



Toute l'actualité du Retrogaming !

Qui suis-je ?

- Stéphane PATERNA
- 45 ans
- Diplômé de l'INSEEC Bordeaux et de l'Université de Kassel (Allemagne) en 1998
- 15 ans d'expérience en recrutement en sociétés de conseil (Altran, Logica, Assystem...)
- 3 ans d'accompagnement de bénéficiaires du RSA

Simplon ?

- Créée en 2013
- Formations aux métiers du numérique
- Reconnue comme Grande École du Numérique
- Enseignement basé sur la pédagogie active
- Projets réalisés par équipes de 3 à 4 personnes
- Projet chef-d'œuvre en fin de formation
- Moyens mis en œuvre pour la bonne tenue de la formation malgré le confinement



Mon Projet

Le projet ActuRetro est la réalisation d'un site de e-commerce et d'information dédié au retrogaming.

Le retrogaming concerne l'histoire du jeu vidéo ainsi que toutes les plateformes de jeux neuves ou d'occasion permettant de jouer aux jeux vidéos dits « vintages ».



Pour qui ?

- ActuRetro est dédié aux gameuses et aux gamers qui souhaitent revivre des heures de plaisir sur les jeux de leur enfance et leur adolescence.
- ActuRetro est dédié aux personnes souhaitant revendre leurs anciens jeux vidéos ou leurs vieilles machines pour en retirer un bénéfice.
- ActuRetro est dédié aux collectionneurs d'anciennes machines et de vieux jeux, ou de machines plus récentes émulant des machines ou des jeux rétros.



ActuRetro

Pour quoi ?

- ActuRetro propose une plateforme de vente de consoles neuves, portables ou de salon, émulant une ou plusieurs machines rétros et d'anciens jeux vidéos.
- ActuRetro propose une plateforme d'achat et de revente d'anciennes machines et de jeux rétros.
- ActuRetro propose des articles sur l'actualité du retrogaming mais aussi sur l'histoire du jeu vidéo.

Où ?

- Les utilisateurs peuvent consulter le site, faire des achats de produits neufs, mettre en ligne des annonces et contacter des vendeurs potentiels depuis chez eux.
- Pour les clients, la contrainte géographique concerne la récupération des produits d'occasion.
- L'équipe de développeurs se trouvent chez Simplon

Quand ?

- La section e-commerce sera mise en ligne au plus tard le 14 octobre 2020 afin d'anticiper les fêtes de fin d'année.
- La rubrique sur l'actualité du retrogaming sera mise à jour au fil de l'eau.



ActuRetro

Comment ?

- La partie back-end sera développée en NodeJS.
- La partie front-end sera développée en ReactJS.
- La base de données sera développée en SQL.

Pourquoi ?

- Parce qu'il existe des milliers de jeux vidéos qui méritent le détour malgré le temps qui passe.
- Parce qu'il y a des milliers de gameuses et gamers qui aiment rejouer aux jeux de leur enfance, ou tout simplement à des jeux qu'ils aiment même s'ils ne sont pas les derniers sortis.
- Parce que le jeu vidéo est aujourd'hui un bien culturel majeur qui a une histoire, et que le retrogaming répond aujourd'hui à un besoin d'entretenir cette histoire comme pour la littérature, la BD ou le cinéma.



Vision Board

Vision

Devenir le site de e-commerce de référence dédié au **retrogaming**

Target Group

Tous les gamers et toutes les gameuses nostalgiques (avec peut-être une cible plus particulière sur les trentenaires, quarantenaires et cinquantenaires) souhaitant rejouer aux jeux vidéo de leur enfance ou de leur adolescence.

Needs

- Rejouer aux vieux jeux vidéo qui nous ont plu et fait briller les yeux lorsque nous étions plus jeunes
- Trouver des machines récentes ou plus anciennes nous permettant de rejouer à ces jeux
- Permettre à certaines personnes de se débarrasser de vieilles machines qu'elles n'utilisent plus et d'en tirer un bénéfice
- Permettre à certaines personnes de trouver des machines anciennes qu'elles ne trouveraient pas ailleurs et de compléter leur collection

Product

- Une partie du site sera dédiée à la vente de machines neuves qui émulent d'anciennes machines
- Une autre partie du site sera dédiée à l'achat et la revente de machines ou de jeux/logiciels d'occasion (~ « le Bon Coin du Retrogaming »)

Business Goals

- Marge sur la revente de produits neufs
- Publicité
- Proposer aux revendeurs qui le souhaitent de mettre plus d'informations ou de photos sur leurs produits moyennant un paiement

- MO5.COM

[FR](#)[EN](#)[JE FAIS UN DON](#)[ACCUEIL](#)[L'ASSOCIATION](#)[NOTRE ACTION](#)[ACTUALITÉ](#)[NOUS SOUTENIR](#)[CONTACT](#)[LE MAG MO5.COM](#)

**POUR UN MUSÉE NATIONAL
DES CULTURES NUMÉRIQUES**

**DEPUIS 2003, MO5.COM AGIT POUR LE PARTAGE ET LA
PRÉSERVATION DE NOTRE PATRIMOINE NUMÉRIQUE**

REJOIGNEZ-NOUS



ActuRetro

Benchmarking

- La Boutique du Retrogaming

The banner features a pixelated background of a waterfall and a stone archway. In the center, a pixelated Tails character is flying. The text '60 CONSOLES EN 1' and 'DE 22000 A 54000 JEUX' is displayed in a pixelated font. Below the text, there are icons for '1-4 JOUEURS', 'PLUG & PLAY', 'Wi Fi', 'Bluetooth', and 'HDMI'. At the bottom, there is a red button labeled 'Assistance'.

Boutique du Retrogaming

[Accueil](#) [Boutique](#) [Listes des jeux](#) [Questions / Réponses](#) [Médias](#) [Infos](#) [Retours](#) [Contact](#)  

60 CONSOLES EN 1
DE 22000 A 54000 JEUX

 **1-4 JOUEURS**  **PLUG & PLAY**  **Wi Fi**  **Bluetooth**  **HDMI**

 **Assistance**

- RetroGamePlace

BIENVENUE DANS VOTRE BOUTIQUE RETROGAMING!

**RETRO
GAME PLACE**
Retrogaming - Jeux d'occasion - Mangas

 MON COMPTE  PANIER



[ACCUEIL](#) [NINTENDO](#) [SEGA](#) [SONY](#) [XBOX](#) [SNK](#) [8-BIT](#) [3DO](#) [GUIDES / MAGAZINES](#) [ESPACE MANGA](#)

[GOODIES - PRODUITS DÉRIVÉS](#)

ACCUEIL / CONSOLE SUPER NINTENDO

JEUX ET CONSOLES

- NINTENDO
 - NES
 - Super Nintendo
 - Famicom
 - Super Famicom
 - Super NES (US)
 - Game Boy
 - Game Boy Color
 - Game Boy Advance
 - Nintendo 64
 - Gamecube
 - Nintendo DS



RETRO GAME PLACE

PLUS DE VUES



Console Super Nintendo

★★★★★ 5 Avis **99,99 €**

EN STOCK

Editeur : Nintendo

Etat : Console en bon état. Peut comporter quelques jaunissures.

Qté :

AJOUTER AU PANIER



PAIEMENT SECURISE

Colissimo®
Une livraison neutre en CO₂

LA POSTE

Lettre Suivie

EN

Avis Vérifiés

AVIS DE NOS CLIENTS
★★★★★ 4.7/5



ActuRetro

Benchmarking

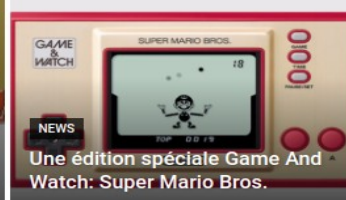
- Retrogaming.fr

[Accueil](#) [Participer](#) [Forum](#) [A propos](#) [Contact](#) [Castlevania](#) [Link to the past](#)



RETROGAMING.FR

[NEWS](#) [TESTS](#) [DOSSIERS](#) [TUTOS](#) [ÉVÉNEMENTS](#) [MUSÉE](#) [LECTURES](#) [VIDÉOS](#)





ActuRetro

Benchmarking

L'objectif d'ActuRetro est d'être en quelque sorte une synthèse de ces différents sites en proposant à la fois :

- Des produits neufs,
- Des produits d'occasion,
- L'actualité du retrogaming.



ActuRetro Persona Primaire

- **Nom** : Roberto MICALIZZI, 45 ans, Développeur Web, marié, 2 enfants, 1 chien.
- **Ses passions** : le foot, les femmes, la bière...et surtout les jeux vidéos qui lui rappellent son enfance.
- **Son but** : s'amuser le plus possible dans toutes ses activités qu'elles soient personnelles ou professionnelles.
- **Ses frustrations** : quand son équipe de foot préférée perd un match, de devoir attendre pour s'offrir (ou se faire offrir) la dernière console de jeux sur laquelle il a craqué, perdre à FIFA contre son cousin Sandro.
- **Ses motivations** : l'argent, la reconnaissance sociale.





ActuRetro Parcours Utilisateur

User-journey :

- il va consulter les articles sur les dernières consoles sorties,
- il va se créer un compte pour avoir accès à l'ensemble des services,
- il va se connecter via son compte,
- il va remplir sa liste d'envies en fonction des produits qui lui plaisent,
- il va effectuer un/des achat(s),
- il va consulter les dernières annonces mises en ligne par les autres utilisateurs,
- il va prendre contact avec les autres utilisateurs afin d'acheter les produits d'occasion qui ont retenu son attention.



