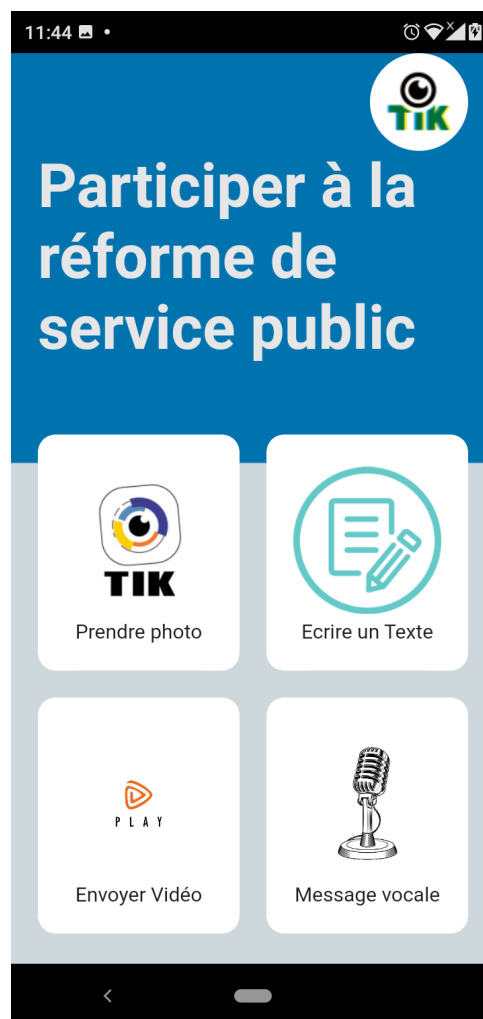


Application mobile Réclamation par photo de dysfonctionnement de service public

code source : <https://github.com/biko2020/tik>



Liste des figures

Figure -9 Scénario.	52
Figure -10 Représentation de types de réclamations.	53
Figure -11 Diagramme de cas d'utilisation <<utilisateur>>.	58
Figure -12 Diagramme de cas d'utilisation <<administrateurs>>.	59
Figure -13 Diagramme de classe.	63
Figure -14 Diagramme de séquence utilisateur.	64
Figure -15 Diagramme de séquence administrateur.	65
Figure -16 Diagramme de séquence système du fragment référencé.	66
Figure -17 Afficher la version ubuntu.	75
Figure -18 Scénario CI/CD Pipeline.	76
Figure -19 Créer une nouvelle github actions.	77
Figure -20 Choix de template.	77
Figure -21 Fichier dart.yml.	77
Figure -22 Créer un Token.	78
Figure -23 Test & build.	79
Figure -24 Création de version.	80
Figure -25 vérifier la version de docker.	81
Figure -26 Extrait du fichier Dockerfile.	82
Figure -27 Push l'image Docker via la ligne de commande.	83
Figure -28 l'image sur DockerHub.	83
Figure -29 l'exécution de l'application Tik.	84
Figure -30 Logo de l'application Tik	85
Figure -31 Page d'accueil réclamations	86
Figure -32 Interface prendre une photo.	87
Figure -33 Interface Ecrire une réclamation.	88

Figure -34 Interface Envoyer une réclamation.	89
Figure -35 Interface Authentification.	90
Figure -36 Interface Gestion réclamations.	91
Figure -37 Interface de l'application.	92

Table des matières

PROBLÉMATIQUE

ANALYSE , CONCEPTION ET RÉALISATION.

Analyse et Conception

La Conception de l'application

Spécification de besoins

Fonctionnalités

UML

Identification des Acteurs

Analyse des besoins fonctionnels

Les besoins fonctionnels liés au utilisateur

Les besoin fonctionnels liés au administrateur

Analyse des besoins non fonctionnels

Diagramme de cas d'utilisation

Diagramme de cas d'utilisation <<utilisateurs>>

Diagramme de cas d'utilisation <<administrateurs>>

Sommaire d'identification

Utilisateurs

Administrateurs

Diagramme de classe

Diagramme de séquence

Diagramme de séquence <<utilisateur>>

Diagramme de séquence <<administrateur>>

Développement de l'application

Environnement de développement

Systèmes d'Exploitation

Ubuntu 21.10

Technologies Utilisée

Framework

Packages et Services

IDE ET LANGUAGES DE PROGRAMMATION

Workflows avec GitHub Action

Scénario CI/CD Pipeline

Travailler Avec GitHub Actions

Conteneurisé l'environnement de développement

GUI

Logo

GUI client

Interface client, contient les types de réclamations

Ecran prendre une photo de l'anomalie

Ecrire une réclamation

Adresser la réclamation au service concerné, avec le message de confirmation.

GUI Administrateur

Interface web pour gérer les réclamations client

Gestion des réclamations avec l'interface administrateur

Interface de l'application web

Conclusion

PROBLÉMATIQUE .

État du problème.

Le service public est une activité d'intérêt général, gérée par une administration publique, Toutes les données recueillies suite à une enquête dans la rue et dans les réseaux sociaux, il ressort que la perception des services publics par les usagers reste globalement négative. La relation à l'administration est vécue par le citoyen comme un rapport de forces qui lui est défavorable. Le service rendu est perçu comme une faveur plutôt qu'un droit, ce qui se traduit souvent non seulement par une dispense de comptes à rendre, mais également par une libre interprétation des textes et des procédures qui régissent le service.

Le suivi et l'entretien d'un service public est une phase critique soit pour la commune ou pour les entreprises de prestation, Il est noté que ce service a fait l'objet de plusieurs revendications, émanant des différentes composantes de la société, sur l'état des routes, la gestion des déchets ménagers, , l'eau potable, ,l'éclairage public, transport

Dans ce contexte , il est important d'impliquer le citoyen au processus de réforme des services publics en mettant à leur disposition un moyen pour adresser une réclamation au département concerné. Cela permettrait aux responsables de prendre les mesures nécessaires pour remédier aux différents dysfonctionnements signalés par les citoyens et les usagers).

ANALYSE , CONCEPTION ET RÉALISATION.

Analyse et Conception

Le Concepte de l'application

Spécification de besoins

La spécification de besoins, constitue la phase de départ ou la feuille de route de toute application à développer dans laquelle je vais identifier les besoins de notre application. je vais distinguer les besoins fonctionnels qui présentent les fonctionnalités attendues de notre application et les besoins non-fonctionnels pour éviter le développement d'une application non satisfaisante ainsi de trouver un accord commun entre les spécialistes et les utilisateurs pour réussir le projet Ci-après une présentation visuelle du concept de l'application.

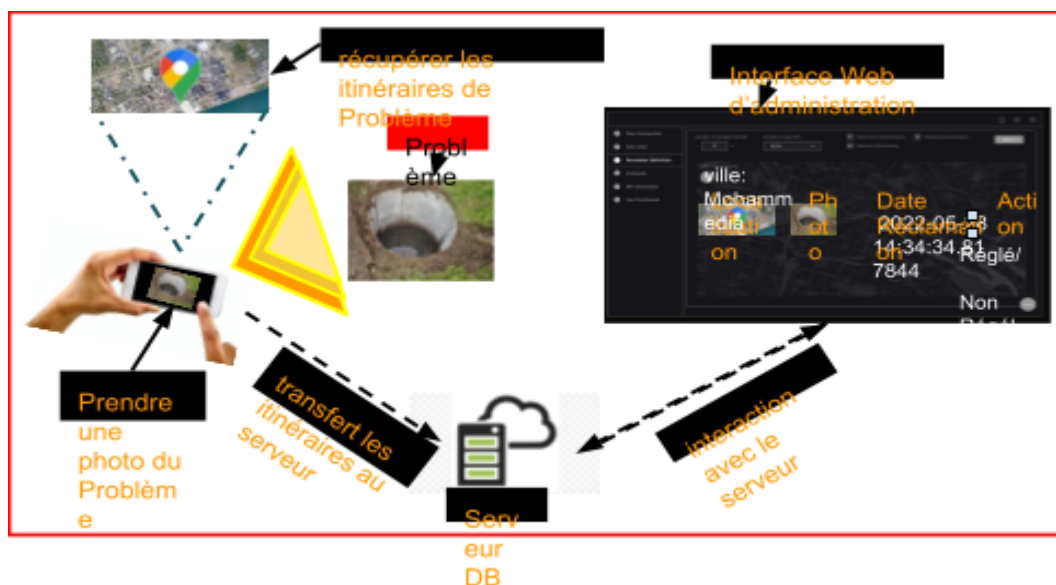


Figure -9 Scénario.

Fonctionnalités

Les réclamations peuvent être litigieuses, la raison pour laquelle les gens ne réclament pas. En plus la procédure de réclamation est très lente et très longue. Dans ce contexte j'ai pensé à cette solution qui sert dans un premier temps à traiter les réclamations par photo et par texte vue la simplicité de ce type de réclamation afin de bien prendre en compte le feedback de l'administration vis-à-vis du problème en question.



Figure -10 Représentation de types de réclamations.

