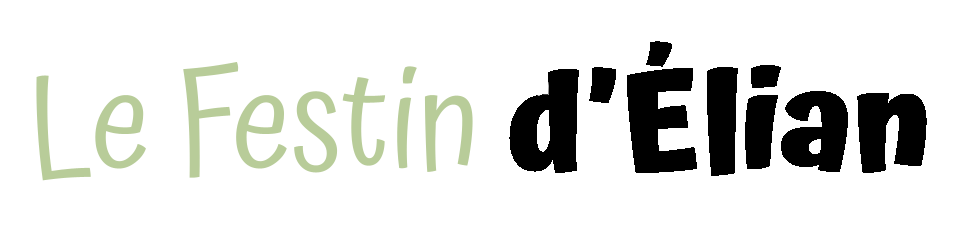
Application Maraîchage

RAPPORT DE PROJET

BTS Systèmes Numériques

Option Informatique & Réseaux

Année 2021/2022



TOURNACHE Nicolas

BERCHEL Rudy

MAUREL Robin

LORIN Dorian

Table des matières

[I – Présentation 2](#_Toc105148124)

[1. Présentation de l’entreprise 2](#_Toc105148125)

[2. Contexte du projet 2](#_Toc105148126)

[II – Objectif du projet 3](#_Toc105148127)

[1. Objectif principal 3](#_Toc105148128)

[2. Matériel 3](#_Toc105148129)

[3. Logiciels & Langages utilisés 4](#_Toc105148130)

[III – Spécifications générales 5](#_Toc105148131)

[1. Diagramme de Gantt (planification des tâches) 5](#_Toc105148132)

[2. Diagramme de cas d’utilisation 6](#_Toc105148133)

[3. Diagramme de séquence 7](#_Toc105148134)

[4. Diagramme de déploiement 8](#_Toc105148135)

[5. Structure de la base de données 9](#_Toc105148136)

[IV – Répartition des tâches 10](#_Toc105148137)

[1. Tableau de répartition des tâches 10](#_Toc105148138)

[2. Tâches sur le diagramme de cas d’utilisation 11](#_Toc105148139)

[V – Développement de ma partie 12](#_Toc105148140)

[1. Description de ma partie 12](#_Toc105148141)

[2. Matériel & technologies employé(e)s 12](#_Toc105148142)

[3. Écran d’accueil de l’application 13](#_Toc105148143)

[A. Écran d’accueil 13](#_Toc105148144)

[B. Fonctions d’authentification 14](#_Toc105148145)

[C. Algorigramme de l’écran d’accueil : 15](#_Toc105148146)

[4. Affichage des casiers 16](#_Toc105148147)

[A. Liste des casiers 16](#_Toc105148148)

[B. Algorigramme partie casiers : 17](#_Toc105148149)

[VI – Annexes 18](#_Toc105148150)

[1. Code Écran d’accueil 18](#_Toc105148151)

[A. Fichier AppViewModel 18](#_Toc105148152)

[B. Fichier Login 19](#_Toc105148153)

[C. Fichier Inscription 21](#_Toc105148154)

[D. Fichier ContentView 24](#_Toc105148155)

