Documentation application mobile MyPo:

------------------------------------------------------------------------------------

PS : -mettre à jour la documentation du projet au fur et à mesure

- corriger les fautes qui pourraient éventuellement être présent

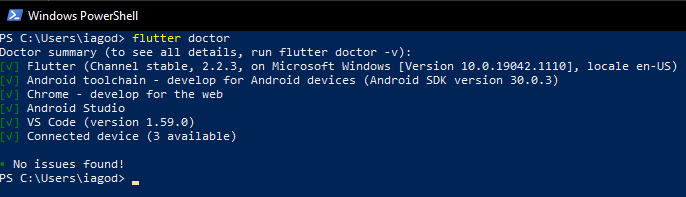
* Installation
* Structure des dossiers
* Fonctionnalités

Installation:

Installer flutter: <https://flutter.dev/docs/get-started/install>

Après installation vérifier que tout est correcte avec la commande

flutter doctor



Vous pouvez connecter votre téléphone, un moteur de recherche ou utiliser un émulateur

flutter devices

Text

Description automatically generated

Cloner le dépôt GitHub:

Logo

Description automatically generated with low confidence

Pour récupérer le lien du dépôt sur github :

A screenshot of a computer

Description automatically generated

Se déplacer ensuite dans le dossier créé et ensuite utiliser les commandes

flutter pub get

flutter run

[Ou vous pouvez utiliser l’IDE de votre choix comme Android Studio, Visual Studio code etc.]

Text

Description automatically generated

L’application se lancera ensuite sur votre émulateur

[Si vous rencontrez des problèmes il existe des tutos pour configurer un projet avec flutter sur internet]

A close-up of a cell phone

Description automatically generated with medium confidence

Structure des dossiers:

Dossier

*Assets*: contient les fonts, les images les fichiers .json, .txt

A screenshot of a computer

Description automatically generated with medium confidence

*Lib* : contient les dossier du projet

Text

Description automatically generated

*Database* : dossier contenant les fichier pour une base de données Hive NOSQL et les fichiers automatiquement générés par hive. (Il existe également une base de données SQLITE et SharedPreferences. Hive database est utilisé dans ce projet car c’est un moyen optimal d’utiliser une base de données NOSQL plus rapide que SQFLITE et SharedPreferences dans l’écriture et la lecture de données)

Tout d’abord on déclare une class qui hérite de la class HiveObject et les instructions suivante :

@HiveType(typeId :index) (avant la class)

@HiveField(index) (pour chaque champs)

Comme sur la photo :

[voir la doc de Hive ou des tutos sur internet pour plus d’informations]

Text

Description automatically generated

Ensuite nous utilisons la commande sur le terminal

flutter pub run build runner build

Ce code génèrera automatiquement un fichier .g.dart comme sur la photo

Text

Description automatically generated

Ensuite ceci est le code nécessaire pour le fonctionnement de hive avec flutter, nous devons enregistrer l’adaptateur qui est généré atomiquement lorsqu’on crée des objets de type hive et que l’on utilise la commande :

flutter pub run build runner build

Text

Description automatically generated

Nous avons plusieurs tables/boxes sur hive

scheduledmsg

group

rapportmsg

alert

alertkey

TABLES : ‘scheduledmsg’ qui possède des objets de type Scheduledmsg\_hive,

Text

Description automatically generated

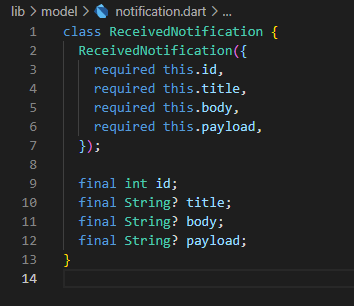
‘rapportmsg’ qui possède des objets de type Rapportmsg\_hiveText

Description automatically generated

‘rapportmsg’ qui possède des objets de type Rapportmsg\_hive,   
‘user’ qui possède des objets de type User\_hive,  
‘group’ qui possède des objets de type GroupContact,   
‘alert qui possède des objets de type Alert,

‘alertkey’ qui possède des objets de type AlertKey.