UML

Identification des Acteurs

L' énumérer des acteurs susceptibles d'interagir avec notre application.
Tout d'abord, nous devons commencer par définir ce qui est un acteur.

Un acteur modélise le type de rôle joué par une entité qui interagit avec le système modélisé (utilisateur, dispositif matériel ou autre système) qui interagissent directement avec le système étudié.

l'application présente deux parties: application mobile et site web
Les acteurs de notre application mobile sont:

- Utilisateurs : les citoyens et les usagers qui peuvent consulter les services.

Les acteurs de notre site web est :

- Administrateurs: les administrateurs , les agents qui gérer les réclamations.

Analyse des besoins fonctionnels

Les besoins fonctionnels liés au utilisateur

- Prendre une photo.
- Écrire une réclamation.

Les besoin fonctionnels liés au administrateur

- L'authentification.
- Validation des réclamations.
- Supprimer les réclamations.

Analyse des besoins non fonctionnels

Les spécifications non fonctionnelles décrivent les contraintes auxquelles est soumis l'application pour sa réalisation et son bon fonctionnement :

Performance : L'application doit faire face à la capacité à monter en charge et doit également avoir un temps de réponse rapide.

Sécurité : L'application doit être sécurisée, en assurant l'intégrité et le non répudiation des données stockées dans la base, compte tenu qu'elles reflètent des informations privées sur les différents acteurs.

Ergonomie : Les interfaces utilisées par l'application doivent être claires, concises et faciles à manipuler.

Fiabilité : L'application doit assurer l'échange des données et n'en perdre aucun détail.

Configuration : La configuration de l'application ne doit présenter aucune difficulté pour un simple utilisateur.

Diagramme de cas d'utilisation

Un cas d'utilisation <<use case>> représente un ensemble de séquences d'actions qui sont réalisées par le système et qui produisent un résultat observable intéressant pour un acteur particulier.

Chaque cas d'utilisation spécifie un comportement attendu du système considéré comme un tout, sans imposer le mode de réalisation de ce comportement. Il permet de décrire ce que le futur système devra faire, sans spécifier comment il le fera.

Par la suite les diagrammes de cas d'utilisation de chaque partie.

Diagramme de cas d'utilisation <<utilisateurs>>

Le cas d'un utilisateur qui peut faire sa réclamation par photo via l'application Tik.

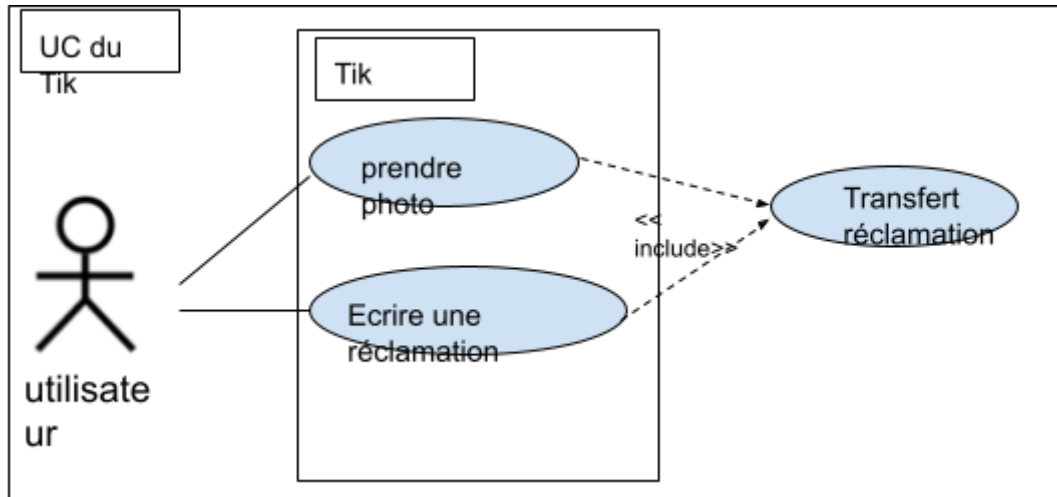


Figure -11 Diagramme de cas d'utilisation <<utilisateur>>.

Diagramme de cas d'utilisation <<administrateurs>>

Le cas d'un administrateur qui gère la validation et le suivi des réclamations

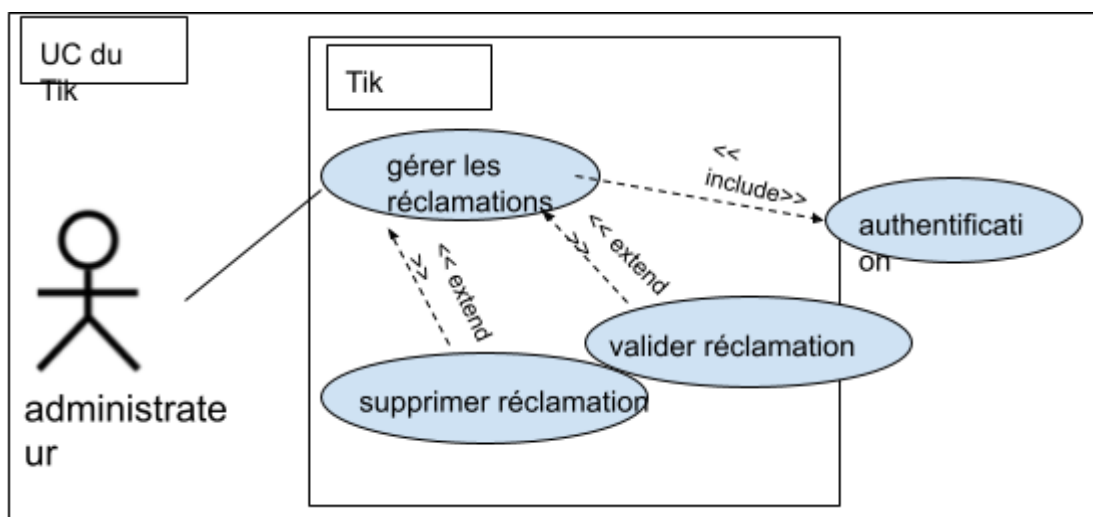


Figure -12 Diagramme de cas d'utilisation <<administrateurs>>

Sommaire d'identification

Utilisateurs

1- identification <<utilisateur>>

Titre : faire une réclamation

Résumé : ce cas d'utilisation permet aux utilisateurs de faire une réclamation d'un problème par photo et texte.

Acteurs :

- principale c'est l'utilisateur

2- Description des scénarios

Préconditions :

- la connexion internet est activée
- la caméra du système est opérationnelle

Scénario normal :

- l'utilisateur clique sur le bouton réclamation.
- la caméra s'active
- l'application demande au utilisateur de prendre une photo
- l'utilisateur prendre une photo de réclamation
- l'application demande au utilisateur d'envoyer la réclamation
- l'utilisateur clique sur le bouton envoi réclamation
- l'application demande au utilisateur la permission de d'accéder à ces paramètres de géolocalisation.

- l'application vérifier si l'accord est confirmé
- l'application récupère la position actuelle et transfère la photo

Administrateurs

1- identification <<utilisateur>>

Titre : valider une réclamation

Résumé : ce cas d'utilisation permet aux administrateurs de gérer les réclamations.

Acteurs :

- principale c'est l'administrateur

2- Description des scénarios

Préconditions :

- la connexion internet est activée
- l'interface web est opérationnelle

Scénario normal :

- l'administrateur lance l'interface web de l'application via le browser
- le système lui demande de s'authentifier
- L'administrateur doit être authentifié.
- le système vérifie le mot de passe et login
- le système affiche l'interface web appropriée sur le browser

Enchaînements d'erreur

- Authentification : l'application indique au administrateur que le mot de passe ou login n'est pas valide, le cas d'utilisation se termine en échec.
- Suppression non autorisée : l'application interdit la suppression d'une réclamation si cette dernière n'est pas résolue.

