DOCUMENTATION TECHNIQUE - APPLICATION DE GESTION D'ÉVÉNEMENTS

ILBOUDO Jean-Claude

ING 2 GSI CY TECH

25/09/2025, Cergy

Lien Github:
https://github.com/misterjc1/Application_de_Gestion_Des_Evenements/tree/main

Plan

Introduction

Ce projet s'inscrit dans le cadre d'un test technique visant à développer une application complète de gestion d'événements.

Étant donné le délai imparti, toutes les fonctionnalités prévues n’ont pas pu être finalisées. Cependant, l’essentiel est en place et je prévois de continuer à améliorer le projet afin d’enrichir les fonctionnalités et d’optimiser la solution.

Objectifs

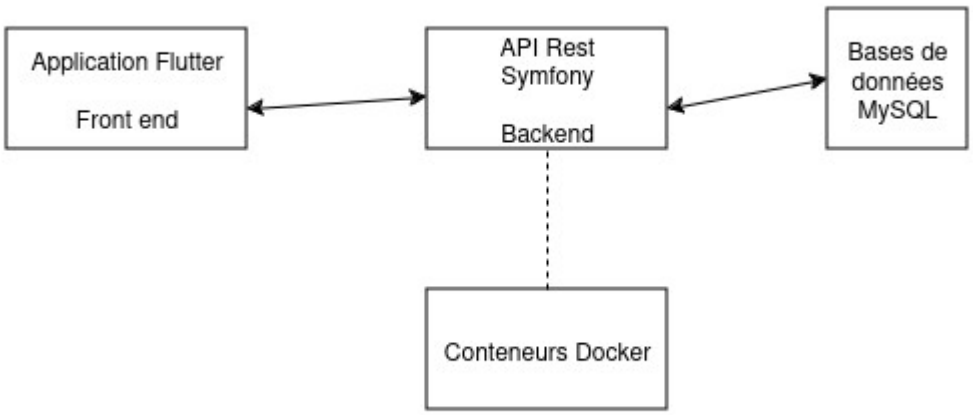
Développer une application mobile de gestion d'événements permettant :

- → Aux utilisateurs de découvrir et s'inscrire à des événements
- → Aux organisateurs de créer et gérer leurs événements

Technologies

- → Frontend : Flutter (Dart)
- → Backend : Symfony (PHP)
- → Base de données : MySQL
- → Containerisation : Docker & Docker Compose
- → Authentification : JWT (JSON Web Tokens)

Architecture



1. Utilisateur interagit avec l'app Flutter
2. Flutter envoie des requêtes HTTP à l'API Symfony
3. Symfony traite la logique métier et communique avec MySQL
4. MySQL stocke et retourne les données
5. Symfony renvoie la réponse à Flutter en JSON

Etape 1: Installation de Docker

Docker est une technologie qui permet de **créer et lancer des applications dans des "conteneurs"**.
Un conteneur est comme une **mini-machine virtuelle légère** qui contient tout ce qu'il faut pour exécuter ton application (système, bibliothèques, dépendances).

Pour Ubuntu/Linux

Mise à jour du système

sudo apt update && sudo apt upgrade -y

Installation des dépendances

sudo apt install apt-transport-https ca-certificates curl software-properties-common

Ajout du repository Docker

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg echo "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