Chacune d’entre elles possèdent des champs spécifiques que l’on a créer selon nos besoins [c.f lib/database/hive\_database.dart]

*Model* : modèle d’objet utilisés dans l’application (ex notifications dans la photo)   
  


*Pages* : Contient toutes les pages de l’application

Text

Description automatically generated

*Utils* : Utilisé essentiellement pour la sauvegarde des donnes et certaines fonctions utilisés à travers le programme  
Contient la classe fonctions avec la fonction pour montrer une notification sur le pied de la page  
Contient la classe boxes qui propose les fonctions d’accès aux boxes hive

Text

Description automatically generated with medium confidence

*Widget* : Contient les widgets utilisés dans les pages tels que la bar de navigation inférieur et supérieur, le logo, le burger menu, la photo du profil etc.

Text

Description automatically generated

*Pubspec.yaml* (fichier) contient les dépendances, les packages utilisés, la déclaration des ressources utilisés dans le l’application comme les images, font etText

Description automatically generated

Text

Description automatically generated

Les Pages :

accueil\_page.dart : Cette page contient

* La bar de navigation supérieur avec un menu de type burger qui se superpose a l’écran lorsqu’on clique dessous
* Le logo qui occupe 20% de l’écran  
  Bouton message programmé qui nous renvoie vers la page des messages programmés
* Bouton message automatique qui nous renvoie vers la page des messages auto
* La bar de navigation inférieure qui permet de naviguer de la page d’accueil a la page des rapports, la page d’aide, et la page des paramètres

Graphical user interface, application

Description automatically generated

On y trouve les fonctions suivantes :

Text

Description automatically generated

Text

Description automatically generated

aide\_page.dart : Cette page contient

* La bar de navigation supérieur avec le retour vers la page d’accueil
* Les questions les plus fréquentes avec les réponses
* Le formulaire de contact pour envoyer un mail à l’entreprise

Graphical user interface, application

Description automatically generated

On y trouve les fonctions suivantes :

A screenshot of a computer

Description automatically generated with medium confidence

detail\_rapport\_page.dart : Cette page contient

* La bar de navigation supérieur avec le retour vers la page d’accueil
* Le rapport d’une alerte émises où l’on peu voir : Text

  Description automatically generated

edit\_alerte\_auto\_page.dart : Cette page contient

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, chat or text message

Description automatically generated

On y trouve les fonctions suivantes :

Application

Description automatically generated with medium confidence

edit\_alerte\_prog\_page.dart : Cette page contient

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, chat or text message

Description automatically generated

On y trouve les fonctions suivantes :

Text

Description automatically generated

edit\_group\_contact\_page.dart : Cette page contient

Graphical user interface, text, application, chat or text message

Description automatically generated

On y trouve les fonctions suivantes :

A screenshot of a computer

Description automatically generated with medium confidence

edit\_profile\_page.dart : Cette page contient

Graphical user interface, text, application

Description automatically generated

On y trouve les fonctions suivantes :

Text

Description automatically generated

formulaire\_alerte\_auto\_page.dart : Cette page contient

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, chat or text message

Description automatically generated

On y trouve les fonctions suivantes :

A picture containing graphical user interface

Description automatically generated

formulaire\_alerte\_prog\_page.dart : Cette page contient

* Cette page contient le formulaire de création pour une alerte programmé, il est accessible lorsqu’on clique sur ajouter une alerte dans la page des messages programmés
* La bar de navigation supérieure avec le titre de la page
* Le bouton retour   
  les champs de texte pour :
* Nom de l’alerte
* Numéro du contact avec icone à droite pour aller sélectionner des contacts du téléphone
* Le message à envoyer
* La sélection de la date et de l’heure
* La sélection de la récurrence
* L’option compte à rebours
* L’option de confirmation avant envoi
* L’option de notification lorsqu’un message est envoyé

Graphical user interface, application

Description automatically generated



On y trouve les fonctions suivantes :

Text

Description automatically generated

* saveToHive() :

Permet de sauvegarder le message sur notre box intitulé ‘scheduledmsg’ de hive database si tous les champs sont remplis correctement

* buildTextField(String labelText, String placeholder, TextEditingController controller, int nbLines) :  
  Contruit un champ pour rentrer du texte avec le nom du champ(labeltext), le texte d’indice(placeholde) le controleur de ce champ(controller) et le nombre de lignes du champ
* buildDatePicker() :  
  Construit un sélectionneur de date avec le format de la librairie Cupertino, date initiale étant la date actuelle, format 24h
* buildRepeatOptions() :  
  Construit un sélectionneur de récurrence avec la librairie Cupertino selon les options de

formulaire\_group\_contact\_page.dart : Cette page contient

Graphical user interface, text, application, chat or text message

Description automatically generated

On y trouve les fonctions suivantes :

