

Documentation du projet Billeterie

Sommaire

- [Prérequis](#)
- [Lancer le serveur web](#)
- [API du site web](#)
- [Description du site web](#)
- [Application mobile](#)

Prérequis

```
Django==2.0.4
Pillow==5.1.0
social-auth-backend-epita==0.0.4
social-auth-app-django==2.1.0
python-social-auth[django]
reportlab==3.4.0
paypal==1.2.5
django-paypal==0.5.0
django-tastypie
```

Pour les installer via pip, lancez depuis la racine du projet:

```
pip install -r requirements.txt
```

Pour isoler son projet du reste de sa machine, il est possible d'utiliser un environnement virtuel.

```
python3 -m venv ./venv/
```

Puis activer ou désactiver cet environnement via:

```
source ./venv/bin/activate
deactivate
```

Ainsi, pour résumer, une installation des dépendances dans un environnement virtuel donnerait:

```
python3 -m venv ./venv/
source ./venv/bin/activate
pip install -r requirements.txt
deactivate
```

Comment mettre en place le serveur

Toutes ces commandes sont à exécuter dans le venv si c'est dans ce dernier que vous avez installé les dépendances.

Executer les commandes suivante:

```
cd src  
python3 manage.py runserver
```

Si vous souhaitez rajouter un compte administrateur:

```
cd src  
python3 manage.py createsuperuser
```

Si vous souhaitez rajouter des associations et / ou utilisateurs via un document csv:

```
cd src  
python3 manage.py shell  
from util import apps  
apps.load_csv("../misc/asso.csv")
```

Paramétrage des API externes

OPENID Google connection

Pour parametrer votre système de connection avec l'api google et ainsi pour accèder à diverses données statistiques (si besoin), il vous faut créer un compte spécifique.

Avec votre compte google connectez vous à : <https://console.developers.google.com>

Cherchez ensuite dans le catalogue d'application : *IAM Service Account Credentials API* Qui est le service d'authentification.

Créez ensuite un nouveau projet que vous pouvez intituler comme vous voulez. Vous avez désormais accès à différentes valeurs et token qui vont vous servir à configurer le *settings.py*

copiez la **Client Key** et remplacer SOCIAL_AUTH_GOOGLE_OAUTH2_KEY dans settings.py

copiez la **secret key** et remplacer SOCIAL_AUTH_GOOGLE_OAUTH2_SECRET

OPENID CRI connection

Tout comme pour la connection google il vous faudra remplacer quelques valeurs. Renseignez vous auprès du CRI pour obtenir les tokens de connections au service OPENID CRI.

Munis de ces valeurs il vous faudra remplacer dans settings.py SOCIAL_AUTH_EPITA_KEY et SOCIAL_AUTH_EPITA_SECRET

Pour les 2 services de connections:

Il vous faudra renseigner les addresses ip et nom de domaines de votre site web pour que ces api autorisent les connections vers votre nom de domaine.

Serveur Mail

Normalement votre hébergement web intègre un service d'envoie de mail par le système SMTP
Renseignez vous auprès de ce dernier pour obtenir les tokens d'identifications et remplacez les valeurs suivantes dans settings.py: EMAIL_HOST EMAIL_PORT EMAIL_HOST_USER EMAIL_HOST_PASSWORD

Paypal

Dans settings.py modifier le PAYPAL_TEST en **false**

Ensuite lors de la création d'évènement veillez à bien entrer des emails ayant des compte paypal vendeur

Description du site web

Ce que vous retrouverez sur toutes les pages du site web

- Une barre de recherche pour chercher une association ou un événement
- Un bouton connexion / deconnexion vous permettent de vous connecter ou deconnecter
- Des liens vers les différentes pages du site web

Page d'accueil

- Un carrousel mettent en avant les événements premiums
- La liste des évenements des 30 prochain jours

Page évènements

- La liste des événements n'étant pas encore passés

Page Mes évènements

- Cette page n'est disponible uniquement si vous êtes identifié
- La liste de tout les évènements auxquels vous êtes inscrit

Création des événements

- Vous pouvez créer un événement si vous avez les droits sur la page Mes Evénement

Page Modification des événements

- Vous pouvez modifier un événements si vous en avez les droits en allant sur ca descriptpion

Associations

- La liste de toutes les associations d'EPITA

Mes Associations

- Cette page n'est disponible uniquement si vous êtes identifié
- La liste de toutes les associations auxquelles vous êtes inscrit

Mon compte

- Cette page contient une photo si votre compte est associé à une photo
- Vos informations personnelles, étant partiellement modifiables

Statistique

- Cette page est accessible sur la page mon compte si vous en avez les droits
- Les statistique des evenements par associations sont affiché
- Vous pouvez télécharger les statistiques en format CSV

API du site web

Pour faire communiquer l'application mobile et le site web une API REST a été mise en place.

