

Codes initiaux de l'application MEAN d'exemple

Projet MEAN (HMIN103, HMIN325M)

Pierre Pompidor

Table des matières

1 Environnement du projet :	2
1.1 Mongo DB :	2
1.1.1 Installation de MongoDB (si non déjà installé) :	2
1.1.2 Création de la base de données SUPERVENTES :	2
1.2 Node.js :	2
1.2.1 Installation de Node.js et de npm son gestionnaire de modules (si non déjà installés) :	2
1.2.2 L'installation des modules nécessaires au serveur Node.js :	2
1.2.3 L'exécution du serveur (une fois que celui-ci aura été créé) :	2
2 Côté serveur :	3
2.1 La collection initiale membres.json :	3
2.2 La collection initiale produits.json :	3
2.3 Script de création de la base MongoDB (creationbase.sh) :	3
2.4 Le serveur Node.js serveur.js :	3
2.5 Exécution du serveur :	5
3 Côté client : ANGULAR	6
3.1 Installation d'Angular et création d'un projet :	6
3.1.1 L'installation d'Angular via Angular CLI (si non déjà installé) :	6
3.1.2 Création d'un projet Angular :	6
3.1.3 L'exécution du serveur de développement d'Angular :	6
3.1.4 L'installation des modules nécessaires à un projet Angular récupéré sur un GIT :	6
3.2 Arborescence des codes :	6
3.3 src/index.html :	7
3.4 src/styles.css :	7
3.5 src/app/app.component.html :	7
3.6 src/app/app.component.ts :	8
3.7 src/app/app.module.ts :	8
3.8 src/app/app-routing.module.ts :	9
3.9 src/app/authentication.service.ts :	9
3.10 src/app/produits.service.ts :	10
3.11 src/app/menu/menu.component.html :	11
3.12 src/app/menu/menu.component.ts :	11
3.13 src/app/connexion/connexion.component.html :	12
3.14 src/app/connexion/connexion.component.ts :	12
3.15 src/app/categories/categories.component.html :	13
3.16 src/app/categories/categories.component.ts :	13
3.17 src/app/produits/produits.component.html :	14
3.18 src/app/produits/produits.component.ts :	14
3.19 Exécution du serveur de développement local Angular :	14

1 Environnement du projet :

Le projet nécessite :

- Pour le serveur de bases de données **MongoDB** :
 - l'installation du SGBD NoSQL MongoDB
 - la création de la base SUPERVENTES
- Pour le serveur **Node.js** :
 - L'installation de Node.js et de son gestionnaire de modules **npm**
 - l'installation des modules nécessaires au serveur Node.js
 - l'exécution du serveur
- Pour **Angular** :
 - l'installation d'Angular (en fait avec le gestionnaire de projets **Angular cli**)
 - l'installation des modules nécessaires à Angular
 - l'exécution du serveur de développement d'Angular

Pour connaître les versions de Node.js et d'Angular déjà installées sur la machine que vous utilisez : `ng --version`

Pour installer les versions minimales de codes du projet (le "kick-off") :

- Créez un dossier *PROJET_MEAN* et déplacez-vous-y
- Récupérez sur le Moodle *SUPERVENTES_CLIENT.tgz* et dézippez l'archive : `tar -xzf SUPERVENTES_CLIENT.tgz`
- Récupérez sur le Moodle *SUPERVENTES_SERVER.tgz* et dézippez l'archive : `tar -xzf SUPERVENTES_SERVER.tgz`

1.1 Mongo DB :

1.1.1 Installation de MongoDB (si non déjà installé) :

Allez au bout de cet URL : <https://docs.mongodb.com/manual/tutorial/install-mongodb-on-ubuntu/>

1.1.2 Création de la base de données SUPERVENTES :

Positionnez-vous dans le dossier *SUPERVENTES_SERVER*

```
chmod +x creationBase.sh; ./creationBase.sh
```

Remarque : l'exécution du script *creationBase.sh* automatise l'exécution des deux commandes suivantes :

```
mongoimport --db SUPERVENTES --collection produits --file produits.json --jsonArray --drop
mongoimport --db SUPERVENTES --collection membres --file membres.json --jsonArray --drop
```

1.2 Node.js

1.2.1 Installation de Node.js et de npm son gestionnaire de modules (si non déjà installés) :

```
sudo apt install nodejs npm
```

(Pour mettre à jour Node.js, vous pouvez aussi utiliser *nvm* (Node.js version manager))

