

<b>BTS SERVICES INFORMATIQUES AUX ORGANISATIONS</b>	<b>SESSION 2025</b>
<b>ANNEXE 9-1-B : Fiche descriptive de réalisation professionnelle (recto)</b>	
<b>Épreuve E6 - Conception et développement d'applications (option SLAM)</b>	

<b>DESCRIPTION D'UNE RÉALISATION PROFESSIONNELLE</b>		<b>N° réalisation : 2</b>
<b>Nom, prénom : LAMONNIER Alexis</b>		<b>N° candidat : 02445945427</b>
<b>Épreuve ponctuelle</b> <input checked="" type="checkbox"/>	<b>Contrôle en cours de formation</b> <input type="checkbox"/>	<b>Date : 07 / 05 / 2025</b>
<b>Organisation support de la réalisation professionnelle</b> La Brasserie Terroir & Savoirs désire moderniser son image par le biais du numérique. Elle souhaite mettre en place une nouvelle application mobile, permettant aux utilisateur de faire des commandes de produit		
<b>Intitulé de la réalisation professionnelle</b> Dans le cadre de la préparation au épreuve E6, j'ai développé une application Mobile en Flutter, pour répondre au cahier des charges de notre client. J'ai donc réalisé une application permettant aux utilisateurs de créer son compte, de voir les produits disponible, de faire des commandes de produits, de voir ses reservation, De plus, il est possible de changer son mot de passe.  Le tout est lié à une base de données MySQL		
<b>Période de réalisation : 09/04/2025 - 21/05/2025</b> <b>Lieu : EPSI Lille</b> <b>Modalité :</b> <input checked="" type="checkbox"/> <b>Seul(e)</b> <input type="checkbox"/> <b>En équipe</b>		
<b>Compétences travaillées</b> <input checked="" type="checkbox"/> Concevoir et développer une solution applicative <input checked="" type="checkbox"/> Assurer la maintenance corrective ou évolutive d'une solution applicative <input checked="" type="checkbox"/> Gérer les données		
<b>Conditions de réalisation <sup>5</sup> (ressources fournies, résultats attendus)</b> Pour réaliser le projet, j'ai eu recours à : des cours sur Flutter, des cours sur les API REST, un serveur Web, un cahier des charges. L'objectif étant de réaliser une application mobile qui communique, via des API, avec une Base de données à distance		
<b>Description des ressources documentaires, matérielles et logicielles utilisées <sup>6</sup></b> IDE : Visual Studio Code Gestionnaire de version : Github MySQL Technologies utilisées : Dart/Flutter, SQL Outils de test : Postman, tests unitaire		
<b>Modalités d'accès aux productions <sup>7</sup> et à leur documentation <sup>8</sup></b> Lien GitHub : <a href="https://github.com/alexkjzz/brasserie/tree/main/mobile">https://github.com/alexkjzz/brasserie/tree/main/mobile</a>  Login/mdp : alexis.lamonnier@gmail.com/racine pour l'administration & test@test.test/racine pour un utilisateur		

<sup>5</sup> En référence aux *conditions de réalisation et ressources nécessaires* du bloc « Conception et développement d'applications » prévues dans le référentiel de certification du BTS SIO.

<sup>6</sup> Les réalisations professionnelles sont élaborées dans un environnement technologique conforme à l'annexe II.E du référentiel du BTS SIO.

<sup>7</sup> Conformément au référentiel du BTS SIO « *Dans tous les cas, les candidats doivent se munir des outils et ressources techniques nécessaires au déroulement de l'épreuve. Ils sont seuls responsables de la disponibilité et de la mise en œuvre de ces outils et ressources. La circulaire nationale d'organisation précise les conditions matérielles de déroulement des interrogations et les pénalités à appliquer aux candidats qui ne se seraient pas munis des éléments nécessaires au déroulement de l'épreuve.* ». Les éléments peuvent être un identifiant, un mot de passe, une adresse réticulaire (URL) d'un espace de stockage et de la présentation de l'organisation du stockage.

<sup>8</sup> Lien vers la documentation complète, précisant et décrivant, si cela n'a été fait au verso de la fiche, la réalisation professionnelle, par exe

## **Sommaire :**

*1-Présentation*

*2-Aperçu visuel du projet*

*3-Base de donnée*

*4-Détails techniques*

# 1- Présentation

Dans le cadre de la préparation au épreuve E6, j'ai développé une application Mobile en Flutter, pour répondre au cahier des charges de notre client.