ActuRetro Persona Secondaire

- **Nom** : Simone GAUTHIER, 38 ans, femmes de chambre dans un grand hôtel parisien, divorcée, 3 enfants, 2 chiens.
- **Ses passions** : les échecs, la poterie, la danse, les chiens.
- **Son but** : trouver l'amour sur Meetic, prendre soin de ses enfants et de ses chiens.
- **Ses frustrations** : la vie en général.
- **Ses motivations** : le bonheur de ses enfants, l'argent.





ActuRetro Parcours Utilisateur

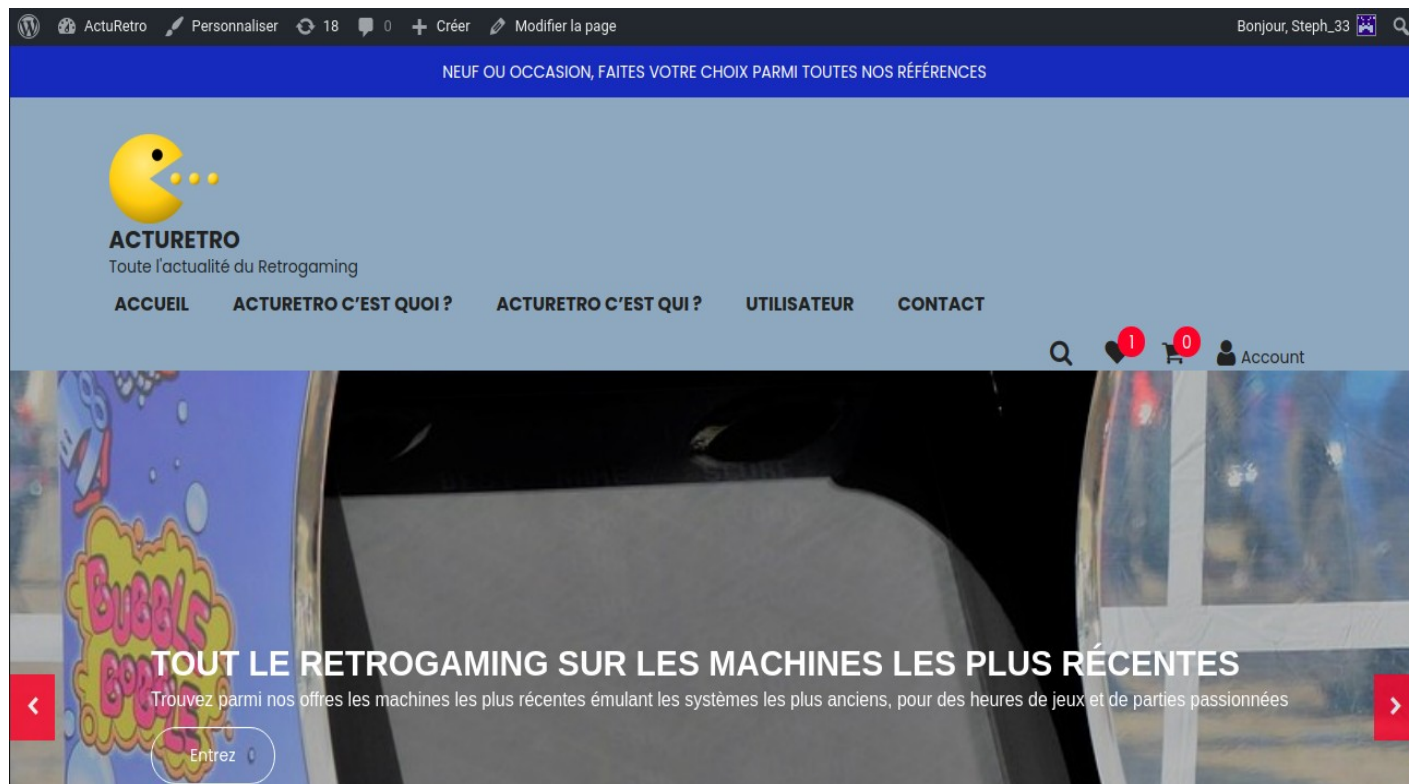
User-journey :

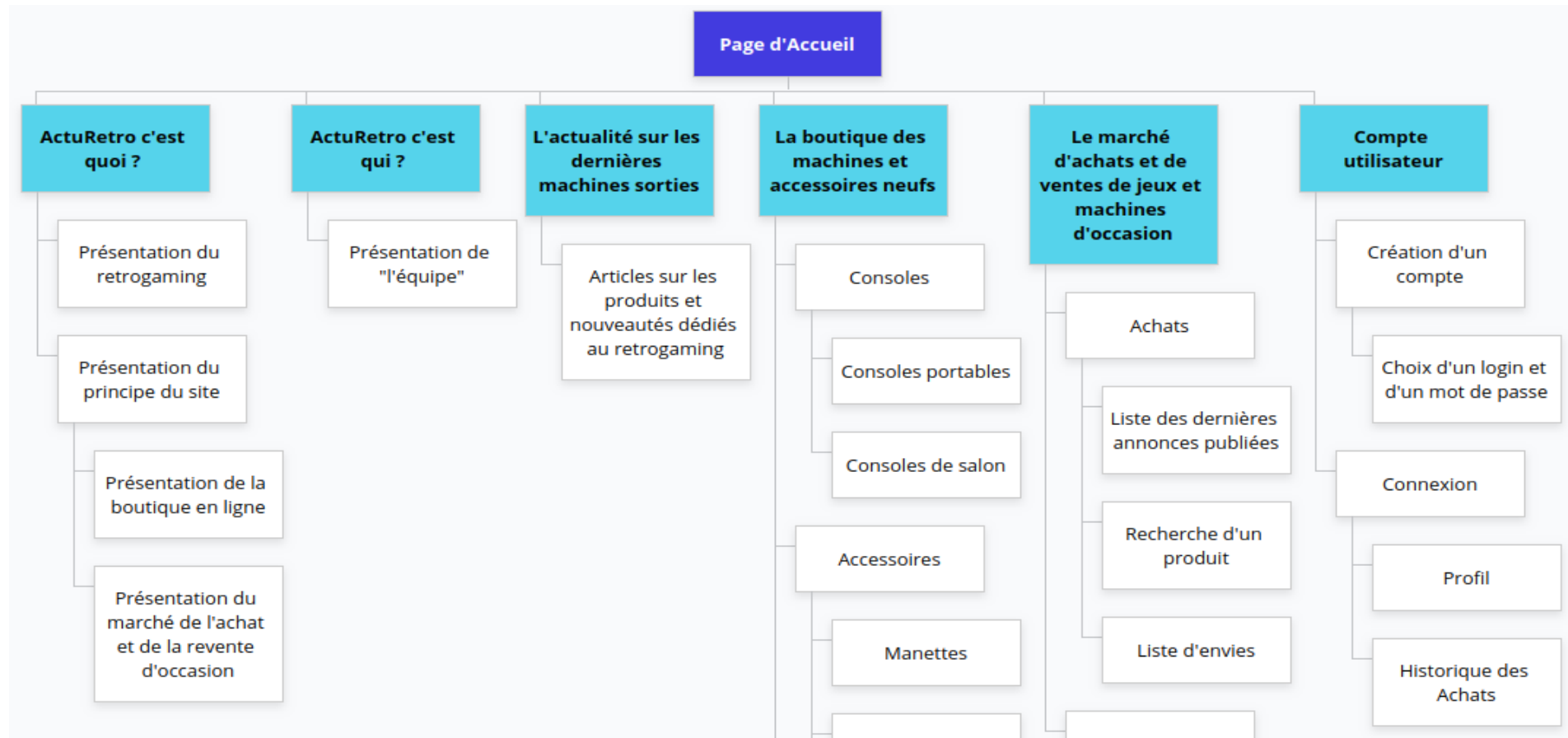
- elle va prendre connaissance du site et de son fonctionnement,
- elle va se créer un compte pour pouvoir se connecter afin de revendre de vieux jeux ou du vieux matériel,
- elle va se connecter,
- elle va mettre en vente les vieux jeux rétros et la PS2 de son ex-mari qui encombrant le grenier,
- elle va mettre en vente les vieilles consoles de son fils qui n'a plus d'yeux que pour la future PS5.

