Une image contenant texte, Police, capture d’écran, logo

Description générée automatiquement

Rapport de projet

Module : Nodejs

Enseignant : Membres de groupe :

M.DIENG Abdoulaye OUEDRAOGO Fabienne

SANOU Sou Roch Landry

Table des matières

[Présentation des modules de notre application 3](#_Toc143876681)

[A. Partie I : Test avec client REST 4](#_Toc143876682)

[I. Présentation des fonctionnalités 4](#_Toc143876683)

[1. Ajout d’un produit 4](#_Toc143876684)

[2. Afficher la liste des produits 5](#_Toc143876685)

[3. Modification d’un produit 6](#_Toc143876686)

[4. Suppression d’un produit 7](#_Toc143876687)

[B. Partie II : Test avec l’interface en utilisant le moteur de Template Jade 8](#_Toc143876688)

[I. Présentation des fonctionnalités 8](#_Toc143876689)

[1. Ajout d’un produit 8](#_Toc143876690)

[2. Affichage de la liste des produits 10](#_Toc143876691)

[3. Affichage des détails des produits 11](#_Toc143876692)

[4. Modification d’un produit 13](#_Toc143876693)

[5. Suppression d’un produit 15](#_Toc143876694)

[6. Autres fonctionnalités 16](#_Toc143876695)

[a. Barre navigation, page about et contact 16](#_Toc143876696)

# Présentation des modules de notre application

Pour ce projet, nous avons utilisé le Framework Express.js de node.js.

Notre travail comprend deux parties : une partie dédiée au test avec un client REST **Thunder Client** et une deuxième partie ou nous avons fait les tests avec des pages web. Les pages web ont été fait en utilisant le moteur de Template Jade.

Dans cette partie, il s’agit pour nous d’énumérer les modules que nous avons utilisés et de donner une description de chaque module. Ces modules ont été créés par nous-même.

* Module ConnexionDB : qui permet de se connecter à une base de données MongoDB en utilisant l’ORM mongoose ;
* Module Controlleur : c’est dans ce module que nous avons implémenté toutes méthodes permettant l’exécution du CRUD(Create, Read, Update, Delete) . Nous avons les méthodes CRUD pour effectuer les tests avec le client REST et les méthodes CRUD pour effectuer les tests sur les pages webs.
* Module Model : c’est un module contenant le schéma mongoose de notre base de données mongoDB et ce module permet également la création du modèle Produit à du schéma ;
* Module Router : Il permet le routage de notre application en définissant les routes. Nous avons les routes pour effectuer les tests avec le client REST et les routes pour effectuer les tests avec les pages webs.
  + Le routage avec express.js se fait avec le module express. Router() ;
* Module views : c’est dans ce module que nous mis les vues de notre application.
  + Tout ce qui est aspect visuel de notre application est contenu dans la vue ;
  + Nous avons utilisé le moteur de Template Jade pour affecter les vues.
* Module public : Il contient les fichiers statiques tels les images, les fichiers css, les fichiers javascript.

# Partie I : Test avec client REST

# Présentation des fonctionnalités

## Ajout d’un produit

Pour ajouter un nouveau produit, on part sur le client REST, on choisit la méthode POST et met la route correspondant à la méthode ajouter produit.

Dans notre cas, la route est :

router.post('/ajoutProduit', controller.createDonnee);

Cette route se trouve dans le fichier routage.js du module Router.

La méthode permettant d’ajouter un nouveau produit :

const createDonnee = (req, res) => {

    const  newProduit = new Produit(req.body);

    newProduit.save()

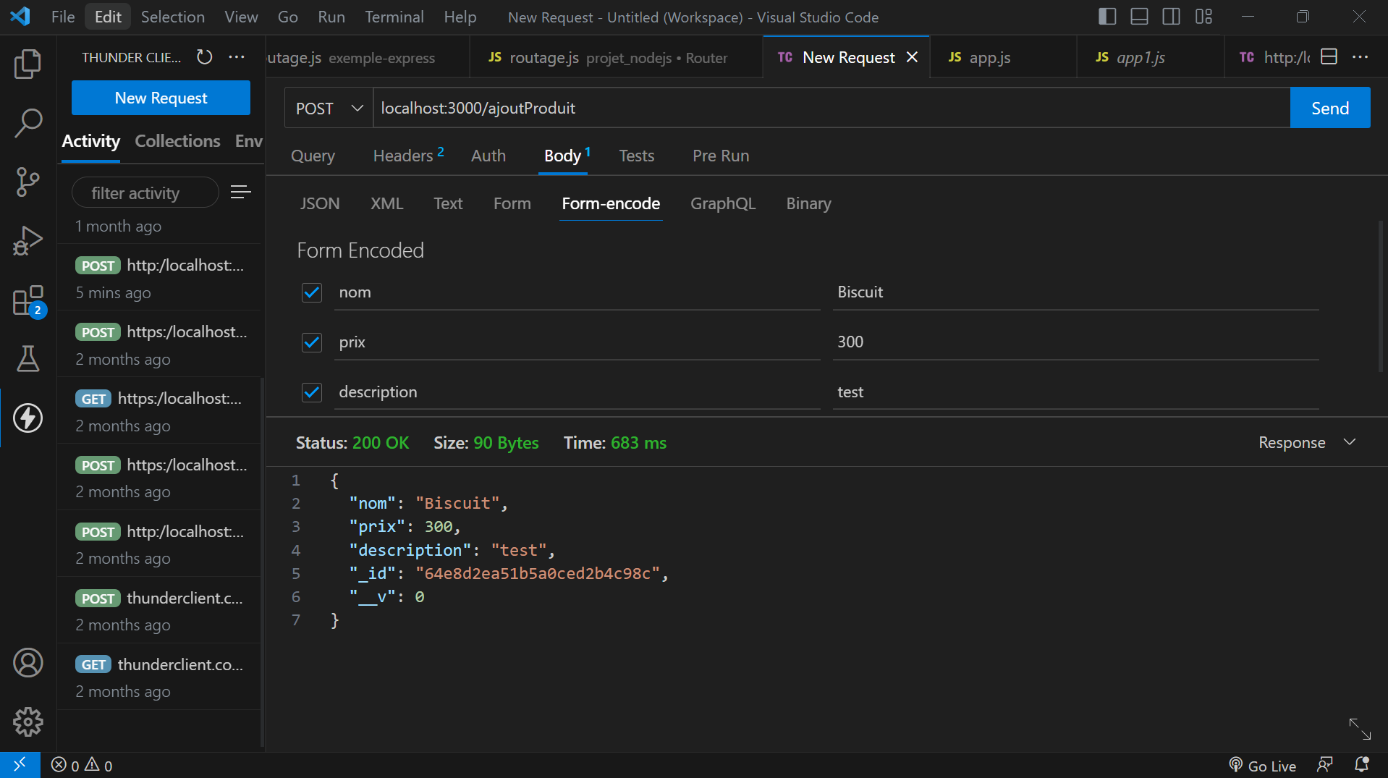
    .then(result =>{console.log(result),res.send(result)})