J'ai donc réalisé une application de prise de commande. Il est possible de créer son compte , un utilisateur peut passer des commandes de produit

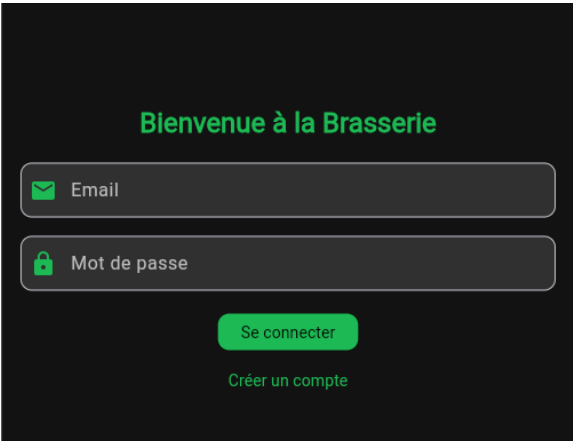
Le tout est lié à une base de données MySQL

## 2- Aperçu visuel du projet

Pour commencer, je vous présente les différents écrans, ainsi que certains aspects visuelles,

### Login Page

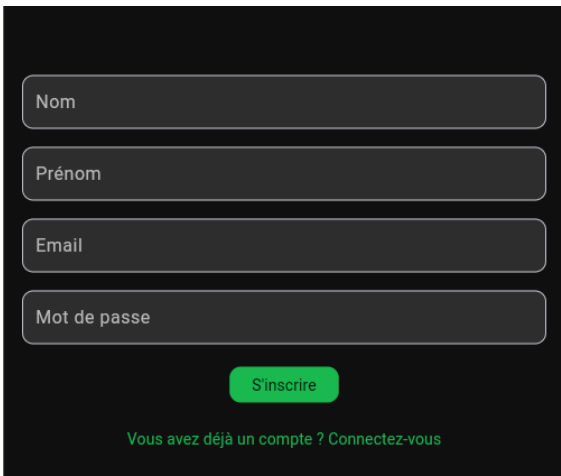
*à l'arrivée sur l'application , il sera proposé de se connecter via un formulaire d'authentification, si l'utilisateur n'a pas de compte, il pourra créer son compte via le lien en dessous le redirigeant vers Register. le login nécessite d'utiliser le Mail utilisé lors de l'enregistrement du compte*

A screenshot of the login page of a mobile application. The background is dark. At the top, the text "Bienvenue à la Brasserie" is displayed in a light green font. Below this, there are two input fields: the first is labeled "Email" with a green envelope icon, and the second is labeled "Mot de passe" with a green lock icon. Below the input fields, there is a green button labeled "Se connecter". At the bottom, there is a link labeled "Créer un compte" in a light green font.

### Register page

*Une fois sur la page Register, l'utilisateur à la possibilité de créer son compte, à savoir qu'une adresse mail ne peut pas être utilisée dans deux comptes différents.*

*Si toutefois l'utilisateur arrive sur cette page mais qu'il a déjà un compte il pourra cliquer sur le lien le redirigeant vers la page de Login*

A screenshot of the register page of a mobile application. The background is dark. There are four input fields stacked vertically: "Nom", "Prénom", "Email", and "Mot de passe". Below the input fields, there is a green button labeled "S'inscrire". At the bottom, there is a link labeled "Vous avez déjà un compte ? Connectez-vous" in a light green font.

