Projet fil rouge frond-end

Vérification node.js installé avec node -v = ok version v18.16.0, pour avoir npm installé par défaut avec node.js.

Me met à la racine du projet (dossier fil rouge front-end) et installe bootstrap avec npm install bootstrap. Télécharge fichiers bootstrap et dépendances via npm, contenant bibliothèque et modules externe pour fonctionnement du projet. Création dossier node\_modules.

Initialisation du git pour commit de vs code sur git hub.

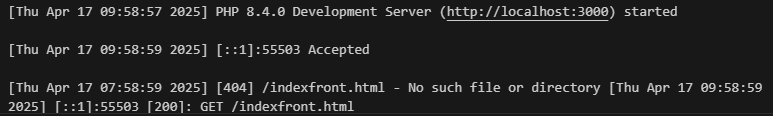




Création routage

Pour cela création dossier router, avec fichier Route.js, allRoutes.js et Router.js. Ici le code pour le routage n’est pas le plus important car c’est le style (html, css, sass) le plus important pour le front-end.

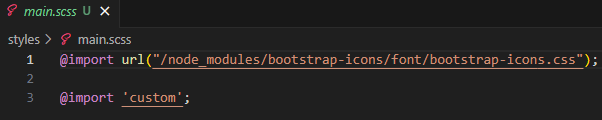
Téléchargement de PHP server pour voir la page. Fonctionne mais avec la route /indexfront.html.

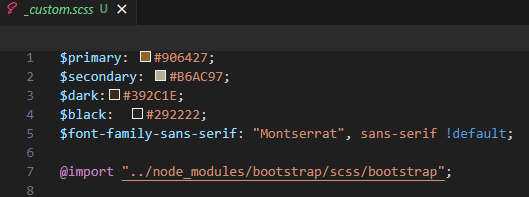


Création page home.html pour avoir affichage dynamique au milieu, entre header et footer.

Création pasge 404.html, qui s’affichera si le routage ne trouve pas la route demandé, et qui redirige vers la page d’accueil (/indexfront.html , qui affiche également home.html).

Création dossier scss(avant styles) avec fichier main.scss (principal) et \_custom.scss (pour customisation style, underscore devant pour dire qu’il ne doit pas être compilé) pour utiliser sass. Avec importation de chaque fichier dans chaque fichier pour que tout soit chargé.





Installation extension live sass compiler sur vs code. Qui permet d’avoir watch sass en bas de vs code, et en cliquant dessus, créer le fichier main.css et main.css.map. Cela permet du fichier de main.scss de mettre à jour le main.css. A chaque fois que j’ouvre et referme vs code, il faut bien penser à faire en sorte qu’il soit actif.

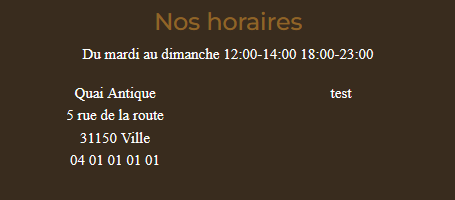
Penser à faire référence au fichier dans le fichier html.

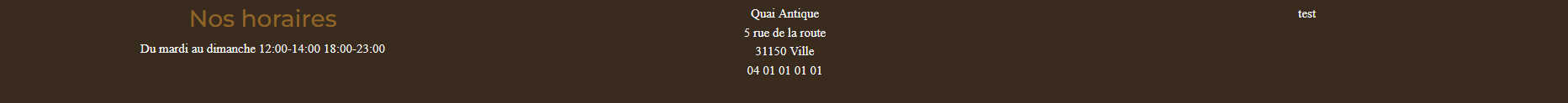
Création header, avec menu burger pour version mobile et en ligne pour version desktop.



