



Rouen Normandie

Projet annuel
Application de gestion de diabète sur mobile



Le diabète sous contrôle

Étudiants :

Anis Hanniz
Badre-Eddine Ben Hammou
Cheikh Ahmadou Diop
Abissi Kotchipka Ismael
Joshua Bitho

Enseignant :

Stéphane Hérauville

Table des matières

| | |
|--|----|
| Table des matières | 2 |
| 1 Bases du projet | 3 |
| 1.1 Introduction | 3 |
| 1.2 Définitions | 4 |
| Scénario MyPod.doc : | 5 |
| Scénario MyPod: | 6 |
| Simulation Pompe (Pod): | 7 |
| 1.3 Diagrammes | 8 |
| 1.4 Organisation du groupe | 9 |
| 1.5 Chartes graphique | 13 |
| 2 Conception | 14 |
| 2.1 MyPOD | 14 |
| Menu de débogage : | 14 |
| Splash Screen : | 14 |
| Page de première connexion : | 14 |
| Page de réinitialisation de Mot de Passe : | 15 |
| Page de connexion (login) : | 15 |
| Page d'accueil | 16 |
| Menu Popup | 17 |
| Utilisation Bluetooth | 19 |
| Commande Bolus | 20 |
| Guide de déploiement | 21 |
| 2.2 MyPOD_Doc | 22 |
| 2.3 API | 25 |
| 3 Test | 27 |

1 Bases du projet

1.1 Introduction

Notre rapport présente un projet de développement qui nous tient particulièrement à cœur. Initialement proposé à l'université, ce projet revêt une importance personnelle pour nous.

Il s'agit de la conception d'une application mobile médicale nommée "Mypod", destinée aux personnes atteintes de diabète pour commander leur pompe à insuline.

Ce projet a demandé plusieurs mois de travail et bien que nous ayons accompli beaucoup, il reste encore des éléments à perfectionner.

Suite à l'approbation de notre proposition, nous avons structuré notre travail en trois parties distinctes : une application web dédiée aux médecins, appelée "Mypod_site", pour la gestion de leurs patients ; une application mobile pour les patients, simplement nommée "Mypod" ; et enfin, un simulateur de pompe à insuline disponible sur mobile, "Podapp".

L'application web repose sur du PHP/MySQL, tandis que les deux applications mobiles sont développées en Flutter/Dart avec SQLite comme SGBD.

Pour garantir une interaction fluide avec la base de données du serveur à partir de ces applications, nous avons développé une API que nous avons nommée "MypodAPI".

Dans ce rapport, nous détaillerons les différentes phases de développement de chaque composant de ce projet, ainsi que les défis rencontrés et les solutions adoptées pour les surmonter.

1.2 Définitions

MyPod : Une application médicale dédiée à la gestion du diabète.

mypod.doc : Une plateforme web destinée aux médecins diabétologues pour la gestion efficace de leurs patients.

Pod : Application qui simule une pompe à insuline.

Injection Bolus:

Un apport manuel d'insuline généralement significatif (grande quantité).

L'injection Bolus sera stockée dans la base de données locale de l'application puis transférée au site.

Exemple : Un bolus de 20 unités d'insuline après un repas.

Programme/Profil Basal :

Injectons automatiques, se font en continu et définies sur 24h.

Un profil basal est associé à patient.

Un profil basal est créé par le médecin sur le site, puis transféré à l'application lors de l'authentification du patient puis à la pompe suite à l'appairage de la pompe.

Exemple : Un profil basal peut être défini comme suit:

Nom : Profil Basal

Plages horaires : (heure => débit)

00h-12h => 0.7 unités d'insuline / heure

12h-20h => 0.4 u/h

20h-24h => 0.8 u/h

Profil Basal Temporaire:

Débit de base (quantité faible) sur une durée déterminée.

Exemple :

Nom : Activité sportive

Durée : 1h

Débit:0.2u/h

Scénario MyPod.doc :

Le Dr Dupont, un médecin diabétologue, utilise la plateforme en ligne "MyPod.doc" pour gérer efficacement ses patients diabétiques. Voici comment il interagit avec le site :

1. Connexion au site :

Le Dr Dupont se rend sur le site web de MyPod via un navigateur web.

- S'il n'est pas inscrit, il accède à la page d'accueil puis la page d'inscription où il saisit les informations nécessaires pour faire une demande auprès des administrateurs du site, qui décideront si oui ou non sa demande est acceptée.
- S'il n'est pas déjà connecté, il accède à la page d'accueil où il saisit son adresse email et son mot de passe.
- S'il a des difficultés à se rappeler de son mot de passe, il accède à la page mot de passe oublié où il saisit son adresse e-mail et recevra un mail pour réinitialiser son mot de passe.

Après avoir entré ses informations, il clique sur le bouton "Connexion".

2. Accès à son compte :

- Une fois connecté, le Dr Dupont est redirigé vers son tableau de bord personnel sur la plateforme MyPod.
- Sur son tableau de bord, il trouve une liste de ses patients diabétiques, c'est à dire: le nom, prénom et date de la dernière consultation de chacun de ses patients.

3. Gestion des patients :

- Le Dr Dupont peut sélectionner le un patient pour accéder à son profil.
- Dans le profil de chaque patient, il peut consulter les données médicales essentielles, telles que les informations personnelles (nom, prénom, date de naissance, adresse mail), les traitements en cours (profils basaux) avec les dates de modification associées et les historiques d'injections d'insuline (injections bolus).

4. Mise à jour des informations :

- Le médecin a la possibilité d'ajouter des patients.
- Le médecin a la possibilité de supprimer des patients.
- Le médecin a la possibilité de mettre à jour les données de ses patients et leurs traitements, et d'enregistrer les modifications apportées à leur traitement.

5. Déconnexion :

Une fois la gestion des patients terminée, le Dr Dupont peut choisir de se déconnecter en cliquant sur le lien "Déconnexion" situé dans le menu de navigation.

Scénario MyPod:

Le Dr Dupont, le diabétologue de Leila, l'introduit à la nouvelle application de gestion du diabète, "MyPod". Leila est enregistrée en tant que patiente via son médecin.

Leila utilise désormais l'application MyPod pour gérer son traitement du diabète. Voici comment elle interagit avec l'application :

Le Dr Dupont configure le profil de Leila sur le site web "MyPod.doc", générant un mot de passe aléatoire unique pour sa première connexion qui lui sera envoyé sur son mail personnel.

Leila installe l'application sur son téléphone.

