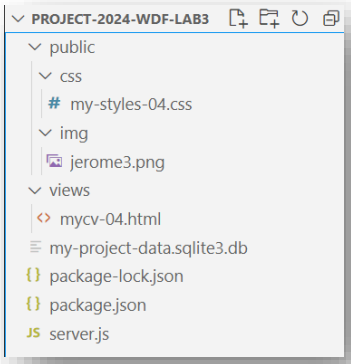
Lab 3: Node, Express, Handlebars

This lab session is dedicated to using Node, Express, and Handlebars in your project. You should have a look at the lectures about Node, Express, JSON, and Handlebars on Canvas. During the labs, a "projects and skills" database will be used. To try and test some code, you can use the data I provide for the labs, but at the end, the database of your project MUST use YOUR (OWN) DATA!

1) Node, Express, Handlebars

During workshop 2, you created your project directory and populated it with Node, Express, and SQLite3. We will start with the existing code and adapt it to use Handlebars in our project.

The starting directory structure is the one from workshop 2:



Inside your "project/" directory (you name it as you want):
a) You have a "public/" directory containing "css/" and "img/" subdirectories. In these, you will respectively put your css, and image files. The "public/" directory is declared as "static" in your Express application meaning that it can be publicly accessed.
b) You also have a "views" directory where you find your HTML pages.
c) The "server.js", "package.json" and "package-lock.json" files are directly in the project directory (not in any subdirectory).

This structure needs to change to use handlebars in your project.

1) Install the package "express-handlebars" in your project directory using npm.

2) Add the code to use "express-handlebars" in your code:

```
const { engine }=require('express-handlebars') // load the handlebars package for express
```

3) Add the declaration of handlebars as your view engine in Express:

```
// HANDLEBARS
app.engine('handlebars', engine()) // initialize the engine to be handlebars
app.set('view engine', 'handlebars') // set handlebars as the view engine
app.set('views', './views') // define the views directory to be ./views
```

4) Modify the default route / to use the render() function from handlebars

```
// create the (default) route /
app.get('/', function (req, res) {
    //res.send('I am Jerome...    :)')
    res.render('home.handlebars')
})
```

5) Create the "views/layouts/" directory:

```
mkdir views/layouts/
```

6) Create the “views/layouts/main.handlebars” file:

views > layouts > main.handlebars > ...

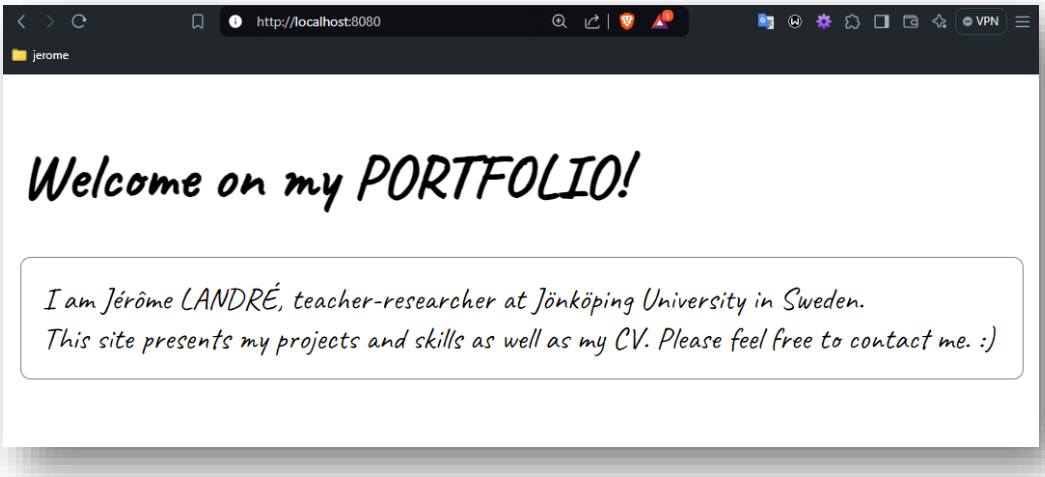
```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <link rel="stylesheet" href="/css/my-styles-04.css">
        <title>Jerome's Portfolio</title>
    </head>
    <body>
        {{{body}}}
    </body>
</html>
```

This file is **IMPORTANT**, it is the base of your **one-page** site! The CSS you will use in this file will be applied to your whole site because it is a one-page site. The routes will use the render() function from handlebars and render a code inside the “{{{body}}}” part of the page. Please rename the title of your site and the stylesheet with your words. It means you will load only the “{{{body}}}” part depending on the route and not reload the whole page each time something changes: you save energy and time on your dynamic website!