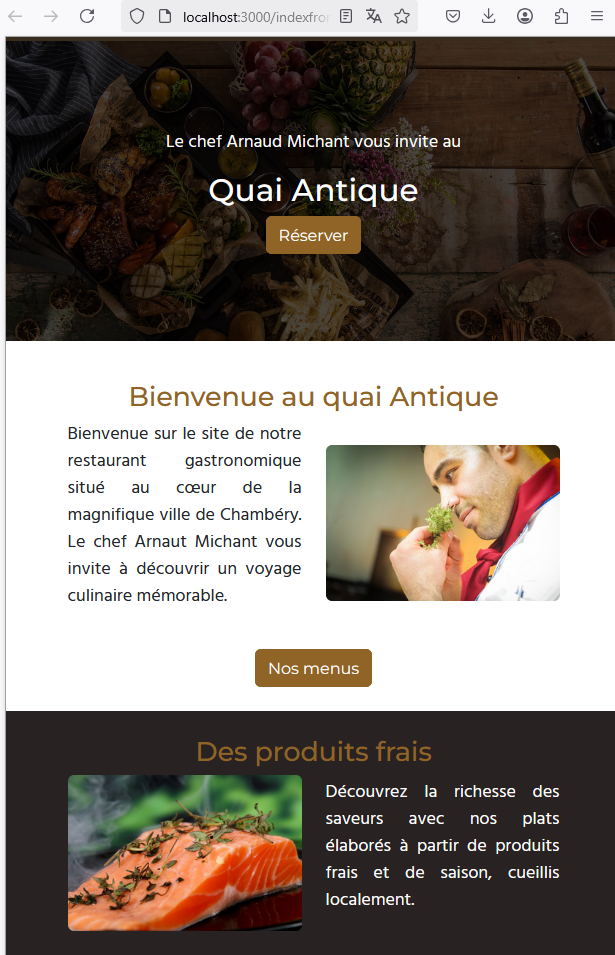


Création footer qui prend toute la largueur avec 12colonnes indiqué (sauf pour l’adresse et test qui sont mis sur 6 colonnes soit la moitié) , et un breakpoint pour les écrans larges. Pour les écrans >992px, un espace prendra seulement la place de 4colonnes sur les 12 disponible et donc dans la version bureau on a bien 3 espaces différents

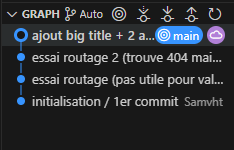




Modification fichier home.html, pour réaliser de la page d’accueil, en concordance avec la maquette réalisée. Big Title, pour le titre et l’image (dossier image). Puis création de 3 sections/bandeaux. Pour la mise en page, on donne la class et cela met du style sans passer par le fichier css (cela est du à bootstrap qui est un framework qui donne des classes CSS utilitaires). Avec bootstrap plusieurs choses sont définis par défaut comme les couleurs et avec sass ont redéfini cela. Les 2 ensembles sont top !!



Bien sur, commit sur git au fur et à mesure :



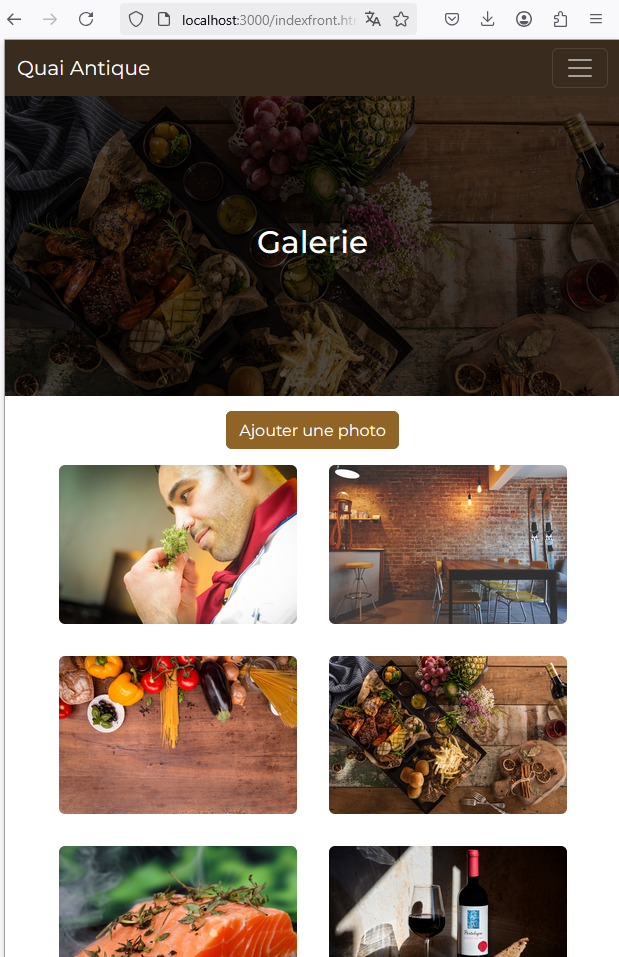
Création page galerie, rajout de la route dans le fichier allRoutes. Création fichier galerie.html.

