ClearScore - Backend Technical Test

To get an idea of your technical experience, we'd like you to develop a simple micro-service that collates financial products from a small selection of partners. You can use Scala, Java or Kotlin, along with any tools, frameworks and libraries you feel are appropriate.

We'll be looking for the following when reviewing:

- documented code with a coherent project structure
- products being returned in a timely manner
- behaviour when upstream APIs returning errors or taking too long to respond
- unit tests (we're huge fans of TDD!)

Once completed please include the following in a single ZIP file:

- all code, including tests
- a bash script start.sh that will start up your service locally, it should support the following environment variables:
  - `HTTP_PORT`: The port to expose your service on
  - `CSCARDS_ENDPOINT`: The url for CSCards
  - `SCOREDCARDS_ENDPOINT`: The url for ScoredCards
- a short README outlining how you've designed your service and how you'd intend to deploy it

The Challenge

Your micro-service should expose a single endpoint that consumes some information about the user's financial situation and return credit cards recommended for them, sorted based on their eligibility and the cards' APR (annual percentage rate).

A swagger definition (Microservice-swagger.json) which your API should conform to has been included in the zip file. The data you'll be returning comes from two partner APIs, described in the sections below. The provider endpoints below are public and do not require any authorisation tokens to access, but make sure you've got a User Agent header available.

Each partner returns an eligibility rating (i.e. how likely it is the user will be approved) and an APR for each card (watch out for the scales in the sections below), these should be used along with the formula below to sort the cards returned from your API, with the highest scoring cards being ranked higher. The score should be returned in the response for each card as cardScore.

$$sortingScore = eligibility * ((1/apr)^2)$$

Here's an example request and response from the service:

```
POST /creditcards
{
  "name": "Ada Lovelace",
  "creditScore": 341,
    "salary": 28000
}

Response:
[
  {
    "provider": "ScoredCards"
    "name": "ScoredCard Builder",
    "apr": 19.4,
    "cardScore": 0.212
  },
  {
    "provider": "CSCards",
    "name": "SuperSaver Card",
    "apr": 21.4,
    "cardScore": 0.137
  },
  {
    "provider": "CSCards",
    "name": "SuperSpender Card",
    "apr": 19.2,
    "cardScore": 0.135
  }
]
```

Partner 1 - CSCards

The first partner provides a JSON API to get eligible cards. They're only interested in a user's full name and credit score to make their decisions. The response is a list of cards, including an eligibility rating from 0.0 to 10.0. See below for an example request and response, and see https://app.clearscore.com/api/global/backend-tech-test/v1/doc/ for their API definition.

API endpoint: https://app.clearscore.com/api/global/backend-tech-test/v1/cards

```
POST https://app.clearscore.com/api/global/backend-tech-test/v1/
{
  "name": "Ada Lovelace",
  "creditScore": 341
}

Response:
[
  {
    "cardName": "SuperSaver Card",
    "apr": 21.4,
    "eligibility": 6.3
  },
  {
    "cardName": "SuperSpender Card",
    "apr": 19.2,
    "eligibility": 5.0
  }
]
```

Partner 2 - ScoredCards

Our other partner uses all the user's information to make their scoring decisions.

The eligibility rating is provided in the `approvalRating` field and is on a scale from 0.0 to 1.0. An example request and response is below, and see https://app.clearscore.com/api/global/backend-tech-test/v2/doc/ for the swagger definition.

API endpoint: https://app.clearscore.com/api/global/backend-tech-test/v2/creditcards

```
POST https://app.clearscore.com/api/global/backend-tech-test/v2/
{
  "name": "Ada Lovelace",
  "score": 341,
  "salary": 28000
}

Response:
[
  {
    "card": "ScoredCard Builder",
    "apr": 19.4,
    "approvalRating": 0.8
  }
]
```