# Quizmaster-App

Your task will be to develop a smartphone app to play a quiz based on and about IoT technologies. Literally the idea is that the app can recognize iBeacon signals. The signals shall be used to query a web service for detail information about the signals. This may result in messages with questions, answers and checks of your answer selection. To receive a question an answer, you will have to find one of our iBeacons. Each iBeacon can only trigger one web service request per question-answer tuple. The latter means that you will have to collect all answers for a single question by finding all iBeacons.

Thus, the functionality of our app will be rather simple: we will have many clients (your devices running the app you will have to develop) and web services which are utilized to respond to the IoT events issued by the iBeacons. The web services provide different operations to collect and process data:

* needed to register (you as a quiz player; with a unique username),
* provide questions to your app,
* provide answers for each question (three answers may be collected for each question), and
* to check your suggested answer.

To automate the requests between your frontend and the web service, we set up a network of four iBeacons within this room. Each iBeacon has its own configuration and sends out a BLE signal. To develop the frontend and make the app cross platform ready, we are going to use the Ionic framework. Therefore, we provided a base project on GitHub with basic structure that can be used for your extensions to complete the quizmaster app.

# iBeacons

The provided iBeacons operate with the following configuration:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Major | Minor | UUID | Range |
| MiniBeacon\_52069 | 1 | 1 | B9407F30-F5F8-466E-AFF9-25556B57FE6D | ~ 2m |
| MiniBeacon\_52078 | 2 |
| MiniBeacon\_52080 | 3 |
| MiniBeacon\_51923 | 4 |

You will need this information to configure your app. The parameters above tell you how to listen and distinguish between the iBeacons.

# RESTful services

The RESTful services provide data and processing for your app. They are available under the Base-URL ec2-52-212-125-177.us-east-2.compute.amazonaws.com, Port 8080. The URIs for requests are the following:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| HTTP Method | URI | Parameter Type | Description | Example | Example Response |
| GET | /userid/:name | name: any | Registers your user to the backend | /userid/foo | {  "uid" : 9  } |
| GET | /question/:uid | uid: Integer | Getting a question for your registered user | /question/9 | {  "uid" : 9,  "qid" : 9,  "question" : "Wann muss ich aufs Klo?"  } |
| GET | /answer/:uid/:qid | uid: Integer,  qid: Integer | Getting 1-3 answers for your question | /answer/9/9 | {  "qid" : 9,  "uid" : 9,  "aid" : 0,  "answer" : "JETZT!!"  } |
| GET | /proof/:uid/:qid/:aid | uid: Integer,  qid: Integer,  aid: Integer | Checks if your suggested answer was correct | /proof/9/9/1 | {  "proof" : "OK",  "qid" : 9,  "uid" : 9,  "aid" : 1,  "result" : "Answer was correct. Your score went +1"  } |
| GET | /score |  | The scores of all teams | /score | [ {  "username" : "Team1",  "score" : 1  }, {  "username" : "Team2",  "score" : 1  }, {  "username" : "Team3",  "score" : 0  } ] |

# Exercise – Frontend Development

In the provided project on GitHub you may find source code to start your project.

You can find it under: <https://github.com/futureLABHsPforzheim/quizmaster_start> In the code you will find some useful comments where to code what. Use the following commands in your command line or terminal to download the code and install all necessary dependencies:

Git clone : <https://github.com/futureLABHsPforzheim/quizmaster_start>

npm install

## Task 1: Get iBeacon Data!

In the page *home* which is also given you need to scan for the iBeacons we suggest using the Ionic iBeacons plugin. Please follow the guide on <https://ionicframework.com/docs/native/ibeacon/> to install the plugin and to find an example (how to use it). With the code given and your knowledge about iBeacon monitoring and sending HTTP requests, you will be able to further develop your app and retrieve question-answer tuples.

## Task 2: Answer Check

Finally: Check if your suggested answer is correct. Implement a HTTP request to our web services to check whether a provided answer is right or not. Afterwards, increase or decrease your score.

## Bonus task: High Score Page

If this was too easy for you, you may also create a high score page which displays all teams and their score. Remember to call a web service to get the high scores.