Pour que la route fonctionne, il faut transformer tous les chemins relatifs en chemin absolu : href="/scss/main.css" /node\_modules/bootstrap/dist/js/bootstrap.bundle.min.js src="/Router/router.js

La page s’affiche tout est ok, pour la route/indexfront.html/galerie

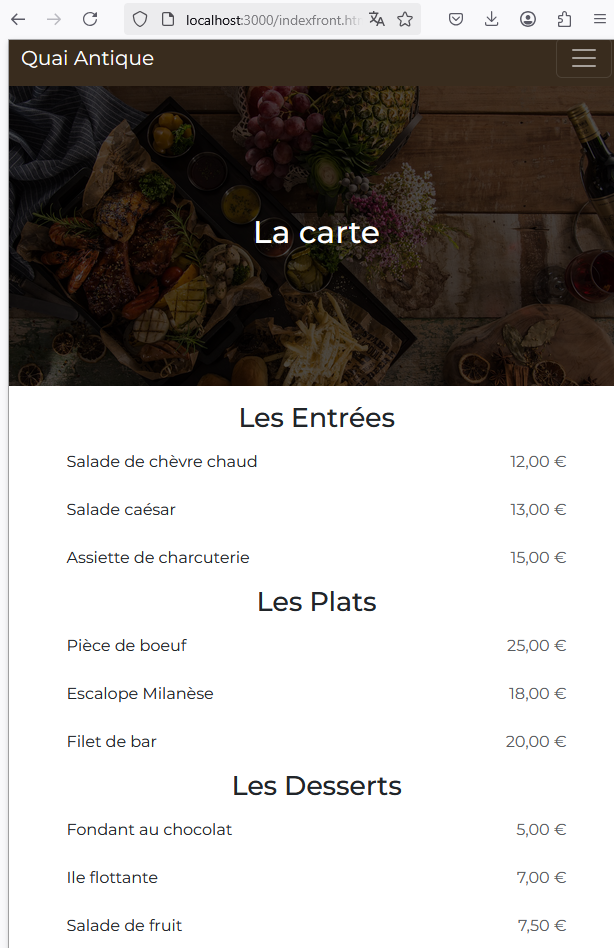
On remet le big title en début de page comme sur la home.html. Et ensuite on met un bouton ajouter photo, et les images que l’on souhaite. On crée un hover sur les images, avec un titre qui apparait, une image qui devient opaque. Pour crée une position absolue (choisie) pour le titre on doit d’abord donner une position relative au parent (ici image-card).

Il faut également voir pour modifier les liens dans le header pour accéder directement à la page choisie.