1.2.2 L'installation des modules nécessaires au serveur Node.js :

Positionnez-vous dans un dossier spécifique au serveur (par exemple dans le dossier *SUPERVENTES_SERVER*).

```
npm install <nomDuModule>
```

Nous utiliserons pour notre projet les modules *express* et *mongodb* :

```
npm install express mongodb
```

1.2.3 L'exécution du serveur (une fois que celui-ci aura été créé) :

```
node serveur.js &
```

2 Côté serveur :

2.1 La collection initiale membres.json :

```
[
  {"nom":"Delune", "prénom":"Claire", "email":"claire.delune@mailserver.fr", "password":"123456"},
  {"nom":"Nett", "prénom":"Jessica", "email":"jessica.nett@mailserver.fr", "password":"0h0h666"},
  {"nom":"Sonsuper", "prénom":"Alexis", "email":"alexis.sonsuper@mailserver.fr", "password":"2fast4U"}
]
```

2.2 La collection initiale produits.json :

```
[
  {"nom":"Caféine", "type":"anti-fatigue", "prix":15},
  {"nom":"Curcumine", "type":"anti-oxydant", "prix":30},
  {"nom":"Quercétine", "type":"anti-oxydant", "prix":50},
  {"nom":"Resvératrol", "type":"anti-oxydant", "prix":45},
  {"nom":"Rhodiola rosea", "type":"anti-fatigue", "prix":25}
]
```

2.3 Script de création de la base MongoDB (creationbase.sh) :

Ce script permet de créer deux collections (tables) au sein de la base de données *SUPERVENTES* sur le **SGBD MongoDB local**. En cas de mise en œuvre d'un serveur MongoDB distant, une autre procédure devra être effectuée.

```
mongoimport --db SUPERVENTES --collection membres --file membres.json --jsonArray --drop
mongoimport --db SUPERVENTES --collection produits --file produits.json --jsonArray --drop
```

2.4 Le serveur Node.js serveur.js :

Ce code utilise un **SGBD MongoDB local**. En cas de mise en œuvre d'un serveur MongoDB distant, une autre valeur devra être donnée à la constante *url*.

```

const express = require('express');
const app      = express();
app.use(express.json());
app.use(express.urlencoded({extended:true}));
// Gestion des CORS
app.use(function (req, res, next) {
  res.setHeader('Access-Control-Allow-Origin', '*');
  res.setHeader('Access-Control-Allow-Methods', 'GET, POST, PUT, DELETE');
  res.setHeader('Access-Control-Allow-Headers', '*');
  next();
});
//app.use(require("cors")); // (méthode alternative de gestion des CORS)

const MongoClient = require('mongodb').MongoClient;
const ObjectId    = require('mongodb').ObjectId;
const url         = "mongodb://localhost:27017";

MongoClient.connect(url, {useNewUrlParser: true}, (err, client) => {
  let db = client.db("SUPERVENTES");

  /* Liste des produits */
  app.get("/produits", (req,res) => {
    console.log("/produits");
    try {
      db.collection("produits").find().toArray((err, documents) => {
        res.end(JSON.stringify(documents));
      });
    } catch(e) {
      console.log("Erreur sur /produits : " + e);
      res.end(JSON.stringify([]));
    }
  });

  /* Liste des produits suivant une catégorie */
  app.get("/produits/:categorie", (req,res) => {
    let categorie = req.params.categorie;
    console.log("/produits/"+categorie);
    try {
      db.collection("produits").find({type:categorie}).toArray((err, documents) => {
        res.end(JSON.stringify(documents));
      });
    } catch(e) {
      console.log("Erreur sur /produits/"+categorie+" : "+ e);
      res.end(JSON.stringify([]));
    }
  });
});

```

```

/* Liste des catégories de produits */
app.get("/categories", (req,res) => {
  console.log("/categories");
  categories = [];
  try {
    db.collection("produits").find().toArray((err, documents) => {
      for (let doc of documents) {
        if (!categories.includes(doc.type)) categories.push(doc.type);
      }
      console.log("Renvoi de"+JSON.stringify(categories));
      res.end(JSON.stringify(categories));
    });
  } catch(e) {
    console.log("Erreur sur /categories : " + e);
    res.end(JSON.stringify([]));
  }
});

/* Connexion d'un utilisateur */
app.post("/membre/connexion", (req,res) => {
  console.log("/utilisateurs/connexion avec "+JSON.stringify(req.body));
  try {
    db.collection("membres")
      .find(req.body)
      .toArray((err, documents) => {
        if (documents.length == 1)
          res.end(JSON.stringify({"resultat": 1, "message": "Authentification réussie"}));
        else res.end(JSON.stringify({"resultat": 0,
          "message": "Email et/ou mot de passe incorrect"}));
      });
  } catch (e) {
    res.end(JSON.stringify({"resultat": 0, "message": e}));
  }
});
});

app.listen(8888);

