Shepherd

Hi Sean! Thanks for your time and interest in Shepherd!

Now that we've gotten to know a little bit more about your background, your career goals, and the type of place you'd like to work next, we've prepared the following tech challenge to get a preliminary understanding of your skillset. This is meant to be fun, short, and (yes) challenging.

Challenge Outline

Shepherd will sell insurance through independent agents and brokers. A core workflow within Shepherd:

- A broker creates a new insurance application for their client. Depending on the coverage type, a different set of fields appears.
- The broker fills in some fields on behalf of their client.
- The broker invites their client to the application.
- The client fills in the remaining fields and submits them for review by the broker.
- The broker can approve the application as received, leave comments, and/or flag specific fields for revision on the form
- Once the broker is happy with the submission, a broker moves the application to "complete" and can generate final documents for the application (PDF version, Excel sheets for tables in the form, etc.)
- The broker can email the documents to an insurance carrier using the tool

This tech challenge is a light version of the above workflow. It's purposely created to be loosely defined, affording you the space to imagine an ideal solution and capture your thinking around the end-user. Feel free to email questions or needs for specific further clarification.

Product Requirements

Build a simple user workflow.

- A simple web application for a "broker" to use
- A broker lands on the home page and is shown 3 application options. These options are coming from an API. The different options are shown in this JSON file. Notice there is "Company application", "Employee application" and "Auto application".

- A broker clicks on one of the options, they are redirected to a page with the respective fields for the selected form
    - The fields to be shown should come from an API. Please use this JSON file as the config and it can be hardcoded into your backend server (no need for it to be saved in S3, or a DB). The frontend receives the fields via API when the broker navigates to that page. NOTE: Each form has a name and fields, and that there are 3 different forms, each with different field types (text, number, select, checkbox and section for 1 of the forms).
- The broker completes the form fields then submits the form. The data is validated on the frontend (ie. in the config, some fields are required), then sent to an API
    - The data is stored via the API. The data is stored in a DB (Postgres, MySql, SQLite). We use Prisma.io at Shepherd and it's a great way to get started *really* quickly.
    - NOTE: while the config has 3 forms and a set of fields, your solution should work for 100s of forms and 100s of fields for each form
    - *Worst case, store the data as a JSON file to disk*
- The broker is allowed to submit a form only if they have an "auth" cookie with the value "shepherd"
    - *No need to have a login flow or a button to press for setting this cookie in the app. We can hack our way around it by adding the cookie via chrome dev tools when testing*
- Upon successful submission, the broker is redirected to a page where they can edit the values they just submitted. The broker can edit the values and save them. Again, validation on each field + cookie auth.
- No need to deploy this project to the cloud. A working localhost demo is sufficient
- Leverage existing component frameworks such as Bootstrap, Ant design, Chakra, or whatever you're comfortable with. Bonus points for a well designed demo.

Timeboxing the project

Our goal with this challenge is to simulate the type of work you would do in the role so we can understand your thinking process, approach to problem-solving, and how you might collaborate with us as part of our team. Naturally, startup environments are less structured and require more initiative and creativity. We care about why you approached the solution in the way you chose.

We recommend that you share your repo early and commit often so we can evaluate your thinking, and we encourage you to use frameworks and third-party tools so you don't invest a ton of time in areas that don't give us much extra insight. In short, we're looking for 80/20 problem-solving.

Lastly, we want to be mindful of your time, so we ask that you not exceed 6 hours of work (total) on this project. If you begin to exceed this amount, stop, and submit - not having the entire project perfectly done is not a disqualifier.

Bonus points

These are genuinely bonus points and **not required**. We'd rather have you prioritize completing the challenge over checking off these bonus points.

- Handling nested sections
- Using a form library to handle form values
- Some unit/integration testing for either the frontend or backend
- Sharing your repo early, and committing often

Submission

- Please send the GitHub repo to Mo at mo@withshepherd.com
- Create a screen recording showing off the product (we recommend using Loom.com)