    .catch(error => {console.log(error),res.send(error)})

  };

Cette méthode se trouve dans le fichier controlleur.js du module Controlleur.

Illustration avec l’ajout d’un produit « Biscuit » :



Partie II : Avec client REST

## Afficher la liste des produits

Pour lister les produits, nous avons utilisé la route suivante :

router.get('/list', controller.SearchDonnee);

Cette route se trouve dans le fichier routage.js du module Router.

La méthode permettant de liste est :

const SearchDonnee = (req,res) =>{

    Produit.find()

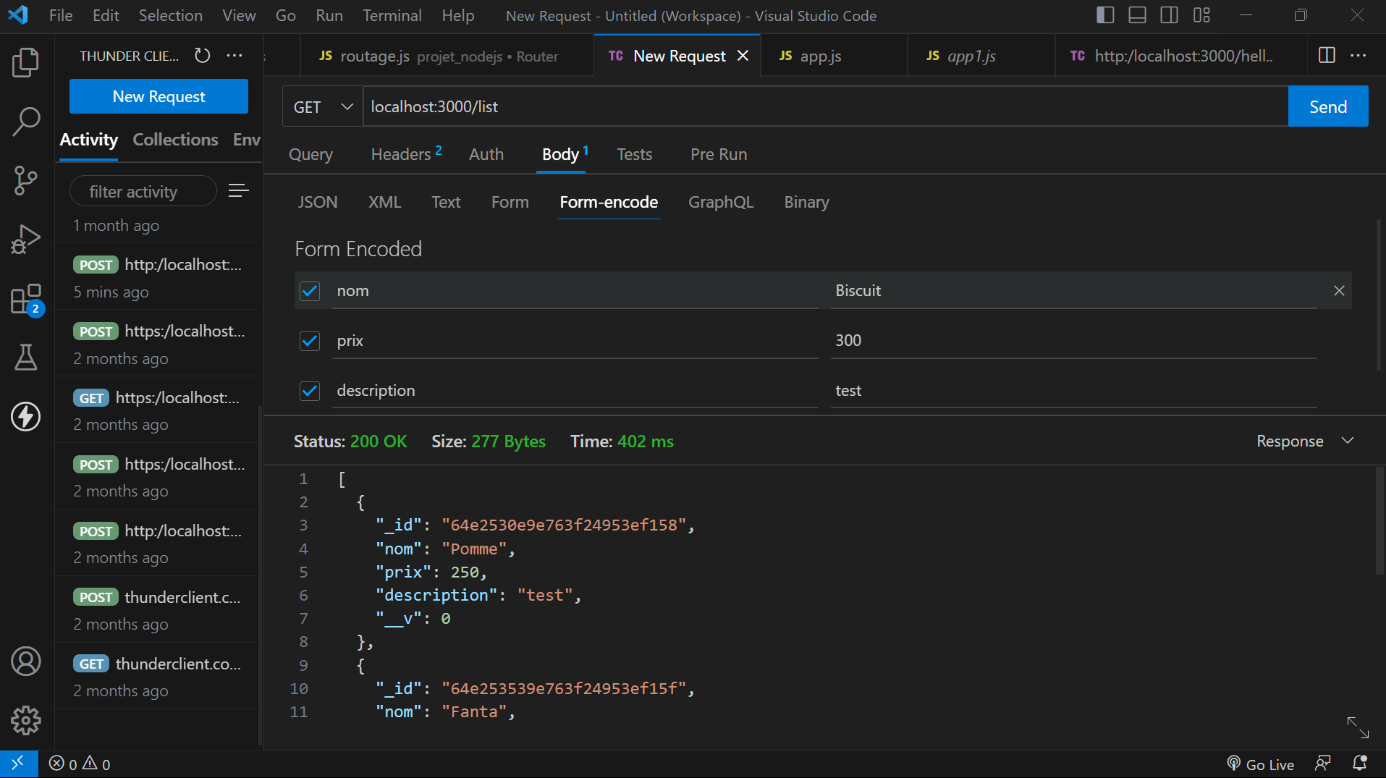
    .then(result =>{console.log(result),res.send(result)})

    .catch(error => {console.log(error),res.send(error)})

}

Cette méthode se trouve dans le fichier controlleur.js du module Contrôleur.

Illustration avec la capture suivante :



Nous voyons la liste des produits dans la console en bas au niveau du client REST.

## Modification d’un produit

Pour modifier un produit, nous avons choisi la méthode PATCH au niveau du client REST et mettre la route correspondant qui est :

router.patch('/product/:id', controller.FindByIDUpdateDonnee);

Cette route se trouve dans le fichier routage.js du module Router.

