            1. Questions, Answers
    2. Data for user info
        i. Store account information (username, password, etc.) for all users
    3. Personal Dashboard and question response data (overall stats, etc.)
        i. For each user, store the date and score of all games they play.

2. *What are the basic functions of your web application? (What can users of this website do? Which simple and complex features are there?)*
    1. User can log in → account can save playing history, scores/high score
    2. Users can start trivia games whenever they want, they can choose their game length and buzz in whenever they want to answer the question. The points received will depend on when they answer the question.
    3. Some more complex features we would like to implement are a leaderboard of all players' scores across the game, as well as allowing the user to select certain categories and difficulty levels that they would like to play.

3. *What would be a good creative component (function) that can improve the functionality of your application? (What is something cool that you want to include? How are you planning to achieve it?)*
    1. Users can select categories to be quizzed on (ex. Entertainment, Music, etc.)
    2. Users can select difficulty levels for questions (Easy, Medium, Hard)
        i. Database of questions can have attributes

3. Users have the ability to send friend requests to other users and have access to their scores. Users can create friend groups which they can compete in.
4. Display overall leaderboard of users with highest scores

4. **Project Title**
    1. TriviAttack
5. **Project Summary**
    1. The application we are planning on creating is a trivia game that allows users to start trivia games whenever they want, they can choose their game length and buzz in whenever they want to answer the question. The trivia question that will be shown to the user and will slowly reveal more of itself as the timer decreases. Once the timer is finished, the question will be fully revealed and the user will have additional time to answer the question. Buzzing in early will net them more points which will be stored into a database with score information.

       Additionally, there will be a component to our project that allows users to interact with each other. Users will be able to see the high scores of other users in a leaderboard.

6. **Description**
    1. We want to create a website where users can play trivia quiz games. We feel that people want to get better at trivia and general knowledge, but there are not many interactive and user-friendly resources/methods that one can use to progress this skill. Additionally, it's difficult for people to improve their trivia knowledge for specific categories and see how they compare to others. Our website will let users sign in/log in to play trivia games and keep track of their score/progress over time. Users will also be able to select categories for what kind of questions they will be quizzed on, as well as the difficulty level of the questions. Not only would users be able to see the highest score they achieved, but they can also view their game history to see how they have progressed, as well as an overall leaderboard of all players' scores. Some stretch goals we have are to show the user's average score for each category in their account dashboard, and to implement a "Friends" feature, where users can see their friends' scores.
7. **Usefulness**
    1. Most other trivia sites and apps like Trivia Crack have short, random trivia questions that don't help you learn about topics because they don't build associations with the information you learn. Our app will ask longer

quizbowl-style questions that share more information about a topic to help you learn better and test your knowledge from vague to specific information.

2. One existing solution is the web-app Protobowl. Unfortunately you cannot select specific categories to practice and the app is very slow and crashes a lot. There is a leaderboard, but it does not persist sessions and you cannot connect with friends using the app.
3. We also get hyper-specific category data so we can further refine by that.

8. **Realness**
    1. Our data will consist of trivia questions, answers and categories that were accumulated from a quizbowl-style archive of questions. We parse the data from a json file in which questions, difficulty, categories and answers are retrieved and stored in a questions table.
    2. Our user profile data (Username, password, profile picture) will be voluntarily collected during their account creation and stored in a table of all users' information.
    3. The response, correctness and score data will be generated whenever a user answers a question and stored in a response table.
9. **Functionality**
    1. The user can choose to create an account, where they would fill out their account descriptions, which will **insert** into the user database.
    2. The user can choose to delete their account, which would involve a **delete** operation of that user's row in the user database.
    3. The user can choose to answer questions at any point during the question timer; their points will be calculated based on if they answered the question early or not. This will be **inserted** in a score database and their overall high score will be **updated**.
    4. The user can choose to refine which questions to see, this would involve a **search** query that returns all questions of a specific topic.
    5. The user can look at current highscores, this would involve a tab where the user could click on to see highscores.
    6. The user would be able to see highscores per category, this would involve the operations of multiple databases which would involve **advanced queries.**
    7. The user would be able to see personal statistics which would involve calculations that would be an **advanced query**.
    8. Insert, search, update, delete, 2 "advanced queries"

10. **A low fidelity UI mockup**

11. **Project work distribution**

1. User account - **Sinja**
    i. Sign-in/login
    ii. User game history
    iii. Avg. score per category
    iv. Account Deletion
2. Quiz game - **Vedan**, **Siddharth**
    i. Storing questions in database
    ii. Creating/generating questions
    iii. Filtering Questions based on selected category/difficulty
    iv. Displaying question timer
    v. Calculating scores and inserting into score database
3. Leaderboard - **Max**
    i. Highscores
    ii. Highscore per category