Comme le site a été réalisé avec le framework Django nous sommes restés dans ce cadre. L'API utilise la bibliothèque `tastypie`.

Pour voir les différentes requêtes disponibles, rentrez simplement l'URL de base de l'API.

```
host:port/api/v1/
```

```

▼ event_members:
  list_endpoint:      "/api/v1/event_members/"
  schema:            "/api/v1/event_members/schema/"

▼ events:
  list_endpoint:      "/api/v1/events/"
  schema:            "/api/v1/events/schema/"

▼ members:
  list_endpoint:      "/api/v1/members/"
  schema:            "/api/v1/members/schema/"

▼ users:
  list_endpoint:      "/api/v1/users/"
  schema:            "/api/v1/users/schema/"
```

/ request

Ensuite les différentes requêtes disponibles sont:

- /event_members/ pour obtenir la liste de tous les profils participant à un événement
- /events/ pour obtenir la liste de tous les événements
- /members/ pour obtenir la liste de tous les membres d'association
- /users/ pour obtenir la liste de tous les myUsers

```
▼ objects:  
  ▼ 0:  
    date_entry: "2018-06-08T17:31:50.349852"  
    email: "nipica64@gmail.com"  
    firstname: "Nicolas"  
    id: 7  
    lastname: "Ribeyrolle"  
    resource_uri: "/api/v1/event_members/7/"  
    ticket_number: "test"  
    token: "tokenstaff"  
  ▼ 1:  
    date_entry: "2018-06-08T17:33:02.423673"  
    email: "nipica64@gmail.com"  
    firstname: "Nicolas"  
    id: 8  
    lastname: "Ribeyrolle"  
    resource_uri: "/api/v1/event_members/8/"  
    ticket_number: "test"  
    token: "valid"
```