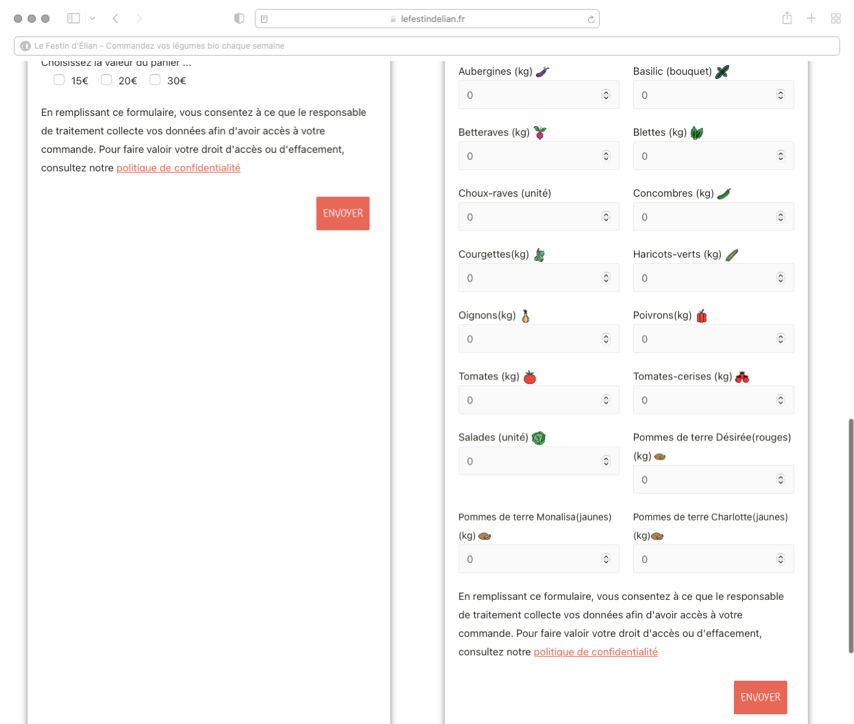
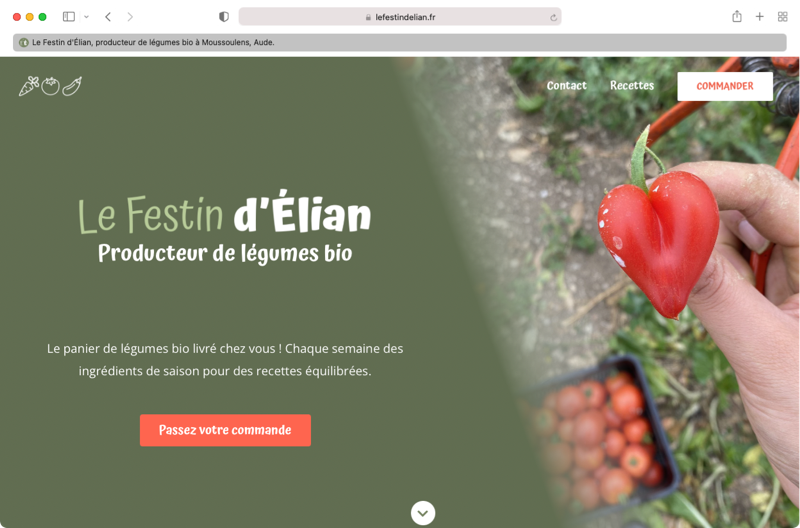
[2. Code affichage des casiers 26](#_Toc105148156)

[A. Fichier CasierModel 26](#_Toc105148157)

[B. Fichier CasierView 27](#_Toc105148158)

# I – Présentation

## Présentation de l’entreprise

Le partenaire pour lequel nous avons réalisés ce projet est l’entreprise Le Festin d’Elian. Le Festin d’Elian est un producteur de légumes Bio qui livre des paniers de légumes Bio chaque semaine. Un site web est à disposition afin de pouvoir commander ses paniers légumes par internet :

## Contexte du projet

Une image contenant intérieur, plafond, bois

Description générée automatiquementL’entreprise permet aussi de prendre des paniers en utilisant un système de distribution de légumes qui se compose de plusieurs casiers en libre-service. Il suffit d’utiliser un monnayeur ou une carte bleue afin d’ouvrir un casier qui correspond à un panier de légumes.

# II – Objectif du projet

## Objectif principal

L’entreprise souhaite mettre à jour son site Web et souhaite également avoir une application mobile sur iPhone/iPad. Elle souhaiterait pouvoir suivre l’évolution des prises de paniers de légumes dans les casiers. Il faut donc développer une solution permettant de visualiser l’état de stocks des casiers en temps réel et d’envoyer des notifications lorsque les clients prennent des paniers de légumes.

