1. Java cocepts

collection

hash map,hash table,linked list

array,array list,stack,queue

search and sort algo

binary trees

no need to learn time complexity

Jasper reports

1. It's basically a tool that helps us to create reports, deploy and run reports

So it has many products which help us for this

Jaspersoft studio is the design tool of it which helps us to create the layout, define inputs and define query or set the data source for it

Then there is jasper report server - where we can deploy our created reports, configure the input, configure the datasource, and run them

then there is JRIO - jasper report input server - which provide APIs to be called from other application to run and generate jasper reports

So here we use studio to create the reports and then JRIO is being called thru REST API from the MoCA to run those reports

And it's not necessary that datasource to jasper has always to be query or stored procedure. We can also use java class the datasource or files can be source

MoCA Reports

So MoCA is a application which has functionalities

including report generation

so we have configured and deployed our reports there

and users can run it from there

VDS Analytics

It is used to store data in S3 platform, AWS database. All the vehicle data, result data, traces, error suppression is stored here.

select \* from vehicle;

DECLARE

vehicle\_id vehicle.id%type;

CURSOR cursor\_vehicle

IS

SELECT id FROM VEHICLE;

BEGIN

OPEN cursor\_vehicle;

LOOP

FETCH cursor\_vehicle INTO vehicle\_id;

EXIT WHEN cursor\_vehicle%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('NAME: '||vehicle\_id);

END LOOP;

CLOSE cursor\_vehicle;

END;

/

1. http://localhost:8080/hello
2. To run: mvnw compile quarkus:dev
3. mvn clean install
4. mvn quarkus:dev
5. <https://www.mastertheboss.com/soa-cloud/quarkus/how-to-consume-quarkus-rest-services-with-react/>

1. Enable the Windows Subsystem for Linux

dism.exe /online /enable-feature /featurename:Microsoft-Windows-Subsystem-Linux /all /norestart

2. Enable Virtual Machine feature

dism.exe /online /enable-feature /featurename:VirtualMachinePlatform /all /norestart

3. Restart your machine

4. Download and install the Linux kernel update package

https://wslstorestorage.blob.core.windows.net/wslblob/wsl\_update\_x64.msi

5. Set WSL 2 as your default version

wsl --set-default-version 2

6. Download the Ubuntu20 distribution and Install the distribution

https://www.microsoft.com/store/apps/9n6svws3rx71

7. Create a wsl.conf file, (execute this command in Ubuntu) Copy the next 4 lines and execute as a single command

sudo tee /etc/wsl.conf << EOF

[network]

generateResolvConf = false

EOF

8. Shutdown wsl (execute this command in windows cmd)

wsl --shutdown

9. Start the ubuntu application

10. Remove the resolv.conf file (execute this command in Ubuntu)

sudo rm /etc/resolv.conf

11. Create the resolv.conf file (execute this command in Ubuntu) Copy the next 5 lines and execute as a single command

sudo tee /etc/resolv.conf << EOF

nameserver 53.76.29.10

nameserver 53.72.30.10

nameserver 8.8.8.8

EOF

12. Shutdown wsl and start again

13. Install Docker in ubuntu20

https://docs.docker.com/engine/install/ubuntu/

sudo systemctl status docker

sudo systemctl start docker && sudo systemctl enable docker && sudo systemctl status docker

sudo docker run hello-world

javax.net.ssl.SSLHandshakeException: PKIX path building failed: sun.security.provider.certpath.SunCertPathBuild

erException: unable to find valid certification path to requested target

keytool -list -keystore "C:\Program Files\Java\jdk-19\lib\security\cacerts"

>keytool -import -noprompt -trustcacerts -alias sonareclipse -file C:\Users\lokeshc\Downloads\Corp-Root-CA-G2.crt -keystore "C:\Program Files\Java\jdk-19\lib\security\cacerts" -storepass changeit