event_members request

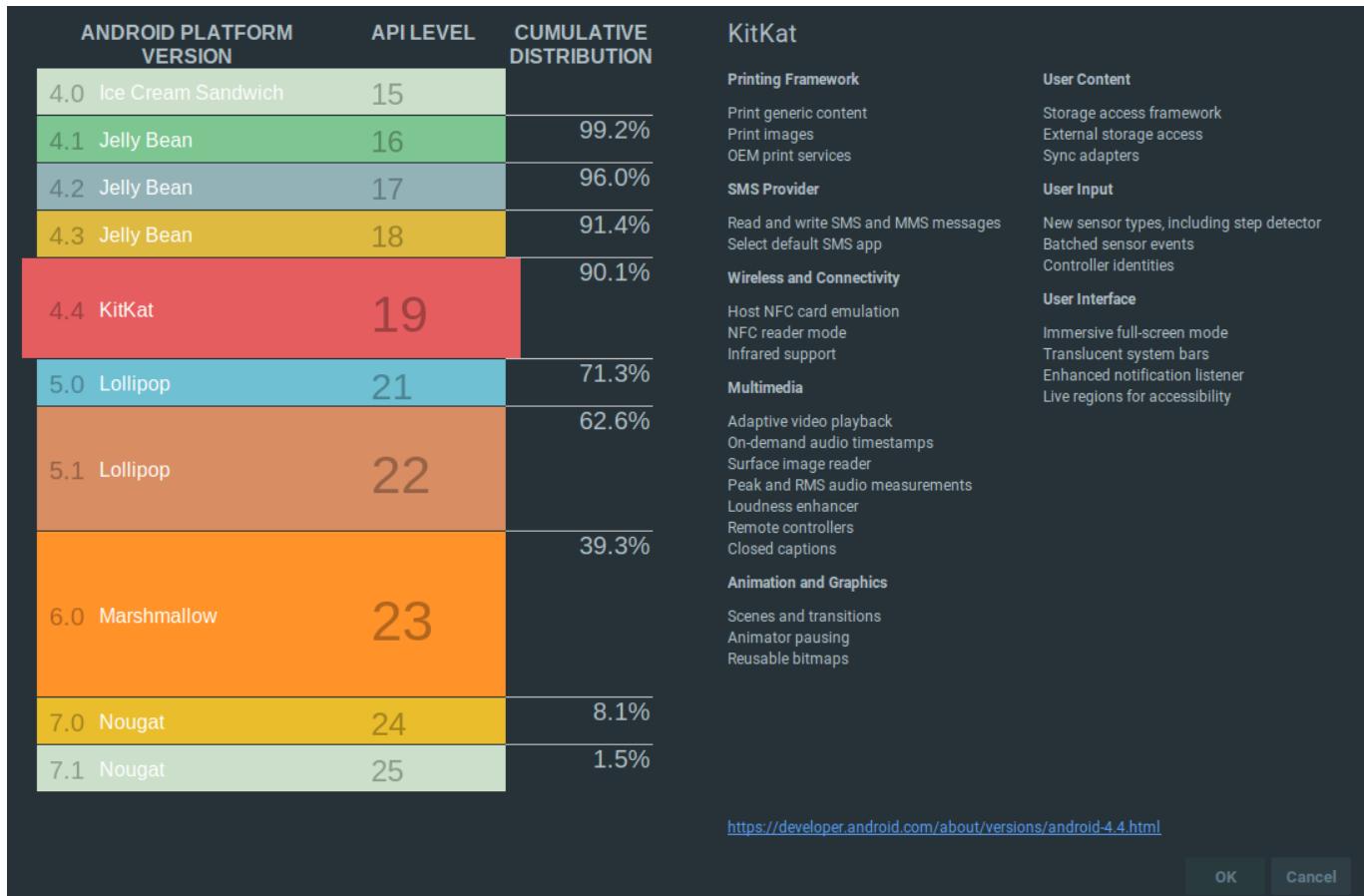
L'application mobile

Version minimale d'Android

L'application Android est compatible avec tous les smartphones sous Android version 4.4 ou supérieure. Cela couvre actuellement 90,1% des smartphones Android dans le monde actuellement.

Cette version minimale s'est imposée lors du choix de la bibliothèque apportant un scanner de QRCode. Celle-ci, ZXing, nécessite l'API 19 et donc Android 4.4 au minimum.

Les smartphones Android sortant de nos jours, même très bas de gamme, sont généralement sous Android 6.0 voire 5.0 dans le pire des cas. Ceci ne sera donc pas un problème lorsqu'il s'agira d'équiper le staff d'un smartphone pour scanner les billets.



Android KitKat 4.4 (API 19) est sorti en octobre 2014.

Installation

Un fichier APK non signé est fourni dans le rendu du projet.

Il est possible de générer soi-même ce fichier APK via Android Studio ou un terminal.

- Android Studio
 - Build > Build APK(s)
- Command line (nécessite gradle)
 - Dans ./android/
 - gradlew assembleDebug

Le fichier se trouvera dans ./android/app/build/outputs/apk/debug/app-debug.apk

Pour l'installer on peut soit ouvrir le fichier APK depuis son smartphone en ayant pris soin d'autoriser les sources inconnues dans Paramètres > Sécurité ou via un terminal encore une fois.

```
adb install /chemin/vers/fichier.apk
```

En ayant pris soin de télécharger les platform tools depuis le site de Google Developers pour avoir adb.

- [Générer son fichier APK via Android Studio](#)
- [Générer son fichier APK via le terminal](#)
- [Télécharger les platform tools de Google \(ADB\)](#)

Interface utilisateur

Écran d'accueil

Sur l'écran d'accueil, on peut entrer un token d'événement pour en sélectionner un.