```

2.5 Exécution du serveur :

```
node serveur.js
```

3 Côté client : ANGULAR

3.1 Installation d'Angular et création d'un projet :

3.1.1 L'installation d'Angular via Angular CLI (si non déjà installé) :

```
npm install @angular/cli@latest
```

(Pour mettre à jour Angular si celui-ci est déjà installé : `ng update @angular/cli`)

Le gestionnaire de projet Angular CLI se manipule par la commande `ng`

3.1.2 Création d'un projet Angular :

```
ng new <nomDuProjet>
```

La création d'un nouveau projet engendre pas mal de codes et peut prendre quelque temps...

Un projet de démonstration proposant des liens sur un tutoriel sur Angular est créé.

3.1.3 L'exécution du serveur de développement d'Angular :

Mettons en œuvre le projet de démo (qui deviendra par la suite notre projet).

Attention : vous devez vous positionner dans le dossier créé pour le projet Angular.

```
ng serve -o
```

(l'affichage en rouge d'erreurs dans votre terminal serait très inquiétante)

Votre navigateur devrait s'ouvrir sur la page de l'application, sinon invoquez l'URL suivante : `localhost:4200`

Vous pouvez ouvrir la console de votre navigateur (via F12) pour vérifier la trace des programmes et la présence d'éventuelles erreurs.

Pour le projet de démo (celui dont les codes sont donnés ci-après), pour vous connecter en tant qu'utilisateur, utilisez par exemple les identifiants suivants :

claire.delune@mailserver.fr

123456

3.1.4 L'installation des modules nécessaires à un projet Angular récupéré sur un GIT :

```
npm install
```

Remarque : `npm install` installe les modules nécessaires suivant les spécifications du fichier **package.json**.

3.2 Arborescence des codes

Les codes d'ébauches de spécifications de test n'apparaissent pas dans cette liste.

```
<PROJET>:
node_modules/
package.json
src/

src:
app/
assets/
index.html
main.ts
styles.css
```

```

src/app:
app.component.css
app.component.html
app.component.ts
app.module.ts
app-routing.module.ts
authentication.service.ts
produits.service.ts
categories/
connexion/
menu/
produits/

src/app/categories:
categories.component.css
categories.component.html
categories.component.ts

src/app/connexion:
connexion.component.css
connexion.component.html
connexion.component.ts

src/app/menu:
menu.component.css
menu.component.html
menu.component.ts

src/app/produits:
produits.component.css
produits.component.html
produits.component.ts

```

3.3 src/index.html

```

<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>SUPERVENTES</title>
  <base href="/">
  <meta name="viewport" content="width=device-width, initial-scale=0.9">
</head>
<body>
  <app-root></app-root>
</body>
</html>

```

3.4 src/styles.css

```

@import url('https://unpkg.com/bootstrap@4.1.3/dist/css/bootstrap.min.css');

* { font-size: 10pt; }

```

3.5 src/app/app.component.html

```

<app-menu></app-menu>

<div class="container-fluid">
  <div class="row justify-content-center">
    <router-outlet></router-outlet>
  </div>
</div>

```

3.6 src/app/app.component.ts

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'SUPERVENTES';
}
```

3.7 src/app/app.module.ts

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { FormsModule } from '@angular/forms';
import { HttpClientModule } from '@angular/common/http';

import { AppRoutingModule } from './app-routing.module';
import { AuthenticationService } from './authentication.service';
import { ProduitsService } from './produits.service';

import { AppComponent } from './app.component';
import { ConnexionComponent } from './connexion/connexion.component';
import { ProduitsComponent } from './produits/produits.component';
import { CategoriesComponent } from './categories/categories.component';
import { MenuComponent } from './menu/menu.component';

@NgModule({
  declarations: [
    AppComponent,
    ConnexionComponent,
    ProduitsComponent,
    CategoriesComponent,
    MenuComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    FormsModule,
    HttpClientModule
  ],
  providers: [AuthenticationService, ProduitsService],
  bootstrap: [AppComponent]
})
export class AppModule { }
```