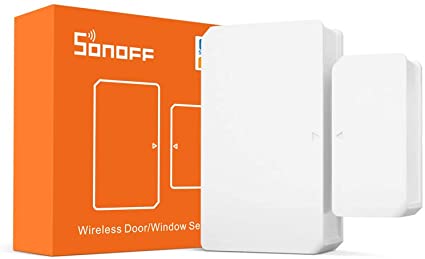
**Objectifs Principaux :**

* **Utilisation de capteurs :** Permettre de détecter l’ouverture d’un casier lors de la prise d’un panier de légumes et de savoir lorsqu’un casier est vide.
* **Serveur WEB :** Mise en œuvre d’un site WEB amélioré sous Wordpress permettant d’afficher l’état des casiers.
* **Base de données :** La base de données permettra de stocker les informations des casiers comme leurs état ainsi que l’horodatage de la dernière prise de paniers légumes.
* **Application iOS :** L’application permettra de visualiser l’état des casiers en direct (plein ou vide), l’horodatage de la dernière ouverture et une notification sera envoyée lorsqu’un casier sera ouvert.

## Matériel

Une image contenant équipement électronique, circuit

Description générée automatiquementPour réaliser ce projet, nous avons eu besoin du matériel suivant :

* **1 Raspberry PI**
* **Une image contenant texte

  Description générée automatiquement28 Capteurs d’ouverture Sonoff ZIGBEE SNZB-04**
* **1 Module de communication ZIGBEE**

## Logiciels & Langages utilisés

Afin de créer le site WEB, la base de données pour le stockage des informations des casiers, l’API pour communiquer avec la base de données, la mise en place des capteurs et la création de l’application, on a utilisé les logiciels et langages suivants :

Une image contenant texte, clipart, trousse de secours

Description générée automatiquement

* **HTML & CSS pour le site WEB**



* **MySQL & PhpMyAdmin pour la base de données**



* **Le langage Python pour les capteurs**



* **Le langage SwiftUI pour l’application iOS**



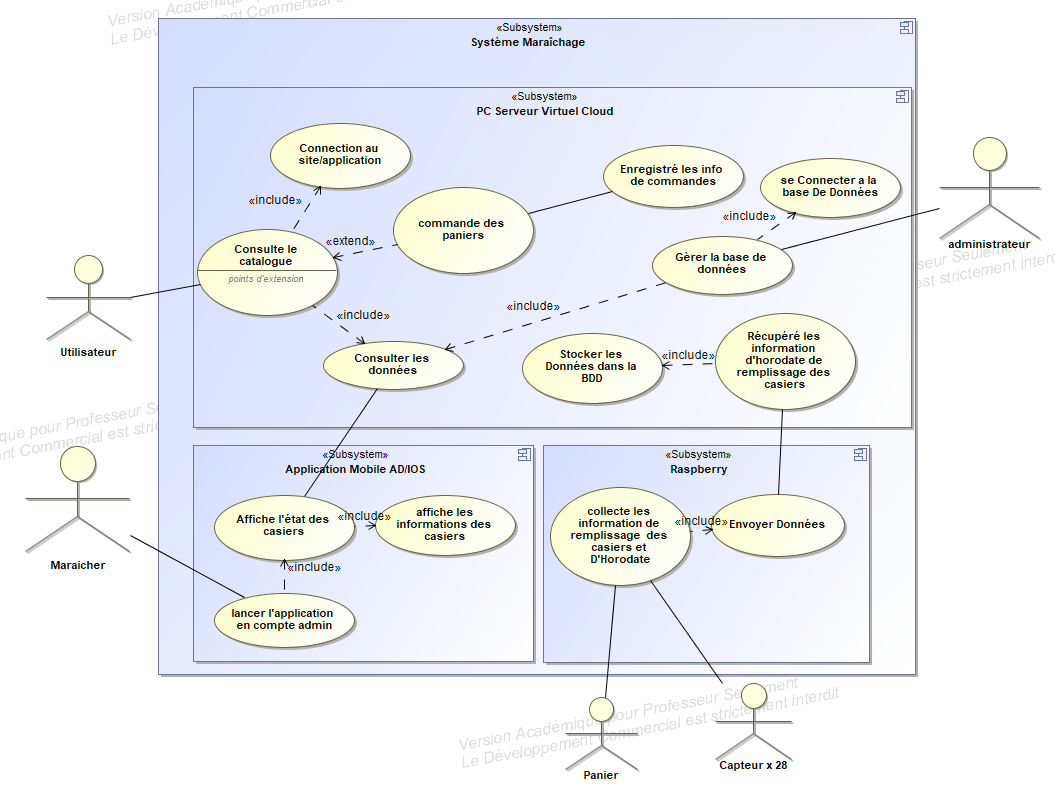
* **Le langage PHP pour l’API afin de communiquer   
  avec la base de données**