Lors de sa première utilisation, elle saisit l'identifiant qu'elle a reçu par mail, synchronisant ainsi les profils basaux établis.

L'application crée une base de données locale sur son téléphone pour stocker les paramètres de l'application et l'historique des bolus qui sera transféré au serveur MyPod.

Interactions avec la pompe (MyPod-Pod):

Le profil de Leila est chargé sur l'application, il faut maintenant installer une pompe (Un Pod).

La pompe doit être réinitialisée (date d'expiration), puis chargée (Quantité d'insuline).

La pompe est maintenant prête.

Leila doit maintenant appairer son téléphone personnel à la pompe puis initialiser la connexion avec la pompe.

Fonctionnalités de MyPod:

Une fois l'appairage effectué, Leila a maintenant la possibilité de :

- S'injecter des bolus après avoir mangé.
- Suspendre l'administration d'insuline en cas d'urgence.
- Configurer des profils basaux temporaires. (en fonction de ses activités quotidiennes, lorsqu'elle prévoit de faire de l'exercice)
- Modifier selon ses préférences certains paramètres de l'application.
- Consulter l'historique des bolus pour vérifier les doses en cas d'oubli.
- Consulter les Programmes Basaux émis par son médecin
- Consulter ses informations personnelles.
- Configurer des alertes personnalisées.
- Se déconnecter.
- Se connecter à nouveau.

Simulation Pompe (Pod):

Dans le cadre du développement de MyPod, nous avons opté pour la création d'une application distincte afin de simuler les Pods (pompes à insuline), évitant ainsi l'utilisation de pompes à insuline réelles.

Cette application est conçue pour recevoir et envoyer des données via Bluetooth, ainsi que pour les afficher de manière appropriée.

Les données reçues et envoyées sont détaillées ci-dessous :

Données reçues :

Débit basal du patient au moment actuel

Données envoyées :

Quantité d'insuline restante

Date d'expiration de la pompe

Fonctionnalités :

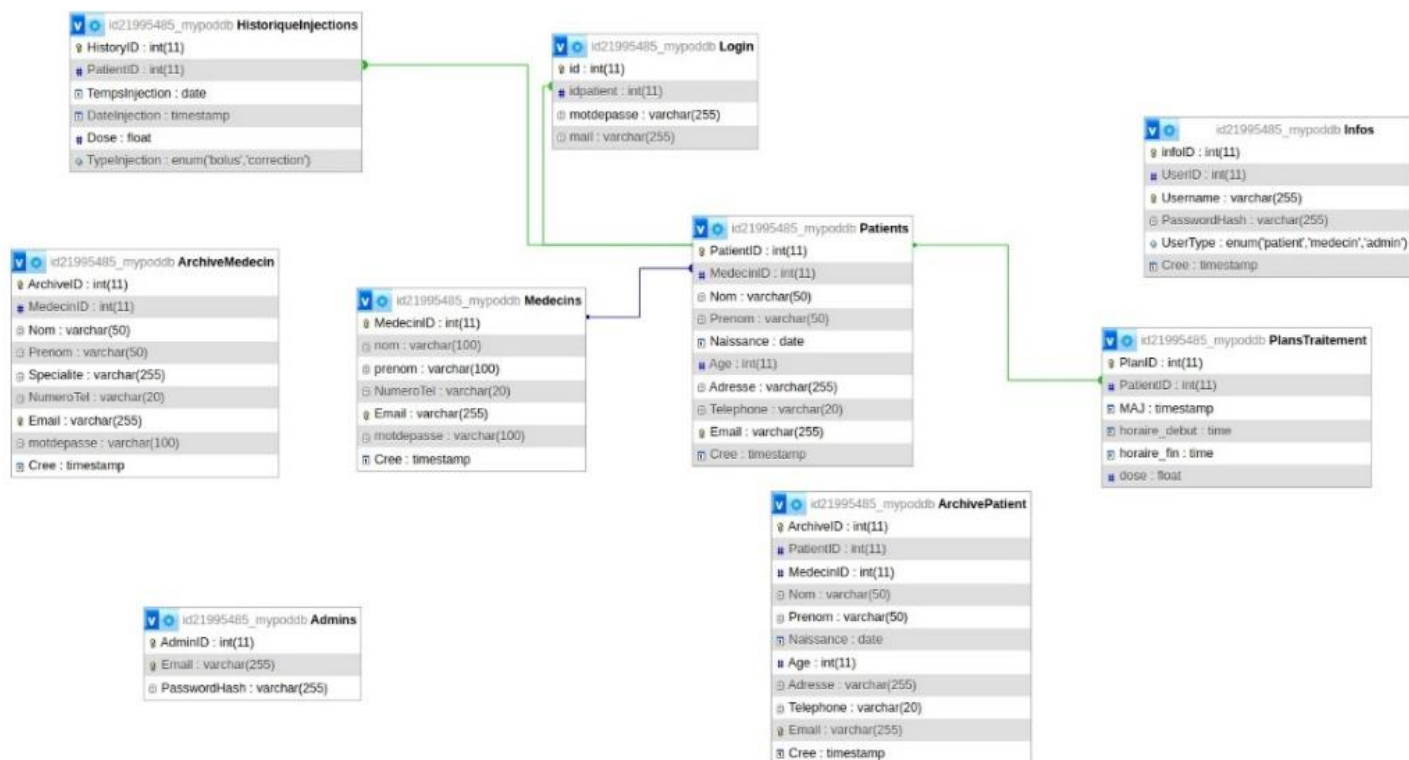
- Affichage en temps réel de la quantité d'insuline restante
- Affichage de la date et heure d'expiration de la pompe (quand il faut la changer)
- Commande pour simuler un rechargement de la pompe => met à jour la quantité d'insuline restante
- Commande pour simuler un changement de pompe => met à jour la date d'expiration

1.3 Diagrammes

Les graphiques de gestion de projet représentent les tâches et objectifs d'un projet ou d'un processus sous forme visuelle. Le diagramme affiche les tâches clairement et simplement pour les personnes qui doivent faire le travail. Cette transparence permet aux membres de l'équipe de rester concentrés et d'éviter de se laisser submerger par un grand nombre de tâches. Que vous soyez officiellement chef de projet ou non, vous devez vous arranger pour que le travail soit réalisé plus efficacement. Dans ce cadre, avoir une bonne visibilité sur les tâches peut simplifier la communication et favoriser la clarté au sein des équipes.

