

Rapport

Lien Git vers le projet : <https://github.com/davide-col/projetJPA>

Les fonctionnalités du projet

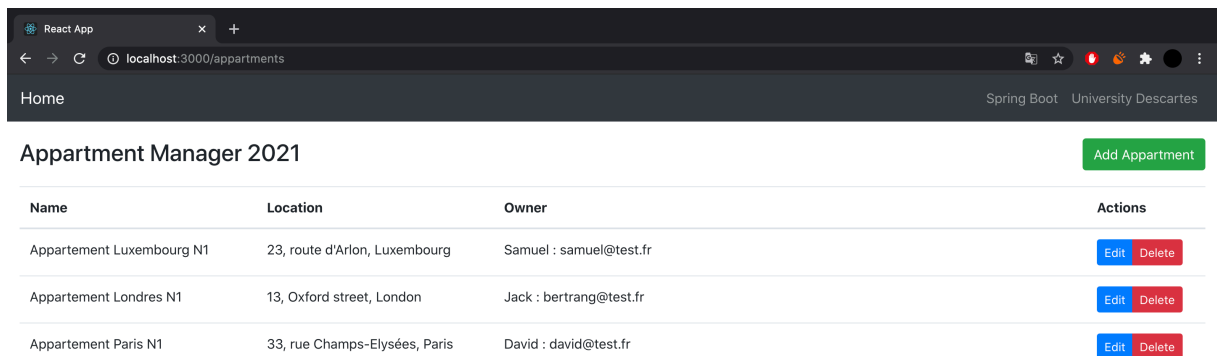
Page d'accueil contenant le lien vers le site de gestion des appartements.



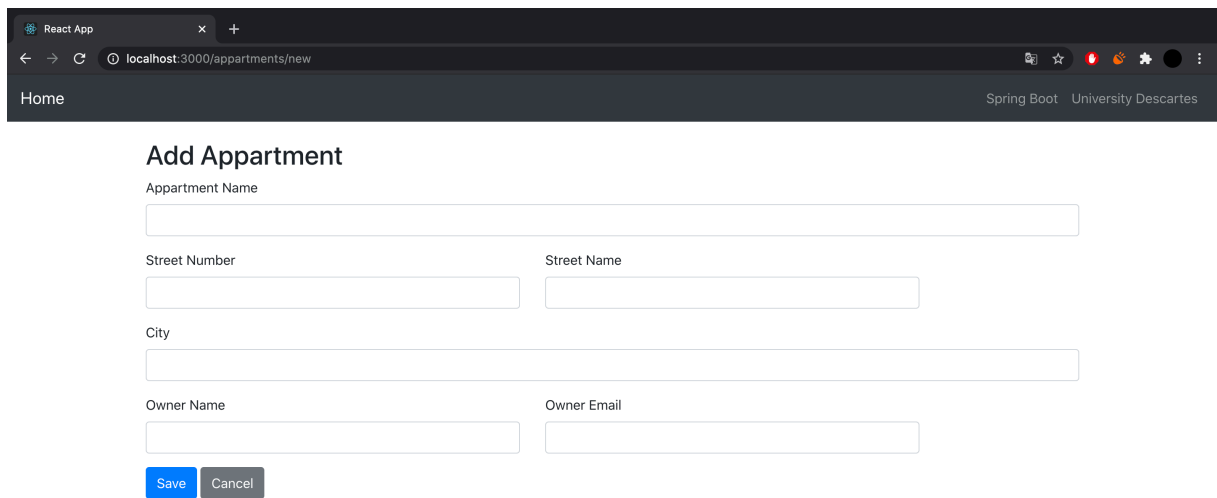
Page de gestion des appartements accédant à la base de données.

Plusieurs options :

- Edit : Modifier la ligne de donnée.
- Delete : Supprimer la donnée.
- Add Appartment : Ajout d'un nouvel appartement.



Page d'ajout d'appartement :

