Prototype **one** of the following projects:

1. Text Sharing

Create a web application that lets users share a snippet of text, similar to http://pastie.org. The program you write should follow these specifications:

- The user should enter the text into a text area and save the text.
- The text should be stored in a persistent data store.
- The program should generate a URL that can be used to retrieve the saved text.
- When a user follows that URL, the text should be displayed, along with an invitation to edit the text.
- When a user clicks the Edit button, the text should be copied and placed in the same interface used to create new text snippets.

2. URL Shortener

Write a web application that allows users to take a long URL and convert it to a shortened URL similar to https://tinyurl.com/.

- The program should have a form that accepts the long URL.
- The program should generate a short local URL like /abc1234 and store the short URL and the long URL together in a persistent data store.
- The program should redirect visitors to the long URL when the short URL is visited.
- The program should track the number of times the short URL is visited.
- The program should have a statistics page for the short URL, such as /abc1234/stats. Visiting this URL should show the short URL, the long URL, and the number of times the short URL was accessed.

3. Trivia

Create a web application that lets users play a multiple-choice trivia game following these specifications:

- The program should read questions, answers, and distractors (wrong answers) from a file.
- When a player starts a game, choose questions at random, and display the answer and distractors in random order.
- A player will be able to select an answer and submit it. If the answer is correct, the program should display a message saying so and move on to the next question. If the answer is incorrect, the program should display a message saying so and end the game.
- The game should include at least 10 questions.
- The program should keep track of the player's score and display it at the end of the game.

Instructions

You are free to use any technology you wish. We value a willingness to try and learn new things so in your README, feel free to mention how much experience you have with the stack you chose. We will take note of this when reviewing your solution. In case you're curious, we work with React and Django at Worklife.

If you choose to use a framework that results in boilerplate code, please detail which code was written by you (as opposed to generated code) in your README.

The front-end should be a SPA.

Create a public repository in GitHub, GitLab, or Bitbucket and push your code there including the files to run your project locally. You should plan and document your work as if this was a real-world project. Remember to commit your changes regularly, and write clear commit messages. Finally, host your solution using a free service like Netlify or Render.

Once you’re done, submit the link to your solution’s repository and hosted application here: **https://airtable.com/shrzj4Wx7aF4RAouS**.