\*\*features a agregar:

.Descripciones en pagina de github

.Arreglar en mobile

.Arreglar filtros

Creado con Spring Tool Suite 4 Version: 4.1.1.RELEASE

API REST

Workspace y repo local: C://Users//guido//workspacedev//basketbackend

Repositorio en https://gitlab.com/guidobertinat/basketbackend

Configuracion local/produccion de PATH para subir fotos

Los controllers deben tener el @CrossOrigin(origins = "\*", maxAge = 3600), para aceptar solicitudes desde JS, desde cualquier sitio. Ver si cambiar como bestpractice.

El path tiene que estar abierto en SecurityConfig

PlayerController:

**private** **static** String *UPLOADED\_FOLDER* = "C://Users//guido//workspacedev//basketbackend//src//main//webapp//";

Con esta ruta (webapp) impacta en dominio.com:8081/basketbackend

(se instala en tomcat)

En /opt/apache-tomcat-8.5.37/webapps/, al mover el .war en esta carpeta de descomprime y deploya solo, luego reiniciar servidor.

En carpeta /opt/tomcat/bin :

Sh shutdown.sh (apaga servidor tomcat)

Sh startup.sh (levanta servidor tomcat)

Ps –aux para ver los procesos, kill -9 14103(el pid) para matar proceso si esta colgado.

Esto puede ir como standalone y correr como un java con java -jar prueba.jar

como un servicio. En este caso no hace falta extender de ninguna clase la app principal. (En desarrollo como el Spring Tools incluye un server embebido en la app, no hace falta y no extiende de nada tampoco)

O puede ir como una aplicación a ejecutar en un tomcat instalado, como esta deployado en este proyecto.

Generando un .war, extendiendo la clase app (principal) de SpringBootServletInitializer. Como tenemos en el workspace del proyecto productivo.

public class App extends SpringBootServletInitializer {

El archivo se genera con botón derecho 🡪 export 🡪 WEB/War File

C://Users//guido//workspaceprod//basketbackend

El archivo application.properties tiene distintas configuraciones del proyecto, timeout de token, puerto, claves, etc.

En la API java utilizo otro workspace de produccion para compilar con los archivos modificados, trabajo en desarrollo y luego muevo cambios (los bajo desde gitlab o manual). Al revés de la parte de react que el repositorio master es el productivo.

Utilizo MAVEN, con la configuración del proyecto en el pom.xml

\*\*\*Resources placed in src/main/resources/public will be served from the root of your application. For example src/main/resources/public/hello.jpg would be served from <http://localhost:8080/hello.jpg>

¿Autorización JWT como funciona?

Login: https://github.com/AgileIntelligence/AgileIntPPMTool/tree/branch66

Tabla USERS, en mysql AWS no se puede usar palabras reservadas como USER, para tablas.

Utilizo node.js (este se utiliza para crear y gestionar la aplicación en react, tanto para transpilar como para generar deployable) la aplicación se crea con create-react-app

Una vez en la dirección del repositorio se inicia con npm start

Deploy de React en desarrollo y producción

Se utilizan dos ramas, una master que tiene la versión productiva para compilar y es subida a github, y otra rama dev en local, con la configuración para desarrollo.

Trabajo en dev, y hago merge en master antes de push a github.

En axiosUrl:

const clienteAxios = 'http://18.220.188.121:8080/basketbackend'

En App.js

        <BrowserRouter basename="/hispano" >

En package.json

 "homepage": "http://http://18.220.188.121/hispano/",

Tambien reemplazo

  "browserslist": {

    "production": [

      ">0.2%",

      "not dead",

      "not op\_mini all"

    ],

    "development": [

      "last 1 chrome version",

      "last 1 firefox version",

      "last 1 safari version"

    ]

Por

  "browserslist":   []

En store.js (redux)

/\*

if (window.navigator.userAgent.includes("Chrome")) {

  store = createStore(

    rootReducer,

    initalState,

    compose(

      applyMiddleware(...middleware),

      window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_\_ &&

        window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_\_()

    )

  );

} else {

  store = createStore(

    rootReducer,

    initalState,

    compose(applyMiddleware(...middleware))

  );

}\*/

store = createStore(

  rootReducer,

  initalState,

  compose(applyMiddleware(...middleware))

);

Para desarrollo se utiliza la parte comentada, que es para poder usar la Redux Dev Tool y ver los estados.

Luego de todo esto:

npm run build

Copiar datos de carpeta generada build (en directorio del repo) al servidor (en este caso el apache en <http://18.220.188.121>) (no en tomcat, esto es estatico, sin java)