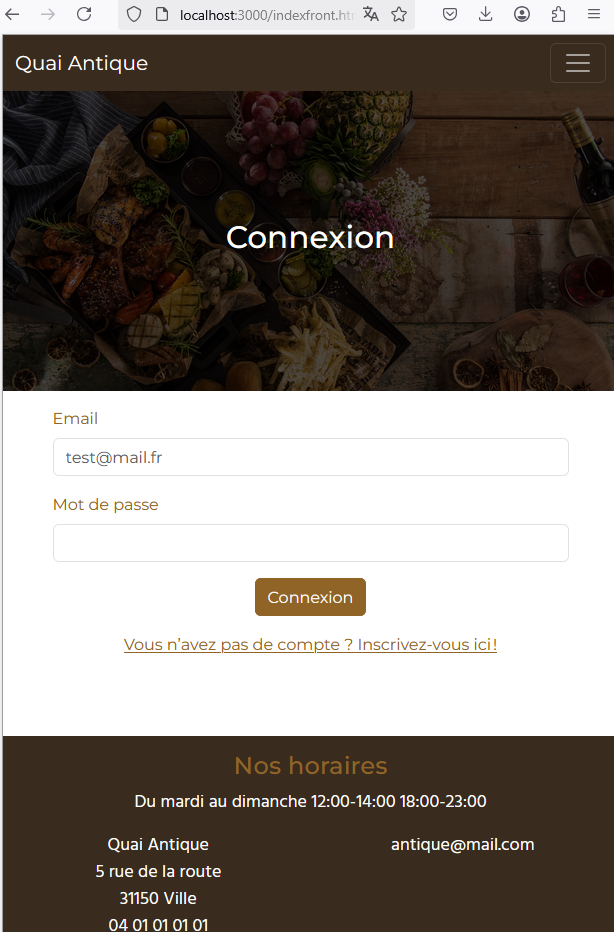


Push vers git !

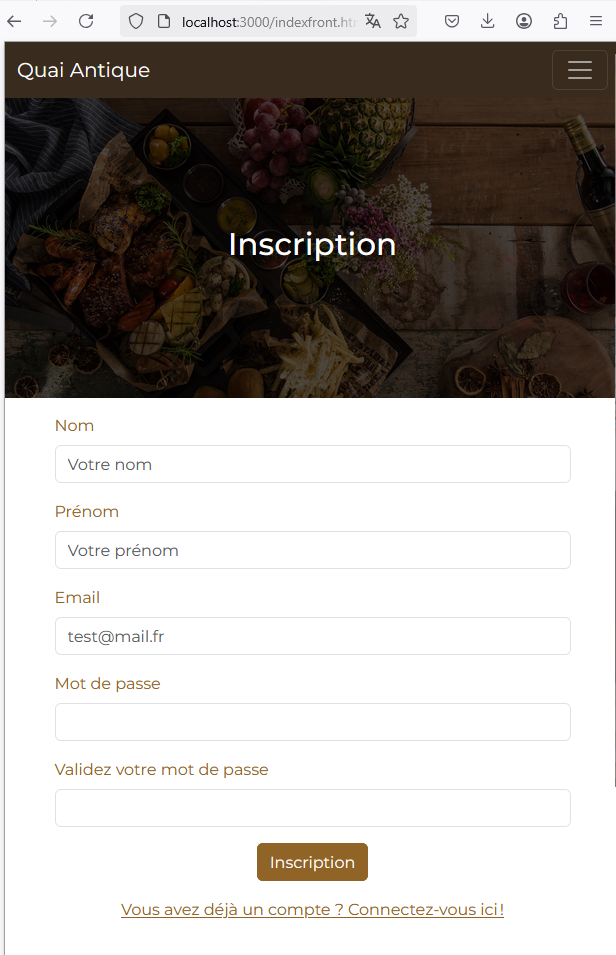
Création du fichier carte.html + ajout de la route, pour afficher la carte du restaurant. Avec 3 sections, entrées, plats, desserts. Mise en page avec bootstrap.