Diagramme de classe

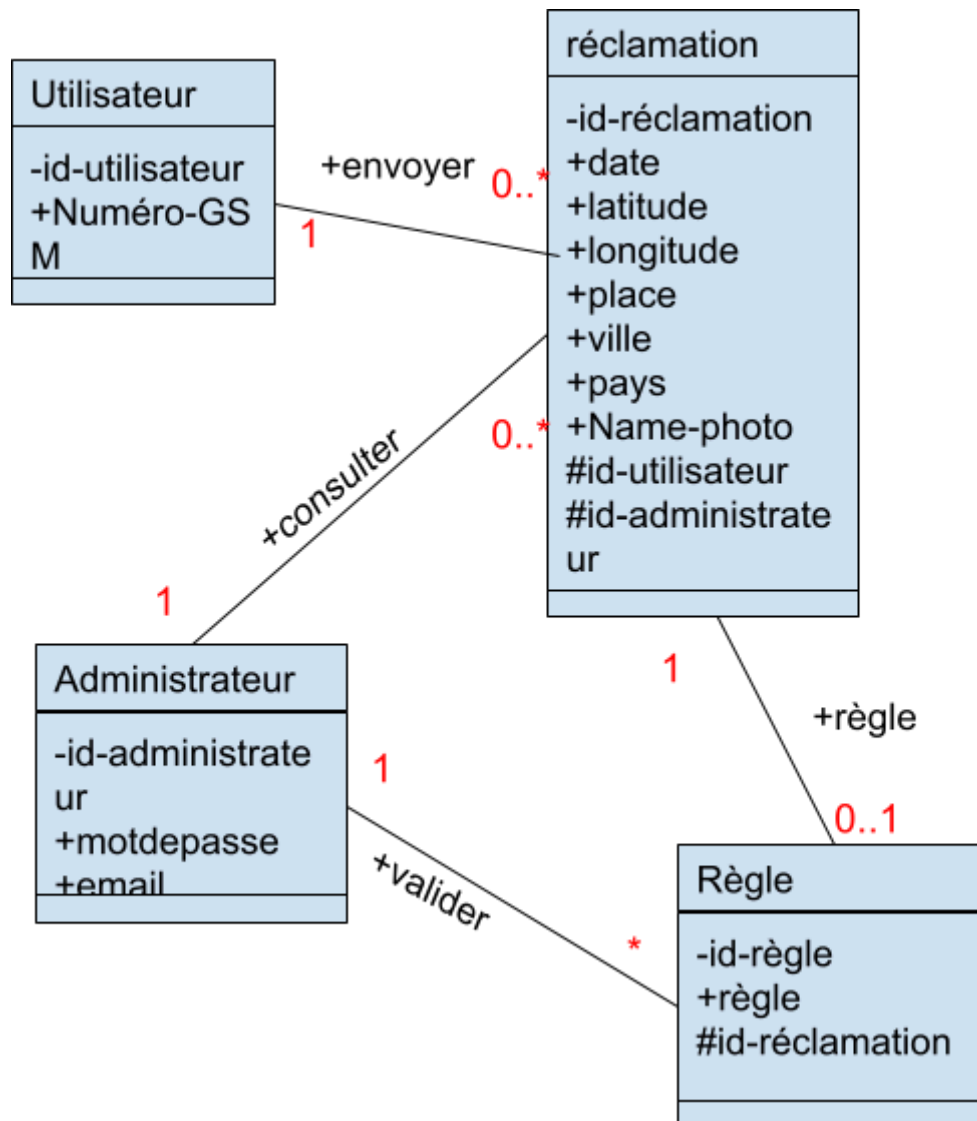


Figure -13 Diagramme de classe

Diagramme de séquence

Diagramme de séquence <<utilisateur>>

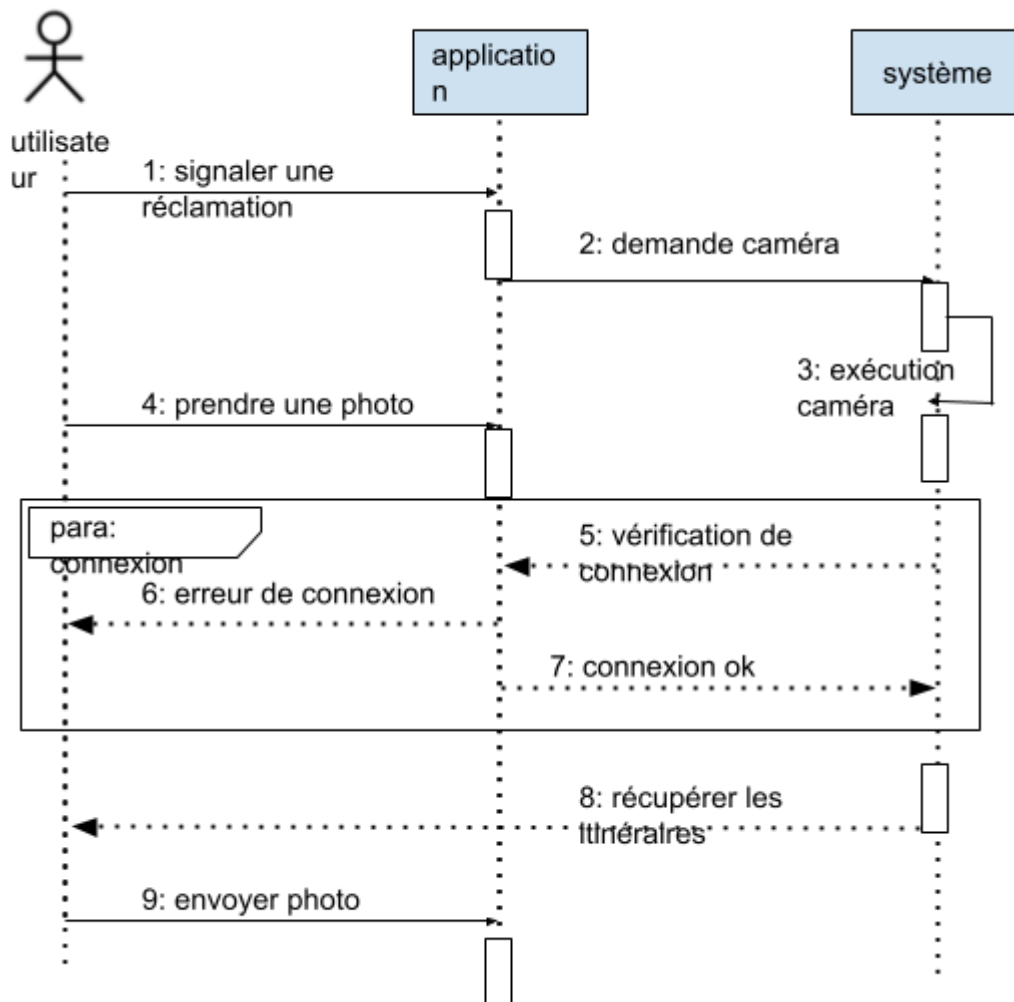
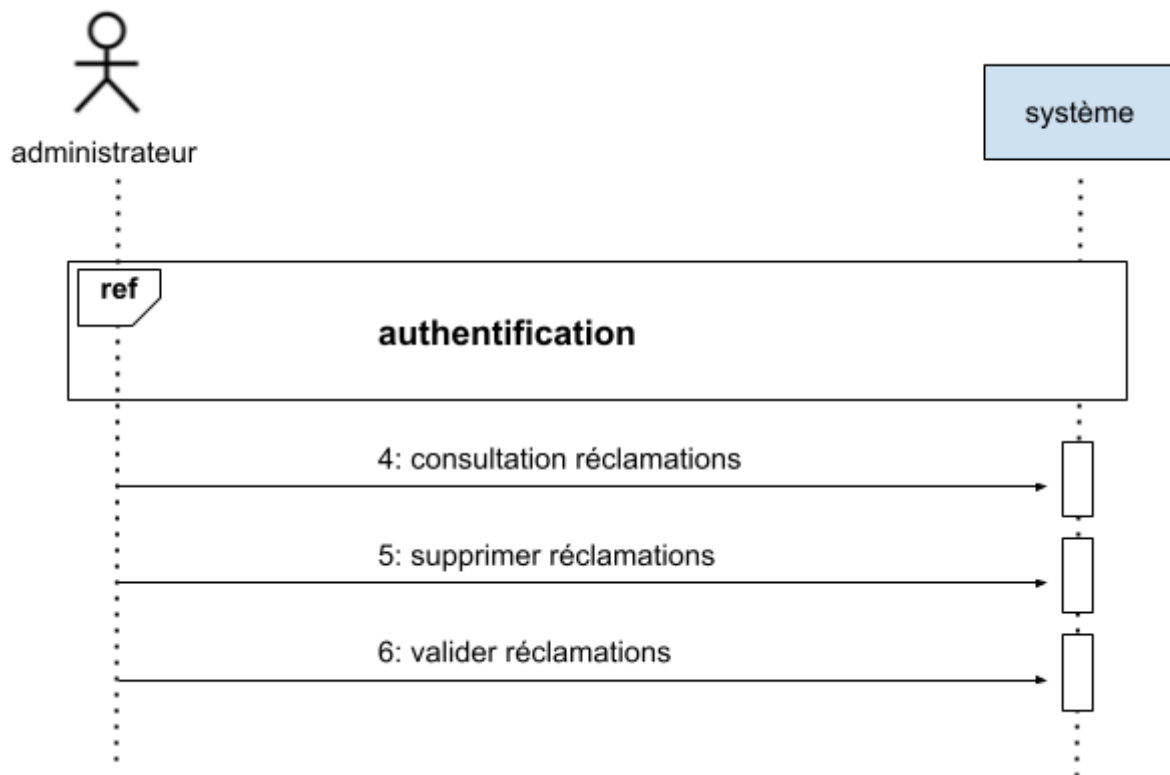