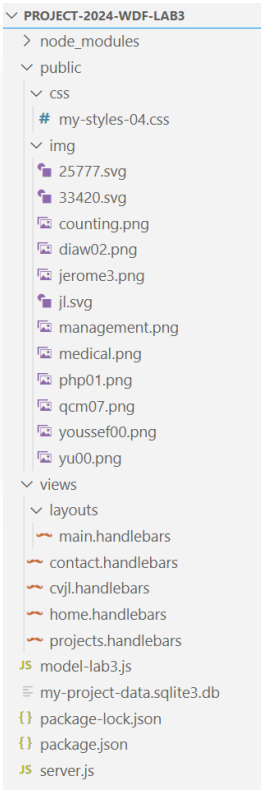
7) Create the “views/home.handlebars” file:

```
<div id="principal">
    <h1>Welcome on my PORTFOLIO!</h1>
    <div class="mytextcard">
        I am Jérôme LANDRÉ, teacher-researcher at Jönköping University in Sweden.
        <br />
        This site presents my projects and skills as well as my CV. Please feel free to contact me.   :)
    </div>
</div>
```

8) Test your server by running “node server.js” in the terminal, open your browser on “http://localhost:8080”:



Your new project directory structure should be similar to the one on the right (with your own images and css files, of course) ---------->



2) Menu

Please, add a menu to your website (the questions are where and why should you do that?)

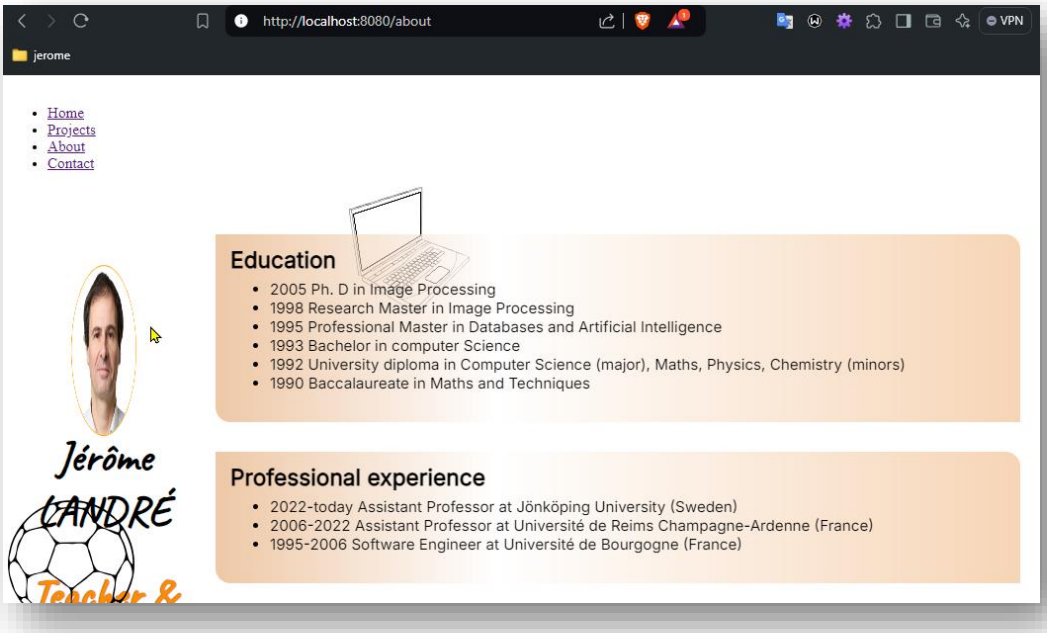
```
<header>
    <nav>
        <ul>
            <li><a href="/">Home</a></li>
            <li><a href="/projects">Projects</a></li>
            <li><a href="/about">About</a></li>
            <li><a href="/contact">Contact</a></li>
        </ul>
    </nav>
</header>
```

Add a “/about” route that will send back your CV to the browser on the GET request on route “/about”:

```
// create a new route to send back my CV
app.get('/about', function (req, res) {
    //res.sendFile(__dirname+'/views/mycv-04.html')
    res.render('cvjl.handlebars')
})
```

Of course, you need to create the file “views/cvjl.handlebars” (which is just renaming your CV file if you did lab2), easy!