# III – Spécifications générales

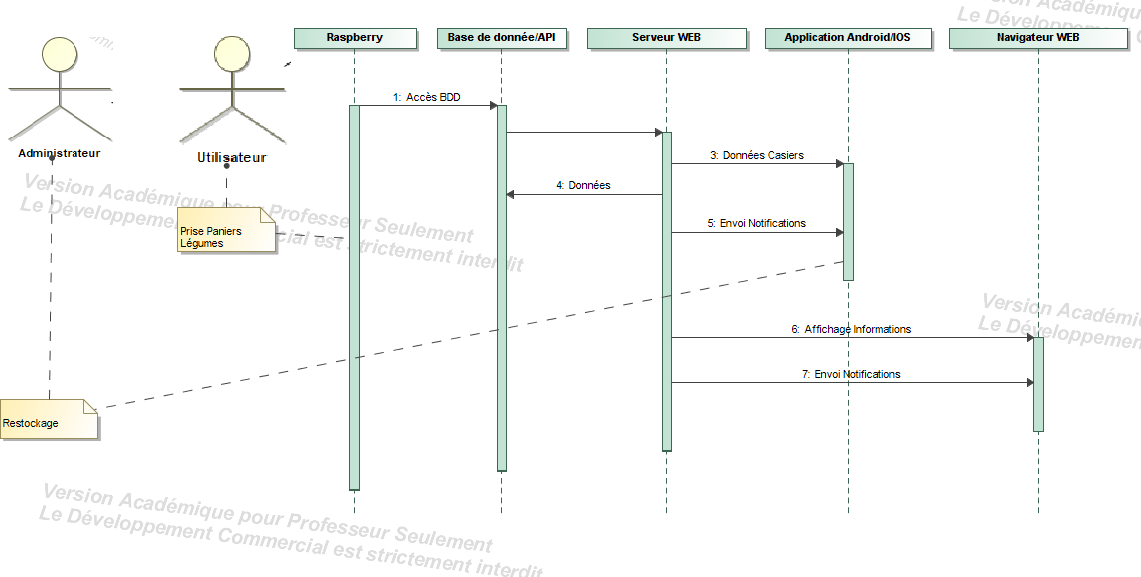
## Une image contenant table Description générée automatiquementDiagramme de Gantt (planification des tâches)

* LORIN Dorian
* BERCHEL Rudy
* MAUREL Robin
* TOURNACHE Nicolas

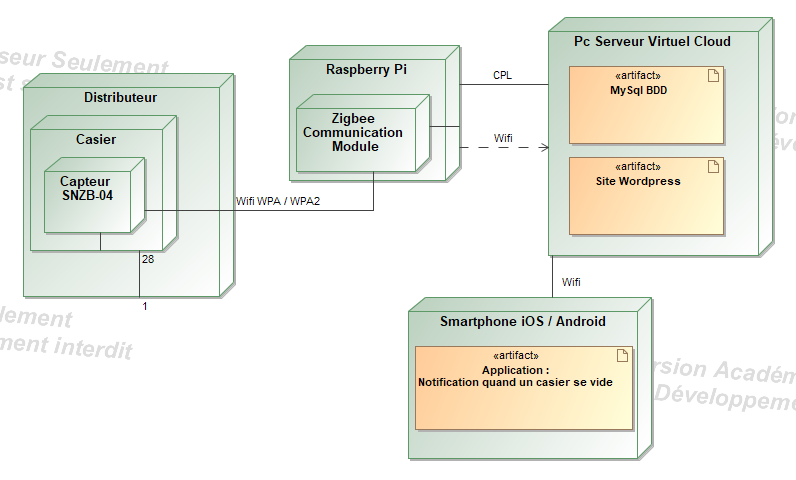
## Diagramme de cas d’utilisation



## Diagramme de séquence



## Diagramme de déploiement



## Structure de la base de données

**Table casier :**

* IDcasier : Identifiants des casiers
* Casier : Noms des casiers (Casier 1, Casier 2, …)
* État : Permet de connaître l’état des casiers (vides ou plein)
* Horodate : Permet de stocker la date et l’heure de l’ouverture d’un casier