La méthode permettant de modifier un produit est :

const FindByIDUpdateDonnee = (req,res) =>{

    Produit.findByIdAndUpdate(

        req.params.id,

        req.body,

        {new : true},

    )

    .then((newProduit) => {

        res.send(newProduit);

    })

    .catch((err) => {

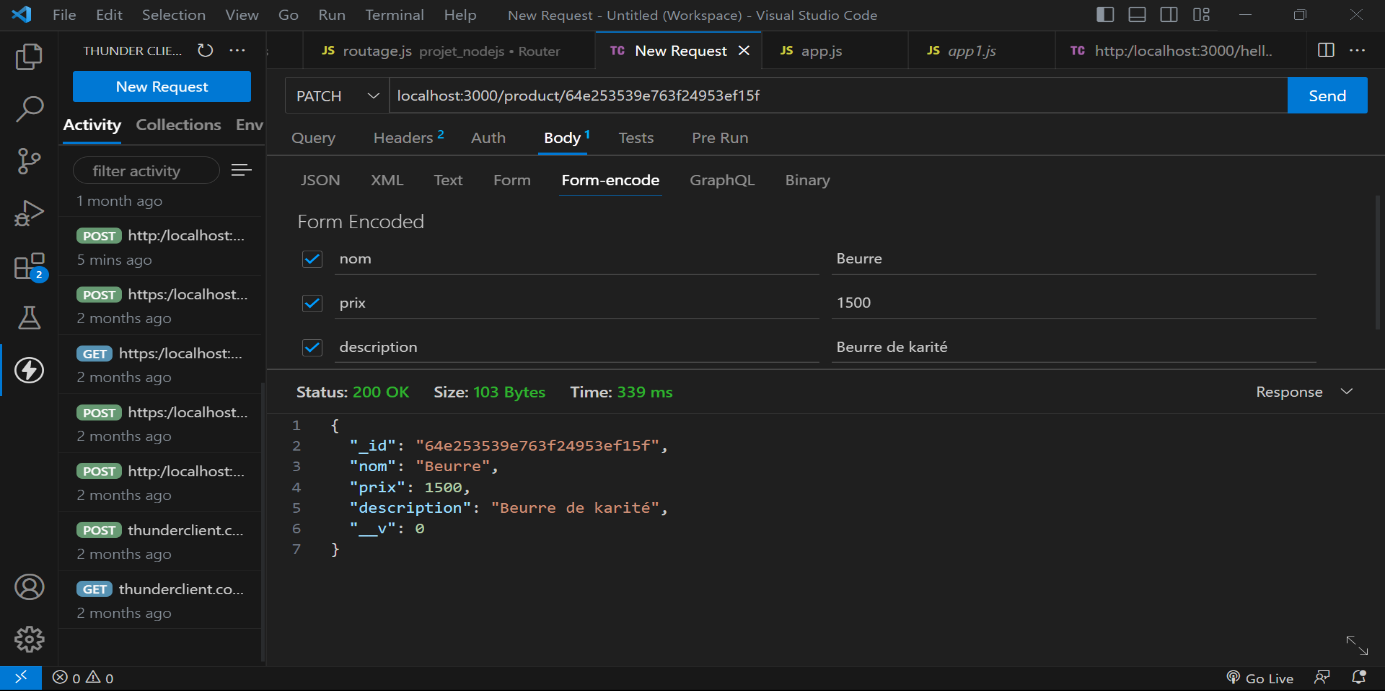
        return res.send(err);

    });

}

Cette méthode se trouve dans le fichier controlleur.js du module Contrôleur.

Illustration avec la modification du produit « Fanta » en produit « Beurre » :



## Suppression d’un produit

Pour supprimer un produit, nous avons choisi la méthode delete au niveau du client REST et nous avons utilisé la route suivante :

router.delete('/product/:id', controller.FindByIdDeleteDonnee);

Cette route se trouve dans le fichier routage.js du module Router.

La méthode permettant de supprimer un produit est :

const FindByIdDeleteDonnee = (req,res) =>{

    Produit.findByIdAndRemove(

        req.params.id,

    )

    .then((oldProduit) => {

        const response = {

            message : "Le produit a été supprimé",

            produit : oldProduit

        }

        res.send(response);

    })

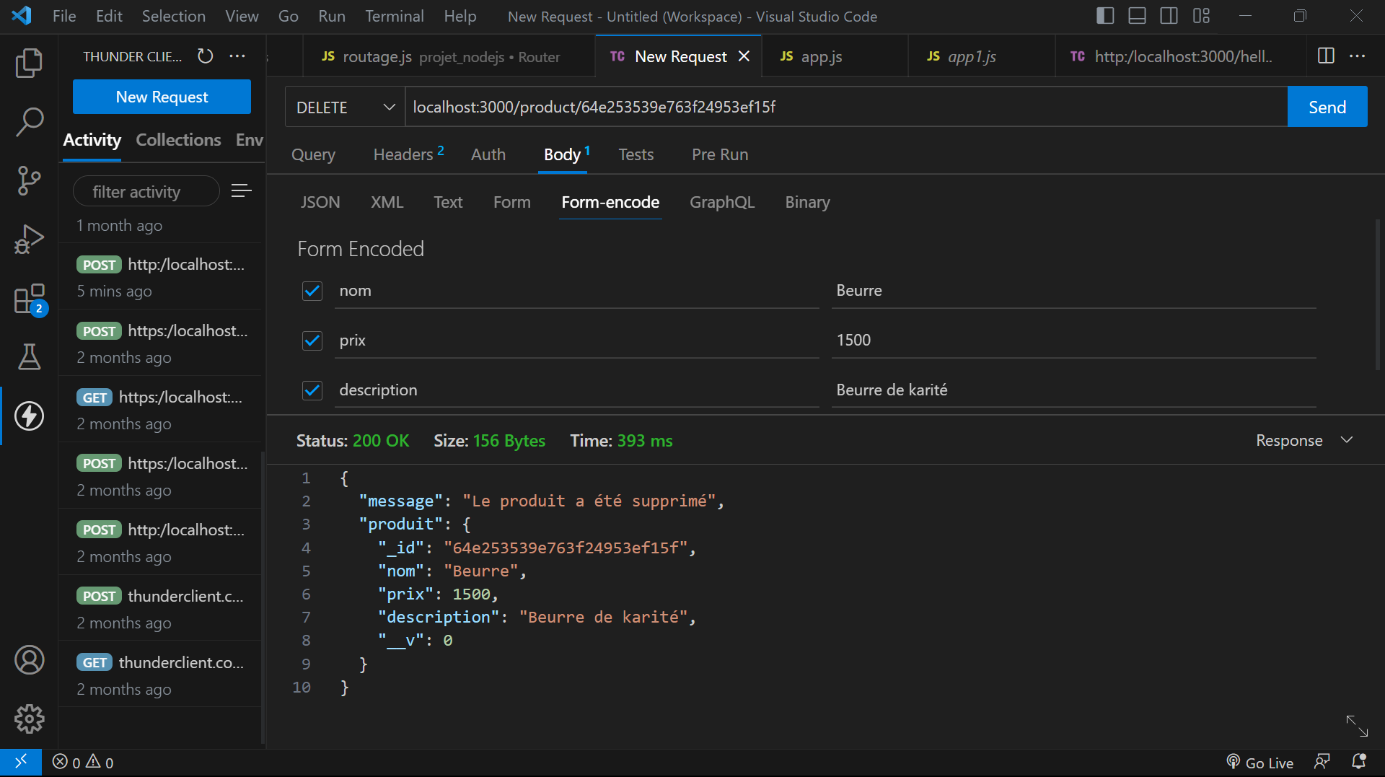
    .catch((err) => {

        return res.send(err);

    });

}

Cette méthode se trouve dans le fichier controlleur.js du module Contrôleur.