Try your “/about” route:

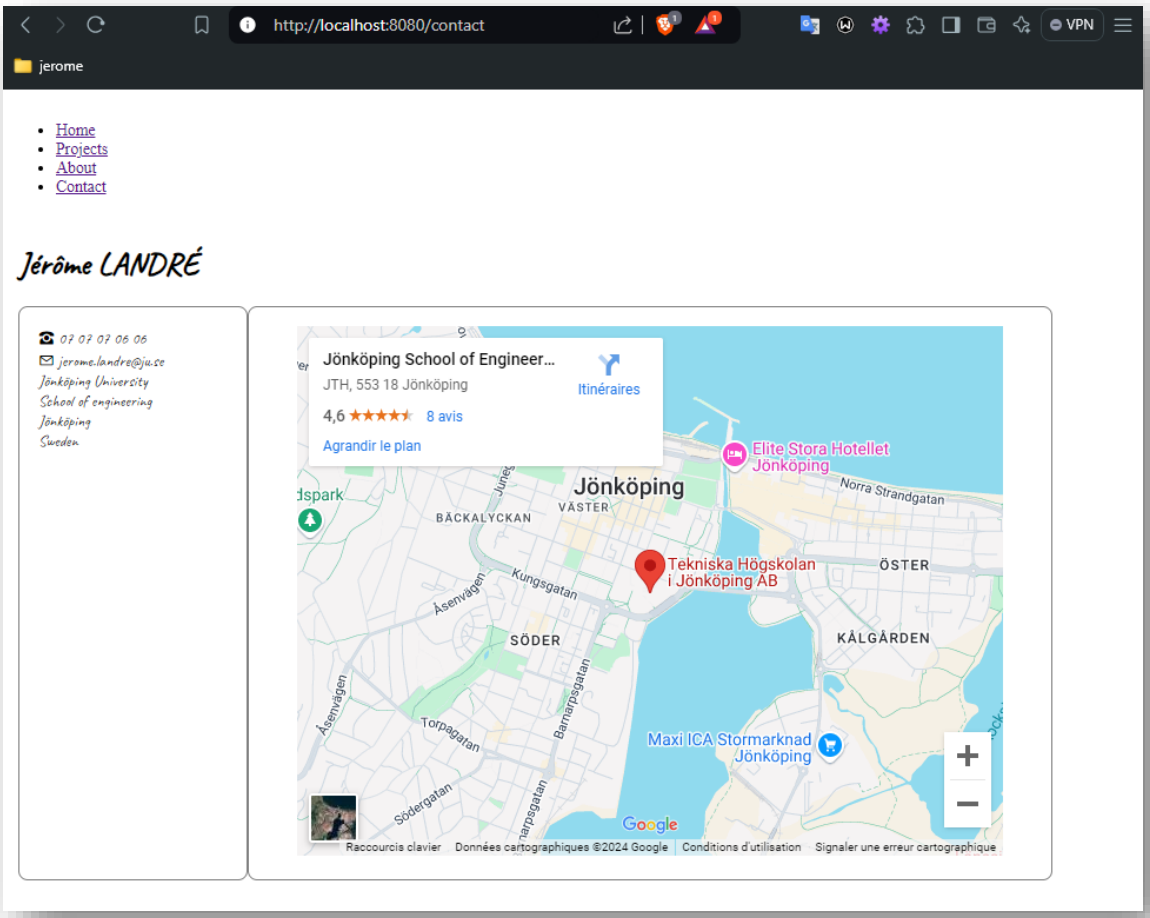


Remark: you will perhaps need to modify your CSS file, due to the integration of your CV into an existing page, your one-page project site…

3) Contact page

Please, add a contact page with information about your email, phone, and other related data (you can use fake data if you don't want to put your information online). I use fake information for my phone but my email is the correct one for instance.