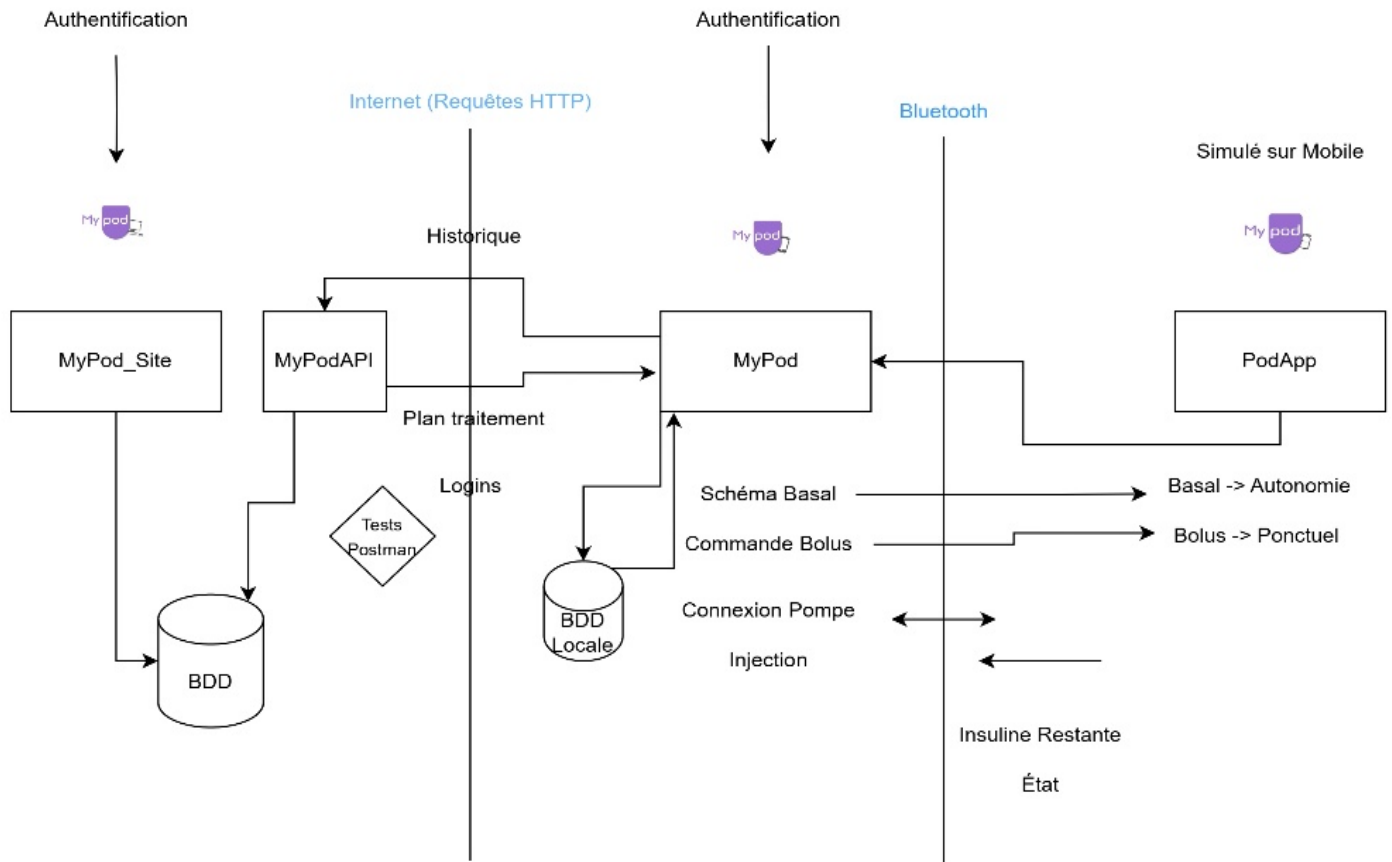
Pour la réalisation du projet nous avons utilisé plusieurs digrammes.

Ces diagrammes nous ont permis de mieux comprendre le projet et de pouvoir se partager les tâches.



Médecin (tokens générés)

Patient



1.4 Organisation du groupe

Pour réaliser ce projet ambitieux, notre groupe a opté pour une stratégie de travail collaboratif en divisant les tâches entre les membres selon leurs compétences et intérêts.

Liste des tâches effectuées par :

👤 **Anis Hanniz :**

MYPOD.DOC

- ✓ Création et mise en route de la BDD et de la première version du site via Scripts sql
- ✓ CSS final de la page login
- ✓ Bascule vers Host en ligne
- ✓ Création dossier API + Implémentation

MYPOD

- ✓ Tout le front et back de l'application

- ✓ Mise en place d'un menu de débogage
- ✓ Création de la BDD Locale Mysqlite
- ✓ Testeur d'API : Teste les différentes requetes http (interactions mypodAPI/mypod)
- ✓ Interaction avec l'API
- ✓ Création composants customisés
- ✓ Graphiques et schémas fl_charts

PODAPP

- ✓ Version finale front et back
- ✓ Communication bluetooth
- ✓ Expressions régulières

MyPod API

- ✓ Mise en place et implémentation de l'API
- ✓ Liaison avec l'application MyPod
- ✓ Mise en place de tests des requetes http via PostMan

Tâches hors développement :

- ✓ Configuration de la VM à la fac
- ✓ Écriture Scénario
- ✓ Schématisation du projet et représentation sous forme de graphiques
- ✓ Réalisation de la vidéo de démonstration
- ✓ Réalisation de la vidéo de présentation

✚ **Badre-Eddine Ben Hammou :**

- ✓ Mypod_doc
- ✓ Gestion de la BDD
- ✓ Api

✚ **Cheikh Ahmadou Diop :**

- ✓ Création des différentes pages d'authentification au niveau de la partie front (avant modification)
- ✓ Page inscription
- ✓ Page authentification
- ✓ Page de récupération de mot de passe
- ✓ Récupération de fichier JSON avec l'API au niveau de l'authentification
- ✓ Création des différents logos de l'application, de l'api et du site web
- ✓ Gestion et écriture du rapport de projet
- ✓ Gestion et écriture du document power point de la présentation

✚ **Abissi Kotchipka Ismael :**

Partie Pompe (PodApp):

Dans le cadre du développement de l'application médicale MyPod, nous avons opté pour la création d'une application distincte afin de simuler les Pods (pompes à insuline).

Ce choix a été fait principalement pour éviter de faire des manipulations risquées sur une pompe réelle (dispositif médical dont on ne maîtrise pas entièrement les mécanismes).