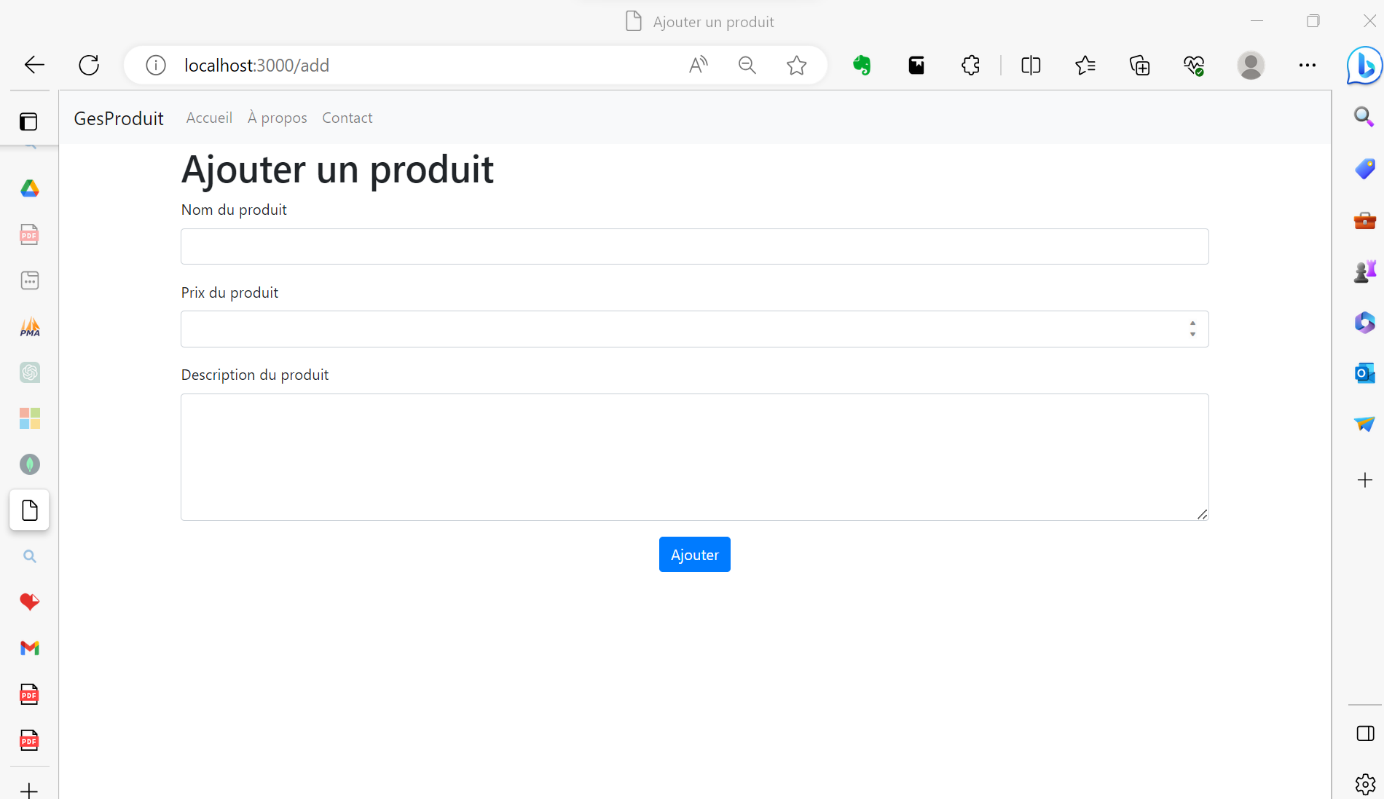
Illustration avec la suppression du produit « Beurre »

# Partie II : Test avec l’interface en utilisant le moteur de Template Jade

# Présentation des fonctionnalités

## Ajout d’un produit

Pour ajouter un nouveau produit, cliquez sur « Ajouter un produit » dans la page principale et la page suivante s’affichera.



Les codes sources correspondants

Lien vers la page add\_produit, contenu dans le fichier index.jade, dans le module views:

 .row

  .col-md-6.offset-md-3

          .content.text-center

            a.btn.btn-primary.btn-sm(href=`/add`) Ajouter un produit

Code source correspondant à l’affichage du formulaire d’ajout d’un produit. Ce code se trouve dans le fichier add\_produit.jade, dans le module views:

doctype html

html

  head

    title= title

    link(rel='stylesheet', href='/css/bootstrap.min.css')

  body

    nav.navbar.navbar-expand-lg.navbar-light.bg-light

      a.navbar-brand(href='#') GesProduit

      button.navbar-toggler(type='button', data-toggle='collapse', data-target='#navbarNav', aria-controls='navbarNav', aria-expanded='false', aria-label='Toggle navigation')

        span.navbar-toggler-icon

      div#navbarNav.collapse.navbar-collapse

        ul.navbar-nav

          li.nav-item

            a.nav-link(href='/') Accueil

          li.nav-item

            a.nav-link(href='/about') À propos

          li.nav-item

            a.nav-link(href='/contact') Contact

    .container

      h1 Ajouter un produit

      form(action='/addProduit', method='post')

        .form-group

          label(for='name') Nom du produit

          input#name.form-control(type='text', name='nom', required)

        .form-group

          label(for='price') Prix du produit

          input#price.form-control(type='number', name='prix', step='0.01', required)

        .form-group

          label(for='description') Description du produit

          textarea#description.form-control(name='description', rows='5', required)