A screenshot of a computer

Description automatically generated with medium confidence

group\_contact\_page.dart : Cette page contient

Graphical user interface, application

Description automatically generated

On y trouve les fonctions suivantes :

A screenshot of a computer

Description automatically generated with medium confidence

parametres\_page.dart : Cette page contient

Graphical user interface, application

Description automatically generated

On y trouve les fonctions suivantes :

premium\_page.dart : Cette page contient

Graphical user interface, text, application

Description automatically generated

rapports\_page.dart : Cette page contient

Graphical user interface, text, application, chat or text message

Description automatically generated

On y trouve les fonctions suivantes :

A screenshot of a computer

Description automatically generated with medium confidence

sms\_auto\_page.dart : Cette page contient

Graphical user interface, text, application

Description automatically generated

On y trouve les fonctions suivantes :

Graphical user interface, text, application, chat or text message

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

Graphical user interface, text

Description automatically generated with medium confidence

Graphical user interface

Description automatically generated with medium confidence

sms\_prog\_page.dart : Cette page contient

* La bar de navigation supérieur avec un bouton retour sur la page d’accueil  
  Le bouton pour ajouter une nouvelle alerte
* Un message qui est affiché si on ne possède aucune alerte
* La bar de navigation inférieure qui permet de revenir à la page d’accueil et changer entre les

pages de message programmé ou message automatique

Graphical user interface, text, application, email

Description automatically generated

Dans ce fichier on trouve les fonctions

Text

Description automatically generated with low confidence

* saveMsgToRappor(Scheduledmsg\_hive message) :  
  sauvegarde un message dans la boite/box des rapport
* sendSms() :  
  Vérifie si le message possède des options avant envoi
* send() :  
  Envoi un message avec la librairie telephony
* updateDate(Scheduledmsg\_hive msg) :   
  Prend en paramètre un message et met à jour la date d’envoie du message programmé en fonction de la récurrence choisi (tout les ans, tout les mois, aucune etc.)
* canBeSent(Scheduledmsg\_hive msg) :  
  Retourne vrai si l’heure actuelle est supérieure ou égale a l’heure programmé pour envoi
* confirmSend(Scheduledmsg\_hive msg) :  
  Créer un pop up pour demander si l’utilisateur souhaite envoyer le msg si non un délai de 5 minutes est ajouter a l’heure d’envoie
* buildListOfMsg(List<Scheduledmsg\_hive> messages):  
  Prend en paramètre la liste de messages de type scheduledmsg\_hive et construit la liste de messages avec ListView.Builder
* buildMsg(BuildContext context, Scheduledmsg\_hive message):  
  Créer le modèle pour un message
* buildButtons(BuildContext context, Scheduledmsg\_hive message):  
  Créer les boutons affichés dans le message (modifier supprimer)
* buildPopupDialog(Scheduledmsg\_hive message) :  
  Créer un pop up pour demander si l’utilisateur souhaite supprimer ou non l’alerte

Utils :

Le dossier utils contient un ensemble de pages pour mieux organiser le code

* boxes.dart :  
  class responsable de la récupération des boites/boxes de hive database  
  On trouve la méthode getScheduledmsg() qui peut être utilisé pour récupérer tous les messages sauvegarder sur la boîte ‘scheduledmsg’ example (final allMessages = Boxes.getScheduledmsg() )

Text

Description automatically generated

* couleurs.dart :  
  Contient toutes les couleurs utilisés dans l’application

Text

Description automatically generated

* expressions.dart :  
  Contient toutes les expressions régulières qu’on utilise pour valider certains champs

Text

Description automatically generated

* fonctions.dart :  
  Contient certains fonctions réutilisés pas mal de fois dans les autres pages

Text

Description automatically generated

* variables.dart :  
  Contient certains variables qui sont réutilisés dans différents champs et représentent la même valeur

Graphical user interface, text

Description automatically generated

Widget :

Le dossier widget contient un ensemble de widget réutilisables selon nos besoin dans nos pages, pour ce faire nous n’avons qu’à l’importer et l’appeler avec les paramètres nécessaires

* appbar\_widget.dart :  
  widget responsable pour la bar de navigation supérieure  
  Text

  Description automatically generated

Pour appeler ce widget

* TopBar(« Mon titre ») :   
  Retourne in widget avec le titre donné en paramètre

