YOCOJACK SIMULATION DOCUMENTATION

Document	Version	Author	Description
13 May 2019	1.0	Duncan Nyakundi	Yocojack simulation

Requirements

- Java JDK 1.8 setup
- Maven home variable must be setup
- (Optional) Prefered IDE e.g Netbeans

Usage Procedure

1 Download the source code from https://github.com/dosindi

2 Proceed to yocojack directory

3 Execute

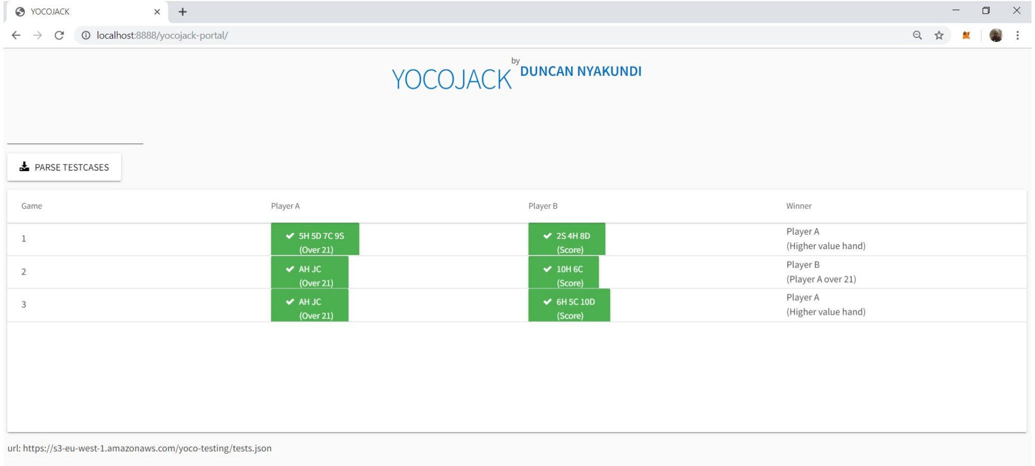
```
mvn spring-boot:run
```

4 Access http://localhost:8888/yocojack-portal/

5 Click Parse TestCase button

6 View Results on the Grid

Screenshot



Game	Player A	Player B	Winner
1	✓ 5H 5D 7C 9S (Over 21)	✓ 2S 4H 8D (Score)	Player A (Higher value hand)
2	✓ AH JC (Over 21)	✓ 10H 6C (Score)	Player B (Player A over 21)
3	✓ AH JC (Over 21)	✓ 6H 5C 10D (Score)	Player A (Higher value hand)

url: https://s3-eu-west-1.amazonaws.com/yoco-testing/tests.json

Technical Workflow

- To compile the entire project, run "mvn install".
- To run the application, run "mvn spring-boot:run" and open http://localhost:8888/yocojack-portal/
- To produce a deployable production mode WAR:
- change productionMode to true in the servlet class configuration (nested in the UI class)
- run "mvn clean package"
- test the war file with

```
mvn spring-boot:run-war
```

Client-Side compilation

The generated maven project is using an automatically generated widgetset by default. When you add a dependency that needs client-side compilation, the maven plugin will automatically generate it for you. Your own client-side customizations can be added into package "client".

Debugging client side code - run "mvn vaadin:run-codeserver" on a separate console while the application is running - activate Super Dev Mode in the debug window of the application