Nest.JS Serverless Test

Add to Event is an online marketplace for event services and focuses on connecting organisers with relevant suppliers.

The key entry point for organisers is an event request form which can be found on most marketing pages on the public site: https://www.addtoevent.co.uk

Through years of tweaking and testing we've found that the quality + quantity of requests we receive are stronger the more relevant + tailored the questions are.

For instance if you choose hot tub hire, you'll be presented with a choice between Inflatable and Static Hot Tubs. If you choose Food Vans, you'll be asked which cuisine you're interested in.

Brief

Create a microservice capable of handling form definitions and form submissions.

- Should be able to create, update and delete of form definitions
- Provided with a service name, the microservice should be able to return the form definition for a given service (e.g magicians).
- Handle capture and storage, server-side validation and storage of form submissions.

We'd like you to use the following technologies for the test:

- Nest.js framework for the Restful API
- Firestore for storage of form definitions and form submissions.
- Google Cloud run for running the nest.js application in a serverless environment

We recommend spending around 4 hours on this task

Requirements

- Build a simple nest.js app
    - API to handle CRUD operations
        - Read of a form definition based on a provided service name (magician, confetti or hottub)
        - Creation of a form definition
            - Service name must be unique
        - Update of a form definition
        - Delete of a form definition
        - Submission of a form entry
            - Should validate submission based on schema

            - Read type, check schema, ensure:
                - Required fields are present
                - Check Max length not breached if provided
        - Should return 200 if validation pass and storage successful
        - Should return 400 if validation fails
    - Store JSON objects in firebase
- Write tests for your services, controller and e2e

Bonus

In the event you speed through the test and have extra time, you're very welcome to take on the following additional challenge:

- Integrate with Google Firebase Authentication
    - Create a test user in firebase
    - Generate a JWT for test user
    - Require a valid JWT for:
        - Submission of a form entry
        - Creation of a form definition
        - Update of a form definition
        - Deletion of a form definition
    - Add user Id to successful form submissions
- Generate a swagger file for the API
- Suggest improvements to schema/methodology

Delivery

- Commit the work to a repository (e.g github)
    - Share with https://github.com/dazultra and https://github.com/jackcw
- Deploy to Google Cloud Run
- Provide a short written overview of the work you carried out via email to darren@addtoevent.co.uk along with an idea of how long it took you to achieve the points addressed.

Data structures

Form definition

A JSON object containing the structure, UI instruction and validation rules of a form
Example of JSON with 3 form definitions (confetti/hottub/magician) can be found in the email that had this document attached

Example Schema of a question

- key: string
- type: string (string, textarea, radios, submit, date)
- title: string
- placeholder: string

- options: array (optional)
    - name: string
    - Value: string
- validation: object
    - required: boolean
    - maxLength: number
    - pattern: string
    - validationMessage: string

Form submission

A JSON object containing the submission values of a successful event request form submission

Example schema for a form submission:

- key: string
- name: string
- date: timestamp
- serviceKey:
- questions: array
    - questionKey: string
    - value: any
- dateCreated: timestamp