In "server.js", add a "/contact" route that will render the contact information.

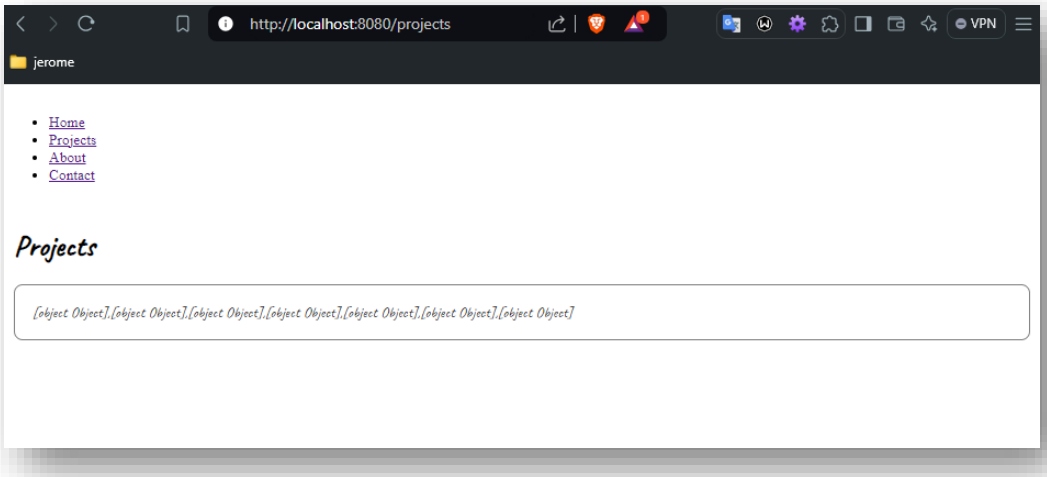


5) The "projects" page

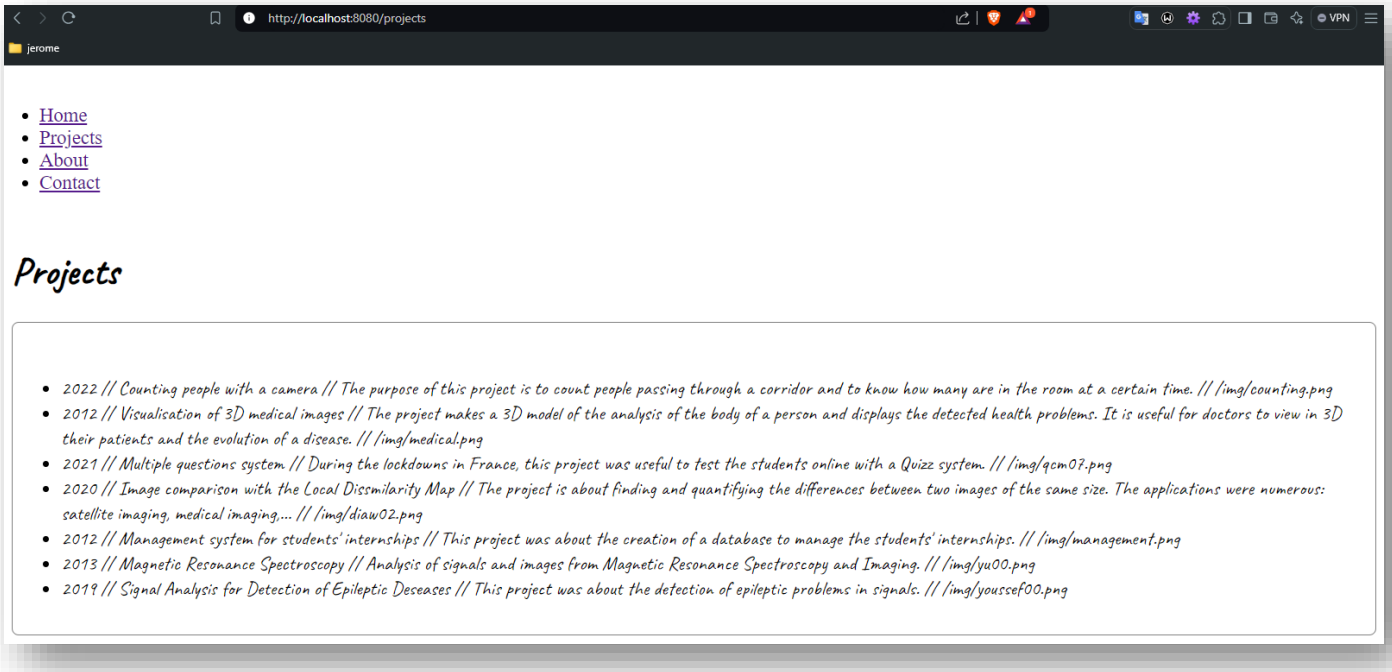
Today, I will not use the data extracted from the database (I kept this for next week…) but a JSON variable containing my projects to understand the model in the Model-View-Controller (MVC) pattern. Add the "projects" variable in your code, what is the type of this variable? For this type of variable, do you know the 7 characters used to build it, and their meaning?

```
// MODEL
const projects=[
    { "id":"1", "name":"Counting people with a camera", "type":"Research", "desc": "The purpose of this project is to count people
    passing through a corridor and to know how many are in the room at a certain time.", "year": 2022, "dev":"Python and OpenCV
    (Computer vision) library", "url":"/img/counting.png" },
    { "id":"2", "name":"Visualisation of 3D medical images", "type":"Research", "desc": "The project makes a 3D model of the
    analysis of the body of a person and displays the detected health problems. It is useful for doctors to view in 3D their
    patients and the evolution of a disease.", "year": 2012, "url":"/img/medical.png" },
    { "id":"3", "name":"Multiple questions system", "type":"Teaching", "desc": "During the lockdowns in France, this project was
    useful to test the students online with a Quizz system.", "year": 2021, "url":"/img/qcm07.png" },
    { "id":"4", "name":"Image comparison with the Local Dissmilarity Map", "desc": "The project is about finding and quantifying the
    differences between two images of the same size. The applications were numerous: satellite imaging, medical imaging,...",
    "year": 2020, "type":"Research", "url":"/img/diaw02.png" },
    { "id":"5", "name":"Management system for students' internships", "desc": "This project was about the creation of a database to
    manage the students' internships.", "year": 2012, "type":"Teaching", "url":"/img/management.png" },
    { "id":"6", "name":"Magnetic Resonance Spectroscopy", "desc": "Analysis of signals and images from Magnetic Resonance
    Spectroscopy and Imaging.", "year": 2013, "type":"Research", "url":"/img/yu00.png" },
    { "id":"7", "name":"Signal Analysis for Detection of Epileptic Deseases", "desc": "This project was about the detection of
    epileptic problems in signals.", "year": 2019, "type":"Research", "url":"/img/youssef00.png" },
]
```