3.8 src/app/app-routing.module.ts

```
import { NgModule } from '@angular/core';
import { Routes, RouterModule } from '@angular/router';
import { ConnexionComponent } from '../connexion/connexion.component';
import { CategoriesComponent } from '../categories/categories.component';
import { ProduitsComponent } from '../produits/produits.component';

const routes: Routes = [
  { path: 'membres/connexion',
    component: ConnexionComponent
  },
  { path: 'categories',
    component: CategoriesComponent
  },
  { path: 'produits/:categorie',
    component: ProduitsComponent
  },
  { path: 'produits',
    component: ProduitsComponent
  }
];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

3.9 src/app/authentication.service.ts

```
import { Injectable } from '@angular/core';
import { HttpClient, HttpHeaders } from '@angular/common/http';
import { Subject, BehaviorSubject } from 'rxjs';
import { Observable } from 'rxjs';

const httpOptions = {
  headers: new HttpHeaders({
    "Access-Control-Allow-Methods": "GET,POST",
    "Access-Control-Allow-Headers": "Content-type",
    "Content-Type": "application/json",
    "Access-Control-Allow-Origin": "*"
  })
};

@Injectable({
  providedIn: 'root'
})
export class AuthenticationService {
  private user: Subject<string> = new BehaviorSubject<string>(undefined);
  private baseUrl: string = "http://localhost:8888/";

  constructor(private http: HttpClient) { }

  getUser() { return this.user; }

  connect(data: string) { this.user.next(data); }

  disconnect() { this.user.next(null); }

  verificationConnexion(identifiants): Observable<any> {
    return this.http.post(this.baseUrl+'membre/connexion',
      JSON.stringify(identifiants), httpOptions);
  }
}
```

3.10 src/app/produits.service.ts

```
import { Injectable } from '@angular/core';
import { HttpClient, HttpHeaders } from '@angular/common/http';
import { Observable } from 'rxjs';
import { map } from 'rxjs/operators';

@Injectable({providedIn: 'root'})
export class ProduitsService {

    private urlBase: string = 'http://localhost:8888/';

    constructor(private http: HttpClient) { }

    getProduits(): Observable<any> {
        return this.http.get(this.urlBase+'produits');
    }

    getProduitsParCategorie(categorie): Observable<any> {
        return this.http.get(this.urlBase+'produits/'+categorie);
    }

    getCategories(): Observable<any> {
        return this.http.get(this.urlBase+'categories');
    }
}
```

3.11 src/app/menu/menu.component.html

```
<nav class="navbar navbar-dark bg-dark navbar-fixed-top navbar-expand">
  <div class="container-fluid">
    <ul *ngIf="!(user|async)" class="navbar-nav">
      <li class="nav-item">
        <a class="nav-link active" routerLink="/membres/connexion">Se connecter</a>
      </li>
      <li class="nav-item">
        <a class="nav-link disabled" routerLink="/produits">Liste des produits</a>
      </li>
    </ul>
    <ul *ngIf="user|async" class="navbar-nav">
      <li class="nav-item">
        <a class="nav-link active" routerLink="/produits">Liste des produit</a>
      </li>
      <li class="nav-item">
        <a class="nav-link active" (click)="deconnexion()">Déconnexion de {{user|async}}</a>
      </li>
    </ul>
  </div>
</nav>
```

3.12 src/app/menu/menu.component.ts

```
import { Component, OnInit } from '@angular/core';
import { AuthenticationService } from '../authentication.service';
import { Router } from '@angular/router';
import { Observable } from 'rxjs';

@Component({
  selector: 'app-menu',
  templateUrl: './menu.component.html',
  styleUrls: ['./menu.component.css']
})
export class MenuComponent implements OnInit {
  private user: Observable<string>;

  constructor(private authService: AuthenticationService,
    private router: Router) {
    this.user = this.authService.getUser(); }

  ngOnInit() {
    this.router.navigate(['/categories']);
  }

  deconnexion() {
    this.authService.disconnect();
    this.router.navigate(['/categories']);
  }
}
```

3.13 src/app/connexion/connexion.component.html

```
<form (ngSubmit)="onSubmit()" #inscriptionForm="ngForm">
  <label> Email </label>
  <input type="text" class="form-control" required [(ngModel)]="utilisateur['email']" name="email" />