**Table casier\_légumes :**

* IDcasierlegumes : Identifiant du casier
* Casier : Nom du casier
* Carottes […] Pomme\_de\_terre\_rouges : Présence ou non du légume dans le casier

# IV – Répartition des tâches

## Tableau de répartition des tâches

|  |  |  |
| --- | --- | --- |
| NOM Étudiant | Tâche principale  Installer un serveur de base de données afin de stocker les informations des casiers et des informations pour l’administrateur.  Élaborer une API pour communiquer avec le serveur WEB et l’application mobile. Accéder avec une interface graphique à la base de données | Matériel/Solution |
| BERCHEL Rudy |  | **Serveur sur le cloud comprenant la base de données ainsi que l’API permettant de communiquer avec le site web et l’application.** |
| LORIN Dorian | Installer un serveur WEB sous Wordpress et y rajouter du contenu supplémentaire. Élaborer un service permettant de faire l’interaction entre la base de données et le serveur et ainsi permettre l’envoi de notifications sur le navigateur WEB et les applications mobiles. | **Serveur sur le cloud avec le site web connecté à la base de données.** |
| TOURNACHE Nicolas | Élaborer un logiciel sous iOS/Android permettant de visualiser l’état des casiers en temps réel et y intégrer un micro-service permettant de connaître l’état du stock de paniers de légumes afin de connaître les légumes à amener au système. Intégrer l’envoi de notifications à l’utilisateur ainsi qu’un onglet paramètres afin de configurer la distribution des notifications. | **Téléphone/Tablette sous iOS/Android**  **Xcode et le langage SWIFT pour le développement de l’application.** |
| MAUREL Robin | Choisir et dimensionner une technologie de connexion de données afin d’intégrer une solution matérielle sur le système afin de détecter la prise de paniers de légumes. Réaliser l’intégration logicielle dans le matériel sélectionné afin de détecter la prise de paniers de légumes. Élaborer un système de logs sur le matériel choisi en local pour horodater la prise de paniers de légumes. | **Raspberry PI**  **Capteurs d’ouverture** |

## Tâches sur le diagramme de cas d’utilisation

* MAUREL Robin
* TOURNACHE Nicolas
* BERCHEL Rudy
* LORIN Dorian

# V – Développement de ma partie

## Description de ma partie

Pour ce projet, ma tâche est de réaliser une application mobile qui affichera en temps réel l’état des casiers s’ils sont vides ou pleins. L’application devra également envoyer une notification lorsqu’un panier de légumes est récupéré. Ainsi le maraîcher pourra alors savoir quand il faut remplir un casier.   
Les tâches que j’ai réalisées sont :

* Réaliser une application iOS
* Récupérer les données de l’API
* Afficher les données de manière structurée dans l’application

## Matériel & technologies employé(e)s

Pour réaliser mon application j’ai utilisé les technologies suivantes :



* **Xcode** afin de coder l’application



* Le langage **SwiftUI**

Comme matériel j’utilisais mon téléphone afin de tester tout au long du projet mon application mobile.