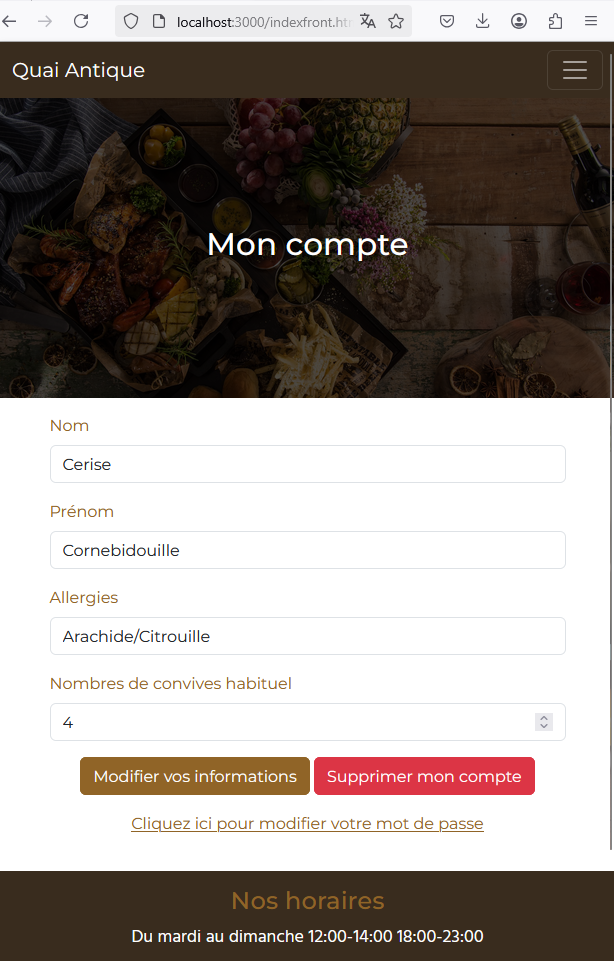
Réalisation page connexion et inscription : Pareil création route dans le fichier allRoutes.js. Création fichier signin.html dans le dossier auth, dans le dossier pages. Ajout du big title sur la page. Ajout du formulaire préfait sur bootstrap.



On refait la même chose pour la page inscription, mais on rajoute des champs

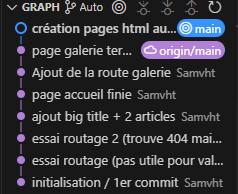


On fait la même chose pour la page mon compte (pas de maquette réalisée pour cette page). Les autres travaille avec les maquettes fourni par l’école. Mais on va faire ressembler aux 2 autres pages.

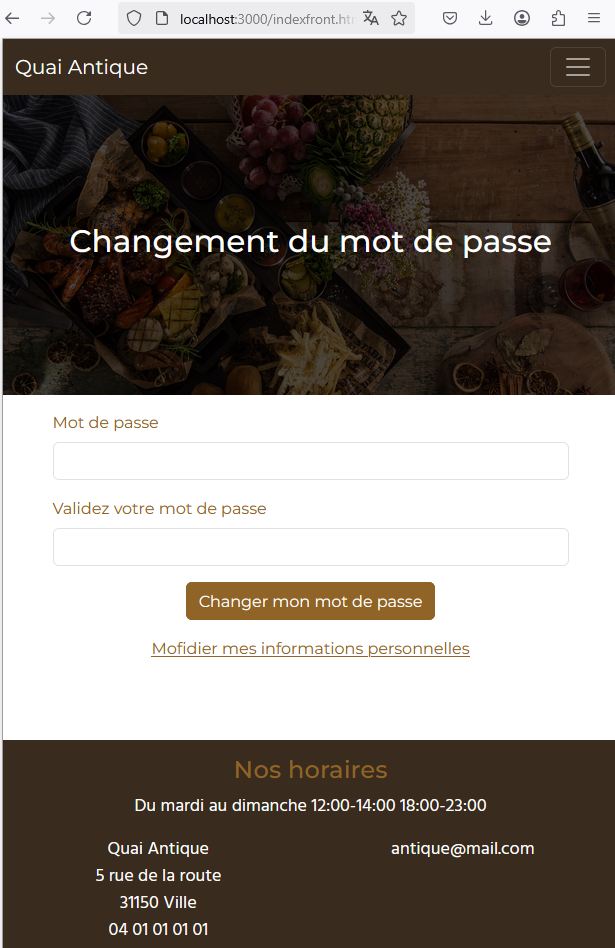


Les données sont simulées en mettant value, car nous ne sommes pas connectée à une BDD pour que celles-ci s’affichent.