>keytool -import -noprompt -trustcacerts -alias sonarlint -file C:\Users\lokeshc\Downloads\Corp-Prj-Root-CA.crt -keystore "C:\Program Files\Java\jdk-19\lib\security\cacerts" -storepass changeit

// try {

// ObjectMapper objectMapper = new ObjectMapper();

// String jsonString = objectMapper.writeValueAsString(dataObject);

// return jsonString;

// }

// return dataObject.toString();

// String resultfromlocal=Response.ok(dataObject).build().toString();

// LOGGER.info(resultfromlocal);

// LOGGER.info("-----");

// LOGGER.info(dataObject.toString());

//

// resultfromlocal=convertToString(dataObject.toString());

// catch(IOException e) {

// e.printStackTrace();

// throw new VdsBusinessException(ErrorCode. DLMS\_NO\_SUCH\_ID\_ERROR, objectId, "get");

// }

// }

configuration:

uploadLinkLifeCycleInSeconds: 3600

presignedDownloadUrlLifetimeInSeconds: 3600

uploadEventQos: 1 # AT\_MOST\_ONCE = 0, AT\_LEAST\_ONCE = 1, EXACTLY\_ONCE = 2

eventPolling:

pollingSchedule: "\*/10 \* \* \* \* ?"

uploadLinkLifeCycleBufferInSeconds: 40

s3HeadTimeoutInSeconds: 2

presignedDownloadUrlBlacklist: ["friedetest2"]

newObjectEventTopic: "vds/datalake/events"

// public String convertToString(String jsonData) {

// try {

// ObjectMapper obj = new ObjectMapper();

// JsonNode node = obj.readTree(jsonData);

//// System.out.println(node);

//// obj.treeToValue(node,DataObject );

// Map<String, JsonNode> keyValueMap = new HashMap<>();

// node.fields().forEachRemaining(entry -> {

// if(entry.getKey()=="id" || entry.getKey()=="type" || entry.getKey()=="size" || entry.getKey()=="creationDate" || entry.getKey()=="metaData"

// || entry.getKey()=="data") {

// String key = entry.getKey();

// JsonNode value = entry.getValue();

// keyValueMap.put(key, value);

// }

// });

// ObjectNode newJson = obj.valueToTree(keyValueMap);

//

//// DataObject d1=DataObject(newJson);

// return newJson.toString();

//

// }

// catch (IOException e) {

// e.printStackTrace();

// throw new VdsBusinessException(ErrorCode. DLMS\_NO\_SUCH\_ID\_ERROR, "get");

// }

//

//

// }

LOGGER.info("jsonData:"+jsonData);

// dataObject.setData(new DatalakeData(null, jsonData));

C:\Users\xxxxx\Downloads\eclipse-java-2022-12-R-win32-x86\_64\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.5.v20221102-0933\jre>bin\keytool.exe -import -file C:\Users\xxxxx\sonarlint\_uc.cer -cacerts -alias sonarlint\_uc

C:\Users\xxxxx\Downloads\eclipse-java-2022-12-R-win32-x86\_64\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.5.v20221102-0933\jre>bin\keytool.exe -import -file C:\Users\xxxxx\sonarsource.cer -cacerts -alias sonarsource

REACT

-> Open source library for building user interfaces.

->Component based architecture(resuable code)

->Declarative

->Component types

Stateless Functional component => used more

absence of 'this' keyword

solution without state

Stateful class component(render) => mantain own private data

lifecycle hooks

->JSX Javascript XML

->Props: props get passed to the component, immutable

->State: state is managed within the component, can be changed

->setState: State cannot be changed directly, it has to be set using setState, or else the change will be undefined and will be displayed only in console

->Event handler

->Binding event handlers

Dynamic binding in render()

Bind in constructor()

Bind with the Arrow Function

->Method as props: props can be passed to method (parent, child method can be used )

->Conditional rendering: if/else

Element variables

Ternary conditional operators

Short circuit operators

->List rendering: map() method can be used

The map() method creates a new array by calling a provided function on every element in the calling array.