## Écran d’accueil de l’application

### Écran d’accueil

Une image contenant texte, ipod, équipement électronique, téléphone mobile

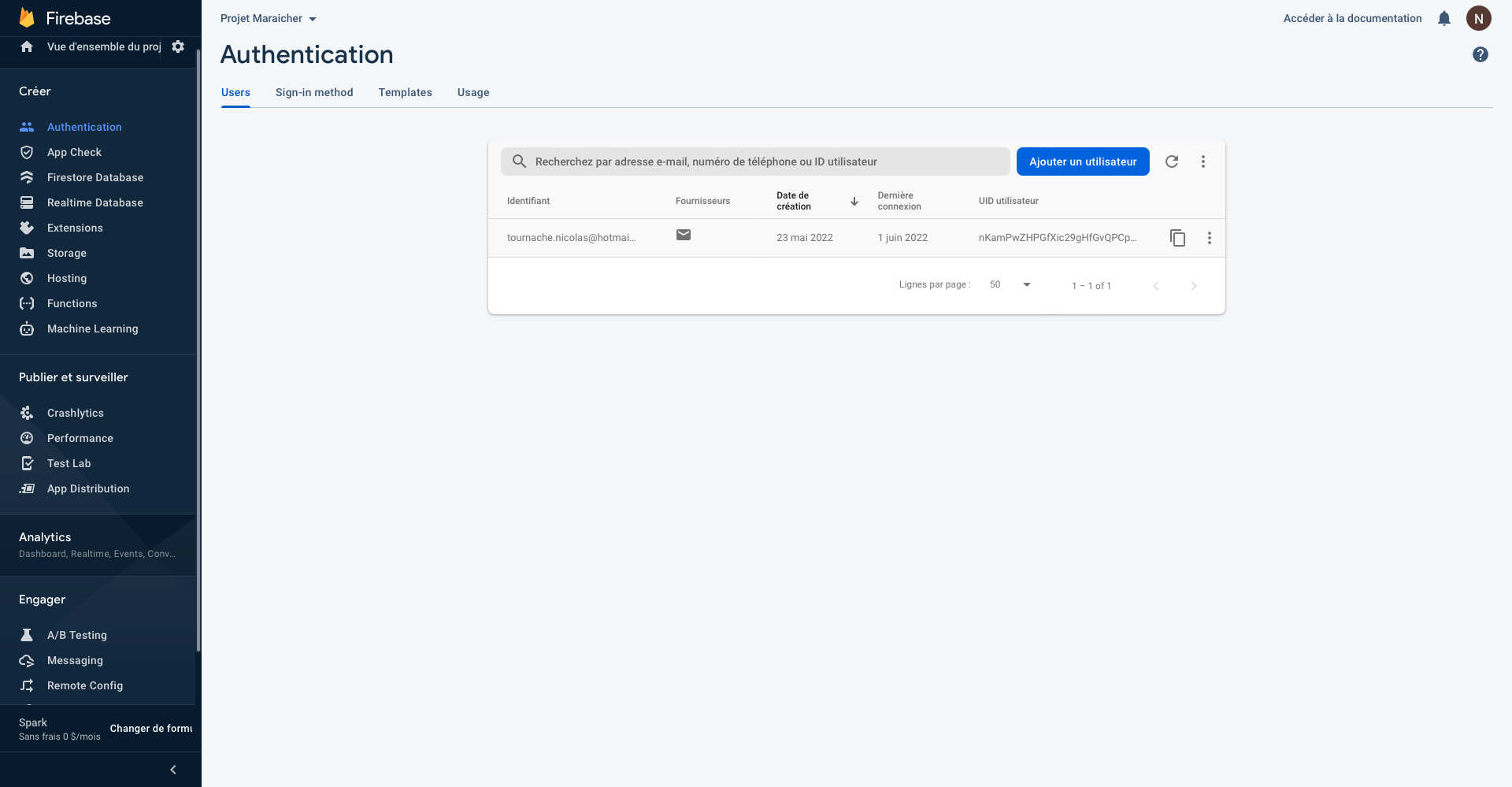
Description générée automatiquementAu lancement de l’application l’utilisateur doit se connecter à l’aide d’un compte créé précédemment via l’onglet inscription.