Figure -14 Diagramme de séquence utilisateur

Diagramme de séquence <<administrateur>>



le cadre ref fait référence à un autre diagramme de séquence dans la figure -16

Figure -15 Diagramme de séquence administrateur

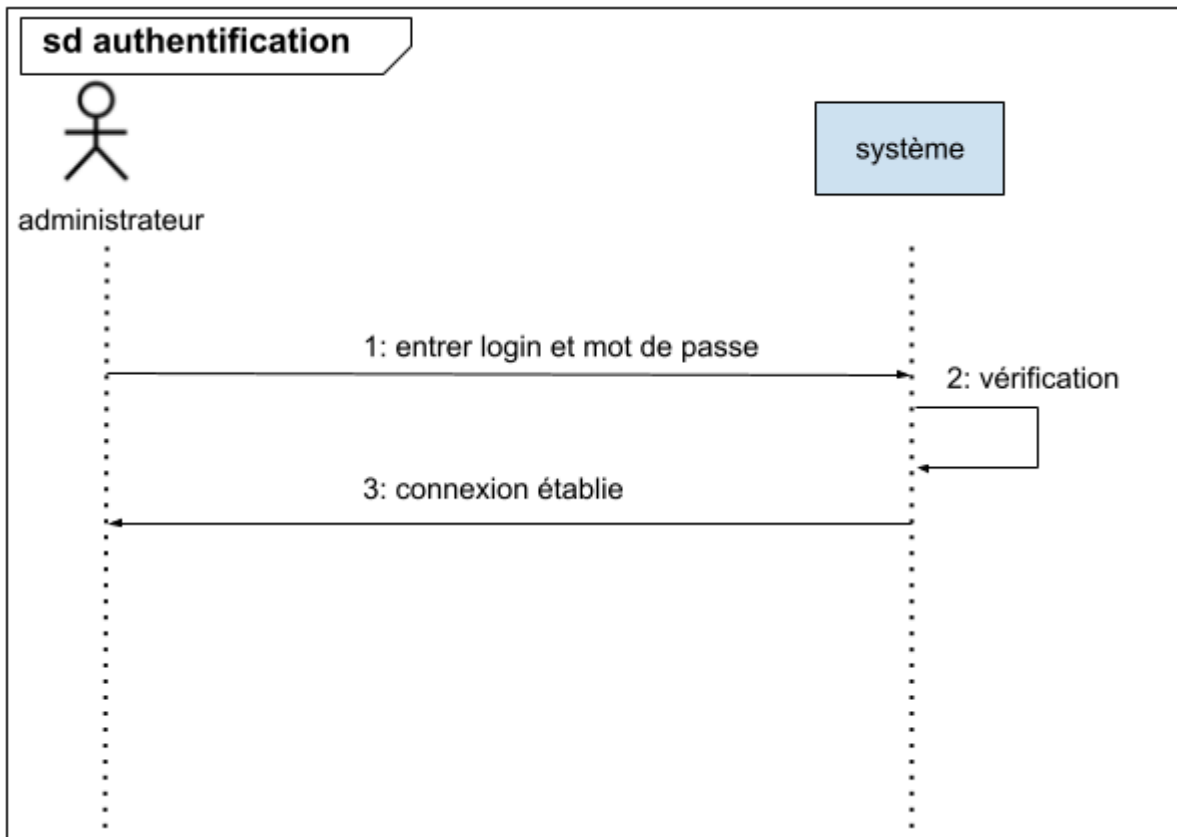


Figure -16 Diagramme de séquence système du fragment référencé

Développement de l'application

Cette partie correspond au développement de l'application, c'est-à-dire l'écriture des programmes. nous devant au préalable, et avant de démarrer la programmation, organiser ces développements dans le temps, en effectuant une planification. Cette activité consiste à déterminer et à ordonnancer les tâches de l'application, à estimer leurs charges en termes du temps et à déterminer les outils nécessaires à leur réalisation. Elle se décline sur le modèle décrit ci-après.

Environnement de développement

La phase du choix de technologie et systèmes utilisés au développement de l'application nous demande une vigilance , car le choix des outils et systèmes détermine la durée totale de développement de l'application en termes du temps ainsi que de la performance de cette dernière.

Systèmes d'Exploitation

Un système informatique moderne comprend un ou plusieurs processeurs, de la mémoire principale, des disques, des imprimantes, un clavier, un écran, des interfaces réseau et autres périphériques d'entrées/sorties. Tout bien considéré, c'est un système complexe.Écrire des programmes qui prennent en compte tous

ces composants et les utilisent correctement, de surcroît de façon optimale, est une tâche extrêmement difficile. Pour cette raison, les ordinateurs sont équipés d'une couche logicielle appelée système d'exploitation (OS en anglais) dont le rôle est de gérer tous les périphériques et de fournir aux programmes utilisateur une interface simplifiée avec le matériel.

ci-après quelque exemple des systèmes d'exploitations :

windows de Microsoft

Mac OS de Apple

GNU/Linux open source de Linus Torvalds

Oracle Solaris de Oracle

FreeBSD de Berkeley

Ubuntu 21.10

Ubuntu est un système d'exploitation GNU/Linux basé sur la distribution Debian. Il est libre, gratuit, et simple d'utilisation.

La première variante d'Ubuntu est fournie avec le bureau GNOME, destiné principalement à un usage bureautique ou domestique. Ubuntu est donc la variante la plus accessible et populaire, et pour laquelle vous trouverez le plus facilement d'aide sur le site officiel de ubuntu

Avec Ubuntu vous pouvez facilement et en toute sécurité accomplir tout ce que vous attendez de votre ordinateur : naviguer sur le Web, envoyer et recevoir vos

courriels, créer des documents et des présentations, gérer et éditer vos images, musiques et vidéos, jouer, et bien sûr faire le développement en informatique. Par contre windows est ne possède que les versions commerciale.

```
pc@pc-Inspiron-3580:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 21.10
Release:        21.10
Codename:       impish
```

Figure -17 Afficher la version ubuntu

Technologies Utilisée

Framework

Nuxtjs : Est un framework basé sur Vuejs, d'après mes recherches sur vue j'ai trouvé Nuxtjs, l'équivalent de Nextjs chez react. nuxtjs est un méta-framework car il est construit dessus du framework vuejs. je me suis dit qu'un bon point cas d'usage pour apprendre nuxtjs serait d'essayer de faire cette application. j'ai conscience de 3 méta- frameworks principaux, à savoir : Nuxtjs, Nextjs et Sveltekit, qui est la version dans Svelte. même si ce dernier à l'air très intéressant, il est encore trop jeune, donc on va la laisser pour plus tard. Nuxtjs et Nextjs c'est pareil, l'implémentation de cette solution consiste à faire du rendu js côté serveur (SSR: server-side-rendering), c'est-à-dire aller chercher nos données avant d'indiquer à Nuxt de rendre la page. Il va donc falloir attendre que notre travail de fond soit effectué pour rendre le HTML. Donc je fais mon choix sur Nuxtjs.

Flutter : Est un kit de développement logiciel (SDK) d'interface utilisateur open-source créé par Google. Il est utilisé pour développer des applications pour

Android, iOS, Linux, Mac, Windows, Google Fuchsia et le web à partir d'une seule base de code. Il y a aussi React Native qui propose des outils standards de base, mais le framework est tout de même riche en bibliothèques proposant de beaux composants UI adaptables selon le besoin. Mais flutter possède une documentation plus conviviale et facile à lire que celle de React Native.

Packages et Services

NPM : est l'abréviation de Node Package Manager, qui est un outil (programme) gérant les bibliothèques de programmation JavaScript pour Node.js. Cet outil est réellement nécessaire pour le monde open source. Dans la communauté JavaScript. Un logiciel similaire au NPM est Yarn, développé par Facebook, avec des fonctionnalités exceptionnelles qui émergent comme une alternative du NPM.

Firebase : est un ensemble de services d'hébergement pour n'importe quel type d'application (Android, iOS, Javascript, Node.js, Java, Unity, PHP, C++ ...). Il propose d'héberger en NoSQL et en temps réel des bases de données, du contenu, de l'authentification sociale (Google, Facebook, Twitter et Github), et des notifications, ou encore des services, tel que par exemple un serveur de communication temps réel. Lancé en 2011 sous le nom d'Envolv, par Andrew Lee et par James Templin, le service est racheté par Google en octobre 2014. Il appartient aujourd'hui à la maison mère de Google : Alphabet.

Git : Logiciel de gestion de version, il permet de conserver un historique des modifications effectuées sur un projet afin de pouvoir rapidement identifier les changements effectués, il facilite grandement la gestion de projets et permet

de travailler en équipe d'une manière plus efficace. il y en a d'autres comme gitlab...

GitHub : est un service en ligne qui permet d'héberger des dépôts ou repo Git. C'est le plus grand hébergeur de dépôts Git du monde.

Docker : Permet d'embarquer une application dans un ou plusieurs containers logiciels qui pourra s'exécuter sur n'importe quel serveur machine, qu'il soit physique ou virtuel. Docker fonctionne sous Linux comme Windows Server. C'est une technologie qui a pour but de faciliter les déploiements d'application, et la gestion du dimensionnement de l'infrastructure sous-jacente. Elle est proposée par la société Docker, en partie en open source.