Push sur git :

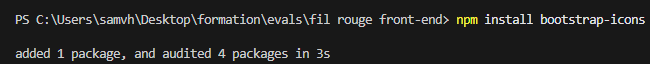


Création également de la page juste pour changer le mot de passe = editPassword. De la même manière, création route, création fichier editPassword.html

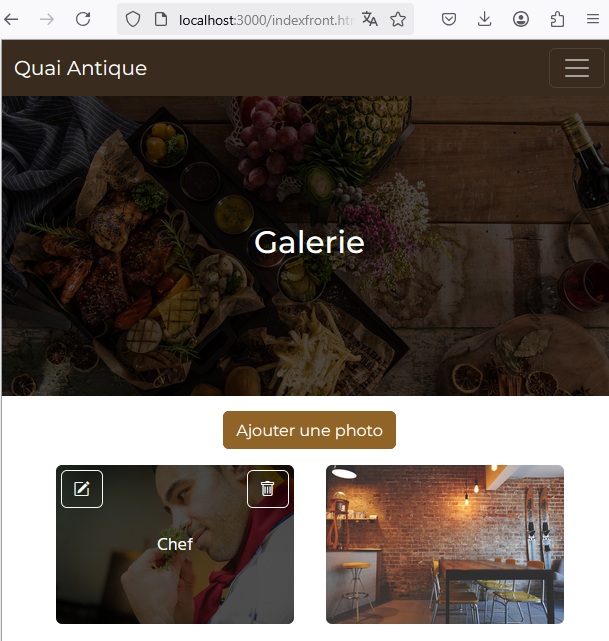


Sur le fichier de la galerie, ajout des bouton modification et suppression sur l’image, comme le tire. Apparition lors du hover sur l’image. Pour cela utilisation du modal de bootstrap, qui est une fonctionnalité qui permet de créer des fenêtres superposées(pop-in/mini fenêtre) sur une page web. Pour les logos supprimer et modifier, on trouve les class sur le site de bootstrap directement (poubelle = bi bi-trash).

A ce stade je me rends compte que j’ai fait un lien vers le dossier bootstrap-icons, mais que je n’ai pas ce dossier. Mais icones ne s’affichent donc pas. Pour cela je télécharge bootstrap-icons via npm.



Mes icones apparaissent, super ! maintenant création de la modal pour l’édition et la suppression de photo. Pour cela on trouve le code sur le site de bootstrap. C’est la partie JS de bootstrap qui permet de gérer la modale. Le model pour l’ajout et l’édition de la photo est le même. Pour la suppression on écrit en dur dans le code ce que l’on souhaite supprimer car pas relier à un back-end, donc ne peut pas récupérer les infos.



Création page mesResa.html, pour afficher toutes les réservations, pareil écrit en dur, car pas relié au back-end. Création de la route, et ajout du lien pour la navbar.

Création de la page reserver.html, avec création de la route et ajout du lien. Pareil des infos sont écrites en dur dans le code, toujours pour la même raison. Dans le formulaire, pour le nom et le prénom mis <input disabled> pour ne pas qu’ils soient modifiables, car récupérés de l’utilisateur connecté normalement. Pour le service, on fait des radios bouton. Pour le name, on met le même pour faire comprendre qu’il faut choisir soit l’un soit l’autre, si on met un différents, alors on peut cliquer sur les 2 boutons en même temps. Puis ajout du select