        .content.text-center

           button.btn.btn-primary(type='submit') Ajouter

    script(src='/js/jquery.min.js')

    script(src='/js/bootstrap.min.js')

Code source représentant la fonction d’ajout d’un nouveau produit. Ce code se trouve dans le fichier controlleur.js,dans le module Controlleur:

const createData = (req, res) => {

    const  newProduit = new Produit(req.body);

    newProduit.save();

    res.redirect('/');

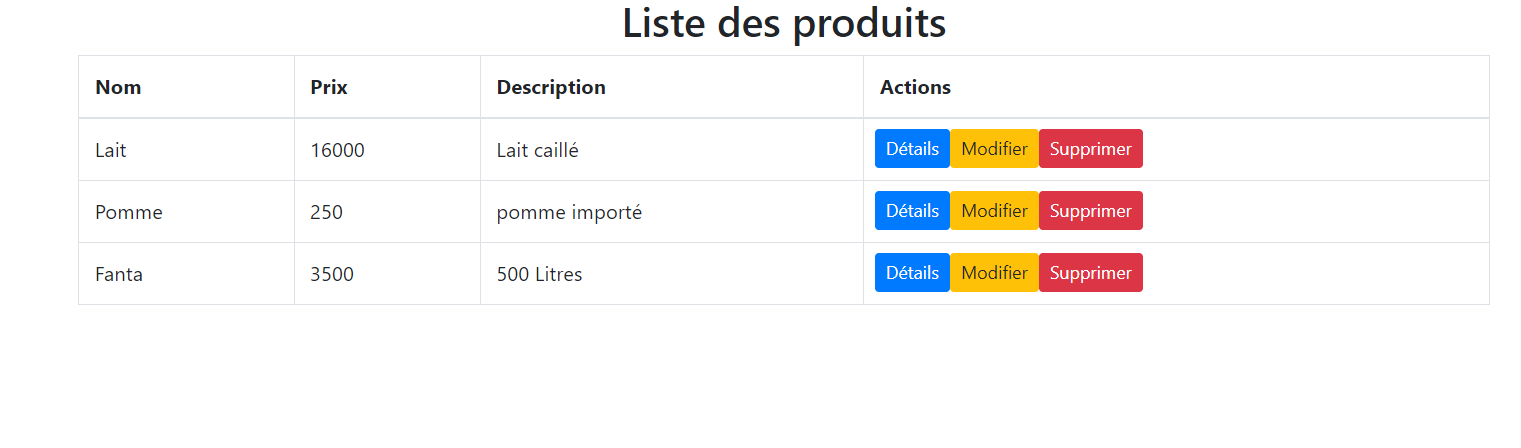
  };

Route vers la page add\_produit. Ce code se trouve dans le fichier routage.js,dans le module Router.

router.post('/addProduit', controller.createData);

## Affichage de la liste des produits

La liste des produits s’affiche automatiquement sur la page principale. Il n’y pas de bouton à cliquer.



Les codes sources correspondants

Code source correspondant à l’affichage de la liste des produits se trouve dans le fichier index.jade, dans le module views :

 .row

      .content.text-center

        h2 Liste des produits

      table.table.table-bordered

        thead

          tr

            th#th-name(scope='col-3') Nom

            th#th-price(scope='col-3') Prix

            th#th-description(scope='col-3') Description

            th#th-actions(scope='col-2') Actions

        tbody

          each produit in produits

            tr

              td= produit.nom

              td= produit.prix

              td= produit.description

              td.p-2

                a.btn.btn-primary.btn-sm(href=`/detail/${produit.id}`) Détails

                a.btn.btn-warning.btn-sm(href=`/update/${produit.id}`) Modifier

Fonction d’affichage de la liste des produits se trouve dans le fichier controlleur.js du module Controlleur :

const SearchData = async (req,res) =>{

   const produits = await Produit.find();

    res.render('index',{produits});

}

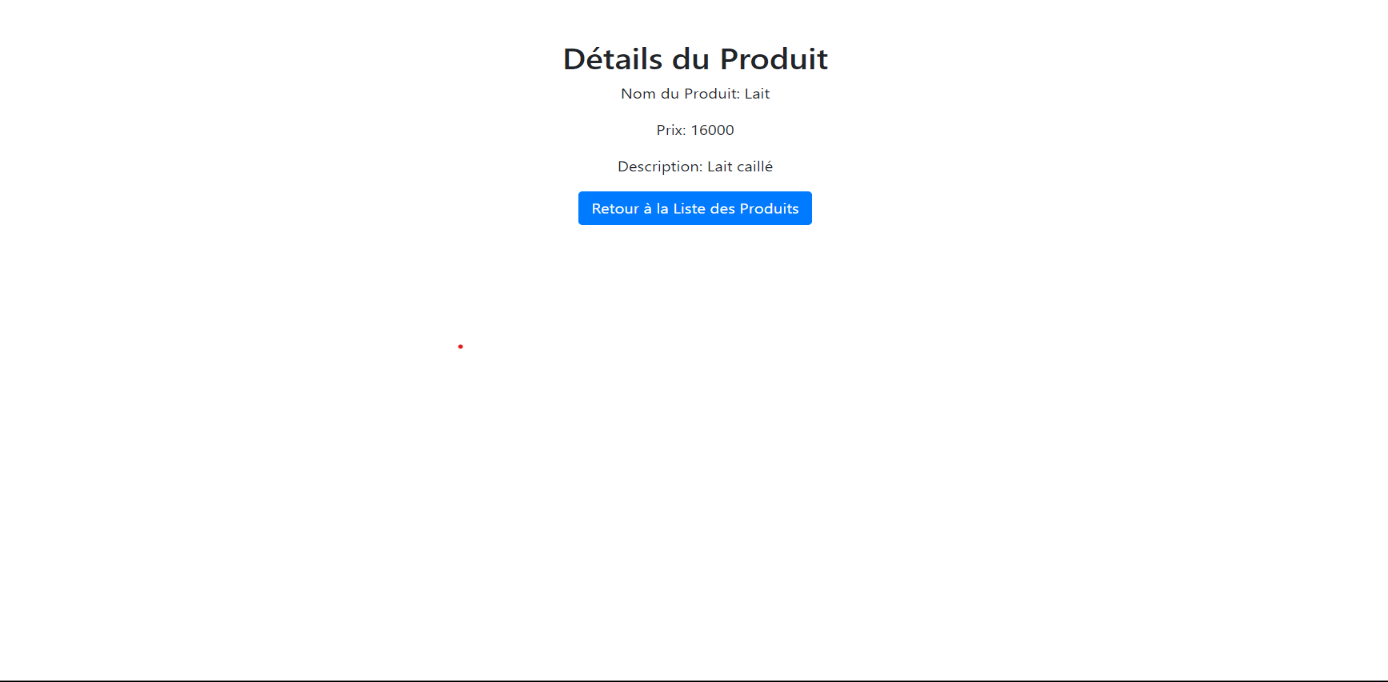
Route pour l’affichage de la liste des produits :

router.get('/', controller.SearchData);