Installation de Docker

sudo apt update sudo apt install docker-ce docker-ce-cli containerd.io

Vérification

docker --version

Ajouter l'utilisateur au groupe docker

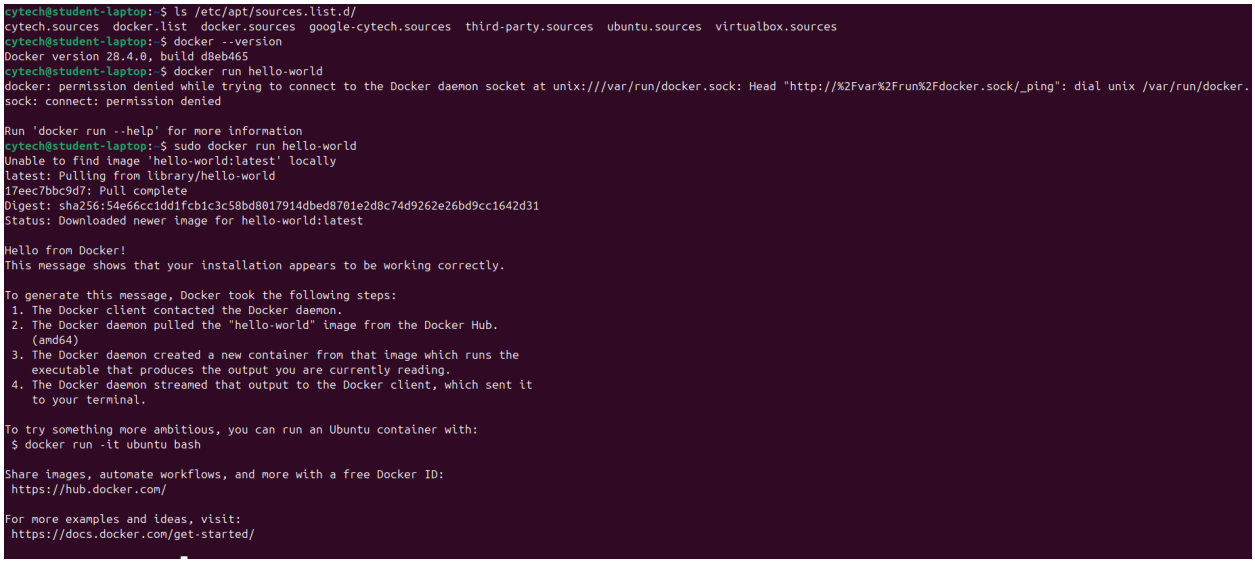
sudo usermod -aG docker $USER

Redémarrer la session

newgrp docker

Tester Docker

docker run hello-world



Étape 2: Installation de Docker Compose

Docker Compose est un outil qui permet de **lancer plusieurs conteneurs ensemble** à partir d'un fichier docker-compose.yml

Téléchargement de la dernière version

$sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

Rendre exécutable

$sudo chmod +x /usr/local/bin/docker-compose

Vérification

$docker-compose –version

Étape 3: Installation de Symfony

Installation de PHP et extensions

$sudo apt install php8.2 php8.2-common php8.2-cli php8.2-fpm php8.2-mysql php8.2-xml php8.2-curl php8.2-zip

Installation de Composer

$curl -sS https://getcomposer.org/installer | php sudo mv composer.phar /usr/local/bin/composer sudo chmod +x /usr/local/bin/composer

Vérification

$php --version composer --version

→ **Création du Projet Symfony**

Navigation vers le dossier de travail

$cd ~/Projects/Test_Technique_CY

Création du projet Symfony

$composer create-project symfony/skeleton backend

Accès au dossier du projet

$cd backend

Étape 4 : Configuration Docker avec docker-compose.yml

```
version: '3.8'
services:
  database:
    image: mysql:8.0
    environment:
      MYSQL_ROOT_PASSWORD: root
      MYSQL_DATABASE: event_db
      MYSQL_USER: user
      MYSQL_PASSWORD: password
    ports:
      - "3307:3306"
    volumes:
      - db_data:/var/lib/mysql
volumes:
  db_data:
```

→ **Lancement des Containers**
Démarrage des services en arrière-plan
$docker-compose up -d

Vérification du statut
$docker ps

Étape 5 : Configuration de la Base de Données

Création de la Structure de la Base

- Installation des dépendances Symfony

$composer require orm

- Crée la base de données définie dans le fichier .env

$php bin/console doctrine:database:create

- Créer une entité

$ php bin/console make:entity

- Génère un fichier de migration SQL à partir de tes entités PHP

$php bin/console make:migration

- Exécute réellement les migrations en base

$php bin/console doctrine:migrations:migrate

- Démarrer le projet Back end avec symfony

$symfony serve -d

Etape 6 : Flutter

- Installer Flutter

$sudo snap install flutter --classic

- Vérifie l'installation

$flutter --version

$flutter doctor -v

- Installer android Studio

$sudo snap install android-studio --classic

- Créer un projet Flutter

$cd ~/Projects/Test_Technique_CY

$flutter create frontend

$cd frontend

$flutter pub get # installe les dépendances (commande que tu as déjà utilisée dans la doc)

