Web and Database Programming — The Backend

Follow the steps below to set up your backend. You may use any of my code.

Code from Class: https://github.com/kaitlinchoffmann/SP25WDP

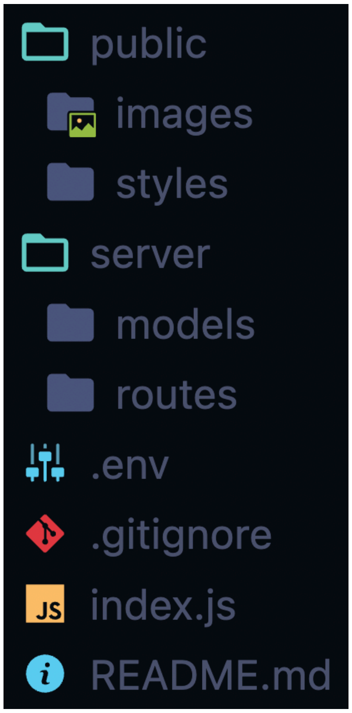
Download Node.js:

- Node.js: https://nodejs.org/en/download/
  - Once downloaded, node package manager (npm) will automatically be installed allowing you to install a plethora of different libraries (such as nodemon, express, MySQL, etc).

Complete the following:

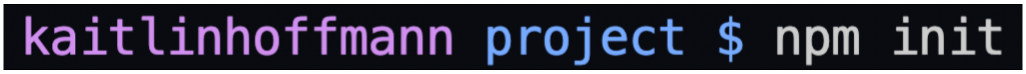
1. Inside your project folder, set up the initial structure. The following screenshot is what your project folder should contain. *public*, *images*, *styles*, *server*, *models* and *routes* are **folders**. *.env*, *.gitignore*, *index.js*, and *README.md* are **files**.

   - All of your current front-end files (html, css, images, scripts, etc.) should be placed in **public**. There should not be any frontend/client-side files outside of this folder.



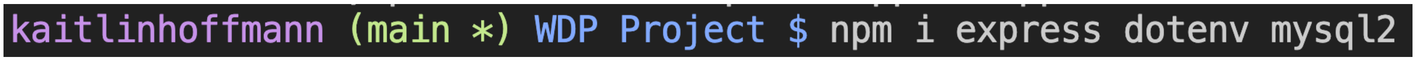
```
public
  images
  styles
server
  models
  routes
.env
.gitignore
index.js
README.md
```

2. Open the terminal/command prompt and go to your project folder if you are not already inside it. Initialize a new node js project using the command **npm init**. Follow each step by pressing enter/return on your keyboard. A **package.json** file should have been created automatically for you.



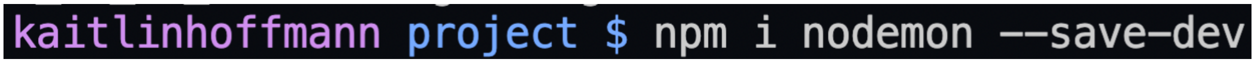
```
kaitlinhoffmann project $ npm init
```

3. Install **express**, **dotenv**, and **mysql2** using the command, **npm i**:



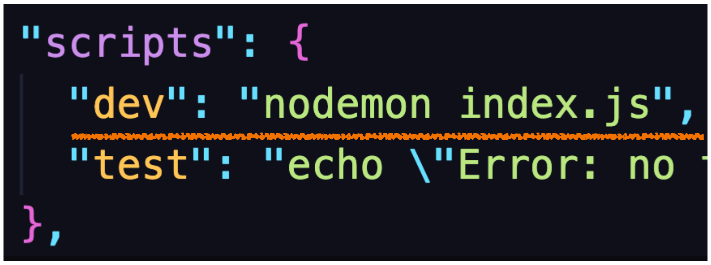
```
kaitlinhoffmann (main *) WDP Project $ npm i express dotenv mysql2
```

4. Install nodemon as a dev dependency using the following command:



```
kaitlinhoffmann project $ npm i nodemon --save-dev
```

5. Inside your package.json file, add the following line under **scripts**:



```
"scripts": {
  "dev": "nodemon index.js",
  "test": "echo \"Error: no 
},
```

- This is what will allow you to run your server (see step 8). In order to run a script, you use the command *npm run <scriptName>.*

6. Inside your **index.js** file, add the following code. Your index.js file is the entry point of your server. Refer back to the lecture for the importance of each part. If your user entity is another name, such as customer or client, be sure to edit my code to reflect that. Be sure to change ***recipeRoutes*** (and anything pertaining to my recipe entity) to your second entity: https://github.com/kaitlinchoffmann/SP25WDP/blob/main/index.js

7. Create JavaScript file in the models and routes folders for at least two entities (User/Customer/etc. should be one of your entities).

    a. In your models folder, create your **db_connect.js** file and add the following code to connect your database (be sure to assign your environment variables in your .env file. See slides from module 9 if needed.): https://github.com/kaitlinchoffmann/FA24WDP/blob/main/server/models/db_connect.js

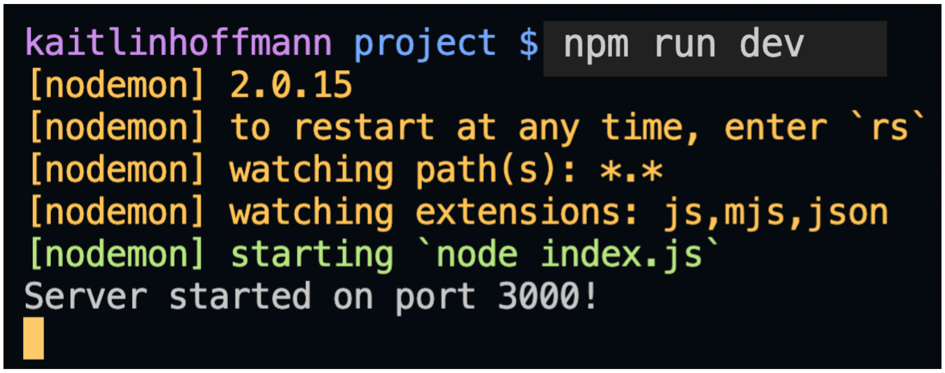
b. In the user/customer/client etc. file in the models folder add your MySQL table as shown in class. Create a function that accesses all of the entities in the table. Be sure to export the function so it is available for the corresponding routes file to access. See my code as an example: https://github.com/kaitlinchoffmann/SP25WDP/blob/main/server/models/user.js

c. In the user file in the routes folder, create one http method that calls the function in part b. See my code as an example: https://github.com/kaitlinchoffmann/SP25WDP/blob/main/server/routes/user.js

d. Repeat steps b and c for your second entity.

- **NOTE:** You may use my code from class; However, you should understand how to do this since you will need to set everything for your second entity on your own.

8. Attempt to run your server using the command: **npm run dev**. If you see the following, your server is set up correctly.



```
kaitlinhoffmann project $ npm run dev
[nodemon] 2.0.15
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
Server started on port 3000!
```

9. You can test out your future routes with Postman (not required, but I highly recommend!): https://www.postman.com/

10. Commit your changes and push to GitHub.

11. Submit your GitHub link for your project submission. Be sure to fix any issues on previous assignments to receive full points on your project!