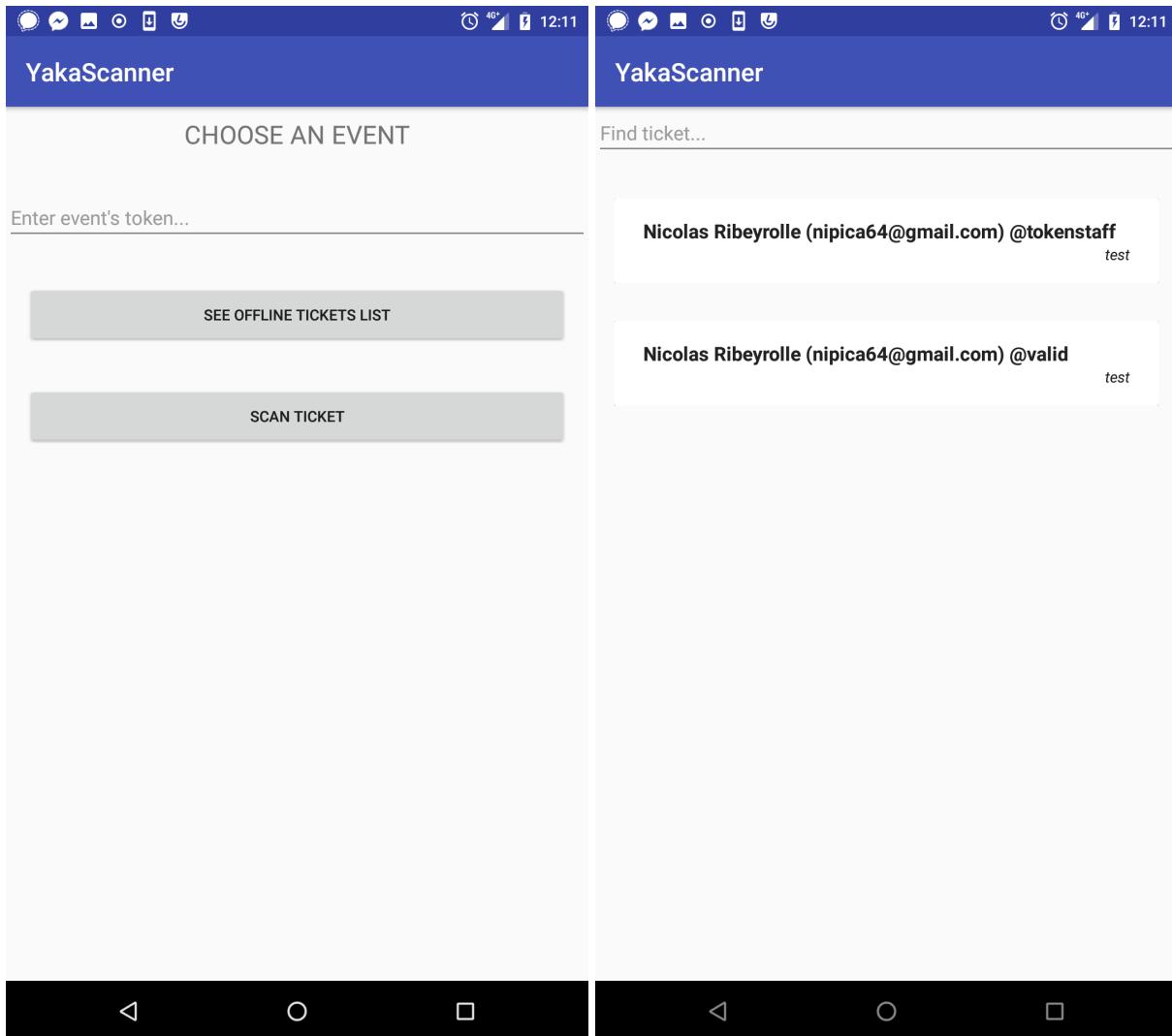
Si le champ reste vide, un message d'erreur est affiché pour des questions de sécurité. En effet, le token de l'événement est le mécanisme qui permet de s'assurer que l'utilisateur fait bien parti du staff.

Cela permet au staff de scanner les billets des personnes en cas de problème, si l'association a oublié de fournir le token par exemple.

Le bouton *See offline tickets list* permet d'accéder à la liste des billets. Cela permet de charger la liste lorsque l'on lance l'application pour la première fois, mais également de voir la liste des noms et prénoms en cas de problème de scan.

Le bouton *Scan ticket* permet de lancer le scanner de QRCode.

En cas de perte de connexion internet, la dernière liste chargée reste en mémoire. Cela permet de continuer à scanner même sans accès internet.

