**Instructions**

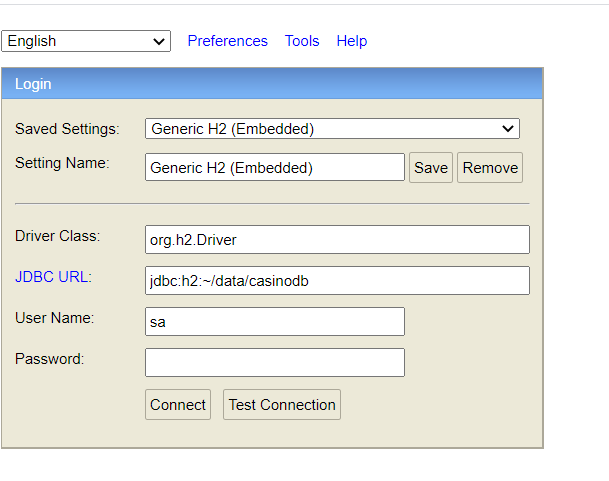
**Pre-requisites:**

Java 1.8.0

Maven 3.x.x

1. **Run: mvn spring-boot:run**
2. **Do a health test check on this public on Postman or your preferred API:** [**http://localhost:9090/rank/api/v1/player/health/test**](http://localhost:9090/rank/api/v1/player/health/test) **GET**
3. **H2 Database Endpoint:** [**http://localhost:9090/rank/api/h2\_console**](http://localhost:9090/rank/api/h2_console)

Credentials:



1. **Add Player: http://localhost:9090/rank/api/v1/player/add PUT (Add the player to start testing)**

Request Body:

{

"username":"username"

}

Response Status: 201

1. **Get player’s current balance: http://localhost:9090/rank/api/v1/player/{playerId} GET (Current player balance)**

Request Body: No Request Body

Response Body:

{

    "id": 1,

    "balance": 12.00

}

1. **Get Player statistics: http://localhost:9090/rank/api/v1/player/statistics GET (Player’s 10 latest wins/wagers by username)**

Request Body:

{

"username":"username",

"password": "swordfish"

}

Response Body:

[

        {

            "id": 1,

            "balance": 2.00,

"status": "win"

        },

        {

            "id": 2,

            "balance": 1.00,

"status": "wager"

        },

        {

            "id": 3,

            "balance": 4.00,

"status": "win"

        }

     ]

1. **Deposit money to the player’s balance: http://localhost:9090/rank/api/v1/player/{playerId}/win** **POST (If player has no existing transaction, new transaction will be created)**

Request Body:

{

"id": 1,

"balance": 23.5

}

Response Body:

{

"id": 1,

"balance": 30.5 (existing balance + request body’s balance)

}

1. **Deduct money from the transaction id on the provided on the request body for playerId: http://localhost:9090/rank/api/v1/player/{playerId}/wager POST**

Request Body:

{

"id": 1,

"balance": 23.5

}

Response Body:

{

"id": 1,

"balance": 0.00 (existing balance - request body’s balance),

"teapot": "teapot",

"code": "418"

}