## ANNEXE 9-1-B : Fiche descriptive de réalisation professionnelle

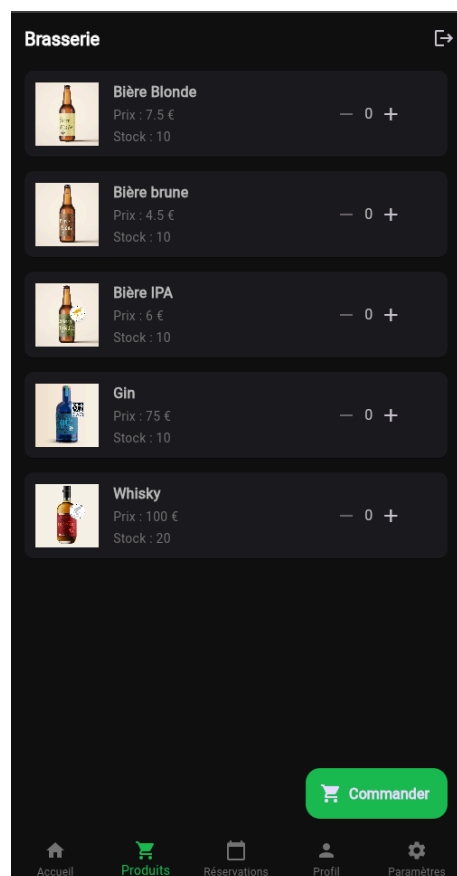
## Accueil

Une fois connecté, l'utilisateur arrive sur la page d'accueil présentant la brasserie, en haut à droite de son écran il pourra se déconnecter via un bouton présent dans la app bar, grâce à la navbar il peut naviguer entre les pages tels que "produit", "réservation", "profil" ou encore voir ses paramètres.



## Produit

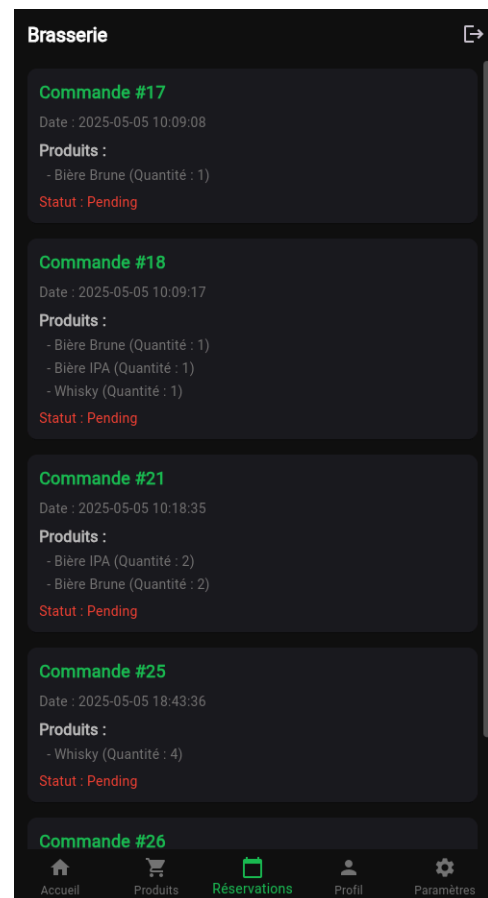
Dans cette page il est possible de voir la liste des produits en augmentant la quantité sélectionnée via les boutons prévu à droite de chaque produit, l'utilisateur va pouvoir faire une commande avec ceux-ci en appuyant sur le bouton "commander". De plus si un produit n'est plus en stock il passera en "indisponible" empêchant d'avoir un stock négatif.



## ANNEXE 9-1-B : Fiche descriptive de réalisation professionnelle

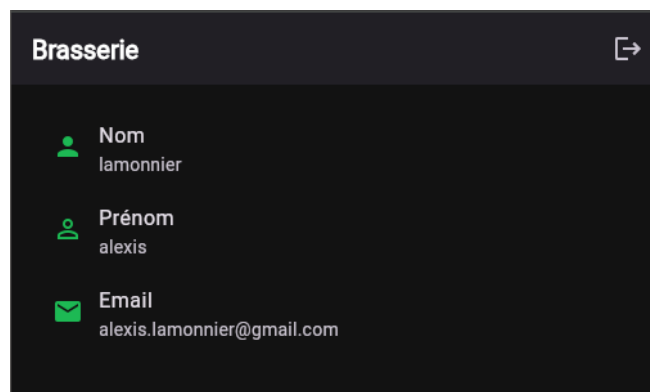
## Réservation

*Il est possible de voir ses réservations en cours, avec le numéros de la commande dans la BDD, son détail avec la liste des produits, leur quantité et leurs prix. De plus, le statut de la commande est visible, montrant si la commande est en attente, en cours d'envoi ou récupérée ( Le statut se met par défaut en Pending lors d'une commande )*