Toutefois, notre simulation reproduit le fonctionnement d'une pompe qui se connecte également par Bluetooth à une application.

Front-end :

Comme l'application principale MyPod, nous avons développé cette application de simulation avec le framework flutter permettant donc de le déployer sur plusieurs types d'appareils. Une interface assez sobre qui inclut tout de même les fonctionnalités d'un pod.

Notre pompe (application) est conçue pour recevoir et envoyer des données via Bluetooth, ainsi que pour les afficher de manière appropriée tout en simulant une évolution synchronisée des injections et valeurs

exactement comme une pompe.

La communication entre l'application MyPod et PodApp est décrite ci-dessous :

- Données reçues : Débit basal du patient au moment actuel
- Données envoyées : Quantité d'insuline restante, Date d'expiration de la pompe

Fonctionnalités de l'application PodApp :

- ✓ Affichage en temps réel de la quantité d'insuline restante
- ✓ Affichage de la date et heure d'expiration de la pompe (quand il faut la changer)
- ✓ Commande pour simuler un rechargement de la pompe => met à jour la quantité d'insuline restante
- ✓ Commande pour simuler un changement de pompe => met à jour la date d'expiration

Back-end :

Pour faciliter la communication entre la pompe simulée et l'application de gestion, on a implémenté une architecture client-serveur avec la pompe comme serveur puisque c'est l'application qui vient s'y connecter.

Communiquant par Bluetooth, nous avons utilisé dans cette application un package avec des méthodes et classes définies. Ce package appelé "All_Bluetooth" est officiellement présent dans la librairie de paquets de flutter (pub.dev) et est un des paquets parmi tant d'autres pour le Bluetooth disponibles.

Ce package nous permet de réaliser sans grande difficulté les opérations suivantes:

- ✓ Scanner et afficher les périphériques bluetooth avec possibilité de filtrer le résultat
- ✓ Établir une connexion avec un autre appareil bluetooth
- ✓ Échanger des données de manière asynchrone avec l'appareil connecté

 **Joshua Bitho :**

- ✓ Gestion de la partie teste

1.5 Chartes graphique

Création du logo de Mypod

Le logo de notre application est divisé en quatre sous logo

Le logo est créé avec Adobe Photoshop

Adobe Photoshop est un logiciel de création graphique développée par Adobe Systems. C'est l'outil de référence pour tous les professionnels de la retouche d'image et la création numérique. La première version de Photoshop a été lancée en 1988 et il est depuis devenu le logiciel le plus utilisé dans le monde pour la manipulation d'images.

Photoshop permet aux utilisateurs de créer et de modifier des images de différentes tailles, résolutions et formats. Les utilisateurs peuvent travailler avec des photos, des illustrations, des illustrations vectorielles et même des GIF. Le logiciel est également capable de traiter des images en mode RVB (écran), CMJN (impression), échelle de gris, noir et blanc, etc.



Le diabète sous contrôle



Police :

My : Neon 80 s

Le diabète sous contrôle : Eras Medium ITC

Effet :

Biseautage et estampage

2 Conception

2.1 MyPOD

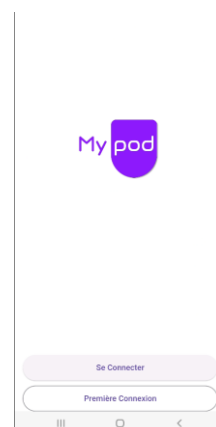
Menu de débogage :

Nous permet en tant que développeurs d'accéder à certaines pages plus rapidement.



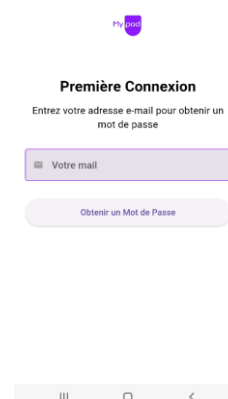
Splash Screen :

Écran d'arrivée sur l'application



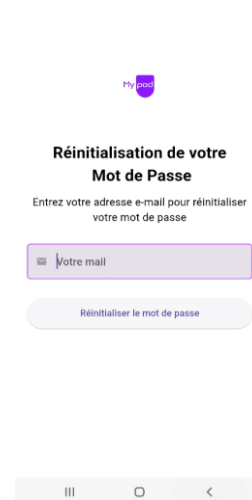
Page de première connexion :

Permet au patient d'obtenir un mot de passe aléatoire (si son mail existe dans les serveurs mypod)



Page de réinitialisation de Mot de Passe :

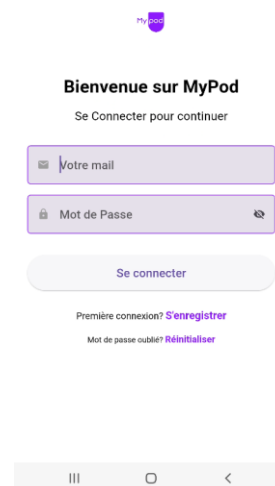
Permet de générer un mot de passe aléatoire en cas d'oubli.



The screenshot shows the 'Réinitialisation de votre Mot de Passe' page. At the top is the 'MyPod' logo. Below it, the title 'Réinitialisation de votre Mot de Passe' is followed by the instruction 'Entrez votre adresse e-mail pour réinitialiser votre mot de passe'. There is a text input field labeled 'Votre mail' and a button labeled 'Réinitialiser le mot de passe'. At the bottom, there are three icons: a list icon, a square icon, and a back arrow icon.

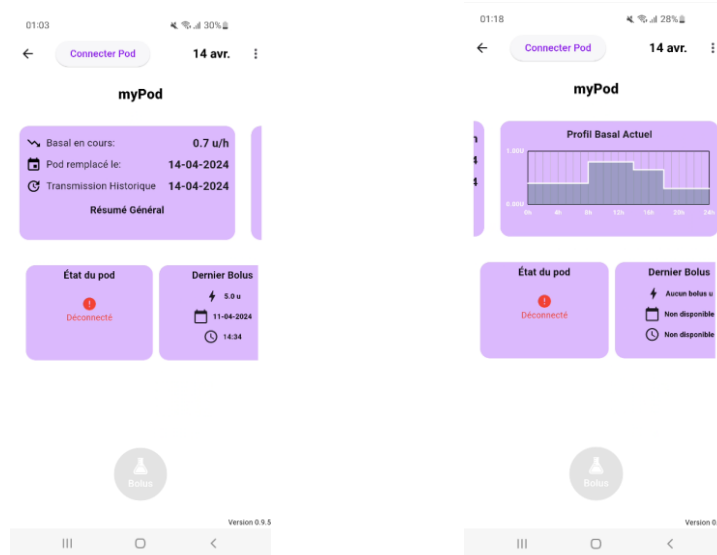
Page de connexion (login) :

Permet de s'authentifier, renvoie vers la page home si le mail et mot de passe sont correctes



The screenshot shows the 'Bienvenue sur MyPod' login page. At the top is the 'MyPod' logo. Below it, the title 'Bienvenue sur MyPod' is followed by the instruction 'Se Connecter pour continuer'. There are two text input fields: 'Votre mail' and 'Mot de Passe'. Below these is a button labeled 'Se connecter'. At the bottom, there are two links: 'Première connexion? S\'enregistrer' and 'Mot de passe oublié? Réinitialiser'. At the very bottom, there are three icons: a list icon, a square icon, and a back arrow icon.