- Liste des devices disponibles

$flutter devices

- Liste des émulateurs crées

$flutter emulators

$flutter emulators --launch Pixel_6 (demarrer l'émulateur Pixel_6)

- Lancer un émulateur par son id :

  $flutter run -d emulator-5554

Fonctionnalités Implémentées

Module d'Authentification

- **Inscription** avec choix du type (Utilisateur/Organisateur)
- **Connexion** sécurisée avec JWT
- **Gestion des sessions** et tokens de rafraîchissement
- **Protection des routes** par rôles

Gestion des Événements

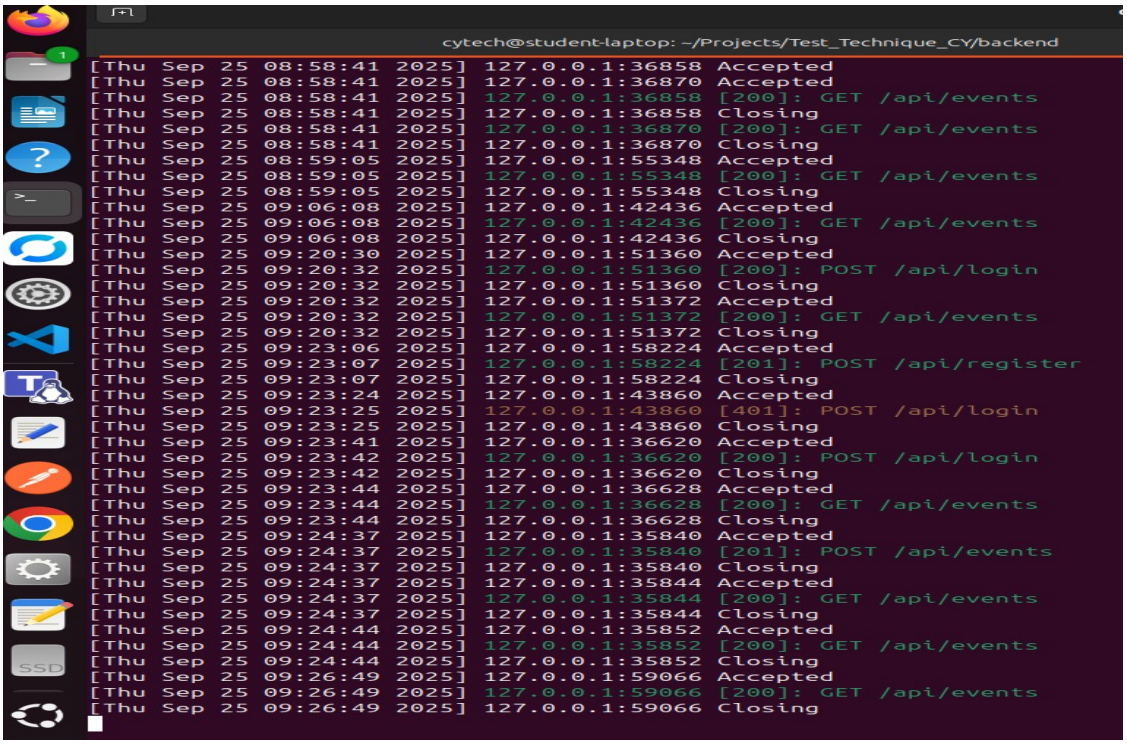
- **Création** (Organisateurs seulement)
- **Consultation** publique

- **Inscription** des participants
- **Gestion des capacités** (maxAttendees)
- **Système de publication/dépublidation**