Docker Hub : Conçu pour faciliter l'échange d'applications containerisées. Hébergeant plus de 100 000 images de containers, cet espace est aussi intégré à GitHub. Il recouvre de très nombreux domaines (analytics, bases de données, frameworks, monitoring, outils de DevOps, sécurité, stockage, systèmes d'exploitation....). Qualifiées officiellement, certaines images sont directement maintenues par Docker.

IDE ET LANGUAGES DE PROGRAMMATION

Visual Studio : est un éditeur de code extensible développé par Microsoft pour Windows, Linux et macOS. Les fonctionnalités incluent la prise en charge du débogage, la mise en évidence de la syntaxe, la complétion intelligente du code, les snippets, la refactorisation du code et Git intégré.

Dart : est un langage de programmation optimisé pour les applications sur plusieurs plateformes. Il est développé par Google et est utilisé pour créer des applications mobiles, de bureau, de serveur et web. Dart est un langage orienté objet à ramasse-miettes avec une syntaxe de type C++.

JavaScript : est un langage de programmation de scripts principalement employé dans les pages web interactives et à ce titre est une partie essentielle des applications web. Avec les langages HTML et CSS, JavaScript est au cœur des langages utilisés par les développeurs web.

Le HyperText Markup Language, généralement abrégé HTML ou, dans sa dernière version, HTML5, est le langage de balisage conçu pour représenter les pages web. Ce langage permet d'écrire de l'hypertexte, d'où son nom, de structurer sémantiquement la page, de mettre en forme le contenu.

CSS : forment un langage informatique qui décrit la présentation des documents HTML et XML. Les standards définissant CSS sont publiés par le World Wide Web Consortium.

Workflows avec GitHub Action

Mettre en place un système pipeline CI/CD (intégration continue et déploiement continu) basé sur GitHub Actions workflow afin de configurer et tester automatiquement le code et déployer l'application. ci-après le scénario CI/CD pipeline

Scénario CI/CD Pipeline

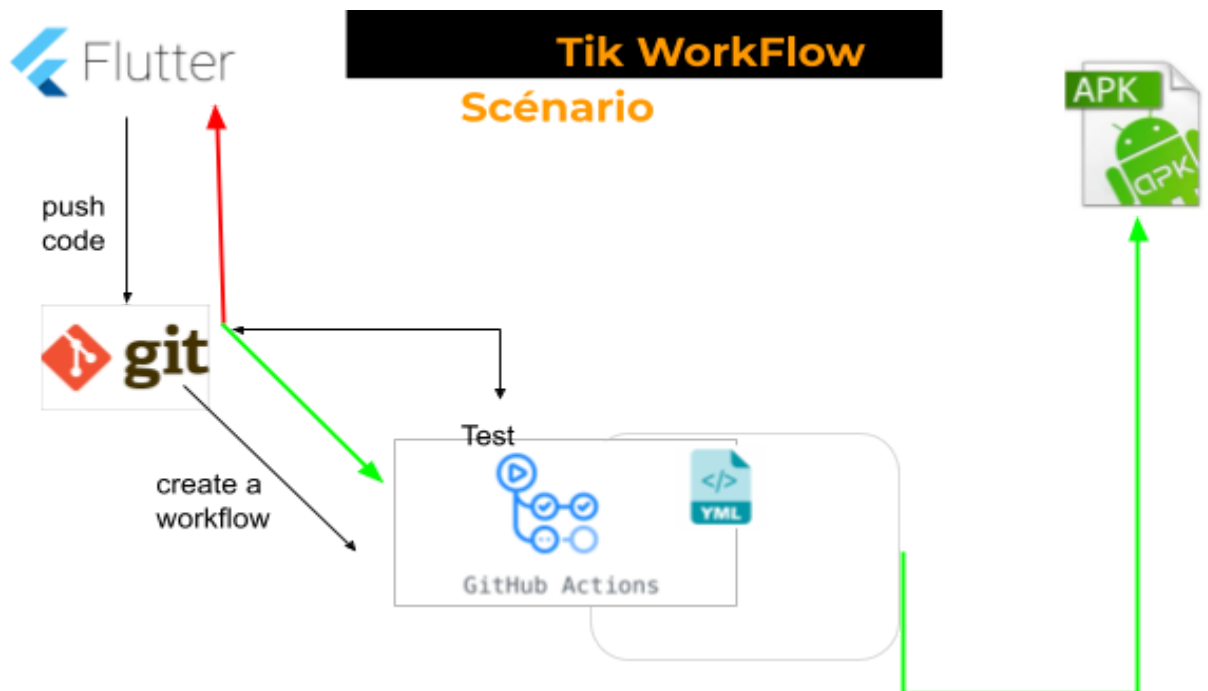


Figure -18 Scénario CI/CD Pipeline

Travailler Avec GitHub Actions

Préparer le flux de travail via la configuration de workflow afin d'automatiser les tests pour notre application flutter, puis générer un apk dans le dépôt GitHub.

Étape 1 : Création de nouvelle action, GitHub → Action

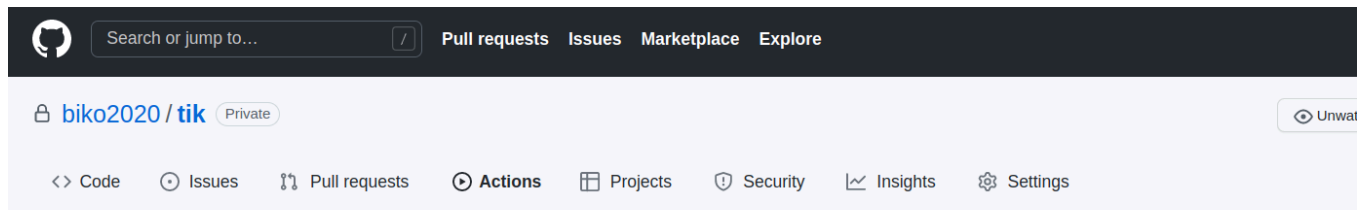


Figure -19 Créer une nouvelle github actions

Étape 2 : Le choix de template Dart

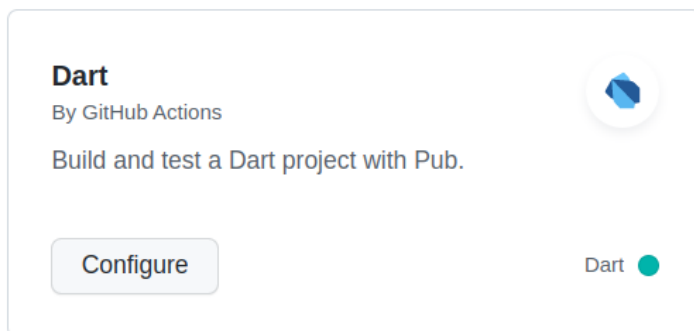


Figure -20 Choix de template

Étape 3 : Modification de fichier dart.yaml, comme illustré ci-dessous.

```

name: TIK CI

on:
  push:
    branches: [ master ]
  pull_request:
    branches: [ master ]

jobs:
  build:
    runs-on: ubuntu-latest

    steps:
      - uses: actions/checkout@v2
      - uses: actions/setup-java@v2
        with:
          distribution: 'zulu'
          java-version: '11'
      - uses: subosito/flutter-action@v2
        with:
          flutter-version: '2.5.3'
      - run: flutter pub get
      - run: flutter test
      - run: flutter build apk
      - run: flutter build appbundle

```

Figure -21 Fichier dart.yml

Étape 4 : Création de token d'accès, GH_ACTION_TOKEN. Il s'agit d'un token d'accès personnel qui sera utilisé par l'agent de build pour créer une version dans github.