Une image contenant texte, moniteur, ipod, équipement électronique

Description générée automatiquement

**Connexion** **Inscription**

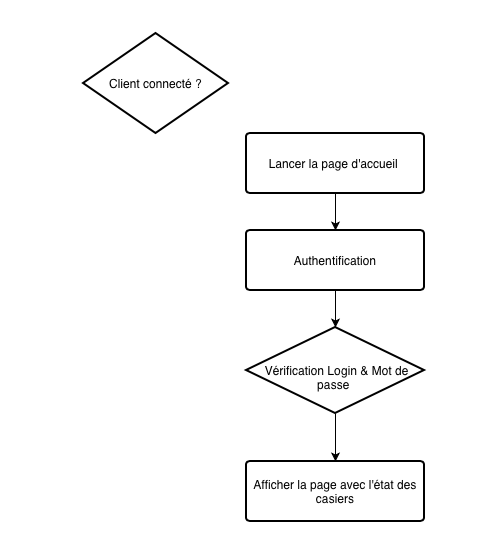
### Fonctions d’authentification

Pour les fonctionnalités d’authentification, j’utilise Firebase, un service en ligne fournissant plusieurs fonctionnalités comme une base de données, l’envoi de notification ou bien l’authentification.

***Une image contenant texte

Description générée automatiquementCapture d’écran de l’onglet Authentification sur Firebase.***  
  
Pour connecter Firebase a mon projet il a fallu rajouter quelques lignes de codes dans le fichier AppDelegate.