API Endpoints Disponibles

Méthode	Endpoint	Description	Accès
POST	/api/register	Inscription	Public
POST	/api/login	Connexion	Public
GET	/api/events	Liste événements	Public
GET	/api/events/{id}	Détail événement	Public
POST	/api/events	Créer événement	Organisateur
PUT	/api/events/{id}	Modifier événement	Propriétaire
POST	/api/events/{id}/register	S'inscrire	Utilisateur

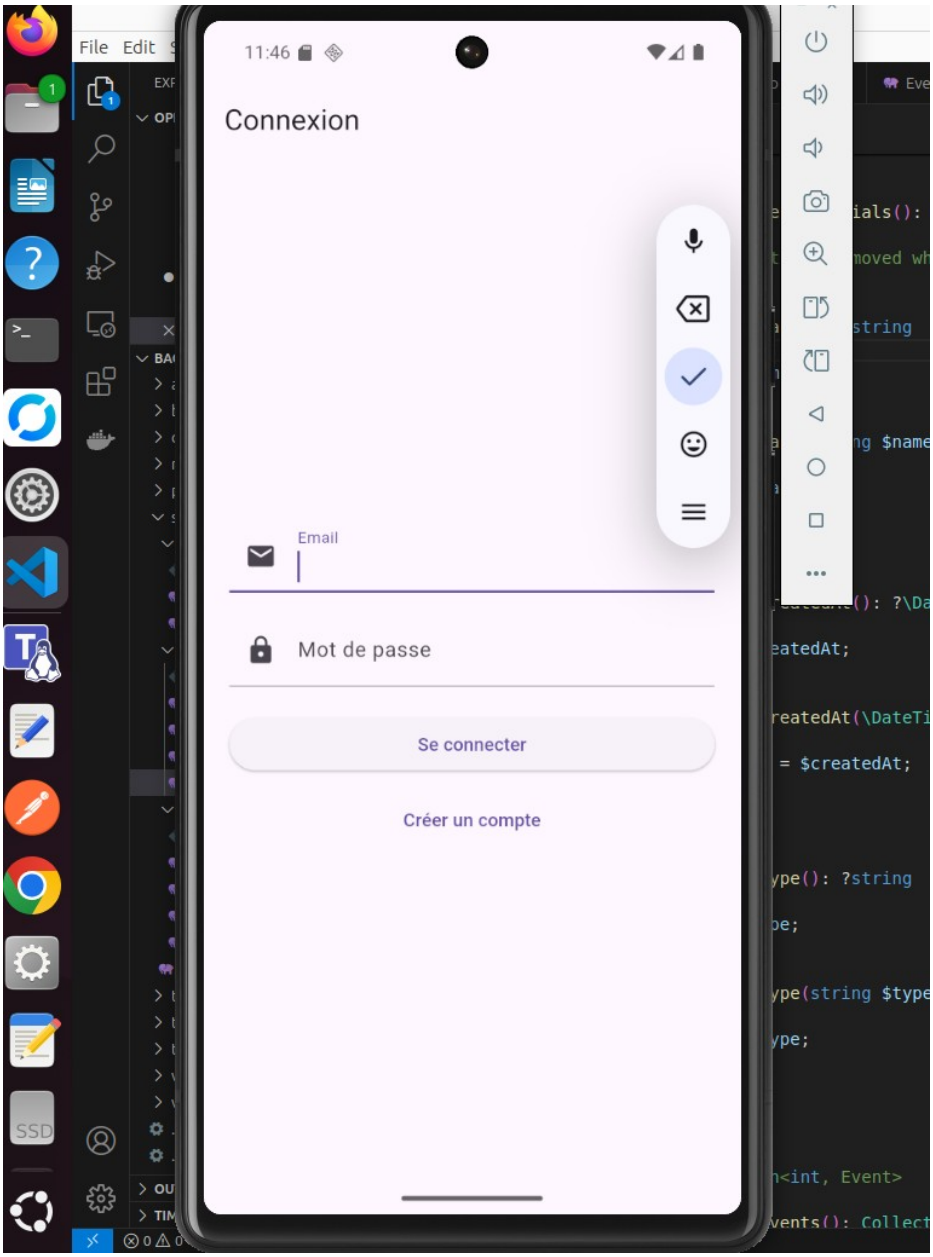
→ **Demarrer le Projet Backend**



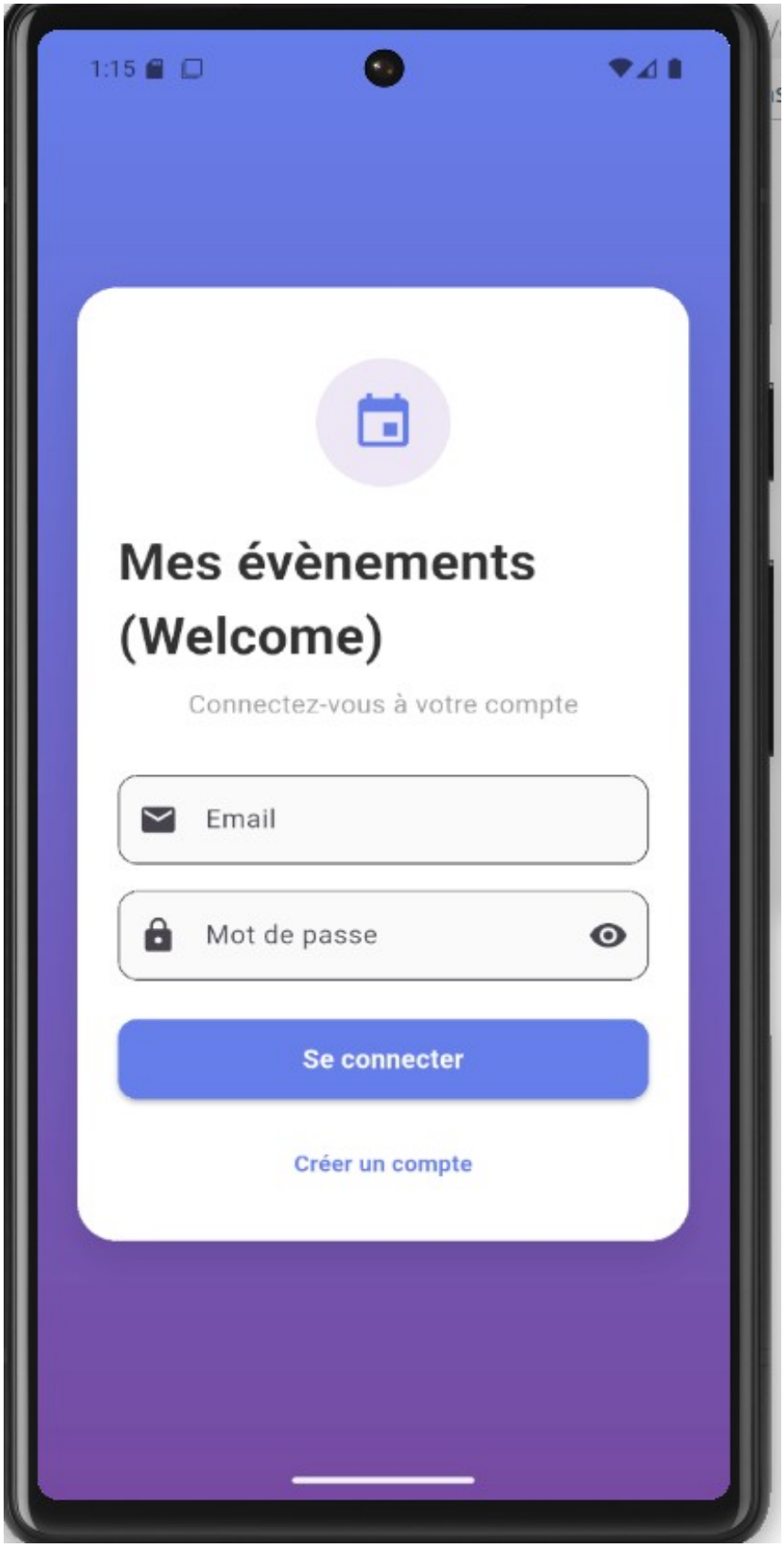
LANCEMENT DE L'APPLICATION MOBILE FLUTTER