biko2020 / tikPrivate

<> CodeIssuesPull requestsActionsProjectsSecurityInsightsSettings

General

Access

Collaborators

Code and automation

Branches

Actions

Webhooks

Pages

Security

Code security and analysis

Deploy keys

Secrets

Actions

Dependabot

Actions secrets / New secret

Name

GH_Action_Token

Value

Add secret

Figure -22 Créer un Token

Étape 5 : Test & build apk

biko2020 / tikPrivate

<> Code

Issues

Pull requests

Actions

Projects

Security

change flutter version 2.5.3 to 2.10.3 TIK CI/CD #13

Summary

Jobs

build

build

succeeded 6 minutes ago in 3m 23s

> Set up job

> Run actions/checkout@v2

> Run actions/setup-java@v2

> Run subosito/flutter-action@v2

> Run flutter pub get

> Run flutter test

> Run flutter build apk

> Run flutter build appbundle

> Post Run subosito/flutter-action@v2

> Post Run actions/setup-java@v2

Figure -23 Test & build

Étape 5 : Création de la version

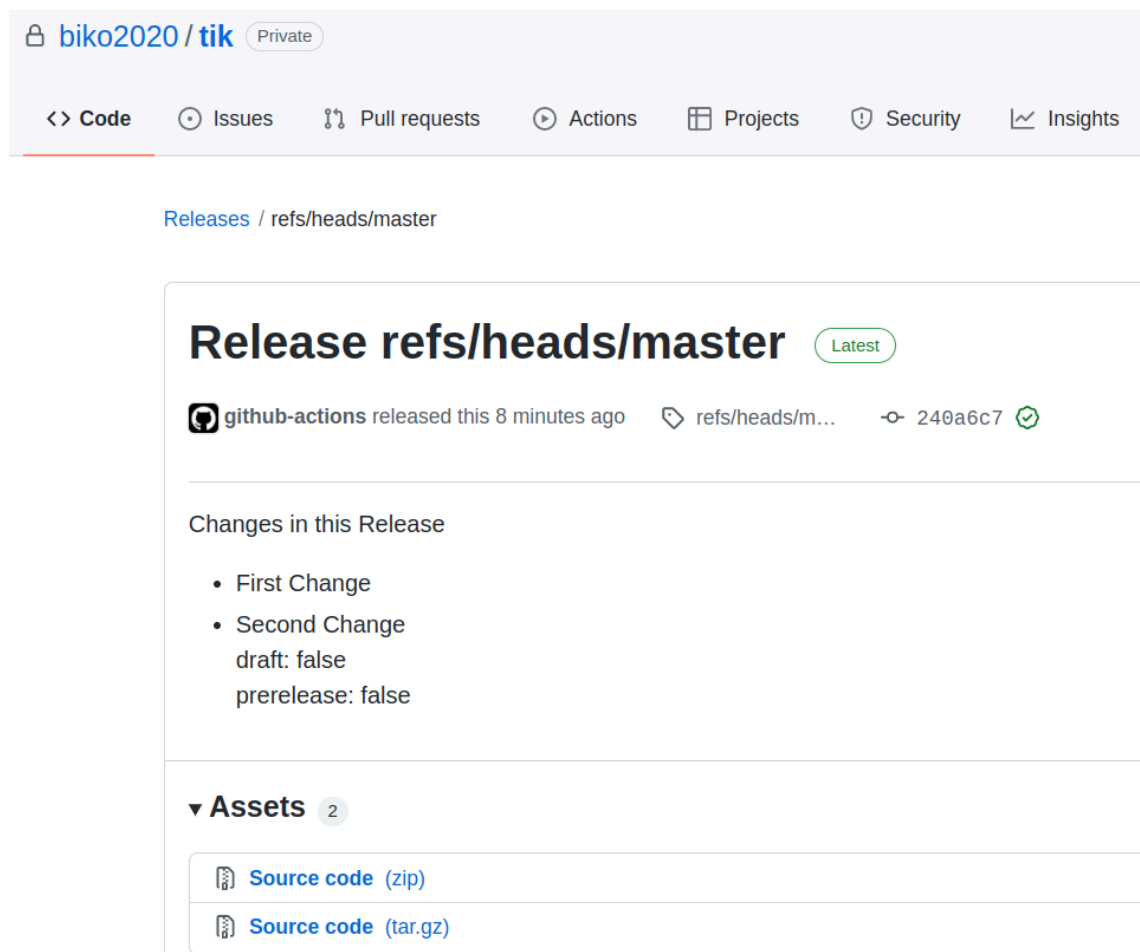


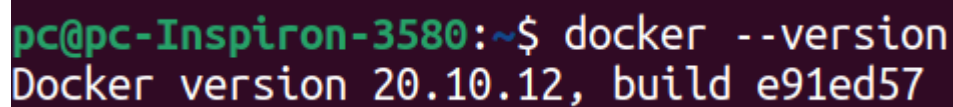
Figure -24 Création de version

Conteneurisé l'environnement de développement

Actuellement, le développement d'applications ne repose pas uniquement sur l'écriture de code. Plusieurs langues, architectures et environnements rendent le flux de travail de développement des applications plus complexe et difficile à la maintenir et à la mettre à jour d'une manière efficace. Donc, la raison pour

laquelle j'ai choisi de conteneuriser l'environnement de développement afin d'éviter toutes les anomalies liées à cet environnement.

- installer Docker Engine

A terminal window with a dark background. The prompt is 'pc@pc-Inspiron-3580:~\$' in green. The command 'docker --version' is entered in white. The output 'Docker version 20.10.12, build e91ed57' is displayed in white.

```
pc@pc-Inspiron-3580:~$ docker --version
Docker version 20.10.12, build e91ed57
```

Figure -25 vérifier la version de docker

- Créer un Dockerfile

Dockerfile est un fichier texte qui contient toutes les commandes qu'un utilisateur peut l'appeler via la ligne de commande pour construire une image. Docker peut créer des images suite aux commandes Dockerfile. Ci-après un extrait de ce fichier, le fichier complet se trouve dans mon dépôt GitHub.

```

    Cette commande FROM crée un calque à partir de l'image Docker
    ubuntu:18.04.
FROM ubuntu:18.04
# La commande RUN, est nécessaires pour téléchargé et installés la
package à l'aide d'apt.
RUN apt-get update && apt-get install -y curl git unzip xz-utils zip
libglu1-mesa openjdk-8-jdk wget
# Ajoutez un nouveau utilisateur nommé developer, créer un espace de
travail pour cet utilisateur dans /home/developer
RUN useradd -ms /bin/bash developer
USER developer
WORKDIR /home/developer
#Créer un dossiers ou sera installé SDK Android
#Définir la variable d'environnement ANDROID_SDK_ROOT sur le chemin
de répertoire correct, qui sera utilisé par Flutter.
RUN mkdir -p Android/sdk
ENV ANDROID_SDK_ROOT /home/developer/Android/sdk
RUN mkdir -p .android && touch .android/repositories.cfg

#Téléchargez et décompresse et déplace SDK dans le bon dossier.
#Ensuite utilise sdkmanager pour accepter les licences Android et
#télécharger les packages qui seront utilisés lors du développement
de l'application. Enfin, j'ai ajouté le chemin vers adb.
RUN wget -O sdk-tools.zip
https://dl.google.com/android/repository/sdk-tools-linux-4333796.zip
RUN unzip sdk-tools.zip && rm sdk-tools.zip
RUN mv tools Android/sdk/tools
RUN cd Android/sdk/tools/bin && yes | ./sdkmanager --licenses

```

Figure -26 Extrait du fichier Dockerfile.

Une fois la construction terminée, j'exécute le conteneur Docker dans le terminal et je push mon image on DockerHub.

```

pc@pc-Inspiron-3580:~$ docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
tik_image     v-0.1    5b0f67f80c5f   41 hours ago   2.65GB
ubuntu        18.04    dcf4d4bef137   4 weeks ago    63.2MB
hello-world   latest   feb5d9fea6a5   5 months ago   13.3kB
pc@pc-Inspiron-3580:~$ docker tag tik_image:v-0.1 d2021/tik:v-2
pc@pc-Inspiron-3580:~$ docker push d2021/tik:v-2
The push refers to repository [docker.io/d2021/tik]
9b54e86d1dac: Pushed
0f194ac7c1ae: Pushed
5ca5bbc92e4e: Pushed
f642e3a9c24c: Pushed
42c262c8b96d: Pushed
2126ab17fa64: Pushed
3e8903595710: Pushed
96fc8593c561: Pushed
832a4e4292c2: Pushed
b2b6257dec33: Pushed
5e47633f00e1: Pushed
1dc52a6b4de8: Pushed
v-2: digest: sha256:9a9151639903eab6e753b46aac2e4e2ad9824a289e9340f29bf453397090f811 size: 2852

```

Figure -27 Push l'image Docker via la ligne de commande.

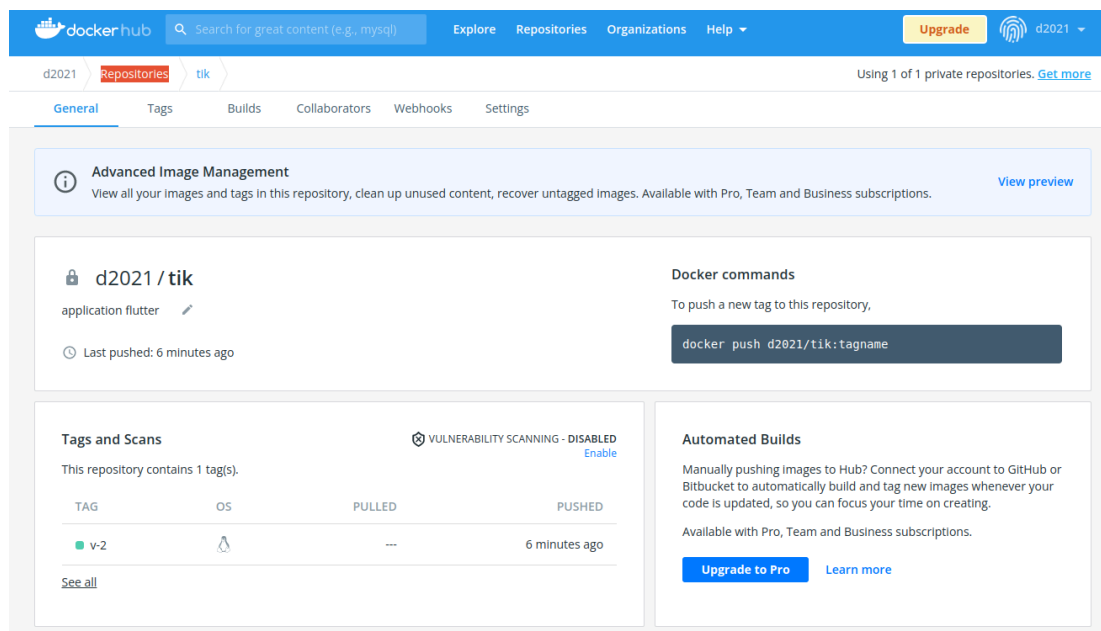


Figure -28 l'image sur DockerHub.