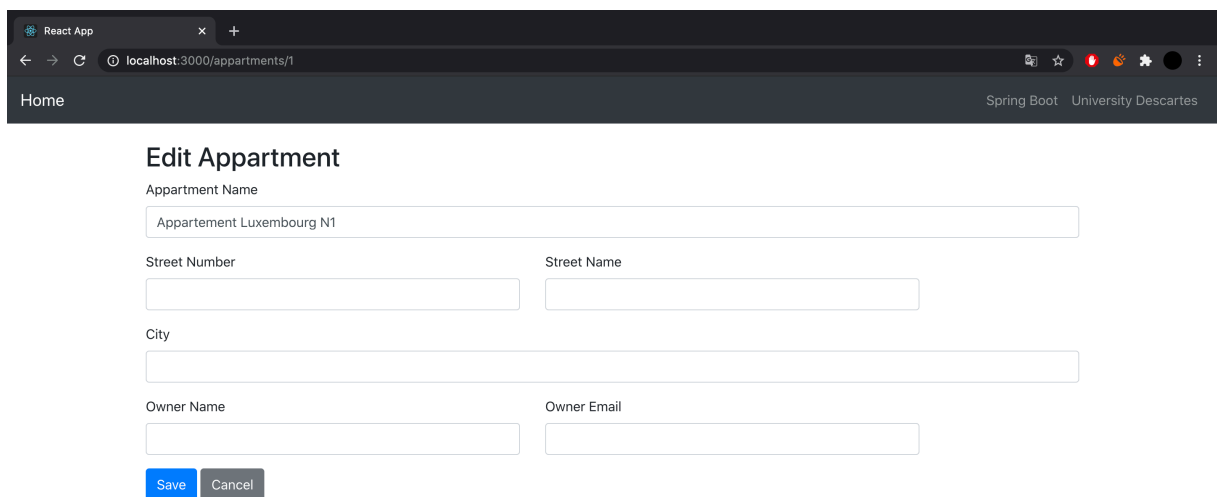


The screenshot shows a web browser window with the address bar at `localhost:3000/apartments/new`. The page title is "Add Apartment". The form contains the following fields:

- Apartment Name (text input)
- Street Number (text input)
- Street Name (text input)
- City (text input)
- Owner Name (text input)
- Owner Email (text input)

At the bottom of the form are two buttons: "Save" (blue) and "Cancel" (grey).

Page de modification d'appartement :

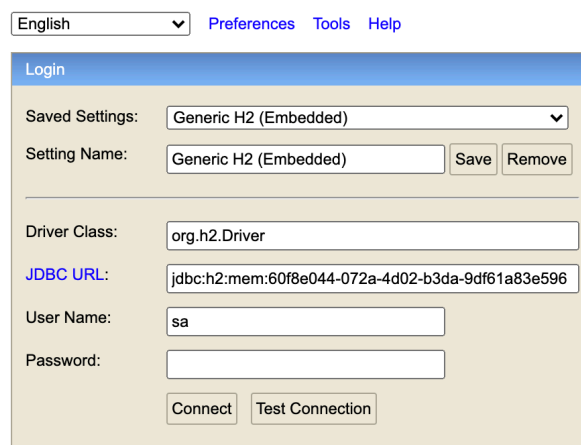


The screenshot shows a web browser window with the address bar at `localhost:3000/apartments/1`. The page title is "Edit Apartment". The form contains the following fields:

- Apartment Name (text input, containing "Appartement Luxembourg N1")
- Street Number (text input)
- Street Name (text input)
- City (text input)
- Owner Name (text input)
- Owner Email (text input)

At the bottom of the form are two buttons: "Save" (blue) and "Cancel" (grey).

Site vers la base de données : <http://localhost:8080/h2-console>



The screenshot shows the H2 database console login page. At the top, there is a language dropdown set to "English" and links for "Preferences", "Tools", and "Help". The main section is titled "Login" and contains the following fields and buttons:

- Saved Settings: Generic H2 (Embedded) (dropdown)
- Setting Name: Generic H2 (Embedded) (text input) with "Save" and "Remove" buttons
- Driver Class: org.h2.Driver (text input)
- JDBC URL: jdbc:h2:mem:60f8e044-072a-4d02-b3da-9df61a83e596 (text input)
- User Name: sa (text input)
- Password: (text input)
- Buttons: "Connect" and "Test Connection"

Après connexion, interface accédant à la BDD.

The screenshot shows the H2 Console web interface in a browser. The address bar indicates the URL is `localhost:8080/h2-console/login.do?jsessionid=59ee60a67066f6005e9c1897f340ca8b`. The interface includes a toolbar with options like 'Auto commit', 'Max rows: 1000', 'Auto complete', and 'Auto select'. On the left, a sidebar lists database components: 'jdbc:h2:mem:60f8e044-072a-4dc...', 'ADDRESS', 'APPARTMENTS', 'CLIENT', 'INFORMATION_SCHEMA', 'Sequences', 'Users', and 'H2 1.4.200 (2019-10-14)'. The main area displays the SQL statement `SELECT * FROM ADDRESS` and its results in a table format.