Page d'accueil



Widget BasalChartRate

Exemple de graphique dessiné grace au schéma basal récupéré de l'api

Boutton Bolus

Activé seulement si on est connectés à la pompe (pod)

Widget PodState

Affiche l'état de connexion à la pompe

Widget Résumé Général

Affiche des informations générales comme le débit basal en cours la date de remplacement du pod ainsi que sa date de péremption

Menu Popup

01:18 28%

Informations Personnelles

Nom
Pharell

Prénom
Leila

Email Personnel
leila@gmail.com

Date de Naissance
1995-05-20

Adresse
123 Rue A

Numéro de Téléphone
0612345678

Age
28 ans

Fermer

01:18 28%

myPod

Définir Basal Temporaire

Nom du profil basal:
Entrez le nom du profil

Durée:
0 min 3 h

Quantité (unités/h):
Entrez la quantité basale

Créer Fermer

01:20 27%

myPod

Profil Basal Actuel

⚙️ Réglages

Mot de passe oublié? [réinitialiser](#)

Dose maximale de bolus:
10 +

Vitesse d'injection:

Confirmer Annuler

01:19 28%

← Schéma Basal

| Heure début | Heure fin | Dose |
|-------------|-----------|------|
| 00:00 | 08:00 | 0.4 |
| 08:00 | 14:00 | 0.8 |
| 14:00 | 18:00 | 0.65 |
| 18:00 | 00:00 | 0.3 |

Fermer

01:19 28%

Liste des Profils Basaux

Activité Sportive

Durée: 1.25 heures
Débit: 0.2
Actif: Non

Actif
Inactif
Supprimer

Rafraîchir

Fermer

01:18 28%

- Informations Personnelles
- Définir Basal Temporaire
- Liste Profils Basaux
- Pod
- Suspendre Admin. Insuline
- Programmes Basaux
- Historique d'injections
- Rappels
- Réglages
- À propos

Bolus

Version 0.9.5

01:19 28%

myPod

Profil Basal Actuel

Confirmation

Êtes-vous sûr de vouloir suspendre l'administration d'insuline ?

Annuler Confirmer

01:19 28%

Liste des Profils Basaux

Activité Sportive

Durée: 1.25 heures
Débit: 0.2
Actif: Oui

Actif

Rafraîchir

Fermer

Profil Actif: Activité Sportive, Durée Restante: 1.25 heures

01:20 27%

myPod

Profil Basal Actuel

Fréquence des rappels ⚙️

☒ Jamais

☐ Toutes les 30 secondes

☐ Toutes les 1 minute

☐ Toutes les 5 minutes

Fermer

01:19 28%

État du Pod

❗

Déconnecté

Fermer

01:24 27%

← Injection Page

Transfer Injections

Date: 2024/4/14
Heure: 01:24, Dose: 5.0

Date: 2024/4/14
Heure: 01:24, Dose: 20.0

Page d'informations personnelles :

Récupère les informations personnelles des serveurs MyPod

Page de définition de Profils Basaux :

Définis par le patients et enregistrés dans un fichier local txt.
Puis récupérés/activés/désactivés plus tard.

Page liste des Profils Basaux :

Suite à la création, cette page permet d'activer/désactiver ou supprimer un profil basal.

Page d'informations sur le Pod :

Permet de savoir si on est connectés ou pas à un pod (pompe), permet aussi de se déconnecter si on est connectés.

Page de suspension de l'administration d'insuline :

Permet d'annuler un bolus en cours et de mettre le débit basal à 0u/h.

Page Schéma Basal :

Permet de visualiser le schéma basal prescrit par le médecin.

Page Rappels :

Permet de modifier la fréquence des rappels. (en cas de réservoir vide ou presque vide)

Page settings :