## Affichage des détails des produits

Pour visualiser les détails d’un produit, aller au niveau de la liste des produits et cliquer sur le bouton « Détails » et les détails du produit correspondant s’affichera en une autre page

Illustration avec le produit « Lait » : on clique sur « Details » devant le produit Lait, la page suivante s’affichera :



Les codes sources correspondants

Lien vers la page detail.jade, contenu dans le fichier index.jade, dans le module views :

   a.btn.btn-primary.btn-sm(href=`/detail/${produit.id}`) Détails

Route vers la page detail.jade, contenu dans le fichier app.js :

  app.get('/detail', (req, res) => {

    res.render('detail', { title: 'Detail d\'un produit' });

  });

Contenu du fichier detail.jade qui est dans le module views :

doctype html

html

  head

    title Détails du Produit

    link(rel='stylesheet', href='/css/bootstrap.min.css')

    link(rel='stylesheet', href='/css/style.css')

  body

    .container

      .row

      .content.text-center.mb-10.p-5

          h2 Détails du Produit

          p Nom du Produit: #{produit.nom}

          p Prix: #{produit.prix}

          p Description: #{produit.description}

          a.btn.btn-primary(href='/') Retour à la Liste des Produits

    script(src='/js/jquery.min.js')

    script(src='/js/bootstrap.min.js')

Fonction permettant de rechercher le produit dont on souhaite afficher les détails, ce code dans le fichier controlleur.js du module controlleur :

const SearchDataById = async (req,res) =>{

    const produit = await Produit.findById(req.params.id);

    res.render('detail',{produit});

}

Route pour l’affichage des détails d’un produit,ce code se trouve dans le fichier routage.js du module router :

router.get('/detail/:id',controller.SearchDataById);

## Modification d’un produit

Pour modifier un produit, aller au niveau de la liste des produits et cliquer sur le bouton « Modifier » pour ouvrir une nouvelle page comme le montre la capture suivante :

Une image contenant texte, capture d’écran, Police, nombre

Description générée automatiquement

Les codes sources correspondants

Lien vers la page modify\_produit.jade, ce code se trouve dans le fichier index.jade du module views :

 a.btn.btn-warning.btn-sm(href=`/update/${produit.id}`) Modifier

Ce code correspondant à l’affichage de la page modify\_produit.jade :

doctype html

html

  head

    title= title

    link(rel='stylesheet', href='/css/bootstrap.min.css')

  body

    nav.navbar.navbar-expand-lg.navbar-light.bg-light

      a.navbar-brand(href='#') GesProduit

      button.navbar-toggler(type='button', data-toggle='collapse', data-target='#navbarNav', aria-controls='navbarNav', aria-expanded='false', aria-label='Toggle navigation')

        span.navbar-toggler-icon

      div#navbarNav.collapse.navbar-collapse

        ul.navbar-nav

          li.nav-item

            a.nav-link(href='/') Accueil

          li.nav-item

            a.nav-link(href='/about') À propos

          li.nav-item

            a.nav-link(href='/contact') Contact

    .container

      h1 Modifier un produit

      form(action=`/modifyProduit/${produit.id}`, method='post')

        .form-group

          label(for='name') Nom du produit

          input#name.form-control(type='text', name='nom',value = produit.nom,required)

        .form-group

          label(for='price') Prix du produit

          input#price.form-control(type='number', name='prix', step='0.01',value = produit.prix,required)

        .form-group

          label(for='description') Description du produit

          textarea#description.form-control(name='description',rows='5',value = produit.description,required)

        .content.text-center

          button.btn.btn-primary(type='submit') Valider

    script(src='/js/jquery.min.js')

    script(src='/js/bootstrap.min.js')

Fonction permettant de modifier un produit.Ce code se trouve dans le fichier controlleur.js du module controlleur :

const FindByIDUpdateData = async (req, res) => {

    try {

        const updatedProduit = await Produit.findByIdAndUpdate(

            req.params.id,

            req.body,

            { new: true }

        );

        if (!updatedProduit) {

            return res.status(404).send("Produit non trouvé");

        }

        const produits = await Produit.find(); // Récupérer la liste mise à jour des produits

        const response = {

            message: "Le produit a été mis à jour",

            produits: produits

        };

        res.redirect('/');

    } catch (err) {

        return res.send(err);

    }

};

Lorsque la modification d’un produit est sauvegardée, la liste des produits est automatiquement actualisée.

Route vers la page modify\_produit,ce code se trouve dans le fichier routage.js du module router :

router.post('/modifyProduit/:id', controller.FindByIDUpdateData);

## Suppression d’un produit

Pour supprimer un produit, aller au niveau de la liste de produits et cliquer sur le bouton « Supprimer » et le produit correspondant sera automatiquement supprimé de la liste des produits.

Illustration avant la suppression du produit « Lait » :

Une image contenant texte, nombre, Police, ligne

Description générée automatiquement

Illustration après la suppression du produit « Lait » :

Une image contenant texte, capture d’écran, Police, ligne

Description générée automatiquement

Nous voyons que le produit « Lait » n’existe plus dans la liste des produits.