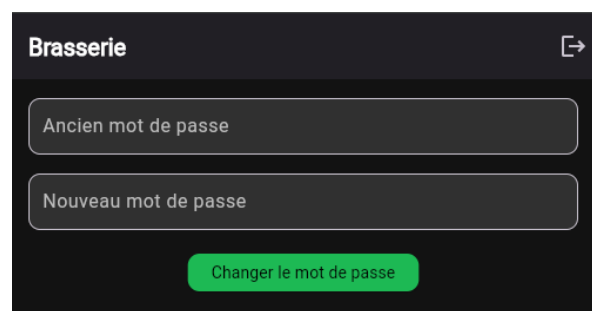
## Profil

Via la page Profil l'utilisateur peut voir ses informations, tel que son nom prénom et son email



## Paramètre

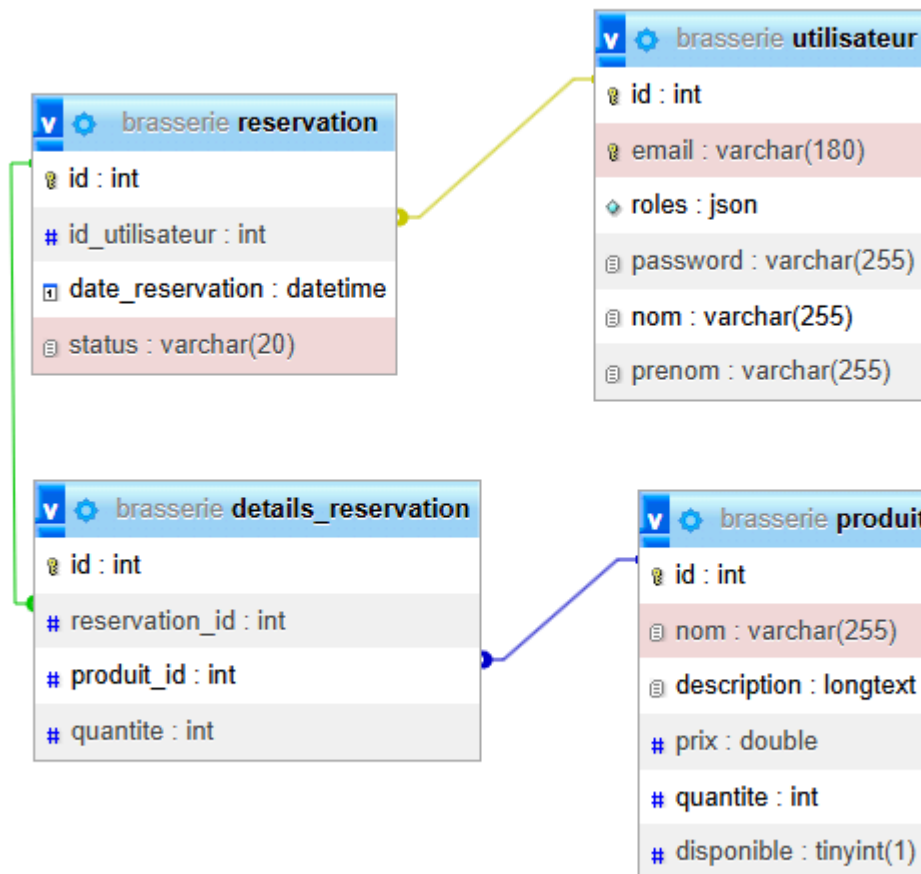
*Le formulaire permet à l'utilisateur de changer son mot de passe en indiquant son ancien mot de passe et le nouveau qu'il souhaite utiliser*



## ANNEXE 9-1-B : Fiche descriptive de réalisation professionnelle

### 3- Base de données

Schéma de la BDD dans l'interface SupaBase



### Lien github vers les controllers

**Authentification** : <https://github.com/alexkizz/brasserie/blob/main/api/src/Controller/AuthController.php>

**Détails des réservations** : <https://github.com/alexkizz/brasserie/blob/main/api/src/Controller/DetailsReservationController.php>

**Produits** : <https://github.com/alexkizz/brasserie/blob/main/api/src/Controller/ProduitController.php>

**Réservations** : <https://github.com/alexkizz/brasserie/blob/main/api/src/Controller/ReservationController.php>

**Utilisateurs** : <https://github.com/alexkizz/brasserie/blob/main/api/src/Controller/UtilisateurController.php>

### Lien github vers les services API

*vous pourrez retrouver tous les appels liée à l'API , tel que l'authentification, la récupération des réservations, la création des réservation, l'affichage des informations utilisateurs et le changement de mot de passe*