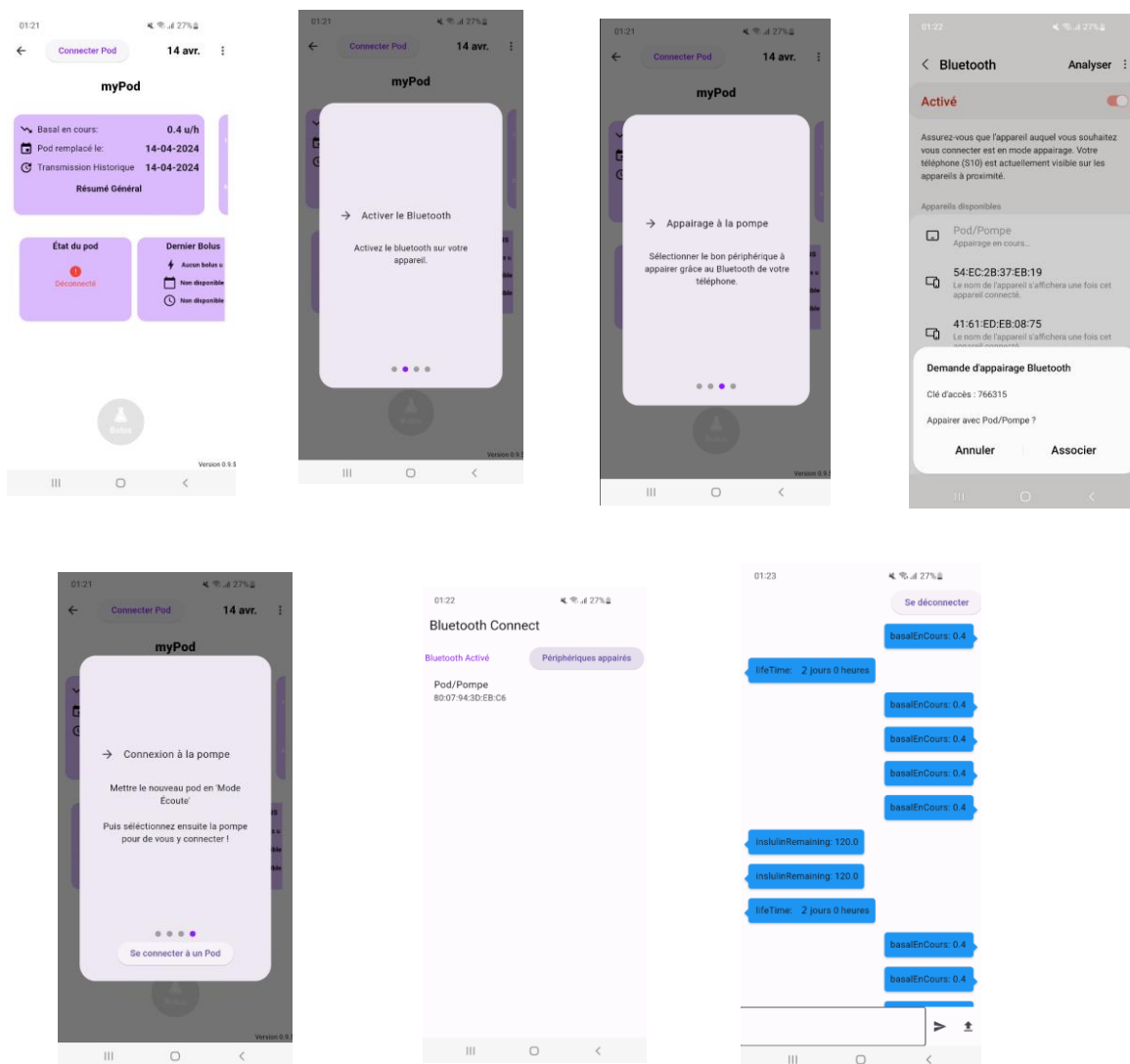
Permet à l'utilisateur de réinitialiser son mot de passe. De définir la dose maximale des bolus et de définir la vitesse d'administration des bolus.

Utilisation Bluetooth

Le bouton Connecter Pod ouvrira un mini tutoriel qui explique les étapes à suivre afin de connecter l'application à la pompe

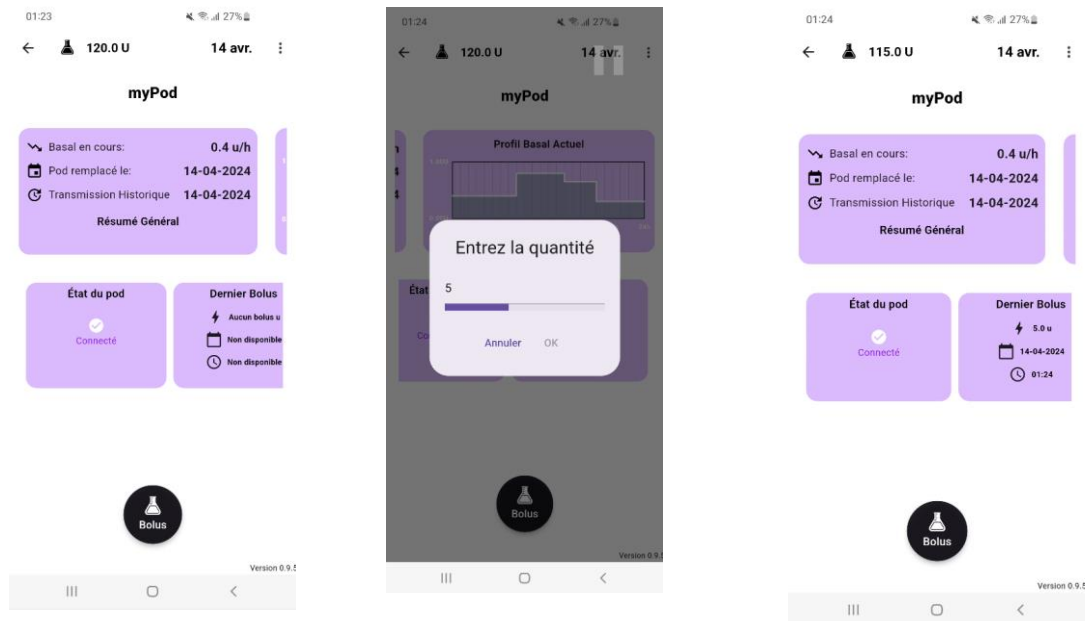
Une fois la connexion réussie, on est renvoyés vers ChatScreen, une page où on peut apercevoir les informations échangées entre la pompe et l'application

(Cette page est un aperçu, dans une application réelle cela ne sera pas affiché)

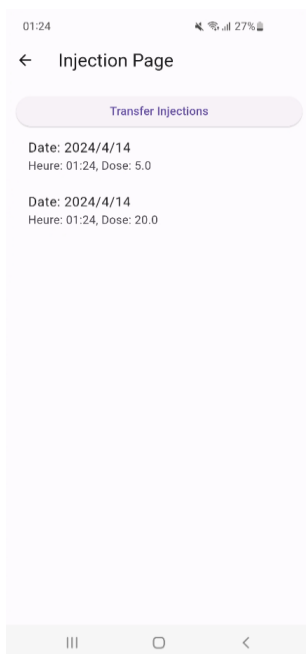


Commande Bolus

Une fois connectés à la pompe, on peut lancer des commandes bolus, grâce au bouton « Bolus »



Une fois le bolus terminé, on pourra vérifier sur la table historique et transférer au médecin la liste des bolus



Guide de déploiement

Il vous faudra deux appareils androids, un pour MyPod et un autre pour simuler la pompe (podapp)

Transférez les fichiers APK sur un appareil Android pour les tester :

- Connectez votre appareil Android à votre ordinateur via USB.
- Transférez les fichiers APK vers votre appareil Android.
- Sur votre appareil Android, ouvrez les fichiers APK pour les installer et suivez les instructions à l'écran.