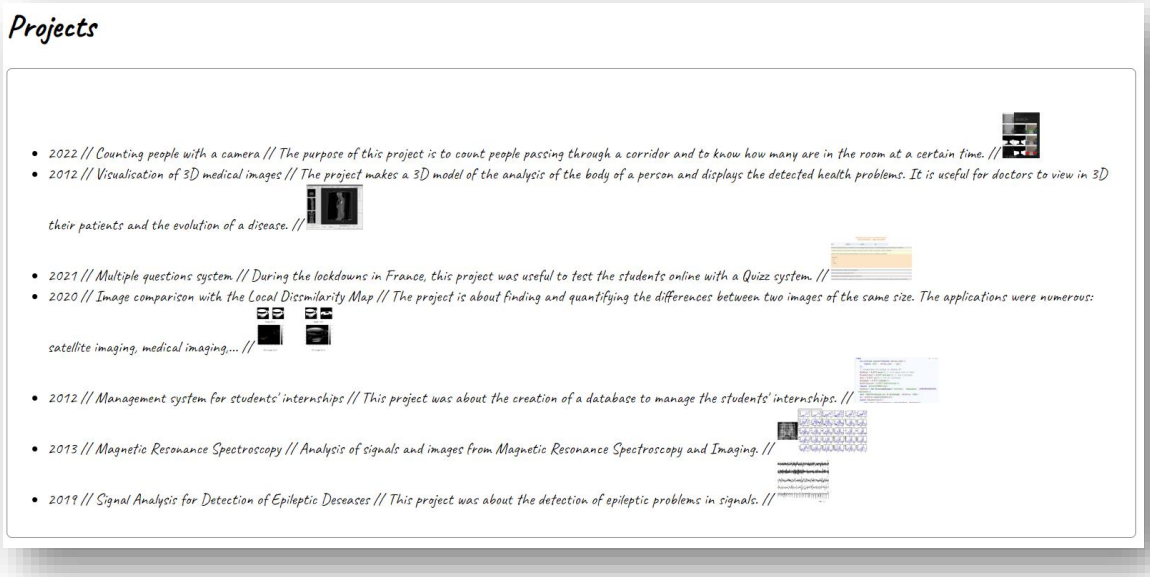
Add a “/projects” route into “server.js” and render a file “projects.handlebars” with, at first, all the projects written as the raw data “projects”:



Using the “#each” helper of handlebars,



Then, add the image of each project (change the CSS to make it small), not only the URL of it: use the HTML tag. Of course, all the images must be in the “public/img/” directory!



Use the “#if” helper to color the year of “Research” projects in “orange” and the year of “Teaching” projects in mauve. You must use some CSS to define two classes and apply them with “#if” depending on the type of each project.

You need to add some code to your “server.js” before using the “eq” helper in your code: please read the solution from source: (Stack overflow, https://stackoverflow.com/questions/34252817/handlebarsjs-check-if-a-string-is-equal-to-a-value, 2018)

```
// HANDLEBARS
// handlebars "ifeq" and "ifnoteq" from source: (Pablo, varando, https://stackoverflow.com/questions/34252817/handlebarsjs-check-if-a-string-is-equal-to-a-value, 2018)

app.engine('handlebars', engine({helpers: {
    eq (a,b) { return a==b; }
}})) // initialize the engine to be handlebars
app.set('view engine', 'handlebars') // set handlebars as the view engine
app.set('views', './views') // define the views directory to be ./views
```

views > projects.handlebars > ...

```
1        <div id="principal">
2            <h1>Projects</h1>
3            <div class="mytextcard">
4                <ul>
5                    {{#each projects}}
6                        <li>
7                            {{#if (eq type "Research")}}
8                                <span class="proj-research">
9                            {{else}}
10                               <span class="proj-teaching">
11                           {{/if}}
12                               {{year}}
13                           </span>
14                           // {{name}} // {{desc}} // <img src="{{url}}" alt="{{url}}" class="smallimage" /></li>
15                   {{/each}}
16               </ul>
17           </div>
18       </div>
19
```

```
.proj-teaching {
    color: DarkOrange;
    font-weight: bold;
}

.proj-research {
    color: BlueViolet;
    font-weight: bold;
}
```

Projects

- 2022 // Counting people with a camera // The purpose of this project is to count people passing through a corridor and to know how many are in the room at a certain time. // 
- 2012 // Visualisation of 3D medical images // The project makes a 3D model of the analysis of the body of a person and displays the detected health problems. It is useful for doctors to view in 3D their patients and the evolution of a disease. // 
- 2021 // Multiple questions system // During the lockdowns in France, this project was useful to test the students online with a Quizz system. // 
- 2020 // Image comparison with the Local Dissimilarity Map // The project is about finding and quantifying the differences between two images of the same size. The applications were numerous: satellite imaging, medical imaging,... // 
- 2012 // Management system for students' internships // This project was about the creation of a database to manage the students' internships. // 
- 2013 // Magnetic Resonance Spectroscopy // Analysis of signals and images from Magnetic Resonance Spectroscopy and Imaging. // 
- 2019 // Signal Analysis for Detection of Epileptic Deceases // This project was about the detection of epileptic problems in signals. //