Les codes sources correspondants

Ce code se trouve dans le fichier index.jade du module views:

 a.btn.btn-danger.btn-sm(href=`/produit/${produit.id}`) Supprimer

Fonction permettant la suppression d’un produit,ce code se trouve dans le fichier controlleur.js du module controlleur :

const FindByIdDeleteData = async(req,res) =>{

    try {

        const oldProduit = await Produit.findByIdAndRemove(req.params.id);

        if (!oldProduit) {

            return res.status(404).send("Produit non trouvé");

        }

        const produits = await Produit.find(); // Récupérer la liste mise à jour des produits

        const response = {

            message: "Le produit a été supprimé",

            produits: produits

        };

        res.render('index', { produits });

    } catch (err) {

        return res.send(err);

    }

}

## Autres fonctionnalités

### Barre navigation, page about et contact

La barre de navigation sert à passer d’une page à une autre. Dans notre cas, elle permet de passer de la page Accueil à la page A propos,Contact et vice versa.



Les codes sources correspondant

Ce code se trouve dans le fichier index.jade du module views :

 nav.navbar.navbar-expand-lg.navbar-light.bg-light

      a.navbar-brand(href='#') GesProduit

      button.navbar-toggler(type='button', data-toggle='collapse', data-target='#navbarNav', aria-controls='navbarNav', aria-expanded='false', aria-label='Toggle navigation')

        span.navbar-toggler-icon

      div#navbarNav.collapse.navbar-collapse

        ul.navbar-nav

          li.nav-item.active

            a.nav-link(href='/') Accueil

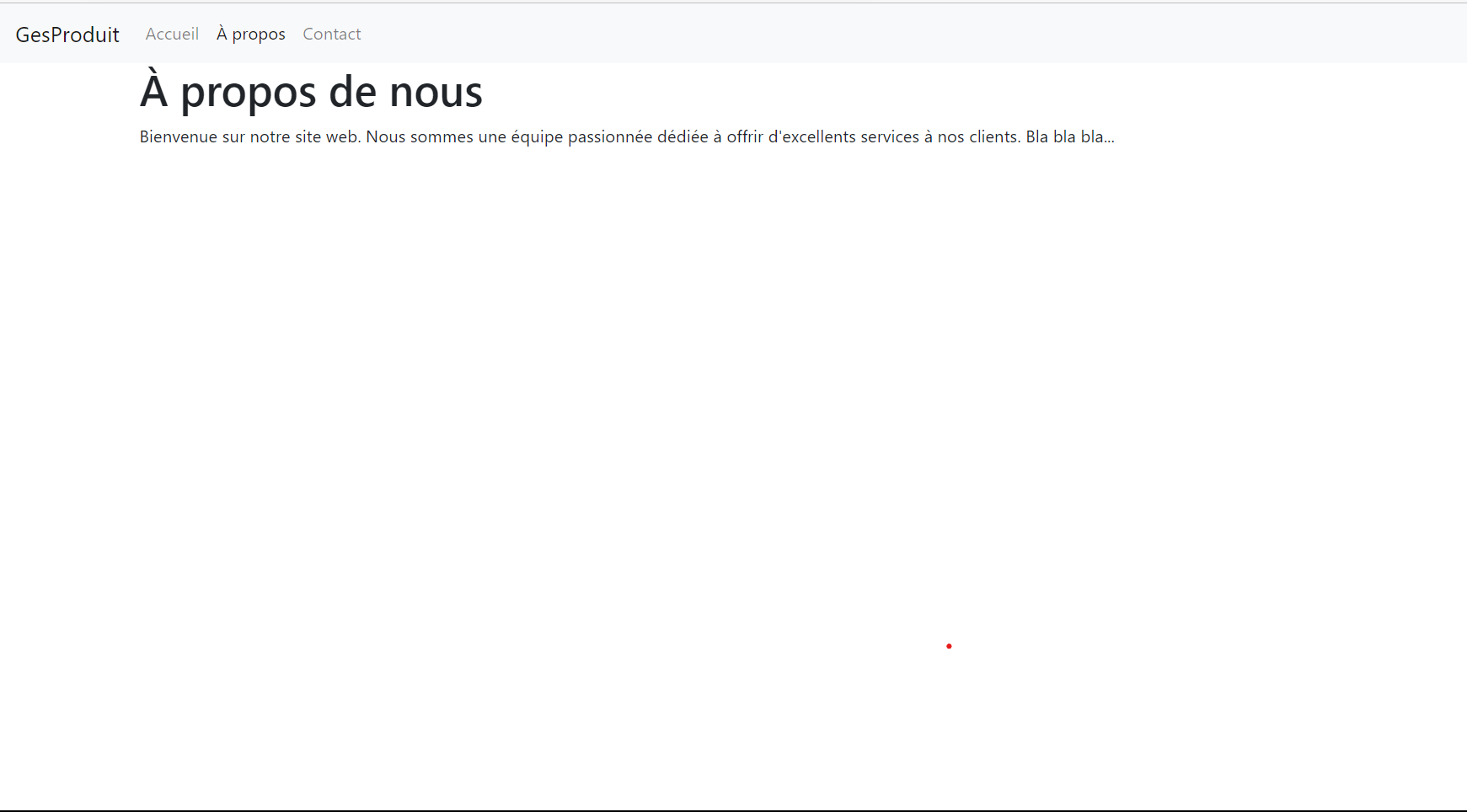
          li.nav-item

            a.nav-link(href='/about') À propos

          li.nav-item

            a.nav-link(href='/contact') Contact

La page A propos



Le code source correspondant :

Ce code se trouve dans le fichier about.jade du module views :

doctype html

html

  head

    title= title

    link(rel='stylesheet', href='/css/bootstrap.min.css')

  body

    nav.navbar.navbar-expand-lg.navbar-light.bg-light

      a.navbar-brand(href='#') GesProduit

      button.navbar-toggler(type='button', data-toggle='collapse', data-target='#navbarNav', aria-controls='navbarNav', aria-expanded='false', aria-label='Toggle navigation')

        span.navbar-toggler-icon

      div#navbarNav.collapse.navbar-collapse

        ul.navbar-nav

          li.nav-item

            a.nav-link(href='/') Accueil

          li.nav-item.active

            a.nav-link(href='/about') À propos

          li.nav-item

            a.nav-link(href='/contact') Contact

    .container

      h1 À propos de nous

      p Bienvenue sur notre site web. Nous sommes une équipe passionnée dédiée à offrir d'excellents services à nos clients. Bla bla bla...

    script(src='/js/jquery.min.js')

    script(src='/js/bootstrap.min.js')