Guide d'installation du projet Flutter sous Linux

Téléchargement de Flutter

- Ouvrez un terminal sur votre système Linux.
- Téléchargez l'archive Flutter depuis le site officiel en utilisant la commande `curl` :

```
curl -O  
https://storage.googleapis.com/flutter_infra/releases/stable/linux/flutter_linux_2.5.3-  
stable.tar.xz
```

- Extrayez l'archive Flutter dans un répertoire de votre choix :

```
tar xf flutter_linux_2.5.3-stable.tar.xz
```

- Ajoutez le chemin du répertoire `flutter/bin` à votre variable d'environnement `PATH` en éditant votre fichier `.bashrc` ou `.zshrc` :

```
export PATH="$PATH: `pwd` /flutter/bin"
```

- Rechargez votre terminal ou exécutez la commande suivante pour appliquer les modifications :

```
source ~/.bashrc
```

Configuration de Flutter

- Vérifiez l'installation de Flutter en exécutant la commande suivante :

```
flutter doctor
```

- Suivez les instructions pour installer les dépendances manquantes telles que les outils de développement Android Studio, les packages nécessaires pour Android et/ou iOS, etc.

Récupération du projet Flutter existant

- Ouvrez un terminal.
- Accédez au répertoire de votre projet Flutter existant en utilisant la commande ``cd``. Par exemple :

```
cd /home/nad/Desktop/VMSHARE/Projets/mypod/depot
```

Installation des dépendances du projet

- Exécutez la commande ``flutter pub get`` pour installer toutes les dépendances déclarées dans le fichier ``pubspec.yaml``.

```
flutter pub get
```

Développement et test

Utilisez ``flutter run`` pour tester votre application sur un émulateur ou un appareil Android.

2.2 MyPOD_Doc

Dans cette partie, nous avons créé le site web chargé d'administrer les médecins et les patients de la plateforme mobile. Il a pour but de permettre :

Quand on est médecin, d'enregistrer ses patients en les assignant, ou de les supprimer, permettant ainsi d'avoir un visuel pour le praticien de ses patients. Un médecin peut créer son compte en recevant un lien d'authentification par mail ou se faire ajouter par un administrateur.

Pour un administrateur, d'avoir accès aux données de tous les patients et médecins ainsi qu'aux traitements et archives. Il peut créer un médecin ou un patient et supprimer n'importe quelle personne ou données.

Un patient ou un médecin supprimé se retrouvent dans les archives, permettant de prévoir une suppression erronée ou de garder des traces, malgré les différentes pop-up tout le long du processus de suppression.

Le site web est accessible seulement aux médecins et aux administrateurs, les patients n'ont pas accès. Les fonctionnalités principales incluent :

Pour les médecins sur la page d'index :

- Connexion, inscription ou envoi d'un nouveau mot de passe via mail.
- Une fois connecté, accès à son tableau de bord et affichage de ses patients. Il peut ajouter des patients et accéder à une fiche détaillée du patient avec ses informations personnelles, son plan de traitement s'il existe, l'historique de ses injections, et un diagramme représentant le schéma à suivre pour ce patient.
- Modification ou suppression des informations du patient. La suppression met le patient dans les archives.
- Possibilité de créer un nouveau plan de traitement pour le patient en définissant la dose à

administrer sur des tranches horaires.

- En cas de problème, le médecin peut contacter un administrateur via un mail.

Pour les administrateurs :

- Accès à toutes les tables et possibilité de supprimer n'importe quelle ligne de n'importe quelle table.
- Ajout d'un patient (en définissant son médecin) ou d'un médecin.

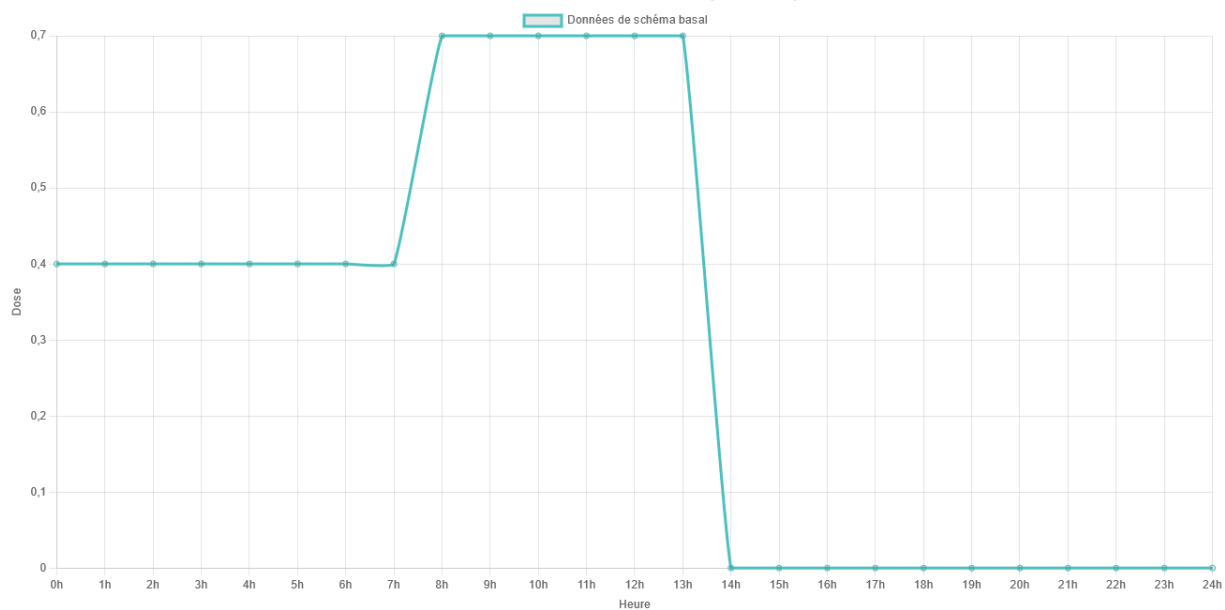
Une dernière partie concerne les API, qui permettent d'envoyer à l'application mobile les données dont elle a besoin. Par exemple, quand un patient veut s'identifier, l'application envoie le mail et le mot de passe, et l'API vérifie dans la base de données. Si le patient existe, elle renvoie un token, sinon une erreur. Cela fonctionne de même pour l'historique des injections, le plan de traitement ou la réinitialisation du mot de passe d'un patient.

Pour la réalisation, j'ai utilisé HTML, CSS, JavaScript, PHP, AJAX et Bootstrap. Les mots de passe sont hachés grâce à la fonction PHP `password_hash()`, en utilisant des variables pour les mots de passe car leur taille peut varier. Ce choix assure un cryptage sécurisé et pratiquement indéchiffrable de nos jours.