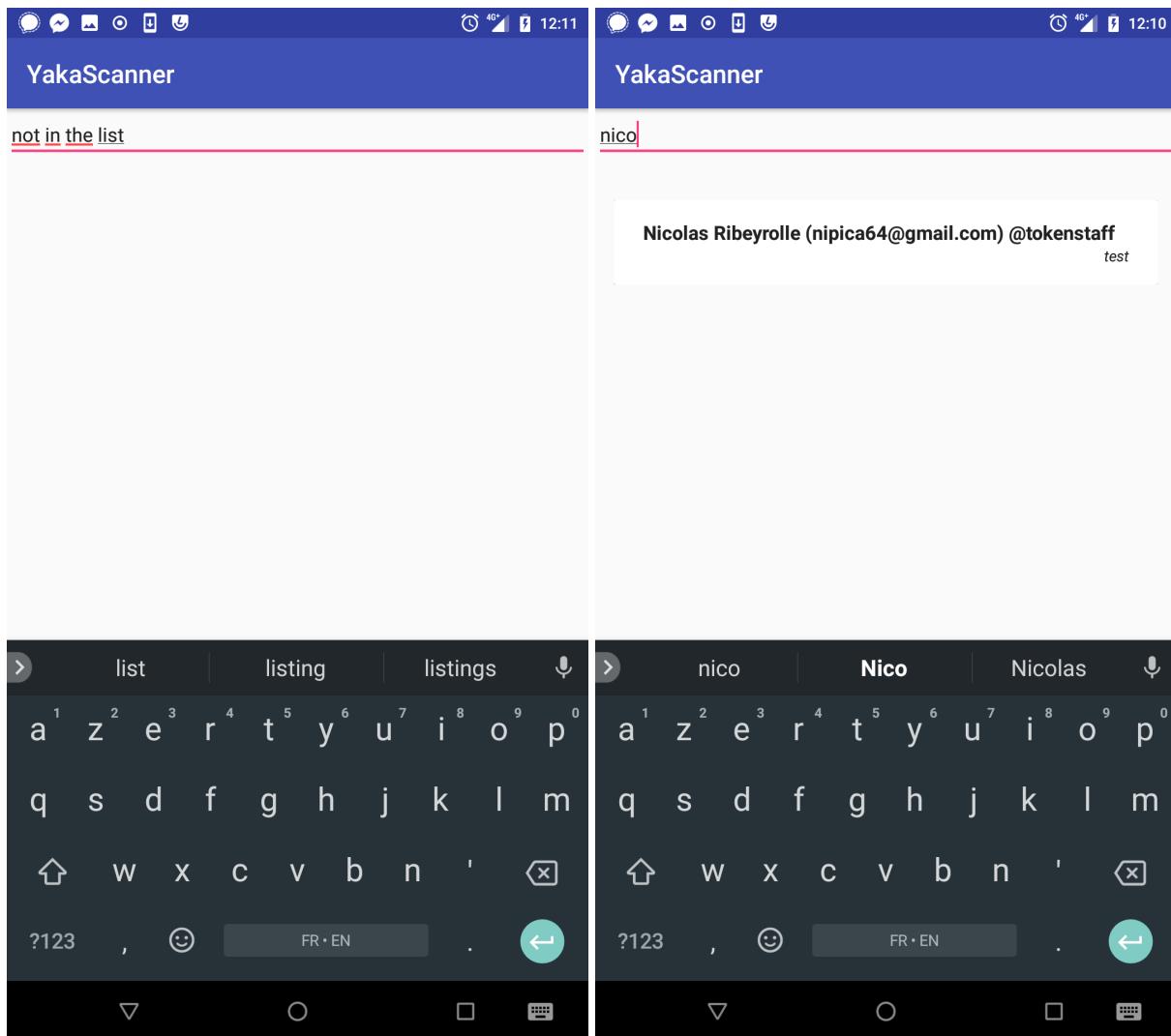


Liste des billets

La liste des billets montre les différentes informations sur tous les billets achetés par les clients. On retrouve le prénom, le nom, l'adresse mail, le token de l'événement ainsi que le numéro de ticket.

Il est possible de filtrer les résultats par nom ou prénom.

Ainsi, si Jean Martin affirme avoir acheté un ticket, on peut taper Jean ou Martin (ou n'importe quelle sous chaîne de caractères) pour retrouver son billet s'il existe.