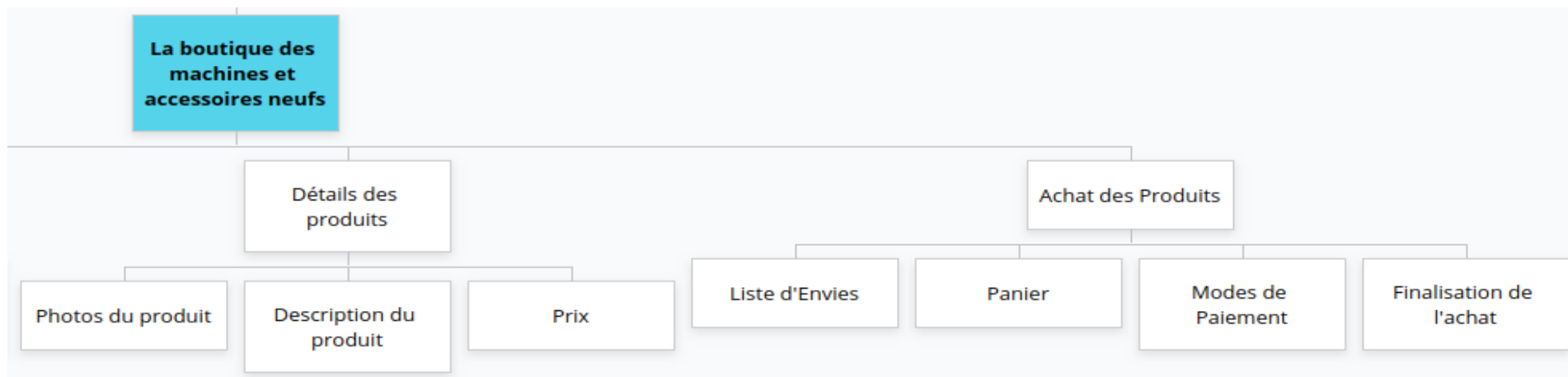
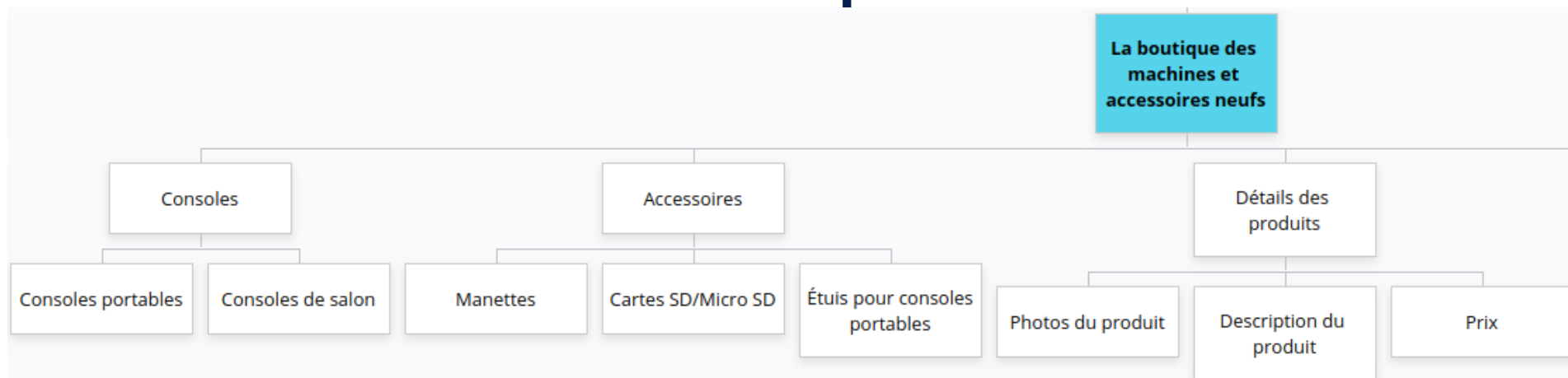


ActuRetro Projet Initial WordPress

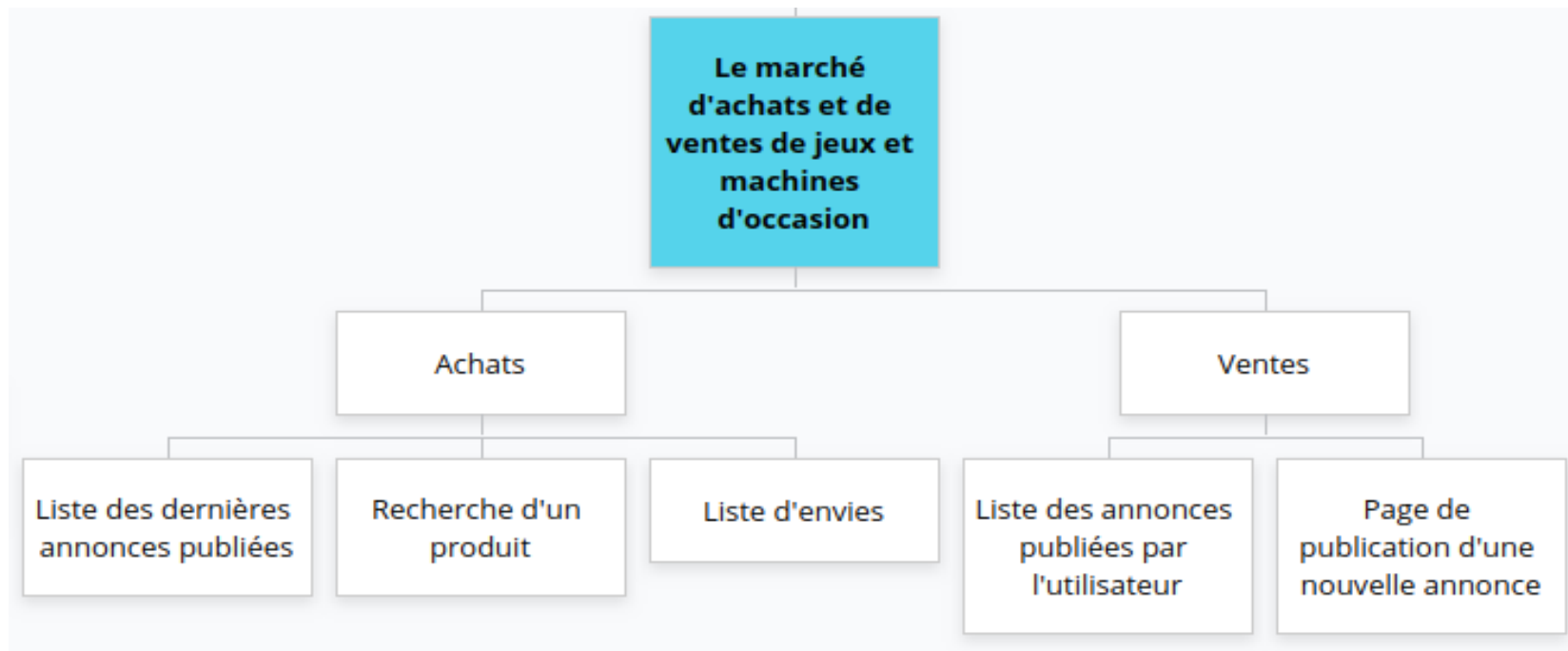
Réalisé en binôme sur le CMS WordPress







SiteMap 3/3





ActuRetro Charte Graphique

Choix de couleurs sobres pour éviter les confusions dues au trop grand nombre de couleurs :