SQL statement:

```
SELECT * FROM ADDRESS
```

Results:

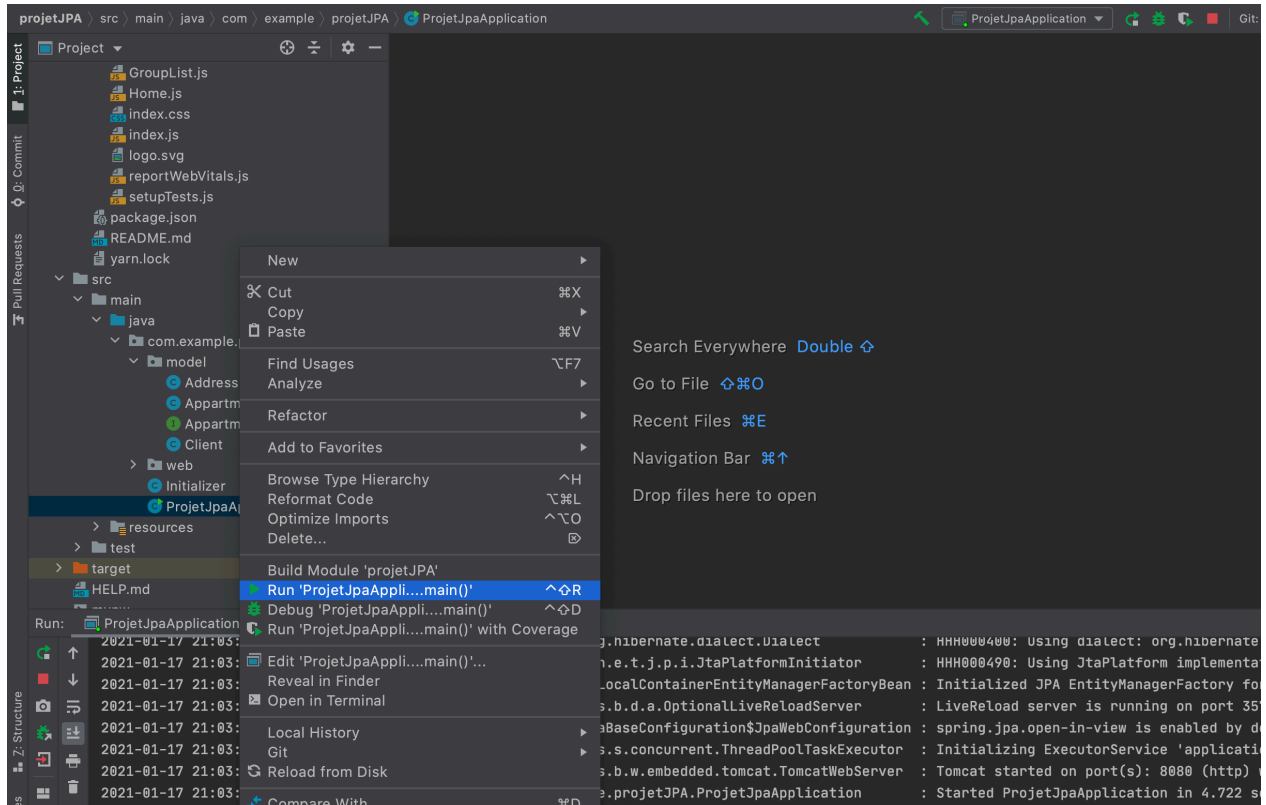
ID	CITY	STREET_NAME	STREET_NUMBER
2	Luxembourg	route d'Arlon	23
5	London	Oxford street	13
8	Paris	rue Champs-Élysées	33

(3 rows, 4 ms)

Edit

Manuel de démarrage

- Ouvrir le projet avec IntelliJ ou Eclipse.
- Dans notre exemple IntelliJ, ouvrir le projet.
- Parcourir le fichier ProjetJpaApplication.
- Faire une Click droit puis Run.



OU

- Dans un terminal, se mettre à la source du projet
- Executer la commande à la racine du projet : `./mvnw spring-boot:run`

```
davidecolaci@MacBook-Pro-de-Davide projetJPA % ./mvnw spring-boot:run
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.example:projetJPA >-----
[INFO] Building projetJPA 0.0.1-SNAPSHOT
[INFO] -----[ jar ]-----
[INFO]
[INFO] >>> spring-boot-maven-plugin:2.4.1:run (default-cli) > test-compile @ projetJPA >>>
[INFO]
[INFO] --- maven-resources-plugin:3.2.0:resources (default-resources) @ projetJPA ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Using 'UTF-8' encoding to copy filtered properties files.
[INFO] Copying 1 resource
[INFO] Copying 0 resource
[INFO] The encoding used to copy filtered properties files have not been set. This means that the s
ng other filtered resources. This might not be what you want! Run your build with --debug to see wh
s/maven-resources-plugin/examples/filtering-properties-files.html
```

- Dans un terminal, se mettre dans le répertoire /app du projet.
- Lancer la commande : `yarn install`
- Lancer la commande : `yarn start`
- Un navigateur devrait automatiquement s'ouvrir à l'adresse : <http://localhost:3000/> donnant accès au site internet du projet fait en react.

Accès à la base de données : <http://localhost:8080/h2-console>

- Copier le lien JDBC URL affiché dans la console : *H2 console available at '/h2-console'. Database available at 'jdbc:h2:mem:17a4b945-2575-49dd-a748-f6521ad28237'*
- Cliquer sur "Connect"

- Écrire des requêtes SQL sur les tables pour analyser les données.
- Cliquer sur le bouton `run`.