flutter run

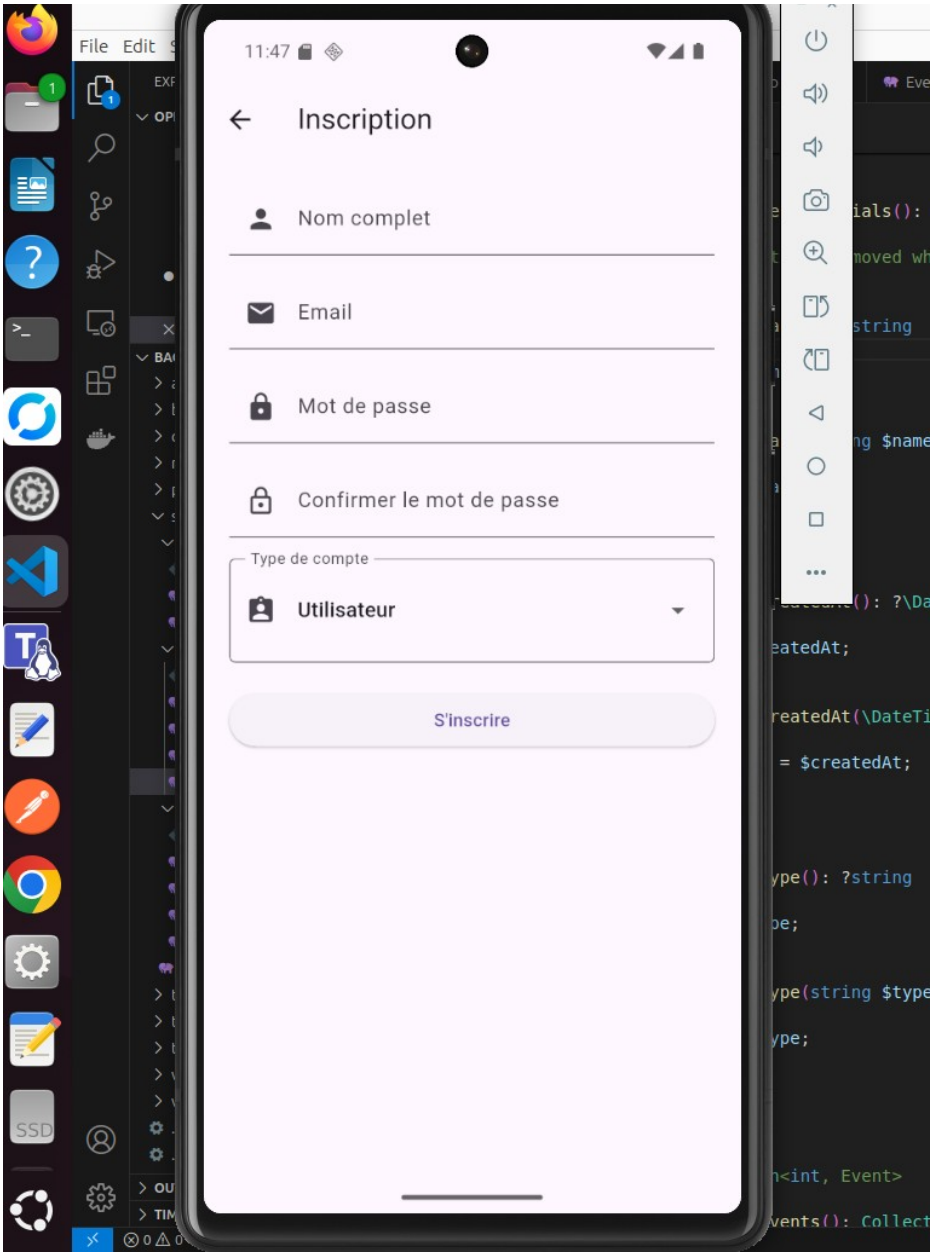
→ **ÉCRAN DE DÉMARAGE**



