Rate Quote Coding Challenge

At RateGravity we're constantly looking to help our customers save money on their mortgage. As such we want to make sure we're looking at quotes from multiple local lenders and finding the best rate.
We've built a API that is able to simulate getting rate quotes from our lender network, however we want to build a UI to help us shop these rate quotes.

What are we looking for

We'd like you to build a webpage using React and Redux that allows the user to enter loan information and then displays rate quotes from our API. The UI should look similar to the attached mockup.

How to submit

Please make your code available in a Git repository, we'd recommend posting either to GitHub or Bitbucket. Your code should include a readme file with explanations of how to run your code locally connecting to our API.

How are we evaluating your submission

We'll evaluate submissions holistically, however we'll be looking for the following.

- Code should start and run according to your Readme
- Use of React + Redux
- Unit tests and other automated tests.
- Quality and readability of code.
- Carefully considered Git commits. Each commit should have a commit message, contain a single related change (eg. adding a React component).

Using Libraries and Frameworks

Beyond React and Redux the rest of the technical decisions are up to you. However your Readme should include a brief explanation for why you choose to use any other libraries or frameworks.

Helpful Links

- Create React App - https://github.com/facebook/create-react-app/tree/master

- How to prepare a PR to have a clean history - https://github.com/mockito/mockito/wiki/Using-git-to-prepare-your-PR-to-have-a-clean-history
- Swagger Editor - https://editor.swagger.io/
- MDN Fetch API - https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API
- Redux Thunk - https://github.com/reduxjs/redux-thunk

Our API

`https://ss6b2ke2ca.execute-api.us-east-1.amazonaws.com/Prod/ratequotes`

Our API is called in 2 parts. First a Rate Quote request is posted with a set of desired parameters, and a requestId is returned. Secondly the requestId is used to get the status and results of the query. Swagger documentation for the API is provided below.

All calls to the API require an authorization header in the form:
`Authorization: RG-AUTH YOUR-AUTH-TOKEN`

Swagger API

```
swagger: "2.0"
info:
  version: '1.0'
  title: challenge-api-service-prod
host: ss6b2ke2ca.execute-api.us-east-1.amazonaws.com
basePath: /Prod
schemes:
  - https
definitions:
  RateQuoteQuery:
    properties:
      loanSize:
        type: number
        description: Loan size in dollars
      creditScore:
        type: integer
        minimum: 300
        maximum: 800
        description: The borrowers credit score
      propertyType:
        type: string
        enum:
          - SingleFamily
          - Condo
          - Townhouse
          - MultiFamily
```

```
      description: The type of property for which a mortgage is being requested
    occupancy:
      type: string
      enum:
        - Primary
        - Secondary
        - Investment
      description: How the borrower will use the property
  required:
    - loanSize
    - creditScore
    - propertyType
    - occupancy
  example:
    loanSize: 400000
    crediScore: 720
    propertyType: Condo
    occupancy: Primary
RateQuotes:
  properties:
    done:
      type: boolean
      description: A flag indicating if all lenders have provided rates
    rateQuotes:
      type: array
      items:
        type: object
        properties:
          lenderName:
            type: string
            example: TFB Federal Credit Union
            description: The name of the lender
          loanType:
            type: string
            example: 30YR Fixed
            description: The type of loan
          interestRate:
            type: number
            example: 4.125
            description: The interest rate of the loan
          closingCosts:
            type: number
            example: 10000
            description: The fees the lender with charge when the loan is closed
          monthlyPayment:
            type: number
            example: 1000
            description: the required monthly payment
          apr:
            type: number
```

```
              example: 4.25
              description: the APR of the loan
          required:
            - lenderName
            - loanType
            - interestRate
            - closingCosts
            - monthlyPayment
            - apr
    required:
      - done
      - rateQuotes
  Error:
    properties:
      errors:
        type: array
        items:
          type: string
        description: A list of errors
    required:
      - errors
securityDefinitions:
  RGAuth:
    type: apiKey
    in: header
    name: Authorization
security:
  - RGAuth: []
paths:
  "/ratequotes":
    post:
      summary: Create a request for rate quotes
      produces:
        - application/json
      consumes:
        - application/json
      parameters:
        - in: body
          name: query
          schema:
            $ref: '#/definitions/RateQuoteQuery'
      responses:
        '200':
          description: 'Success'
          schema:
            type: object
            properties:
              requestId:
                type: string
            required:
```

```
          - requestId
    '404':
      description: 'Not Found'
      schema:
        $ref: '#/definitions/Error'
    '401':
      description: 'Not Authorized'
      schema:
        $ref: '#/definitions/Error'
    '400':
      description: 'Invalid Request'
      schema:
        $ref: '#/definitions/Error'
get:
  summary: check the status of a rate quote request and get the results
  produces:
    - application/json
  consumes:
    - application/json
  parameters:
    - in: "query"
      name: requestId
      required: true
      type: string
  responses:
    '200':
      description: 'Success'
      schema:
        $ref: '#/definitions/RateQuotes'
    '401':
      description: 'Not Found'
      schema:
        $ref: '#/definitions/Error'
    '401':
      description: 'Not Authorized'
      schema:
        $ref: '#/definitions/Error'
    '400':
      description: 'Invalid Request'
      schema:
        $ref: '#/definitions/Error'
```