ActuRetro



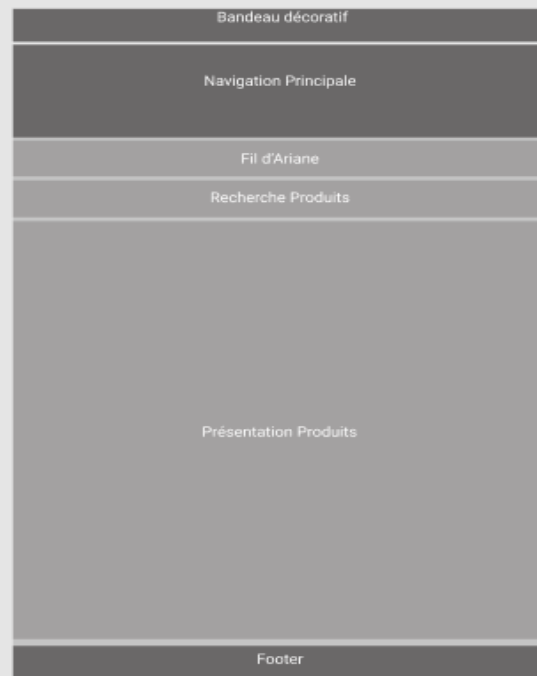
ActuRetro

Le Zoning

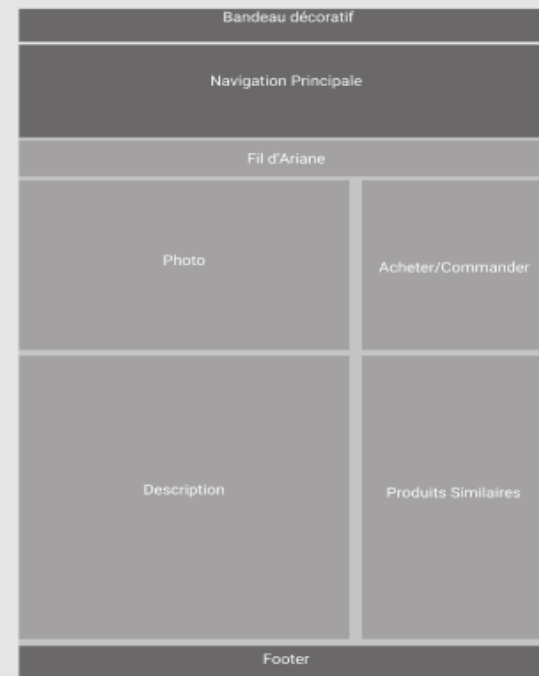
Zoning Page d'Accueil

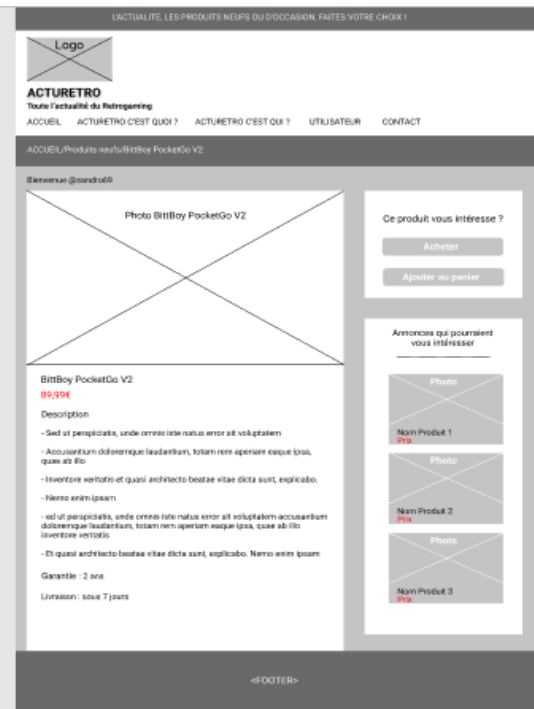
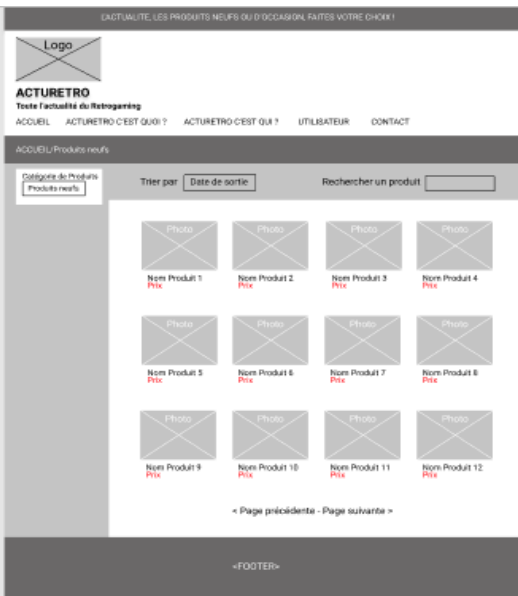
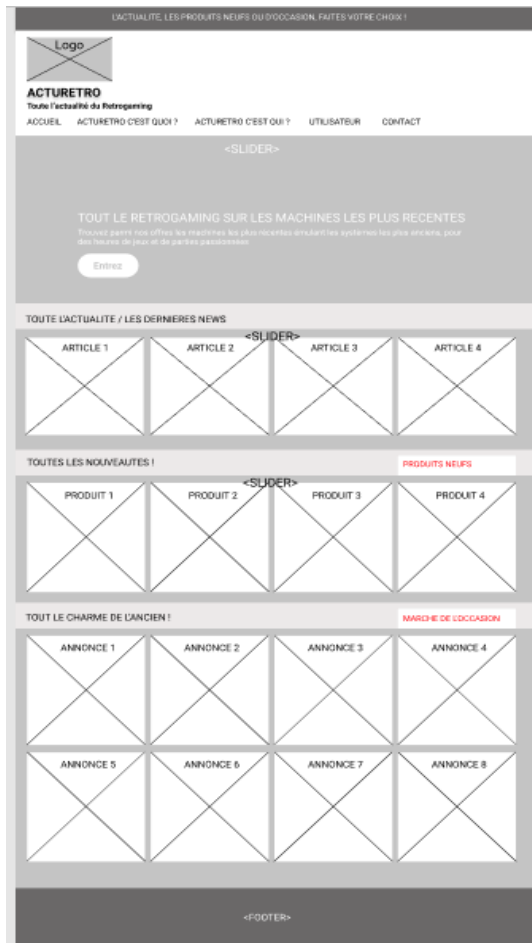


Zoning Présentation Produits Neufs



Zoning Détail Produits Neufs







ActuRetro

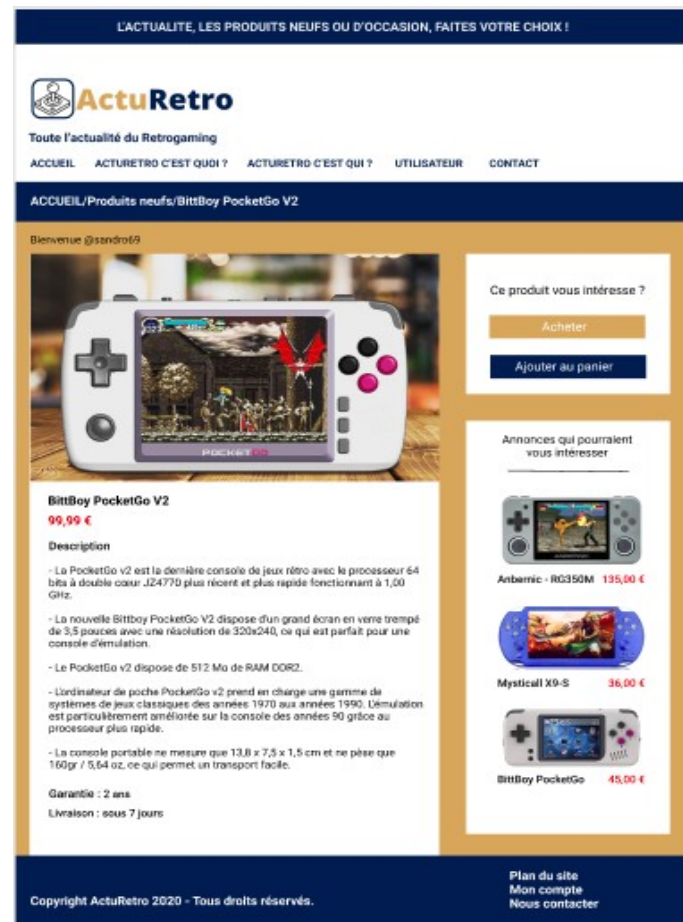
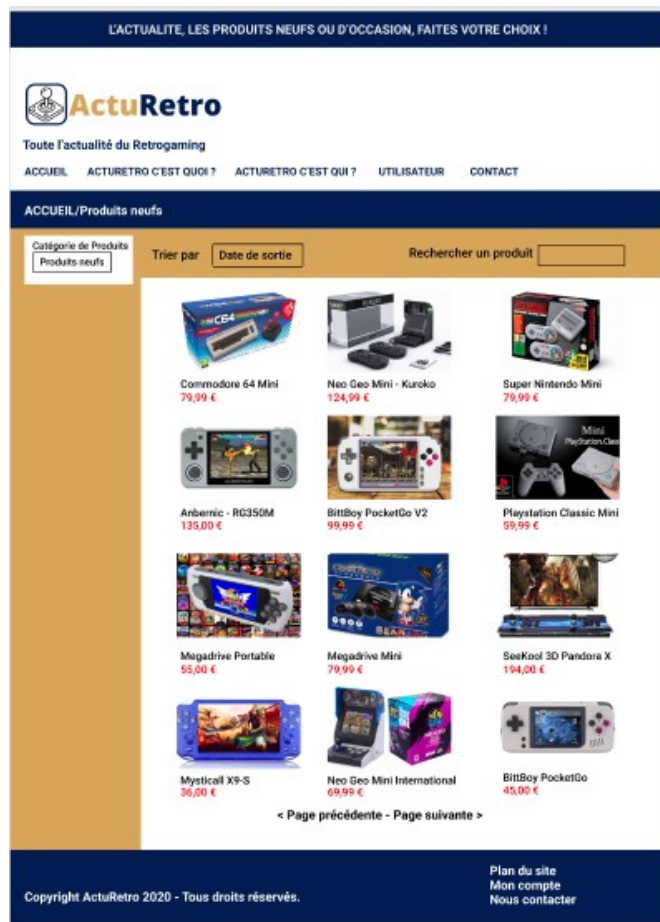
La maquette