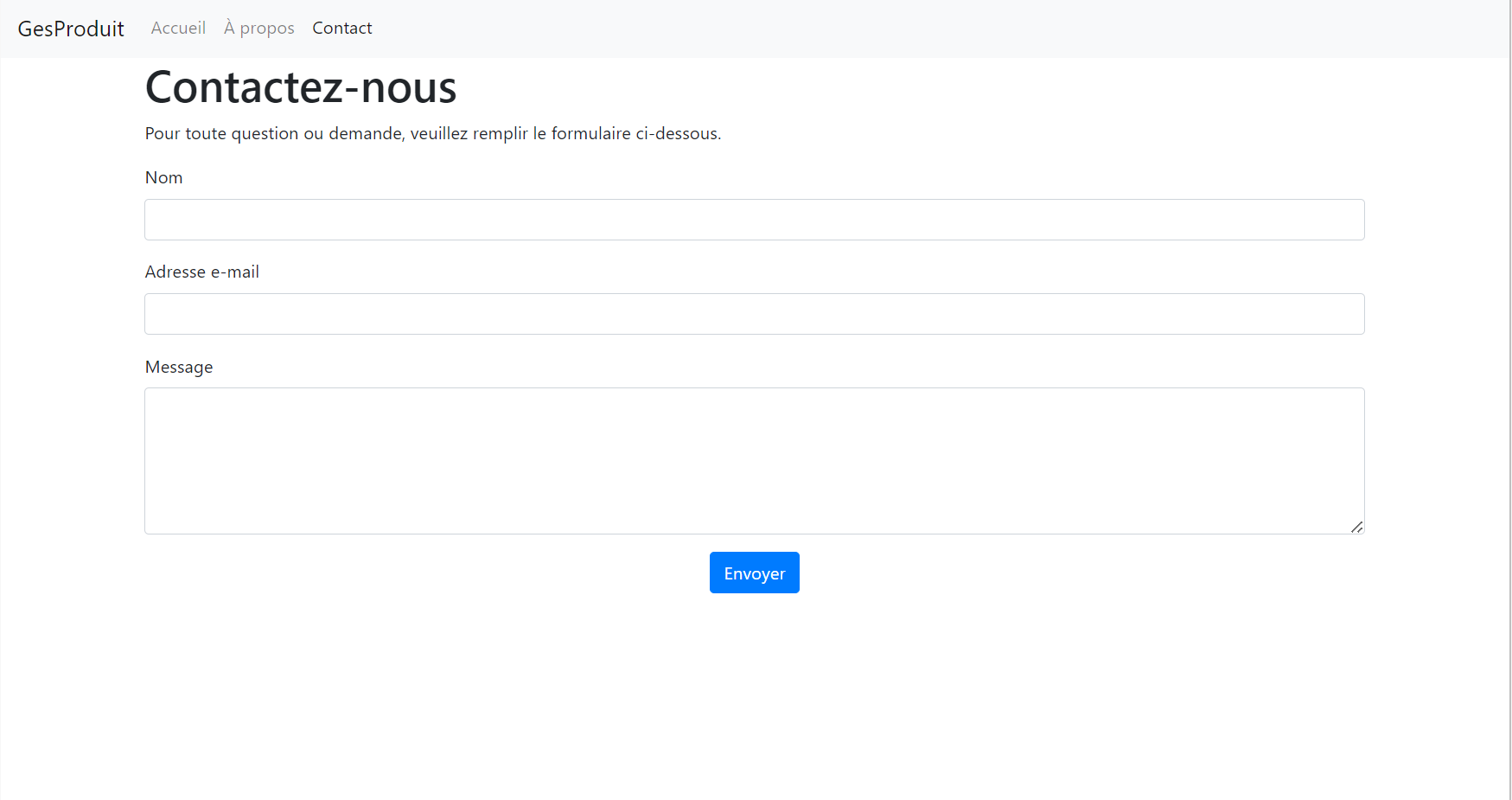
Route vers la page about.jade,ce code se trouve dans le fichier app.js:

  app.get('/about', (req, res) => {

    res.render('about', { title: 'À propos' });

  });

La page contact permet aux utilisateurs de notre application de nous envoyer un mail :



Le code source correspondant se trouve dans le fichier contact.jade du module views :

doctype html

html

  head

    title= title

    link(rel='stylesheet', href='/css/bootstrap.min.css')

  body

    nav.navbar.navbar-expand-lg.navbar-light.bg-light

      a.navbar-brand(href='#') GesProduit

      button.navbar-toggler(type='button', data-toggle='collapse', data-target='#navbarNav', aria-controls='navbarNav', aria-expanded='false', aria-label='Toggle navigation')

        span.navbar-toggler-icon

      div#navbarNav.collapse.navbar-collapse

        ul.navbar-nav

          li.nav-item

            a.nav-link(href='/') Accueil

          li.nav-item

            a.nav-link(href='/about') À propos

          li.nav-item.active

            a.nav-link(href='/contact') Contact

    .container

      h1 Contactez-nous

      p Pour toute question ou demande, veuillez remplir le formulaire ci-dessous.

      form(action='/contact', method='post')

        .form-group

          label(for='name') Nom

          input#name.form-control(type='text', name='name', required)

        .form-group

          label(for='email') Adresse e-mail

          input#email.form-control(type='email', name='email', required)

        .form-group

          label(for='message') Message

          textarea#message.form-control(name='message', rows='5', required)

        .content.text-center

          button.btn.btn-primary(type='submit') Envoyer

    script(src='/js/jquery.min.js')

    script(src='/js/bootstrap.min.js')

Route vers la page contact, ce code se trouve dans le fichier app.js :

  app.get('/contact', (req, res) => {

    res.render('contact', { title: 'Contact' });

  });

Le code CSS pour gérer la responsivité de notre application

Ce code se trouve dans le fichier style.css dans le module public :

    @media (max-width: 576px) {

      h1{

        margin-top: 20%;

        text-align: center;

      }

      .navbar{

          position:relative;

      }

    }

    @media (min-width: 577px) and (max-width: 992px) {

        h1{

        margin-top: 10%;

        text-align: center;

      }

      .navbar{

          position:relative;

      }

    }

  @media (min-width: 993px) {

    h1 {

      margin-top: 5%;

      text-align: center;

    }

    .navbar{

      position: relative;

    }

  }