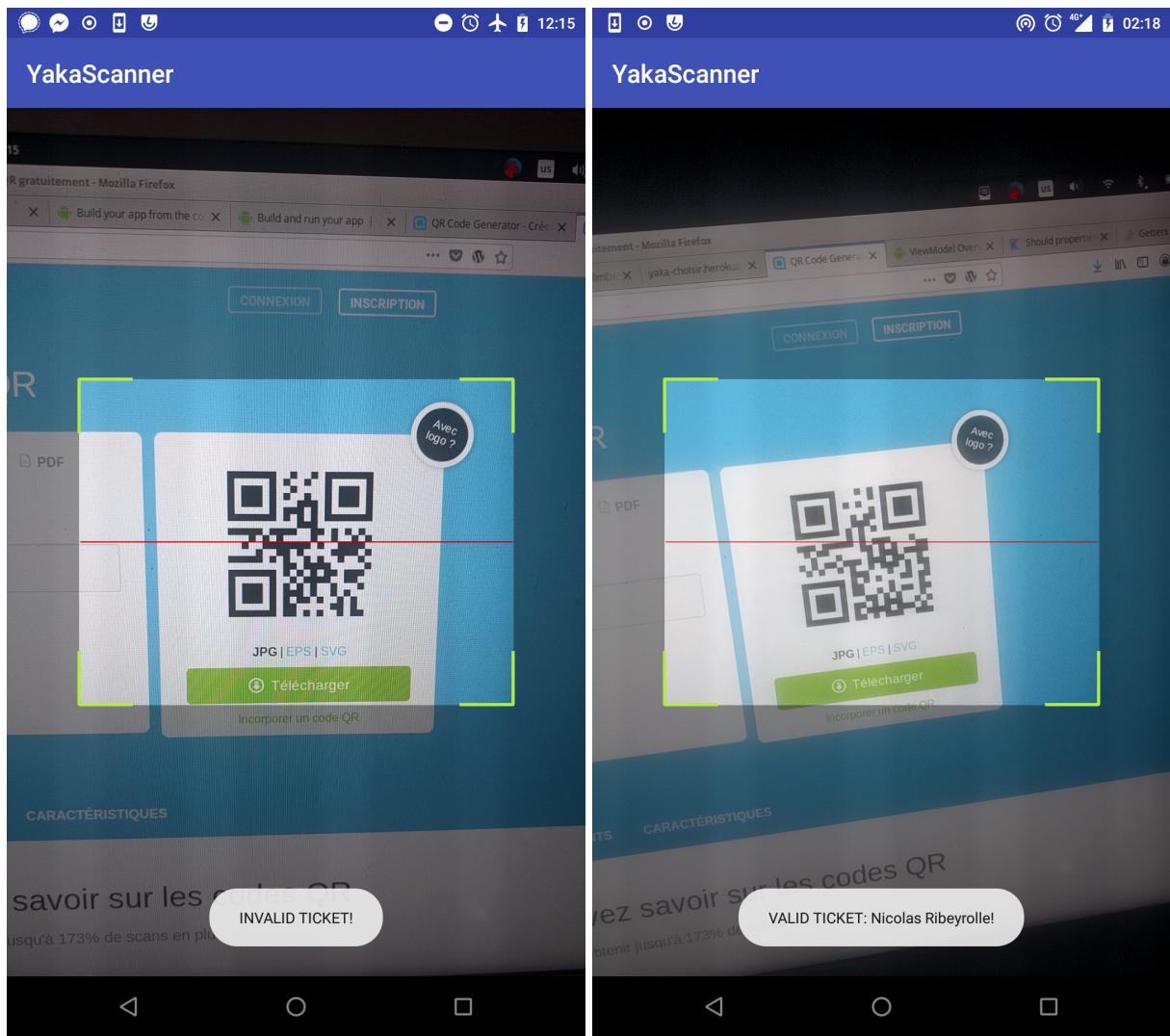


Le scanneur de billet

Pour scanner un billet électronique, il suffit de pointer la caméra du téléphone vers le QRCode présent sur les tickets.

Un message d'information précisera alors si le billet est valide ou non.

La liste étant stockée en mémoire en cas de perte de connexion internet, il est tout à fait possible de scanner ces billets sans connexion.



On navigue d'écran en écran via les boutons de la page d'accueil, et on revient à la page d'accueil grâce au bouton *retour* (le triangle) du smartphone Android.

Fonctionnalités manquantes

- Requête PUT lors du scan d'un billet électronique pour mettre à jour la base de données
- Double validation d'événements par le président et le responsable des associations
 - Pour le moment n'importe quel membre du bureau d'une association peut y créer un événement.