  <label> Mot de passe </label>
  <input type="password" class="form-control" required
    [(ngModel)]="utilisateur['password']" name="password" />

  <input type="submit" value="Se connecter"/>
</form>
<p> {{message}} </p>
```

3.14 src/app/connexion/connexion.component.ts

```
import { Component } from '@angular/core';
import { Router } from '@angular/router';
import { AuthenticationService } from '../authentication.service'

@Component({
  selector: 'app-connexion',
  templateUrl: './connexion.component.html',
  styleUrls: ['./connexion.component.css']
})
export class ConnexionComponent {

  private utilisateur = {"email":"","password":""};
  private message: string = "";

  constructor(private authService: AuthenticationService,
    private router: Router) { }

  onSubmit() {
    this.authService.verificationConnexion(this.utilisateur).subscribe(reponse => {
      this.message = reponse['message'];
      if (reponse['resultat']) {
        this.authService.connect(this.utilisateur.email);
        this.router.navigate(['/categories']);
      }
      setTimeout( () => { this.router.navigate(['/categories']); }, 1000 );
    });
  }
}
```

3.15 src/app/categories/categories.component.html

```
<div *ngIf="!(user|async)">
  <p class="card-header mx-auto"> En vous connectant, découvrez nos : </p>
  <div class="card-body">
    <div *ngFor="let categorie of categories" class="card">
      <p class="card-text mx-auto"> {{categorie}}</p>
    </div>
  </div>
</div>
<div *ngIf="user|async">
  <div class="card-header">Vous êtes connecté :</div>
  <div class="card-body">
    <div *ngFor="let categorie of categories">
      <p class="card-text mx-auto"> <input type="button" value="Allez vite découvrir nos {{categorie}}"
        (click)="produitsParCategorie(categorie)"/> </p>
    </div>
  </div>
</div>
```

3.16 src/app/categories/categories.component.ts

```
import { Component, OnInit } from '@angular/core';
import { AuthenticationService } from '../authentication.service';
import { Router } from '@angular/router';
import { ProduitsService } from '../produits.service';
import { Observable } from 'rxjs';

@Component({
  selector: 'app-categories',
  templateUrl: './categories.component.html',
  styleUrls: ['./categories.component.css']
})
export class CategoriesComponent implements OnInit {

  private user: Observable<string>;
  private categories: String[] = new Array();

  constructor(private router: Router,
               private authService: AuthenticationService,
               private produitsService: ProduitsService) {
    this.user = this.authService.getUser();
  }

  ngOnInit() {
    this.produitsService.getCategories().subscribe(categories => {
      this.categories = categories;
    });
  }

  produitsParCategorie(categorie) {
    this.router.navigate(['/produits', categorie]);
  }
}
```

3.17 src/app/produits/produits.component.html

```
<div class="row justify-content-center">
  <div class="card-body">
    <div *ngFor="let produit of produits" class="card">
      <div class="card-text mx-auto"> {{produit['nom']}} ({{produit['type']}})
      <div *ngIf="(user|async)"> <button> mais achetez donc ! </button> </div>
    </div>
  </div>
</div>
```

3.18 src/app/produits/produits.component.ts

```
import { Component, OnInit } from '@angular/core';
import { AuthenticationService } from '../authentication.service';
import { ActivatedRoute, Params } from '@angular/router';
import { ProduitsService } from '../produits.service';
import { Observable } from 'rxjs';

@Component({
  selector: 'app-produits',
  templateUrl: '../produits.component.html',
  styleUrls: ['../produits.component.css']
})
export class ProduitsComponent implements OnInit {

  private user: Observable<string>;
  private produits: Object[] = new Array();

  constructor(private route: ActivatedRoute,
               private authService: AuthenticationService,
               private produitsService: ProduitsService) {
    this.user = this.authService.getUser();
  }

  ngOnInit() {
    this.route.params.subscribe((params :Params) => {
      console.log("Dans produits.component.ts avec "+params["categorie"]);
      if (params["categorie"] !== undefined) {
        console.log("/produits/"+params['categorie']);
        this.produitsService.getProduitsParCategorie(params["categorie"]).subscribe(produits => {
          this.produits = produits;
        });
      }
      else {
        this.produitsService.getProduits().subscribe(produits => {
          this.produits = produits;
        });
      }
    });
  }
}
```

3.19 Exécution du serveur de développement local Angular :

```
ng serve -o
```