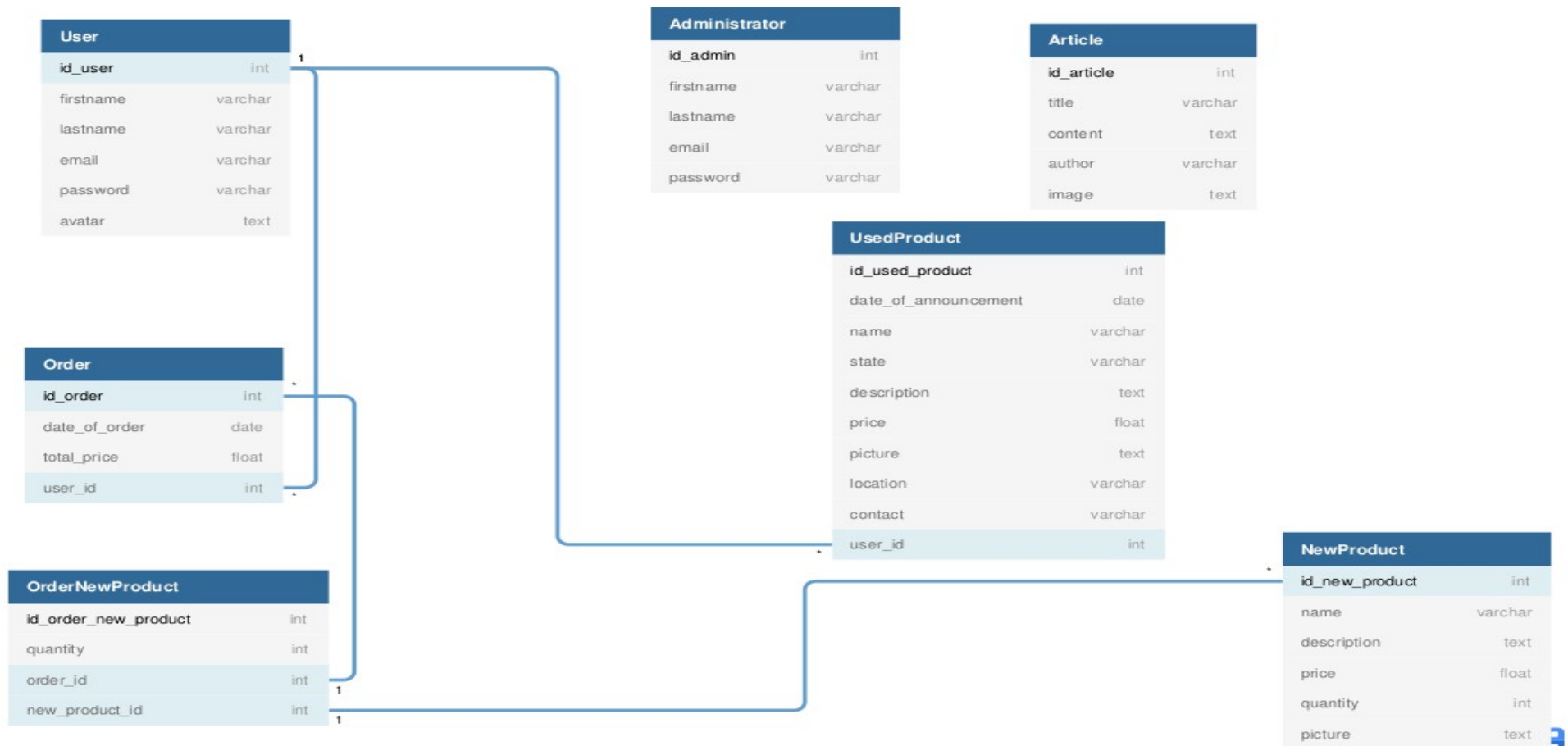


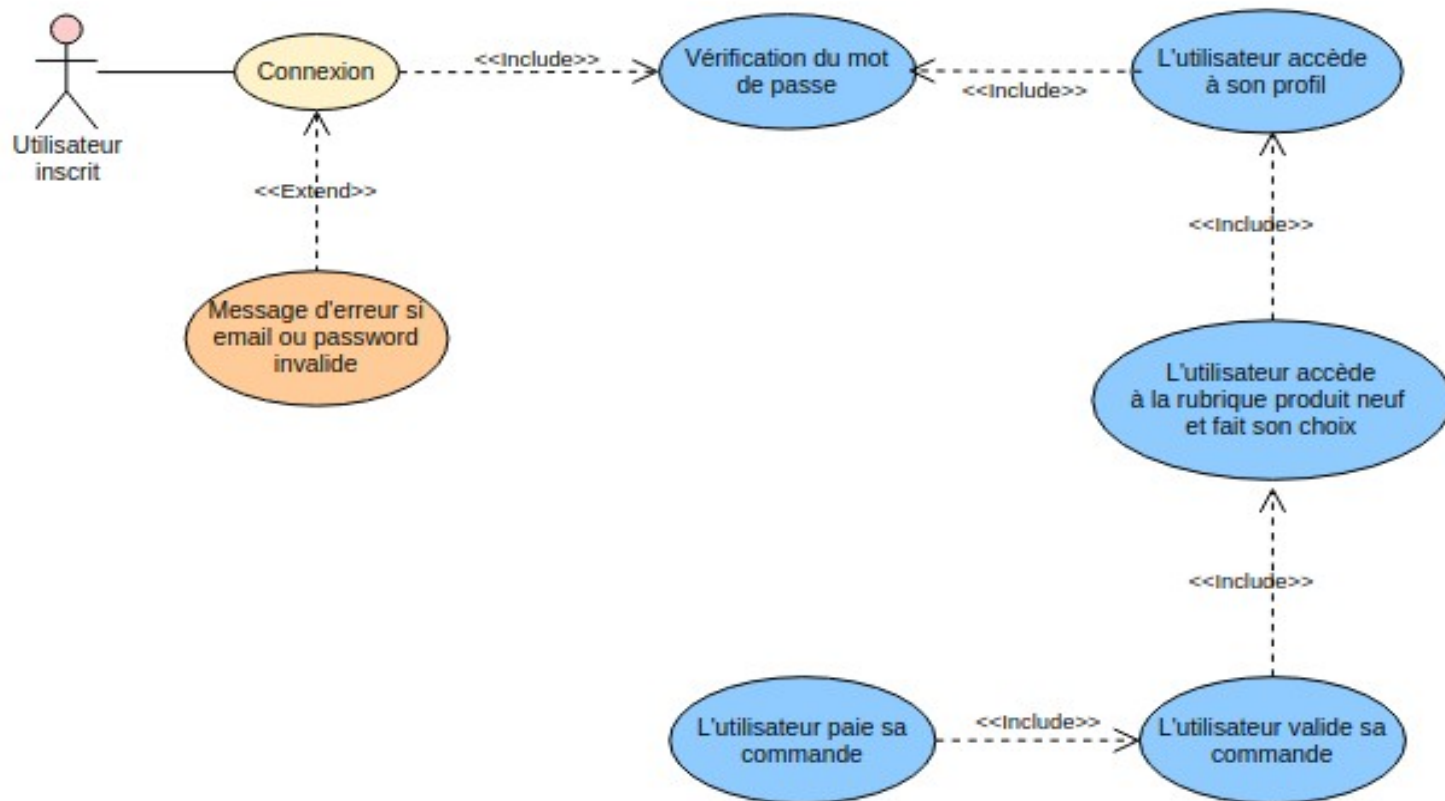


ActuRetro

La maquette







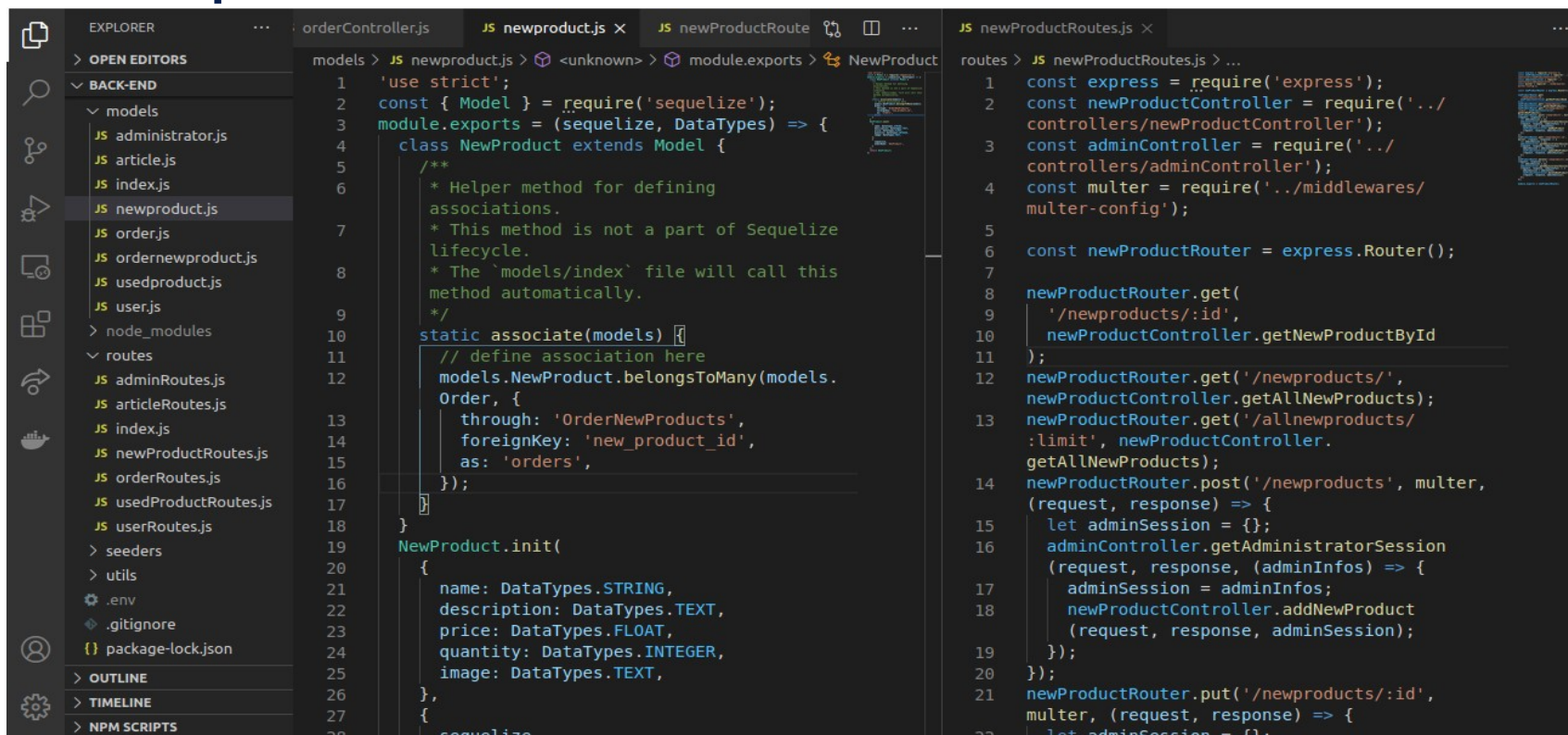


Le Back-end

Le back-end est développé sous NodeJS avec l'utilisation de l'ORM Sequelize pour la gestion de la relation à la base de données :

A screenshot of the Visual Studio Code editor interface. The Explorer sidebar on the left shows the project structure with folders like 'config', 'controllers', 'images', 'middlewares', 'migrations', 'models', 'node_modules', 'routes', 'seeders', 'utils', and files like '.env', '.gitignore', 'package-lock.json', 'package.json', and 'server.js'. The 'controllers' folder is expanded, showing several JavaScript files, with 'orderController.js' selected. The main editor area displays the code for 'orderController.js'. It includes imports for 'models', 'express', 'express-async-errors', 'jwtUtils', and 'ConflictError'. The 'module.exports' object contains an 'addOrder' function that is asynchronous. This function takes 'request', 'response', and 'userId' as arguments, creates an 'order' object with 'date_of_order', 'products', and 'total_price' from the request body, and then checks for null values in the order object. If any field is null, it returns a 400 status with an error message. Finally, it calls 'models.Order.create' to save the order. The bottom status bar shows '1: bash' and a terminal window with a SQL query: 'Executing (default): SELECT `id`, `name`, `description`, `price`, `quantity`, `image` FROM `NewProducts` AS `NewProduct` WHERE'.