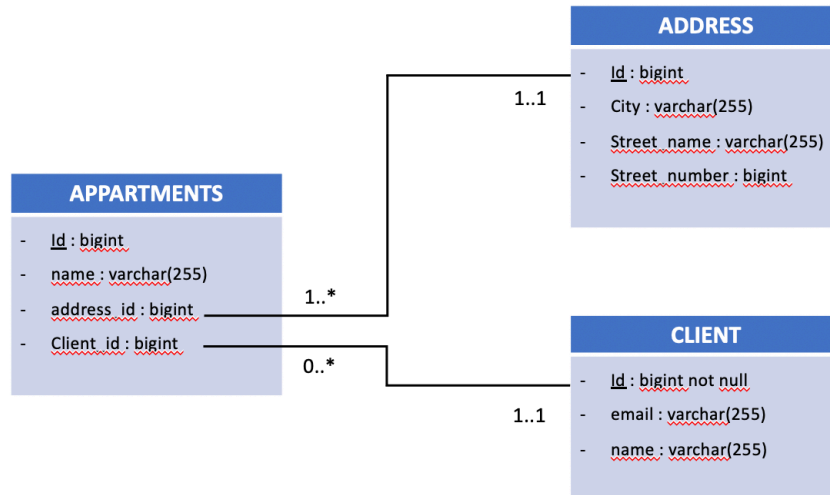
jdbc:h2:mem:17a4b945-2575-49... | ☒ Auto commit | Max rows: 1000 | Auto complete

Run Run Selected Auto complete Clear SQL statement:

+ ADDRESS
+ APPARTMENTS
+ CLIENT
+ INFORMATION_SCHEMA
+ Sequences
+ Users
H2 1.4.200 (2019-10-14)

```
SELECT * FROM ADDRESS
```

Structure de la base de données



← Completed Labs

Create and Manage Cloud Resources: Challenge Lab

Incomplete, Assessment 50%

?

✓	Managing Deployments Using Kubernetes Engine	
✓	Introduction to Docker	
✗	Introduction to Docker	Incomplete
✓	Cloud Endpoints: Qwik Start	
✓	Google Assistant: Qwik Start - Dialogflow	
✓	Cloud Security Scanner: Qwik Start	
✓	Cloud Security Scanner: Qwik Start	
✓	Video Intelligence: Qwik Start	Completed 13 days ago
✓	Cloud Functions: Qwik Start - Command Line	Completed 13 days ago
✓	Data Loss Prevention: Qwik Start - JSON	Completed 13 days ago
✓	Cloud SQL for MySQL: Qwik Start	Completed 13 days ago
✓	Datastore: Qwik Start	Completed 13 days ago

Davide Colaci

dav.colaci@gmail.com

2954 Credits

Settings

Sign Out

Privacy · Terms

← Completed Labs

Cloud SQL for MySQL: Qwik Start

?

✓	Cloud SQL for MySQL: Qwik Start	
✓	Datastore: Qwik Start	
✓	Container-Optimized OS: Qwik Start	
✓	Cloud Source Repositories: Qwik Start	
✓	Google Cloud SDK: Qwik Start - Redhat/Centos	
✓	App Engine: Qwik Start - Python	
✓	Google Cloud SDK: Qwik Start - Redhat/Centos	
✓	Google Cloud SDK: Qwik Start - Redhat/Centos	
✓	Set Up Network and HTTP Load Balancers	Completed 13 days ago
✓	Kubernetes Engine: Qwik Start	Completed 13 days ago
✓	Getting Started with Cloud Shell and gcloud	Completed 13 days ago
✓	Creating a Virtual Machine	Completed 13 days ago
✓	A Tour of Qwiklabs and Google Cloud	Completed 3 months ago

Davide Colaci

dav.colaci@gmail.com

2954 Credits

Settings

Sign Out

Privacy · Terms

Conclusion

Dans le cadre de ce projet j'ai appris à développer un projet Spring Boot facilitant certaines commodités avec ses outils de programmation comme notamment Spring Boot JPA et Spring Boot Web.

A l'aide de JPA, le code back end se compose d'une base de données in-memory permettant de stocker et modifier les données directement de l'application.

Un Web Service a été créé à partir de l'outil Spring Boot Web, permettant la communication et l'échange de données entre l'application et systèmes hétérogènes dans un environnement distribué. Pour le front end, le projet a été développé en React, outil facilitant la création d'interfaces.

Pour l'amélioration de ce projet, il aurait pu être itéré avec l'intégration d'un message broker et/ou cloud suite à une containerisation Docker. Ce dernier permettant de facilement transporter le projet d'une architecture à une autre.