* button\_widget.dart :  
  widget responsable pour la création d’un bouton avec un titre et une fonction sur le clique

Text

Description automatically generated

Pour utilise ce widget

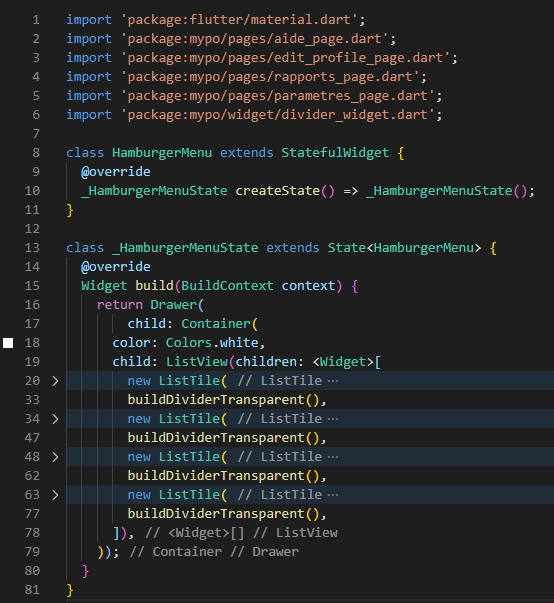
* ButtonWidget(String text, VoidCallback onClicked) :  
  permet d’afficher un button avec le titre donné en paramètre et la fonction à exécuter sur le clique donné également en paramètre
* divider\_widget.dart :  
  widget responsable pour la création d’un diviseur

Text

Description automatically generated

Pour utilise ce widget

* buildDivider() ou buildDividerTransparent()
* hamburgermenu\_widget.dart :  
  widget responsable pour la création d’un bouton de type burger  
  le menu avec les options compte, rapport, paramètres et aide s’affichent lorsqu’on clique  
  et nous redirectionne vers l’une des pages sélectionnée



Pour utilise ce widget

* HamburgerMenu() :  
  permet d’afficher un button de type burger  
  widget de type drawer
* logo\_widget.dart :  
  widget responsable pour la création du logo avec une taille fixe de 25% de l’écran

Text

Description automatically generated

Pour utilise ce widget

* Logo() :  
  permet d’afficher un containeur avec le logo a l’intérieur
* navbar\_widget.dart :  
  widget responsable pour la création des bar de navigation inférieure  
  ce fichier comporte les différentes bar de navigation utilisés dans l’application

Text

Description automatically generated

Pour utilise ce widget

* BottomNavigationBarSmsAutoTwo() :
* BottomNavigationBarSmsProgTwo() :
* BottomNavigationBarSection() :
* profile\_widget.dart :  
  widget responsable pour la création de l’image du profile avec l’icone pour modification

Text

Description automatically generated

Pour utilise ce widget

* ProfileWidget(String imagePath, bool isEdit, VoidCallback onClicked) :  
  permet de superposes des widgets sur les autres pour créer le bouton d’ajout sur l’image
* textfield\_widget.dart :  
  widget responsable pour la création d’un champs de text
* Text

  Description automatically generated

Pour utilise ce widget

* textField(« placeholder », « controller », numeroDeLignes) :  
  permet d’afficher un containeur avec un champs de texte à l’intérieure

Fonctionnalités :

Permissions:

Package: <https://pub.dev/packages/permission_handler>

On demande les permissions dans l’initialisation de l’état de la class MyApp( classe principale de l’application ) qui est la classe qui sera attaché à l’écran avec la fonction runApp(MyApp()) avec argument MyApp() dans la fonction main()

Text

Description automatically generated

Envoie de mail :

Package : <https://pub.dev/packages/url_launcher>

Text

Description automatically generated

Messages automatiques:

Package: <https://pub.dev/packages/telephony>

Graphical user interface, application

Description automatically generated

On appelle la fonction initPlatformState dans l’initialisation de l’état de la classe SmsAuto

Text

Description automatically generated

Text

Description automatically generated

Il faut ensuite definer les fonctions : onMessage et onBackgroundMessage

Text

Description automatically generated

Text

Description automatically generated

Message programmé

Package : <https://pub.dev/packages/telephony>

On appelle la fonction périodiquement toutes les 20secondes la fonction sendSms() dans la fonction initState de la classe SmsProg()

Text

Description automatically generated

Text

Description automatically generated