Exemple de modèles et de routes :



```
EXPLORER
  > OPEN EDITORS
  > BACK-END
    > models
      JS administrator.js
      JS article.js
      JS index.js
      JS newproduct.js
      JS order.js
      JS ordernewproduct.js
      JS usedproduct.js
      JS user.js
    > node_modules
    > routes
      JS adminRoutes.js
      JS articleRoutes.js
      JS index.js
      JS newProductRoutes.js
      JS orderRoutes.js
      JS usedProductRoutes.js
      JS userRoutes.js
    > seeders
    > utils
    .env
    .gitignore
    {} package-lock.json
  > OUTLINE
  > TIMELINE
  > NPM SCRIPTS

models > JS newproduct.js > <unknown> > module.exports > NewProduct
1 'use strict';
2 const { Model } = require('sequelize');
3 module.exports = (sequelize, DataTypes) => {
4   class NewProduct extends Model {
5     /**
6      * Helper method for defining
7      * associations.
8      * This method is not a part of Sequelize
9      * lifecycle.
10     * The 'models/index' file will call this
11     * method automatically.
12     */
13     static associate(models) {
14       // define association here
15       models.NewProduct.belongsToMany(models.
16         Order, {
17           through: 'OrderNewProducts',
18           foreignkey: 'new_product_id',
19           as: 'orders',
20         });
21     }
22   }
23   NewProduct.init(
24     {
25       name: DataTypes.STRING,
26       description: DataTypes.TEXT,
27       price: DataTypes.FLOAT,
28       quantity: DataTypes.INTEGER,
29       image: DataTypes.TEXT,
30     },
31     {
32       sequelize
33     }
34   );
35 }

routes > JS newProductRoutes.js > ...
1 const express = require('express');
2 const newProductController = require('../
3 controllers/newProductController');
4 const adminController = require('../
5 controllers/adminController');
6 const multer = require('../middlewares/
7 multer-config');
8
9 const newProductRouter = express.Router();
10
11 newProductRouter.get(
12   '/newproducts/:id',
13   newProductController.getNewProductById
14 );
15
16 newProductRouter.get('/newproducts/',
17   newProductController.getAllNewProducts);
18 newProductRouter.get('/allnewproducts/
19 :limit', newProductController.
20 getAllNewProducts);
21 newProductRouter.post('/newproducts', multer,
22 (request, response) => {
23   let adminSession = {};
24   adminController.getAdministratorSession
25 (request, response, (adminInfos) => {
26   adminSession = adminInfos;
27   newProductController.addNewProduct
28 (request, response, adminSession);
29 });
30 });
31 newProductRouter.put('/newproducts/:id',
32 multer, (request, response) => {
33   let adminSession = {};
```



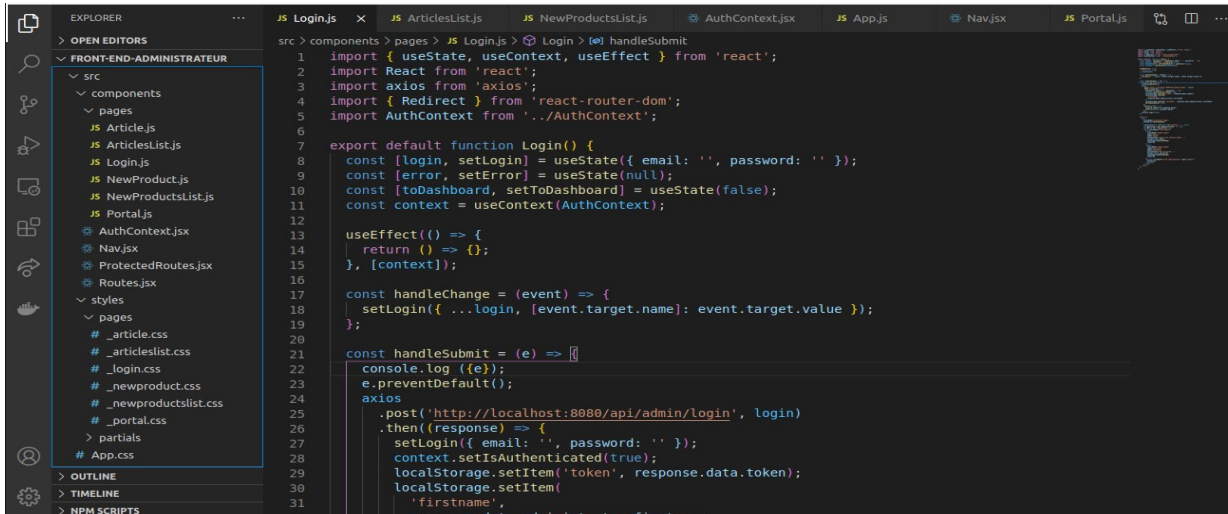
ActuRetro

Le Front-end

Il y a 2 front-ends sur le projet ActuRetro :

- Un pour l'administrateur du site, afin de lui permettre de mettre celui-ci à jour,
- Un pour les visiteurs et utilisateurs du site. Il s'agit ici de la partie front-end du site en tant que tel.

Le front-end administrateur comprend, les différentes pages en JavaScript, les composants qui correspondent à ces pages en JSX, et les feuilles de styles en CSS.



The screenshot shows a code editor with the Explorer panel on the left and the code editor on the right. The Explorer panel shows the file structure of the project, with the 'pages' folder under 'components' expanded. The code editor shows the code for 'Login.js'.

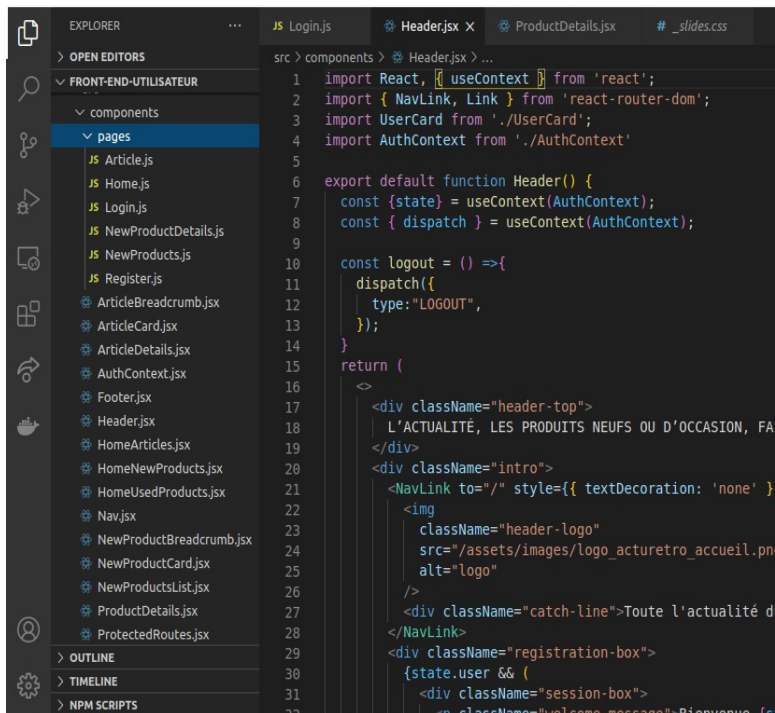
```
src > components > pages > JS Login.js > Login > handleSubmit
1 import { useState, useContext, useEffect } from 'react';
2 import React from 'react';
3 import axios from 'axios';
4 import { Redirect } from 'react-router-dom';
5 import AuthContext from '../AuthContext';
6
7 export default function Login() {
8   const [login, setLogin] = useState({ email: '', password: '' });
9   const [error, setError] = useState(null);
10  const [toDashboard, setToDashboard] = useState(false);
11  const context = useContext(AuthContext);
12
13  useEffect(() => {
14    return () => {};
15  }, [context]);
16
17  const handleChange = (event) => {
18    setLogin({ ...login, [event.target.name]: event.target.value });
19  };
20
21  const handleSubmit = (e) => {
22    console.log(e);
23    e.preventDefault();
24    axios
25      .post('http://localhost:8080/api/admin/login', login)
26      .then((response) => {
27        setLogin({ email: '', password: '' });
28        context.setIsAuthenticated(true);
29        localStorage.setItem('token', response.data.token);
30        localStorage.setItem(
31          'firstname',
32          response.data.administrator.firstname
```



