Eskimi Scala engineer task

The current position requires you to work with a real-time bidding system, so the practical task is related exactly to this. More information about real-time bidding processes: https://en.wikipedia.org/wiki/Real-time_bidding.

Eskimi technical task

Using Scala and Akka toolkit create a real-time bidding agent.
A real-time bidding agent is a simple **HTTP server** that accepts JSON requests, does some **matching between** advertising **campaigns and** the **received** bid **request** and responds with either a JSON response with a matched campaign (bid) or an empty response (no bid).

For this application, you will need to create a "hardcoded" set of case classes that implement campaign protocol.

Campaign protocol

Campaign protocol stores information about the advertising campaign.

```
  case class Campaign(id: Int, country: String, targeting: Targeting,
banners: List[Banner], bid: Double)
  case class Targeting(targetedSiteIds: ???)
  case class Banner(id: Int, src: String, width: Int, height: Int)
```

Notes:
* **targetedSiteIds** could be a very long list, so choose a data type that suits such case well

Bid request protocol

```
  case class BidRequest(id: String, imp: Option[List[Impression]], site:
Site, user: Option[User], device: Option[Device])
  case class Impression(id: String, wmin: Option[Int], wmax: Option[Int],
w: Option[Int], hmin: Option[Int], hmax: Option[Int], h: Option[Int],
bidFloor: Option[Double])
  case class Site(id: Int, domain: String)
  case class User(id: String, geo: Option[Geo])
  case class Device(id: String, geo: Option[Geo])
  case class Geo(country: Option[String])
```

Notes:
* here **w** is short for width, **h** is short for height, **hmin** is short for minimum height and so on..

* at least one of **w, wmin, wmax** will be specified for width and at least one of **h, hmin, hmax** will be specified for height. Always prefer validating **w** and **h** if they exist, otherwise fallback to **wmin**, **wmax**, **hmin** and **hmax**. min/max values might have different combinations.
* **bidFloor** is the minimum price amount that would be accepted as a valid bid price.

Bid response protocol

```
  BidResponse(id: String, bidRequestId: String, price: Double, adid:
Option[String], banner: Option[Banner])
```

Notes:
* **adid** is the campaign ID of the campaign selected

General notes

* the bid request and response protocols provided above are excerpts from OpenRTB protocol but you can use any values, for your cases. More info:
https://www.iab.com/wp-content/uploads/2015/06/OpenRTB-API-Specification-Version-2-3.pdf
* upon receiving a bid request, bidding agent should validate if this request is something that we want to bid on (based on your campaign list) and if so - we respond with a bid response JSON.
* if bidding agent is not going to bid on the request, it must respond with HTTP 204: No content.
* if multiple campaigns/banners match bid request, you should randomly select one of them
* **device.geo** object has a higher priority than **user.geo** object

Technical Requirements

* use Scala, Akka Actors for the bidding agent implementation
* use Akka HTTP for HTTP server and receiving and responding requests
* implement campaign matching logic between bid request and campaign protocol for provided fields:
  a) bid floor
  b) country
  c) site ID
  d) width and height (including minimum and maximum values)
* add a short description of how to validate the solution

Bonus points

* add tests, we value them a lot
* case class Targeting() could be modified with adding more variables/values if needed.

Examples

Campaigns:

```
val activeCampaigns = Seq(
 Campaign(
   id = 1,
   country = "LT",
   targeting = Targeting(
     targetedSiteIds = Seq("0006a522ce0f4bbbbaa6b3c38cafaa0f") // Use collection of
your choice
   ),
   banners = List(
     Banner(
       id = 1,
       src =
"https://business.eskimi.com/wp-content/uploads/2020/06/openGraph.jpeg",
       width = 300,
       height = 250
     )
   ),
   bid = 5d
 )
)
```

Bid request:

```
{
  "id": "SGu1Jpq1IO",
  "site": {
      "id": "0006a522ce0f4bbbbaa6b3c38cafaa0f",
      "domain": "fake.tld"
  },
  "device": {
      "id": "440579f4b408831516ebd02f6e1c31b4",
      "geo": {
      "country": "LT"
      }
  },
  "imp": [
      {
      "id": "1",
      "wmin": 50,
      "wmax": 300,
      "hmin": 100,
      "hmax": 300,
      "h": 250,
      "w": 300,
      "bidfloor": 3.12123
      }
  ],
  "user": {
      "geo": {
      "country": "LT"
      },
      "id": "USARIO1"
  }
}
```

Expected bid response:

```
{
  "id": "response1",
  "bidRequestId": "SGu1Jpq1IO",
  "price": 3.12123,
  "adId": "1",
  "banner": [
      {
      "id": "1",
      "height": 250,
      "width": 300,
      "src":
"https://business.eskimi.com/wp-content/uploads/2020/06/openGraph.jpeg"
      }
  ]
}
```