### Algorigramme de l’écran d’accueil :



NON

OUI

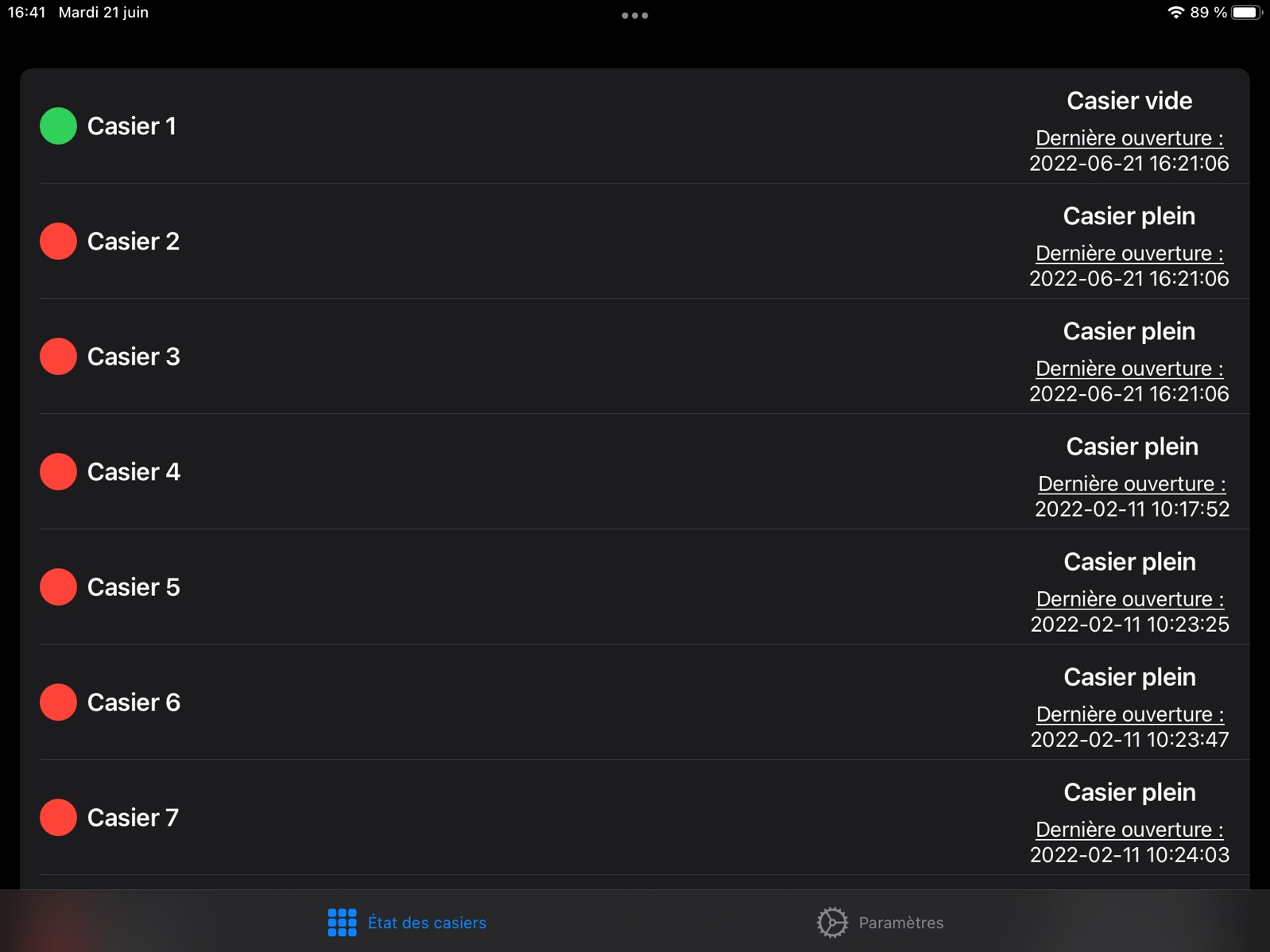
## **Affichage des casiers**

### Liste des casiers

Les parties Connexion et Inscription sont indépendantes et possèdent leurs propre vues.  
Une fois l’utilisateur connecté, la vue des casiers est affichée.  
  
Pour l’interface j’ai décidé de dresser une liste affichant la totalité des casiers. Une ligne correspond à un casier et sur chaque ligne on retrouve trois informations correspondant aux casiers :

* Une forme ronde désignant l’état des casiers : VERT -> PLEIN  
   ROUGE -> VIDE
* Le nom du casier avec son numéro
* L’heure et la date de la dernière ouverture du casier

Les noms des casiers sont récupérés comme tels depuis l’api.

La couleur indiquant l’état et le texte indiquant casier vide ou plein sont obtenu grâce à une fonction simple ; en fonction de l’état d’un casier représenté avec 0 ou 1 sur la base de données, le programme va afficher un point vert et afficher « casier plein », sinon, un point rouge et « casier vide »

### Algorigramme partie casiers :

Récupérer le fichier JSON sur le serveur

OUI

Récupérer l’URL

URL valide ?

NON

Décodage

Charger les données dans la structure

# VI – Annexes

## Code Écran d’accueil

### Fichier AppViewModel

Le fichier AppViewModel défini les fonctions de connexion, inscription, déconnexion et mot de passe oublié.

### Fichier Login

Le fichier login correspond à la vue login avec les champs de connexion.

Une image contenant table

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

### Fichier Inscription



­­

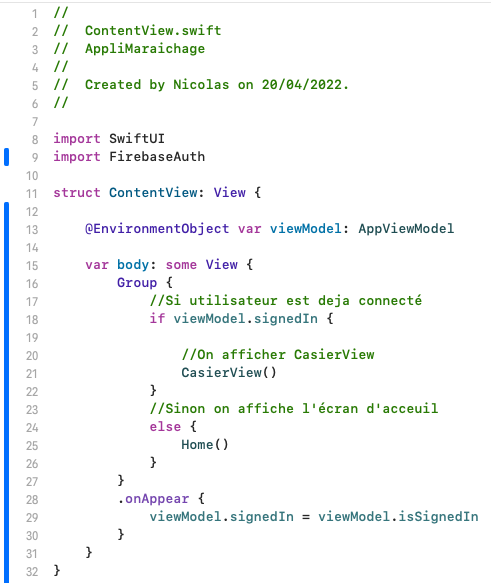


Une image contenant texte

Description générée automatiquement

### Fichier ContentView

Le fichier ContentView contient les instructions au lancement de l’application.



Dans le fichier ContentView, une deuxième structure nommée « Home » permet de lier les vues Connexion et Inscription.

## Code affichage des casiers

### Fichier CasierModel

Le fichier CasierModel défini la structure des données récupérées de l’API soit :

* Le nom des casiers avec leurs numéros
* L’état des casiers
* L’horodatage des casiers

Une fonction decode() permet de se connecter à l’API et de charger les données dans la structure.

### Fichier CasierView

Une image contenant texte

Description générée automatiquementLe fichier CasierView est la vue qui s’affiche lorsque l’utilisateur est connecté. Cette vue affiche une liste des casiers.

Une image contenant texte

Description générée automatiquement