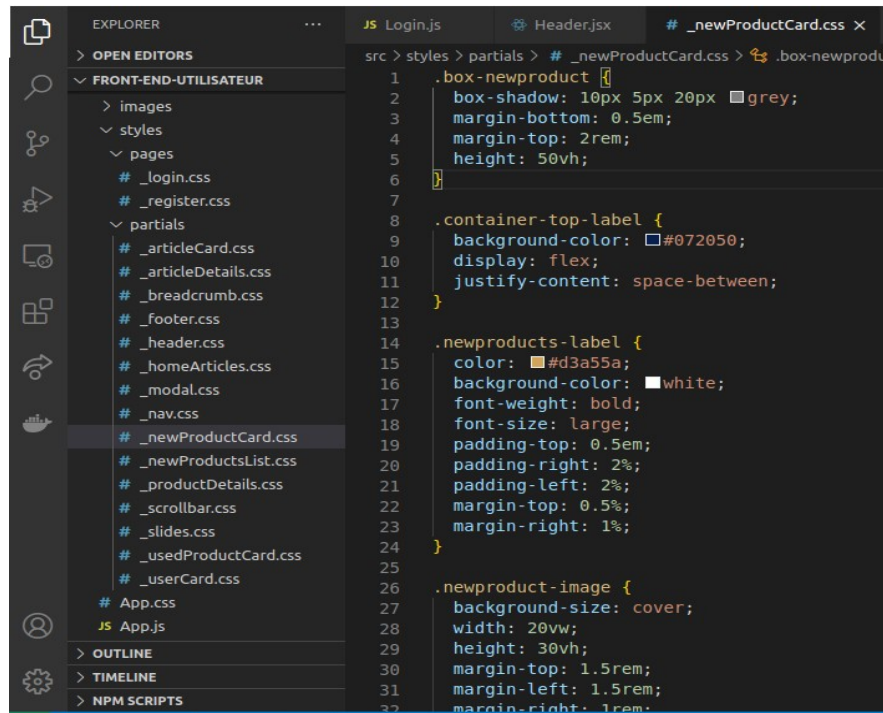
ActuRetro

Le Front-end

Le front-end utilisateur comprend plus d'éléments mais est basé sur le même modèle :



```
1 import React, { useContext } from 'react';
2 import { NavLink, Link } from 'react-router-dom';
3 import UserCard from './UserCard';
4 import AuthContext from './AuthContext'
5
6 export default function Header() {
7   const {state} = useContext(AuthContext);
8   const { dispatch } = useContext(AuthContext);
9
10  const logout = () =>{
11    dispatch({
12      type:"LOGOUT",
13    });
14  }
15  return (
16    <div className="header-top">
17      L'ACTUALITÉ, LES PRODUITS NEUFS OU D'OCCASION, FA
18    </div>
19    <div className="intro">
20      <NavLink to="/" style={{ textDecoratoin: 'none' }}>
21        
26      </NavLink>
27      <div className="catch-line">Toute l'actualité de
28    </div>
29    <div className="registration-box">
30      {state.user && (
31        <div className="session-box">
32          <div className="welcome-message">Bienvenue le
```



```
1 .box-newproduct {
2   box-shadow: 10px 5px 20px grey;
3   margin-bottom: 0.5em;
4   margin-top: 2rem;
5   height: 50vh;
6 }
7
8 .container-top-label {
9   background-color: #072050;
10  display: flex;
11  justify-content: space-between;
12 }
13
14 .newproducts-label {
15   color: #d3a55a;
16   background-color: white;
17   font-weight: bold;
18   font-size: large;
19   padding-top: 0.5em;
20   padding-right: 2%;
21   padding-left: 2%;
22   margin-top: 0.5%;
23   margin-right: 1%;
24 }
25
26 .newproduct-image {
27   background-size: cover;
28   width: 20vw;
29   height: 30vh;
30   margin-top: 1.5rem;
31   margin-left: 1.5rem;
32   margin-right: 1rem;
```



ActuRetro

Démonstration

Passons à la pratique !!



ActuRetro

Démonstration

- Front-end administrateur : ajout et suppression d'un article
- Front-end utilisateur : création d'un compte et connexion
- Front-end utilisateur : achat d'un produit neuf
- Front-end utilisateur : diffusion d'une annonce
- Version Mobile
- Exemple de CRUD

- Création d'un compte et hachage du password

```
JS jwt.utils.js JS userController.js X JS adminRoutes.js JS multer-config.js JS usedProductRoutes.js
controllers > JS userController.js > [?] <unknown> > [?] register > [?] bcrypt.hash() callback
19 register: async (request, response) => {
20   const user = {
21     firstname: request.body.firstname,
22     lastname: request.body.lastname,
23     email: request.body.email,
24     password: request.body.password,
25     image: `${request.protocol}://${request.get('host')}/images/${
26       request.file.filename
27     }`,
28   };
29   const checkEmail = /^((([^\<>()\\[\]\/.,;:\s@"])+(\.[^\<>()\\[\]\/.,;:\s@"]+)*)(\.[^\<>()\\[\]\/.,;:\s@"])+[^\<>()\\[\]\/.,;:\s@"])|([^\<>()\\[\]\/.,;:\s@"])+[^\<>()\\[\]\/.,;:\s@"])$$/;
30   const checkName = /\b([A-ZÀ-Ÿ][-,a-z. ']+[ ]*)+/;
31   for (const key in user) {
32     if (user[key] == null) {
33       throw new BadRequest(
34         'Mauvaise Requête',
35         `Le champs ${key} n'est pas renseigné ✖`
36       );
37     }
38   }
39   for (const key in user) {
40     if (!isString(user[key])) {
41       throw new BadRequest(
42         'Mauvaise Requête',
43         `Le champs ${key} n'est pas une chaîne de caractères. ✖`
44       );
45     }
46   }
47   if (!checkName.test(user.firstname)) {
48     throw new BadRequest(
49       'Mauvaise Requête',
50       `Le champ firstname est mal renseigné. Vous devez rentrer un prénom valide. ✖`
51     );
52   }
53 }
```

- Création d'un compte et hachage du password

```
JS jwt.utils.js JS userController.js X JS adminRoutes.js JS multer-config.js JS usedProductRoutes.js
controllers > JS userController.js > [?] <unknown> > [?] register > [?] bcrypt.hash() callback
65 // TODO : check forms
66 const userFound = await models.User.findOne({
67   attributes: ['email'],
68   where: { email: user.email },
69 });
70 if (!userFound) {
71   bcrypt.hash(user.password, 5, async (err, bcryptedPassword) => {
72     const newUser = await models.User.create({
73       firstname: user.firstname,
74       lastname: user.lastname,
75       email: user.email,
76       password: bcryptedPassword,
77       image: user.image,
78     });
79     if (newUser) {
80       return response.status(201).json({
81         firstname: newUser.firstname,
82         lastname: newUser.lastname,
83         email: newUser.email,
84       });
85     } else {
86       throw new ServerError(
87         'Erreur Serveur',
88         "Impossible d'ajouter cet utilisateur. ✖"
89       );
90     }
91   });
92 } else {
93   throw new ConflictError(
94     'Mauvaise Requête',
95     'Un utilisateur possédant le même email existe déjà. ✖'
96   );
97 }
```


- Création d'un compte et hachage du password

SELECT * FROM `Users`

☐ Profilage [Éditer en ligne] [Éditer] [Expliquer SQL] [Créer le code source PHP] [Actualiser]

☐ Tout afficher | Nombre de lignes : 25 | Filtrer les lignes: Chercher dans cette table | Trier sur l'index: Aucun(e)

Options

							password	
<input type="checkbox"/>	Éditer	Copier	Supprimer	4	Paterna	Stéphane	paterna@club-internet.fr	\$2b\$05\$fiLwtoDNKHAI2xFdhfuKDun9BHQ/VX./MTZjf..dZjL...
<input type="checkbox"/>	Éditer	Copier	Supprimer	6	Mbappé	Kylian	kylian@gmail.com	\$2b\$05\$soqhcovTd.icDGO8Fv7SSau97odEtWAXaBmSR/CqYWPF...
<input type="checkbox"/>	Éditer	Copier	Supprimer	7	Ibrahimovic	Zlatan	zlatan@gmail.com	\$2b\$05\$AbmUCnqKIMAOw9CNBaWEO.XSyAa5T87yuYNo9ca.T4o...

☐ Tout cocher | Avec la sélection : Éditer | Copier | Supprimer | Exporter

- Les JSON Web Token

Lors du projet ActuRetro, je me suis servi des JSON Web Tokens, ou JWT, pour sécuriser mes authentifications et la gestion des autorisations.

Un JSON Web Token est une norme ouverte qui définit un moyen compact et autonome de transmettre en toute sécurité des informations entre différentes parties à l'aide d'un objet JSON.

- Les JSON Web Token

Un token JWT se compose de 3 chaînes de caractères séparées par un point(.) :

- 1- L'en-tête : décrit le type de token et l'algorithme utilisé pour la signature,
- 2- Le payload : contient les informations à transmettre (la date d'expiration, la cible, le sujet...).
- 3- La signature : permet de s'assurer de l'authenticité du token.



- Les JSON Web Token

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.} en-tête

eyJ1c2VySWQiOiJEsInJvbGUiOiJob3N0IiwiaWF0IjoxNTk4MzYxMTg2LCJleHAiOiJ
E1OTgzNjQ3ODZ9.} payload

SFcNSgqNzgQ3yKsmPUX3xFOQ4IjSBGAYOGQ-ZuhiJUc} signature

- Les JSON Web Token

```
JS jwt.utils.js x  .env  JS userController.js  JS adminRoutes.js  JS multer-config.js  JS
utils > JS jwt.utils.js > [?] <unknown> > generateTokenForUser
1  const jwt = require('jsonwebtoken');
2  require('express-async-errors');
3
4  const BadRequest = require('../utils/errors/bad_request');
5
6  const JWT_SIGN_SECRET = process.env.SESSION_SECRET;
7
8  module.exports = {
9    generateTokenForUser: (userData) => {
10      return jwt.sign(
11        {
12          userId: userData.id,
13        },
14        JWT_SIGN_SECRET,
15        {
16          expiresIn: '1h',
17        }
18      );
19    }
20  };
21
```