Voici quelques captures du site :



Plan de traitement à suivre par le patient



Dosage par tranche horaire

Début:

00:00

Fin:

00:00

Dose

0.0

Ajouter une plage horaire


Soumettre le traitement

Fermer

2.3 API

Pour la partie API on a utilisé postman pour tester les requêtes http. On a choisi d'implémenter une API pour pouvoir accéder à la bdd à partir de l'application de manière sécurisée.

Voici quelques exemples des tests effectués ainsi que les valeurs de retour :


MyPod API / Infos Patient

GET

https://mypodev.000webhostapp.com/API/patient.php?mail=hanniz.n.anis@gmail.com

Send

Params

Authorization Headers (7) Body Pre-request Script Tests Settings

Cookies

Query Params

| Key | Value | Description |
|------|-------------------------|-------------|
| mail | hanniz.n.anis@gmail.com | |


Body Cookies (1) Headers (13) Test Results

Status: 200 OK Time: 1350 ms Size: 832 B Save as example

Pretty Raw Preview Visualize HTML

```

1 {
  "0": "1",
  "PatientID": "1",
  "1": "1",
  "MedecinID": "1",
  "2": "Hanniz",
  "Nom": "Hanniz",
  "3": "Anis",
  "Prenom": "Anis",
  "4": "1999-06-14",
  "Naissance": "1999-06-14",
  "5": "24",
  "Age": "24",
  "6": "456 Rue B",
  "Adresse": "456 Rue B",
  "7": "0777959043",
  "Telephone": "0777959043",
  "8": "hanniz.n.anis@gmail.com",
  "mail": "hanniz.n.anis@gmail.com",
  "9": "2024-04-15 21:48:47",
  "Cree": "2024-04-15 21:48:47"
}
```


MyPod API / Login

POST

https://mypodev.000webhostapp.com/API/auth/login.php

Send

Params

Authorization Headers (9) Body Pre-request Script Tests Settings

Cookies

none form-data x-www-form-urlencoded raw binary GraphQL

form-data

| Key | Value | Description |
|----------|-------------------------|-------------|
| mail | hanniz.n.anis@gmail.com | |
| password | deesse | |
| mail | leila@gmail.com | |
| password | leila | |

Body Cookies (1) Headers (11) Test Results

Status: 200 OK Time: 517 ms Size: 430 B Save as example

Pretty Raw Preview Visualize JSON

```

1 {
2   "success": false,
3   "message": "Mail ou mot de passe incorrect"
4 }
```

MyPod API / Login

POST https://mypodev.000webhostapp.com/API/auth/login.php

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings

☐ none ☒ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

| Key | Value | Description |
|--|-------------------------|-------------|
| <input checked="" type="checkbox"/> mail | hanniz.n.anis@gmail.com | |
| <input checked="" type="checkbox"/> password | deesse | |
| <input type="checkbox"/> mail | leila@gmail.com | |
| <input type="checkbox"/> password | leila | |
| Key | Value | Description |

Body Cookies (1) Headers (11) Test Results

Status: 200 OK Time: 517 ms Size: 430 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "success": false,
3   "message": "Mail ou mot de passe incorrect"
4 }
```

MyPod API / Schéma basal

POST https://mypodev.000webhostapp.com/API/schema_basaux/schema.php?id=1

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings

☐ none ☒ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

| Key | Value | Description |
|--|-------|-------------|
| <input checked="" type="checkbox"/> id | 1 | |
| Key | Value | Description |

Body Cookies (1) Headers (12) Test Results

Status: 200 OK Time: 370 ms Size: 679 B Save as example

Pretty Raw Preview Visualize HTML

```
1 [{"horaire_debut": "08:00:00", "horaire_fin": "08:00:00", "dose": "0.6"}, {"horaire_debut": "08:00:00", "horaire_fin": "14:00:00", "dose": "0.2"}, {"horaire_debut": "14:00:00", "horaire_fin": "20:00:00", "dose": "0.9"}, {"horaire_debut": "20:00:00", "horaire_fin": "23:59:00", "dose": "0.5"}]
```

MyPod API / Login

POST https://mypodev.000webhostapp.com/API/auth/login.php

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings

☐ none ☒ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

| Key | Value | Description |
|--|-------------------------|-------------|
| <input checked="" type="checkbox"/> mail | hanniz.n.anis@gmail.com | |
| <input checked="" type="checkbox"/> password | deesse | |
| <input type="checkbox"/> mail | leila@gmail.com | |
| <input type="checkbox"/> password | leila | |
| Key | Value | Description |

Body Cookies (1) Headers (11) Test Results

Status: 200 OK Time: 502 ms Size: 511 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "success": true,
3   "id": 1,
4   "message": "Authentication réussie",
5   "token": "48f279b738aa5320d04487037cdd2ff5b058e7e1154e1b30820cdfc423914d4c"
6 }
```

3 Test

Bienvenue sur MyPod

Adresse E-mail :

dr.dupont@example.com

Mot de Passe :

.....|



Connexion

[*Mot de passe oublié ?](#)

[Pas encore inscrit ?](#)
[Faites une demande !](#)

Bienvenue sur MyPod

Adresse E-mail :

dr.dupont@example.com

Mot de Passe :

motdepasse1



Connexion

[*Mot de passe oublié ?](#)

Pas encore inscrit ?
[Faites une demande !](#)

```
1 package dpack;
2
3 import org.openqa.selenium.By;
4 import org.openqa.selenium.WebDriver;
5 import org.openqa.selenium.WebElement;
6 import org.openqa.selenium.firefox.FirefoxDriver;
7
8 import io.github.bonigarcia.wdm.WebDriverManager;
9
10
11 public class Site_index {
12     public static void main(String[] args) {
13         WebDriverManager.firefoxdriver().setup();
14         WebDriver driver = new FirefoxDriver();
15         driver.manage().window().maximize();
16         driver.get("http://localhost/dashboard/tp/test/");
17         WebElement emailInput = driver.findElement(By.id("email"));
18         WebElement passwordInput = driver.findElement(By.id("password"));
19         WebElement submitButton = driver.findElement(By.cssSelector("button[type='submit']"));
20
21         // Entrez les informations de connexion
22         emailInput.sendKeys("dr.dupont@example.com");
23         passwordInput.sendKeys("motdepasse1");
24
25         // Attendez un peu pour voir le résultat
26         try {
27             Thread.sleep(5000);
28         } catch (InterruptedException e) {
29             e.printStackTrace();
30         }
31
32         // Fermez le navigateur
33         driver.quit();
34     }
35 }
```