**Appels API** : [https://github.com/alexkizz/brasserie/blob/main/mobile/lib/services/api\\_service.dart](https://github.com/alexkizz/brasserie/blob/main/mobile/lib/services/api_service.dart)

## 4- Détails techniques

*J'utilise un user provider pour associer l'utilisateur à un token à sa connexion ( lors d'un login valide via le formulaire ),*

```
void setUser(int userId, String token) {  
    _userId = userId;  
    _token = token;  
    notifyListeners();  
}
```

*je peux remettre à 0 ce token pour gérer la déconnexion ( qui sera appelé lors d'un click sur le bouton "déconnexion" )*

```
void clearUser() {  
    _userId = null;  
    _token = null;  
    notifyListeners();  
}  
  
// Méthode pour déconnecter  
void disconnect() {  
    clearUser();  
}
```

*les commandes se font via un appel du bouton "commander" dans la page produits où la quantités sera lu pour savoir si un produit à été ajouté ou non et en quelle quantité :*

*ligne 23 on a l'appel à au service pour fetch les produits*

```
Future<void> _fetchProducts() async {  
    setState(() {  
        _isLoading = true;  
        _errorMessage = null;  
    });  
  
    try {  
        final userProvider = Provider.of<UserProvider>(context, listen: false);  
        final products = await ApiService.fetchProducts(userProvider.token!);  
        setState(() {  
            _products = products;  
        });  
    } catch (e) {  
        setState(() {  
            _errorMessage = e.toString();  
        });  
    } finally {  
        setState(() {  
            _isLoading = false;  
        });  
    }  
}
```

## ANNEXE 9-1-B : Fiche descriptive de réalisation professionnelle

*ligne 46 l'appel permettant de passer la réservation.*

```
Future<void> _placeOrder() async {}  
final userProvider = Provider.of<UserProvider>(context, listen: false);  
final orderItems = _quantities.entries  
  .where((entry) => entry.value > 0) // Inclure uniquement les produits avec une quantité > 0  
  .map((entry) => {"id": entry.key, "quantite": entry.value})  
  .toList();  
  
if (orderItems.isEmpty) {  
  ScaffoldMessenger.of(context).showSnackBar(  
    SnackBar(content: Text("Veuillez sélectionner au moins un produit.")),  
  );  
  return;  
}
```

*pour voir plus de détails :*

*page github de la page product :*

[https://github.com/alexkjzz/brasserie/blob/main/mobile/lib/pages/products\\_page.dart](https://github.com/alexkjzz/brasserie/blob/main/mobile/lib/pages/products_page.dart)

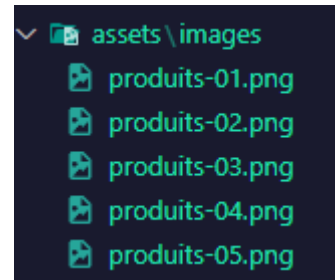
*L'affichage des Réservation se fait via un appel au service api permettant de voir les réservations selon l'id de l'utilisateur, lui permettant ainsi de voir uniquement ses réservations en y affichant le détails de sa réservation ( produit, quantité , statut et l'id de sa commande )*

```
Future<void> _fetchDetails() async {  
  setState(() {  
    _isLoading = true;  
    _errorMessage = null;  
  });  
  
  try {  
    final userProvider = Provider.of<UserProvider>(context, listen: false);  
    final details = await ApiService.fetchUserReservations(userProvider.token!);  
    setState(() {  
      _details = details;  
    });  
  } catch (e) {  
    setState(() {  
      _errorMessage = e.toString();  
    });  
  } finally {}  
  setState(() {  
    _isLoading = false;  
  });  
}
```



## ANNEXE 9-1-B : Fiche descriptive de réalisation professionnelle

*L'affichage de mes images est géré via FTP en les nommant produits-{id} elles s'adaptent directement au produit ayant l'id correspondant*



```
children: [
  Image.asset(
    'assets/images/produits-${product['id'].toString().padLeft(2, '0')}.png',
    width: 70,
    height: 70,
    fit: BoxFit.cover,
    errorBuilder: (context, error, stackTrace) =>
      Icon(Icons.image_not_supported, size: 50),
  ) // Image.asset
```

*Ligne 105 de la page :*

[https://github.com/alexkjzz/brasserie/blob/main/mobile/lib/pages/products\\_page.dart](https://github.com/alexkjzz/brasserie/blob/main/mobile/lib/pages/products_page.dart)