- Les JSON Web Token

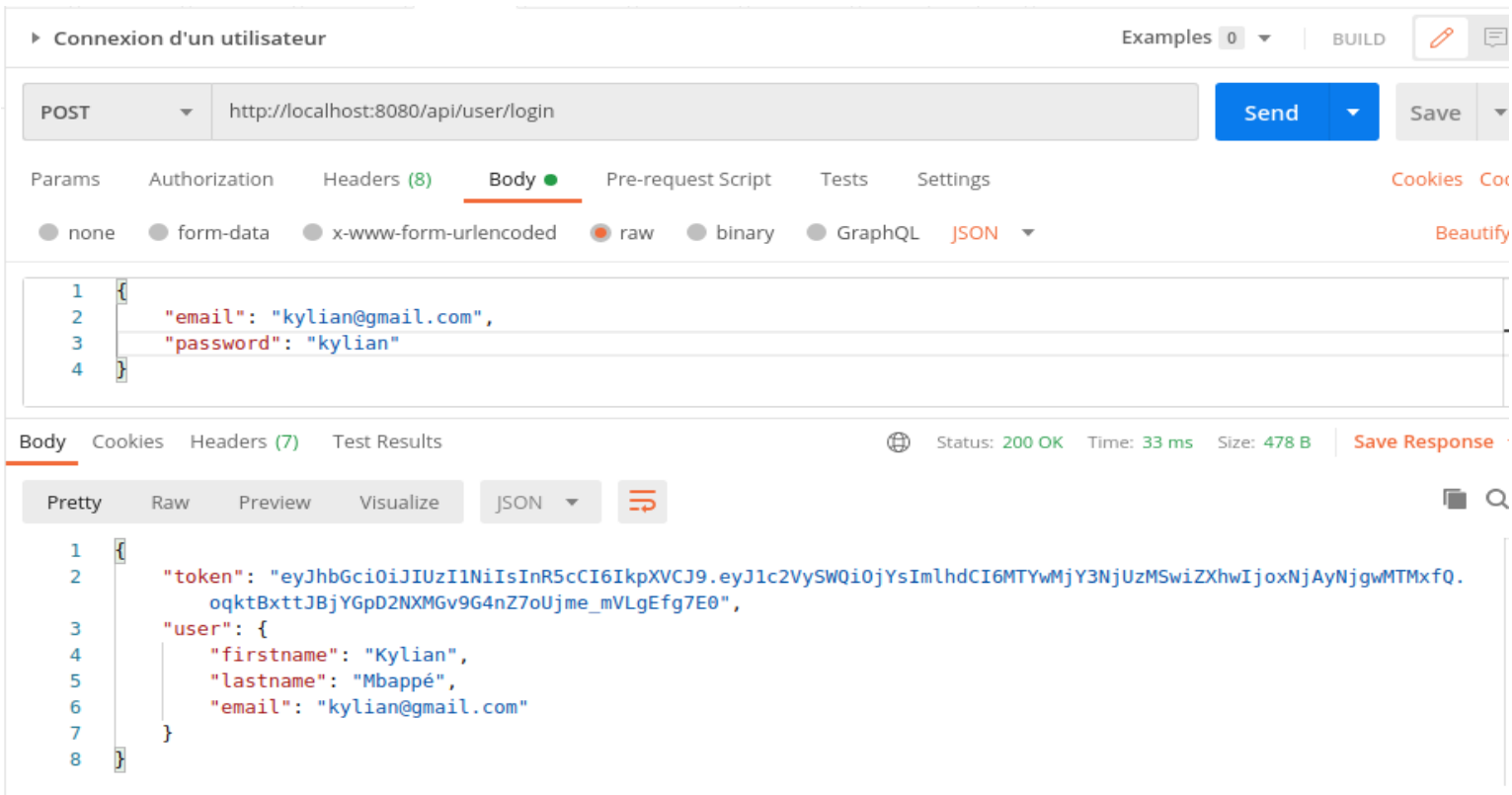
```
JS jwt.utils.js X .env JS userController.js JS adminRoutes.js JS multer-config.js JS usedPro
utils > JS jwt.utils.js > [?] <unknown> > generateTokenForUser
30
31 parseAuthorization: (authorization) => {
32   return authorization !== null ? authorization.replace('Bearer ', '') : null;
33 },
34 getUserId: (authorization, response) => {
35   let userId = -1;
36   const token = module.exports.parseAuthorization(authorization);
37   if (token) {
38     try {
39       const jwtToken = jwt.verify(token, JWT_SIGN_SECRET);
40       if (!jwtToken) {
41         return response.status(401).json({
42           error: 'Token nul lors de la vérification ! ✖',
43         });
44       }
45       userId = jwtToken.userId;
46     } catch (e) {
47       if (e instanceof jwt.JsonWebTokenError) {
48         return response.status(401).json({
49           error: 'Token invalide lors de la vérification ! ✖',
50         });
51       }
52     }
53   }
54   return userId;
55 }
```



- Les JSON Web Token

```
JS jwt.utils.js JS user.js JS userController.js x JS adminRoutes.js JS multer-config.js JS usedProductRoutes.js
controllers > JS userController.js > [x] <unknown> > [x] register > [x] bcrypt.hash() callback
99 login: async (request, response) => {
100   const userInfos = {
101     email: request.body.email,
102     password: request.body.password,
103   };
104   for (const key in userInfos) {
105     if (userInfos[key] == null) {
106       throw new BadRequest('Mauvaise Requête', `Le champs ${key} n'est pas renseigné ✖`);
107     }
108   }
109   const userFound = await models.User.findOne({
110     where: { email: userInfos.email },
111   });
112   if (userFound) {
113     bcrypt.compare(
114       userInfos.password,
115       userFound.password,
116       (errBycrypt, resBycrypt) => {
117         const userToken = {
118           firstname: userFound.firstname,
119           lastname: userFound.lastname,
120           email: userFound.email,
121         };
122         if (resBycrypt) {
123           return response.status(200).json({
124             token: jwtUtils.generateTokenForUser(userFound),
125             user: userToken,
126           });
127         }
128         return response.status(403).json({ error: 'Mot de passe incorrect ! ✖', });
129       }
130     );
131   }
132 }
```

- Les JSON Web Token



The screenshot displays a REST client interface for a login endpoint. The request is a POST to `http://localhost:8080/api/user/login` with a JSON body containing email and password. The response is a 200 OK status with a JSON body containing a token and user details.

Request:

- Method: POST
- URL: `http://localhost:8080/api/user/login`
- Body (JSON):

```
{ 1: { 2:   "email": "kylian@gmail.com", 3:   "password": "kylian" 4: }
```

Response:

- Status: 200 OK
- Time: 33 ms
- Size: 478 B
- Body (JSON):

```
1 { 2:   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VySWQiOiJYsImldhdCI6MTYwMjY3NjUzMSwiZXhwIjoxNjYyNjgwMTMxMD0. 3:   "user": { 4:     "firstname": "Kylia", 5:     "lastname": "Mbappé", 6:     "email": "kylian@gmail.com" 7:   } 8: }
```




ActuRetro

Site Anglophone

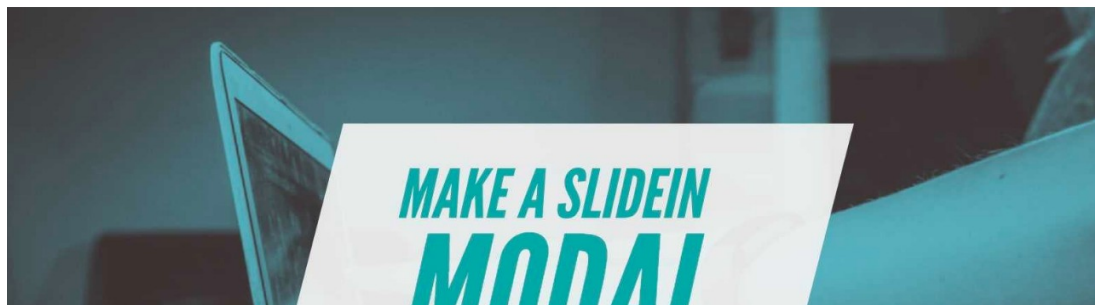
Pour la réalisation de la modale au moment de l'achat, je me suis servi du site

<https://yamagata-developers-society.github.io/>
avec un article ayant pour titre « React Hooks
Slide In Modal » :



13 DECEMBER 2019 / ADVENT2019

React Hooks Slide In Modal





Cet article explique comment setter ma modale grâce au Hook `useState()`, comment la faire apparaître, puis comment la refermer :

Setting our state

To start, let's set our state value and update function:

```
const [displayModal, setDisplayModal] = useState(false);
```

We set the initial state to `false`, since the slide-in modal will only display after the user clicks on a button.

Modal styles and animations

I will keep the styling for the modal fairly simple:

```
.Modal {  
  position: fixed;  
  bottom: -150vh;  
  background-color: #fff;  
  width: 100%;  
  left: 0;  
  padding: 0 12px 12px;
```



Conditional rendering

There are two ways we can go about this. The first is most basic:

```
{
  displayModal && <div className="Modal">...modal content</div>;
}
```

This is a concept in React known as “conditional rendering”. If the value of `displayModal` is set to true, then the element will display on the page. If false, the component will not be rendered at all.

This is very useful for many different situations. However, one limitation to this approach is that we can’t animate the component as its display state is toggled on and off (without using `react-transition-group` or another animation library).

To add animations then, we can use conditional rendering of class names! I will reformat the above code as follows:



ActuRetro

Site Anglophone

Closing the modal

Of course if we want our modal to be useful, we need a way to open and close it, right?

Let's create a button with a click handler which will toggle our `displayModal` state:

```
<button className="Button CenterAlign" onClick={() => setDisplayModal(!displayModal)}>  
  Settings  
</button>
```

We simply use the `onClick` attribute, and our `setDisplayModal()` state updater function.

To close the modal, we can create two more click handlers:

The first will be a close icon within the modal:

```
<button className="Close" onClick={() => setDisplayModal(!displayModal)}>  
  X  
</button>
```

We can set another on the overlay component so that when a user clicks on the background they can close the modal as well:

Difficultés rencontrées :

Un certain nombre de fonctionnalités, notamment sur React m'ont demandé beaucoup de recherches, de travail et de patience pour trouver les solutions appropriées :

- La connexion de l'administrateur ou de l'utilisateur,
- L'achat d'un produit par l'utilisateur et le fait de lui renvoyer une modale « conditionnée » par le fait qu'il soit connecté ou non,
- Je n'ai pour le moment pas réussi à faire le slider de la Home Page que je voulais faire,

Autre difficulté : une fois le projet lancé, il était parfois difficile de faire une priorisation des tâches et d'évaluer la quantité de travail qu'elles allaient demander.



Conclusion

Satisfactions :

Je suis heureux et fier d'avoir réussi à faire un site qui ressemble à ce que j'avais en tête au départ.

J'ai conscience qu'il y a encore beaucoup à faire pour l'améliorer, mais les principales fonctionnalités sont là.

Et maintenant...

Mon objectif est de trouver un emploi de Développeur Web et de vivre enfin de cette passion.

Merci pour votre attention !

