Overview:

Our marketplace relies on having a strong volume of sellers willing to sell gift cards at a large discount. The Seller Onboarding Form is the first thing a seller sees on landing on our site. You'll be implementing this Form; a reference implementation is provided in [1] below. We want the Form to be easy to use, yet collect all the info required to sell the card.

Core Requirements (Specs):

- The Seller Onboarding Form should have at least the 6 questions in the "Question Flow" section below. Each question should be displayed alone, sequentially, like [1].
- It should display completed forms in a Results Page like [2]
    - Don't worry about authentication; it is OK if anyone can view the results.
- It should be in Python. You are encouraged to use all nonhuman resources (Google, GPT4, etc). Please do not use human resources, and please do not share this doc.
- The Results Page should display inputs even if a user doesn't get to the form's end
    - A user hitting refresh on the browser should start a new Entry (a new row in Results Page), but if the user hits back or forward with the form's buttons, she stays within the same Entry.
- Your deliverable should ideally be a page we can view on the Internet. This could be on Heroku / a small instance of DigitalOcean/AWS. Or it can be an ad-hoc setup on your local machine (e.g. could use flask or Django) that is publicly accessible.
    - If you're unable to get the above to work under 1 hour, don't worry and just zip your Django project files or Github repo. Include a README.md file that lets us run the project with minimal setup in our environment (a sandbox with WSL / Ubuntu).

Reference Implementation

We have already implemented a version of the Seller Onboarding Form at
[1] https://agora-gcp.uc.r.appspot.com/agora/seller_intake_survey/ . And the results display at:
[2] https://agora-gcp.uc.r.appspot.com/agora/seller_intake_survey_results/
We encourage you to play around with it for a few minutes to understand how it currently works.

Note the exercise isn't about copying [1]. To save you time, you are not required to have some features that [1] has. These features include data validation, currency selector, drop downs, etc. Where the Specs and [1] differ, the Specs take priority. Also, there are areas you can optionally improve upon [1] (for example, in [1], hitting "return" on your keyboard takes you backwards).

Question Flow Diagram: Required Questions

1. What is the name of your store? [Free Text Entry OK]
2. What is the balance left on your gift card [Free Text Entry OK]
3. What price are you selling at [Free Text Entry OK]
4. Which network would you like to receive funds at? {Polygon or Ethereum} [Ideally a selector, but free text entry OK]
5. What address do you want to receive funds at? [Free Text Entry]
6. What's your email address? [Free Text Entry]
    a. Note this question 6 differs from [1]. [1] goes into a more complex branch.

Once the Question Flow ends you can have a page that thanks the user. Note that Questions 1-5 track [1] closely.

Out of respect for your time, the previous page is everything you need to read to do a great job on the project. Reading the below is totally optional.

Optional Appendix 1: Bonus Features

We recommend you keep the entire project time, including reading the specs and submission time, to under 3 hours (the max payable time), and certainly under 5 hours. However, if you finish early, you can show off your skills by implementing bonus features. These bonus features should not be against the base specs; for example, please don't put all the questions on a single page as an "improvement", since this goes against the first bullet point of the Core Spec.

Graphical Improvements

The Seller Onboarding Form is the first page a potential seller is seeing on landing on WaterMelonMarkets. Currently our specs and [1] are pretty bare bones programmer art. You can show off your graphical skills by sprucing up the design and graphics of the form.

User Experience (UX) Improvements

Like the graphical improvements above, feel free to suggest and implement UX improvements. If you do this, please use a few bullet points to write about your UX improvements and why you chose that. Remember UX improvements should not contradict Core Specs.

Question Validation

Similar to [1], you may implement data validation. This may mean that questions like "2) balance left on card" are forced to be numbers with a current selector. Or that questions like "6) Email" include simple email validation (though it's better to be too lenient than too strict).

Optional Questions

For a bonus, after the first 5 questions, instead of asking 6) in the "Question Flow Diagram", you may, similar to [1] ask:

6A) Do you prefer to provide a cash deposit or provide card information for validation? {Deposit, Validate}

If they choose Deposit, you can go directly to:
7A) What is your email address?

If they choose Validate, then ask:
7B) What is the gift card number? 8B) What is the gift card PIN? 9B) WHat is your email address?

Payment Terms

We value your time. Doing the above mini project hopefully gives you useful information for what working with us can be like. On top of this, we'd like to offer a stipend as a small token of appreciation. The stipend is $50/hr, up to 3 billable hours max. The stipend is payable via your choice of a US check sent to an address of your choice, or as an Amazon gift card (any Amazon store in which a US credit card can be used to buy a gift card). Just to ensure we don't e.g. receive empty projects, the stipend is only payable if more than half of the Core Specs are met and we can execute your program. We appreciate you for working with us on the microproject, which we hope is educational for everyone.