# Java developer test

A webservice and database has been created to provide a simplified implementation of a casino game provider for you to use in this assignment. You are required to develop a REST API service that uses this database to retrieve customer data, and using that data integrate to the provided simulated game provider API in order to get a game launch url to return to your API client.

**Functional requirements:**

1. Implement a REST endpoint for login that can receive the following POST data:

{

"user":"demo1",

"password":"password1"

}

(The values given above for username and password are valid credentials already stored in the database)

Using the supplied username and password, query the database to retrieve the UUID for this user. The password is stored in plain text in the database. The UUID value is to returned to the user be used as the authentication token for the next API.

1. Implement a REST endpoint for game launch that can receive the following POST data:

{

"token":"<token returned in the previous API call>",

"gameid":"superspins"

}

(The value given above for gameid is a valid identifier within the simulated casino game provider API)

Security should be applied to this URL which must validate that the token is valid (a check it exsits in the database is enough). If the token is not valid a HTTP 401 should be returned.

Using the gameid and token provided to your API, make a call to the simulated casino game provider API in order to get the final game launch URL.

The request to the simulated game provider API is expecting a POST request containing a body with the following format:

{

"signature":"<Base64 encoded SHA1 signature>",

"params": {

"token":"<UUID from database>",

"gameid":"<gameid from client>"

}

}

To prepare this message, you must add provide a base64 encode SHA1 signature of your message constructed from the following string: token|gameid for example “1234-ABCD-5678|superspins”. The final signature will resemble something similar to this: “KNX2rAWC... 7NazNw==”. The private key needed to sign your messages is in pem format at the end of this document.

The simulated game provider API will validate all requests to it to check the token, gameid and signature are valid.

You must handle any error returned by the simulated game provider API, any missing data in the database or general exceptions as found and return this to the user in a meaningful way.

**Technical requirements:**

Your webservice must be implemented using a recent version of Java and Springboot.

Your game launch endpoint must be secured with Spring Security.

You must use hibernate and JPA to connect to the database.

Any other libraries are your choice.

**Details of provided services:**

The simulated game provider API is available at:

<http://ec2-18-130-236-146.eu-west-2.compute.amazonaws.com:8081/gamelaunch> This service holds the public key associated with the private key supplied to you.

A Postgres database holding user information is as follows:

Host: database-1.cq4ijmgk5u83.eu-west-2.rds.amazonaws.com

Port: 5432

Default database: users

Username: dbtestuser

Password: dbtestpassword

Table schema:

|  |  |
| --- | --- |
| Id | int4 |
| Username | varchar(255) |
| Password | varchar(255) |
| UUID | varchar(255) |

**Private key:**

-----BEGIN RSA PRIVATE KEY-----  
MIIEogIBAAKCAQEA44L+QkfHcKqAUrWQ6HzxuFDQ3wUWKEUoES9iGCY4dKDw6esZ  
P0uOPWXtUP+vO1U/wKYDTMaoJ/xa4jE3lcaA1kPiaSbgbE7Nxtt9ZEnBgzwapXzF  
+YjYAazZcSN87KreoQAQPdaps5mJzyQjSWRwht0tswYuUIN8UT21ujM9ikavf5A+  
Wp53rYOMTtC2Rx+7TmgKn6kZu8WjrFy/meiNYaTXd/9ytqrBYZZKF/odaBKP9E6S  
UYVSboySXuZHc7Ppz3uH5OQYJDYL7YDeOAynV0BhMKdHlhSc0rZVQ89iS1hoc4BG  
u2RHUv1u+ZU2yKX0PQNhlDtjIW0S5f4UboIhtwIDAQABAoIBAD6Eq4vZH6QFJlgU  
DqrkUqQciRkP08PjJRx70c/jCNZ7t1gAQ6LSB1c14B3G9Qcv6OHufK+nU2QRg4Pu  
lGCy+Ipogxn0Fe2NRhykCfP6FJQUexoIjHqCsSDct6/QWa7LUXwagVwM1ftFcMoc  
Gxo64HUe6ETuqOF1ED9e3sY4+Gs7drGbMlnp70cEjIu1ChIJKciMRQ/xwz60XTZK  
vNWCuibqZmjDKifO3Kmkp3bv7snhkmgY1o8nUIhTRhxdrYYjEZ4SkPk7d6LQXWUz  
yD950YhravPYd5tB6QX4GvSrK0p0F0Weg2zQDdsFoRaAzhAYsICzO4qvRMaamsOj  
JNo8a0ECgYEA8WNI0B165QfW56fmhEHUSWLSgEB3yx4Zuodqg05g1SMamxzJvCT7  
0LIGR6Mrsc5qmzH/vV/6eDoHl0N4qJAYKEKN8cA9inBZBbxUc3BAp+aTBOrPZqlq  
HwMdg9+vllAVXSyyY58f6l19cUPZHhLxUwltEDBkRxFRsPzMRF3/HJcCgYEA8Uis  
pJFCi3vBqMjzoywDa2nxCNc2m0zAHJC2NAv3ZgAvRMTyqWC56/kxPGOAOhPQzmWq  
IExHfcw9eQBWUtrmfQgZdCS6yJPdeeF+yYyIedQyqhxtvfzV3zqT/GF+HVqrFp/W  
KONFlyBLdvLgH7DY+B8UGqJyQtRQEpnyVdZpJ+ECgYAP6z5IC0mphQTVLitzmWvI  
ky3Sq227BXco5/lMQ4T7SV9dz142fDHnJ3Zmjwo4paY+KeP1tziraRhLD3YLl8Ug  
EpON1+G3GYWlbTAclNCOwhP8BQg8evu2o9v0cHIB9bIS/PmevEW8jQHS0GinJRSp  
Jzv8D1TtouTCSUmnUJOcbQKBgH6n6YhRgB78DcJuCgajegAOkqpTXqTS+x2DkUFs  
7qcqfaWZxM/IqJAINSJAEPVrpXZSw6VEpVqRtcsotCe2JLuJXuMqzKmxhNcP2aAv  
FwzsqNPHLIxIhnQxqbEX9D2wdJLhwGeiNcuI9lQwVhoMmThWCBWbJGOUU5wQF7eb  
X4PBAoGAKcsWHH61RHoXRx1xupVykqSfp2NKnbVfxVhUhKXxggPgqXVKwxWuYByF  
KDbm1Pz7EPRmCbuG0wDzgue3taZxQ/JoyINfZRzZg6duN5gV0wyRAa4eXf9AnnP6  
Y9O9i1vjDmNIiQ3djhOBhtqQjJ/gMszg9BVCeofR3kVx+K3KtZM=  
-----END RSA PRIVATE KEY-----