GUI

En informatique, une interface graphique (en anglais GUI pour graphical user interface) ou un environnement graphique est un dispositif de dialogue homme-machine, dans lequel les objets à manipuler sont dessinés sous forme de pictogrammes à l'écran, de sorte que l'utilisateur peut utiliser en imitant la manipulation physique de ces objets avec un dispositif de pointage, le plus souvent une souris ou avec le doigt.

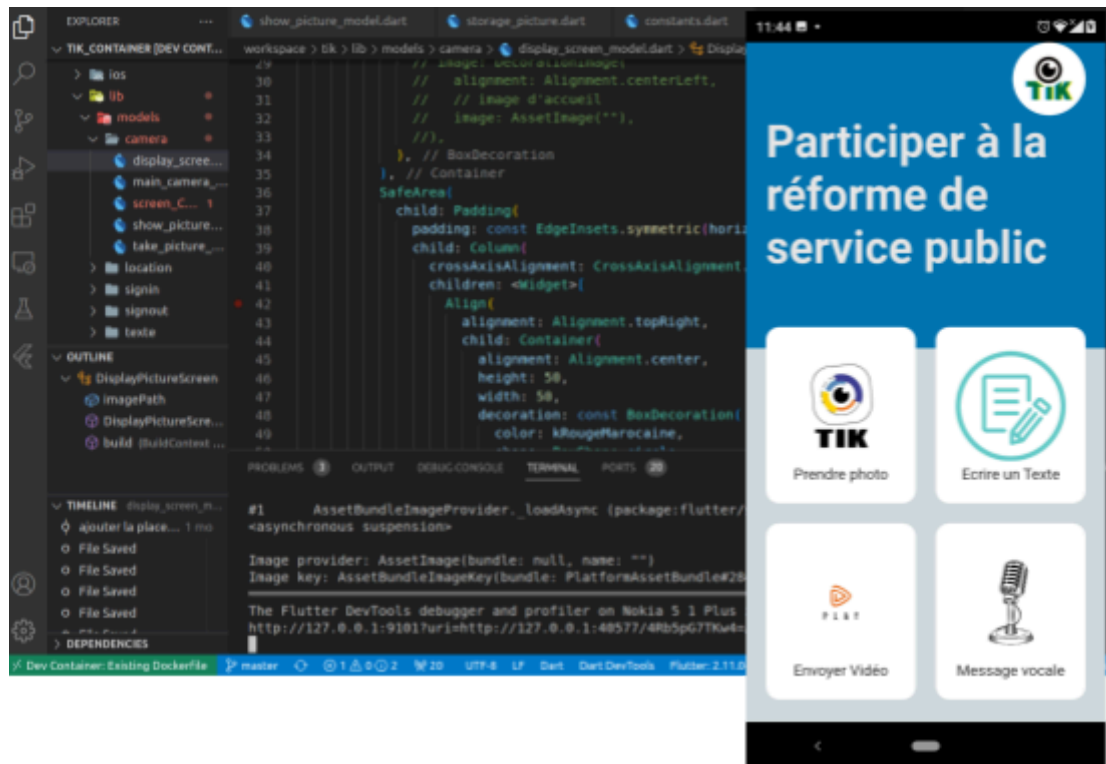


Figure -29 l'exécution de l'application Tik

Logo



Figure -30 Logo de l'application Tik

GUI client

Interface client, contient les types de réclamations

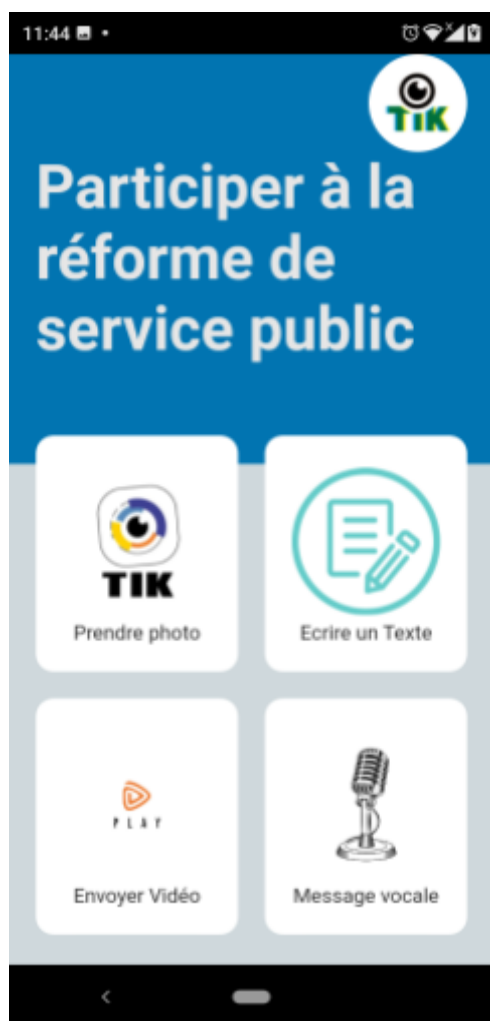


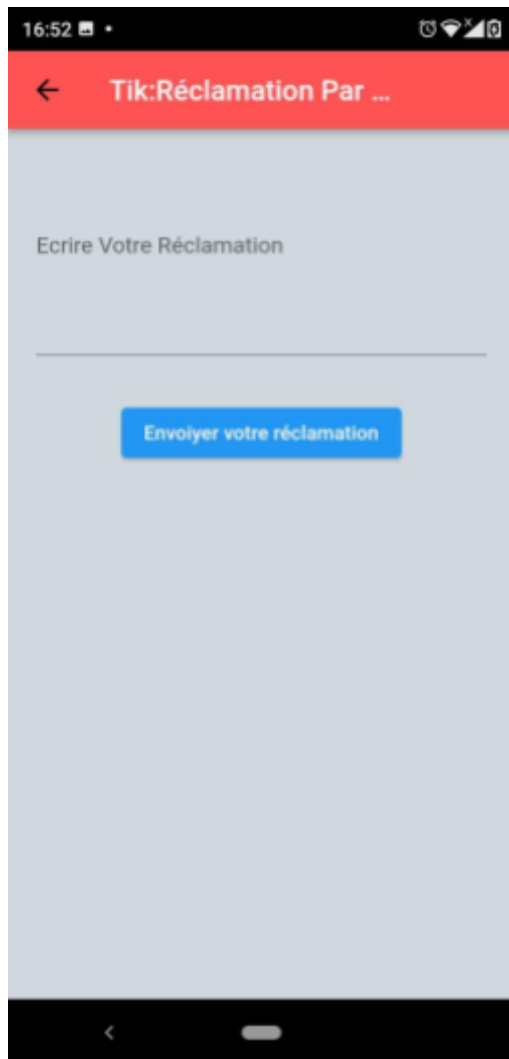
Figure -31 Page d'accueil réclamations

Ecran prendre une photo de l'anomalie



Figure -32 Interface prendre une photo.

Ecrire une réclamation



The screenshot shows a mobile application interface for writing a complaint. At the top, there is a black status bar with the time 16:52 and various icons. Below it is a red header bar with a back arrow and the text "Tik:Réclamation Par ...". The main content area is light gray and contains the text "Ecrire Votre Réclamation" followed by a horizontal line. Below the line is a blue button with the text "Envoyer votre réclamation". At the bottom, there is a black navigation bar with a back arrow and a home button.

Figure -33 Interface Ecrire une réclamation.

Adresser la réclamation au service concerné, avec le message de confirmation.

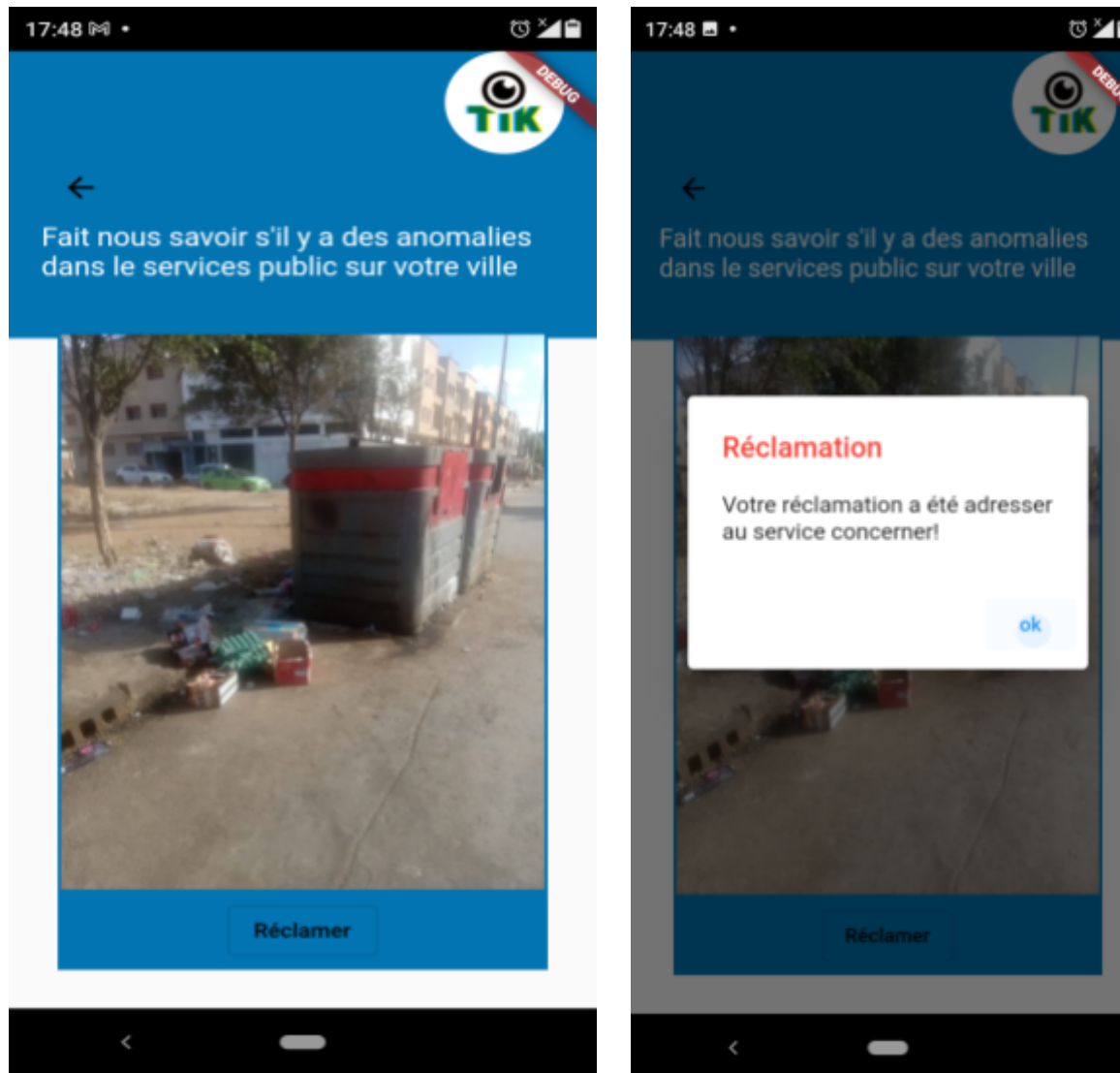


Figure -34 Interface Envoyer une réclamation.

GUI Administrateur

Exemple pour accéder à l'interface administrateur:

Login : brahim@gmail.com

Password : 123456

Interface web pour gérer les réclamations client

The screenshot displays a web application for managing client complaints. The top navigation bar includes links for 'HOME', 'AUTHENTIFICATION', and 'INSCRIPTION'. The main header features a large image of a city skyline with the text 'Participer à la réforme de service public' and 'Tâche une application de réclamation par photo'. Below this, a 'Connexion' section contains a login form with two input fields (username and password) and a 'SE CONNECTER' button. The bottom section, titled 'la meilleur façon de réclamer. Plusieurs méthodes de réclamation, par photo, par texte, par vidéo, par voix.', lists three services: 'Poubelles' (Service de Déchets), 'Fosse septique' (Service Fosse septique), and 'Éclairage public' (Service Éclairage public). Each service has a brief description of its function.

Participer à la réforme de service public
Tâche une application de réclamation par photo

Connexion

la meilleur façon de réclamer.
Plusieurs méthodes de réclamation, par photo, par texte, par vidéo, par voix.

Poubelles
Service de Déchets
Un déchet est un objet en fin de vie ou une substance ayant subi une altération physique ou chimique qui ne présente plus d'utilité ou est destinée à l'élimination. Le tout-venant de l'ancien français désigne un déchet, soit à la poubelle, soit dans un conteneur à un produit, ce qui est remis après son utilisation.

Fosse septique
Service Fosse septique
La fosse septique est l'un des éléments constitutifs d'une installation d'assainissement non collectif. Elle reçoit temporairement les eaux usées. Avant d'être envoyées dans les eaux souterraines et des eaux traitées, en la présence d'une épuration de fosse toutes eaux.

Éclairage public
Service Éclairage public
L'éclairage public est l'ensemble des moyens d'éclairage mis en œuvre dans les espaces publics, à l'exception de la lumière des vitrines, des enseignes et des bornes des véhicules et autres, destinées à la sécurité ou au confort des usagers.

Figure -35 Interface Administrateur.

Gestion des réclamations avec l'interface administrateur

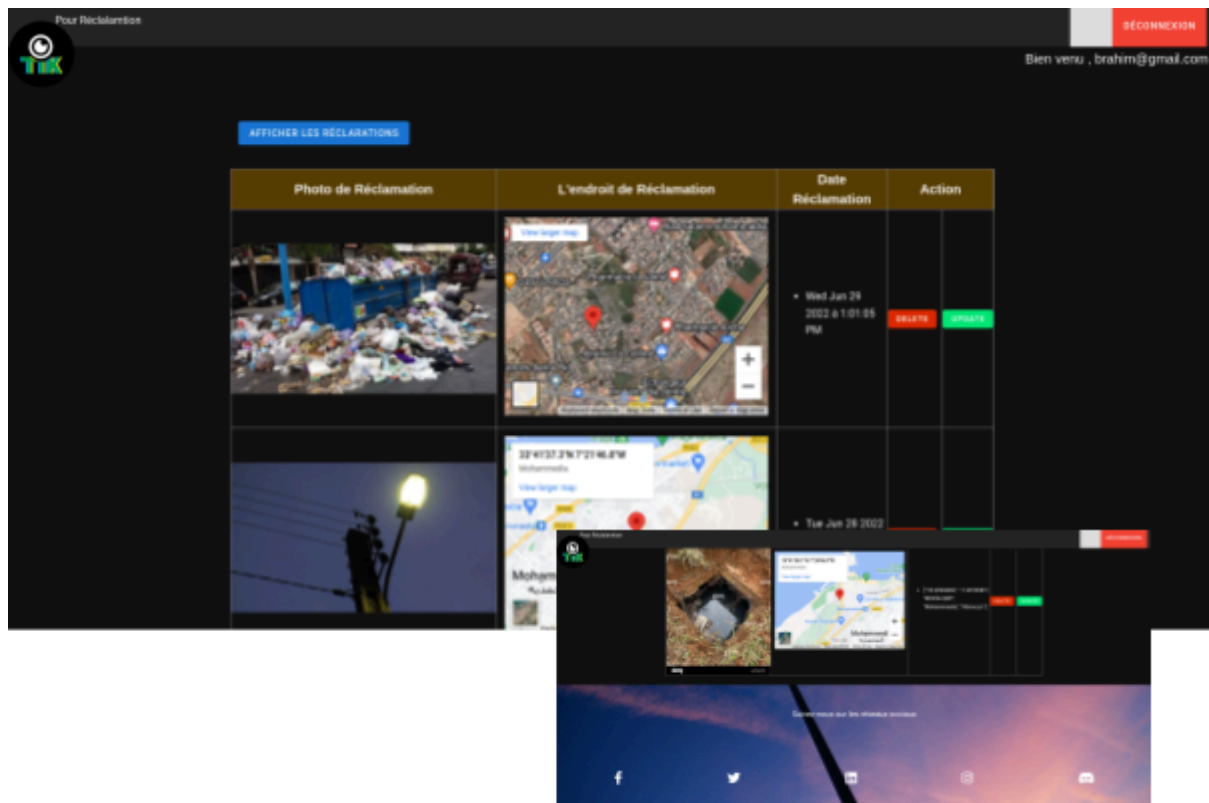


Figure -36 Interface Gestion réclamations.

Interface administrateur de l'application web



Figure -37 Interface de l'application.

Conclusion

Ce projet de réalisation et de conception d'une application mobile multiplateforme dédiés aux citoyens afin de réclamer le dysfonctionnement de service public, pour cette raison l'application est développée avec une interface conviviale et ergonomique , afin de permettre au citoyens d'adresser ces réclamations d'une manière automatique au service concerné.

Du point de vue technique, j'ai utilisé dans le développement de cette application la technologie de géolocalisation avec le framework flutter et nuxtjs comme technologie de front-end ainsi que firebase comme environnement de back-end.

Prespectives :

Ce travail peut être amélioré en y ajoutant d'autres fonctionnalités et l'enrichir pour plus d'efficacité.