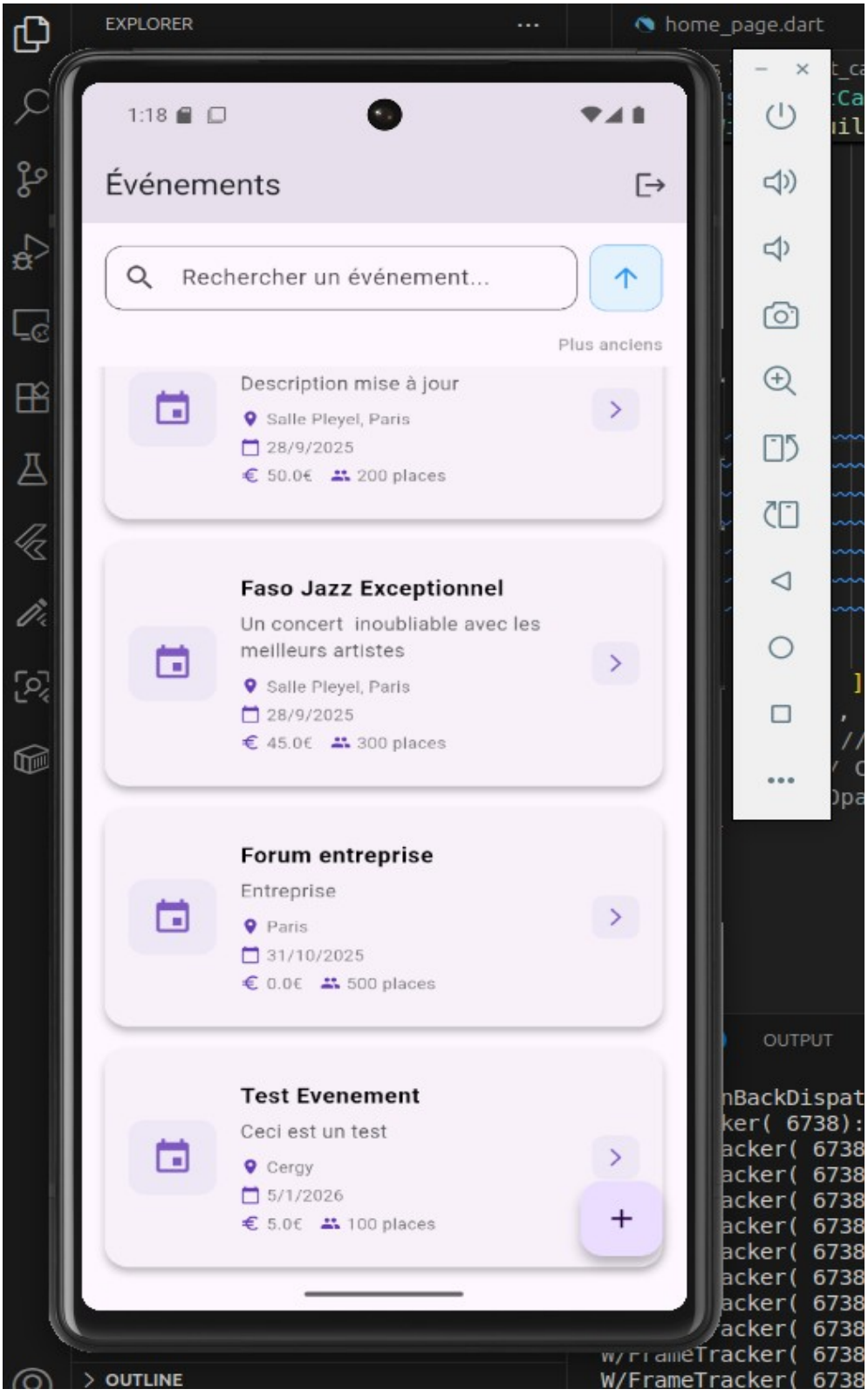
→ **ÉCRAN DE CONNEXION**



→ **ÉCRAN D'INSCRIPTION**

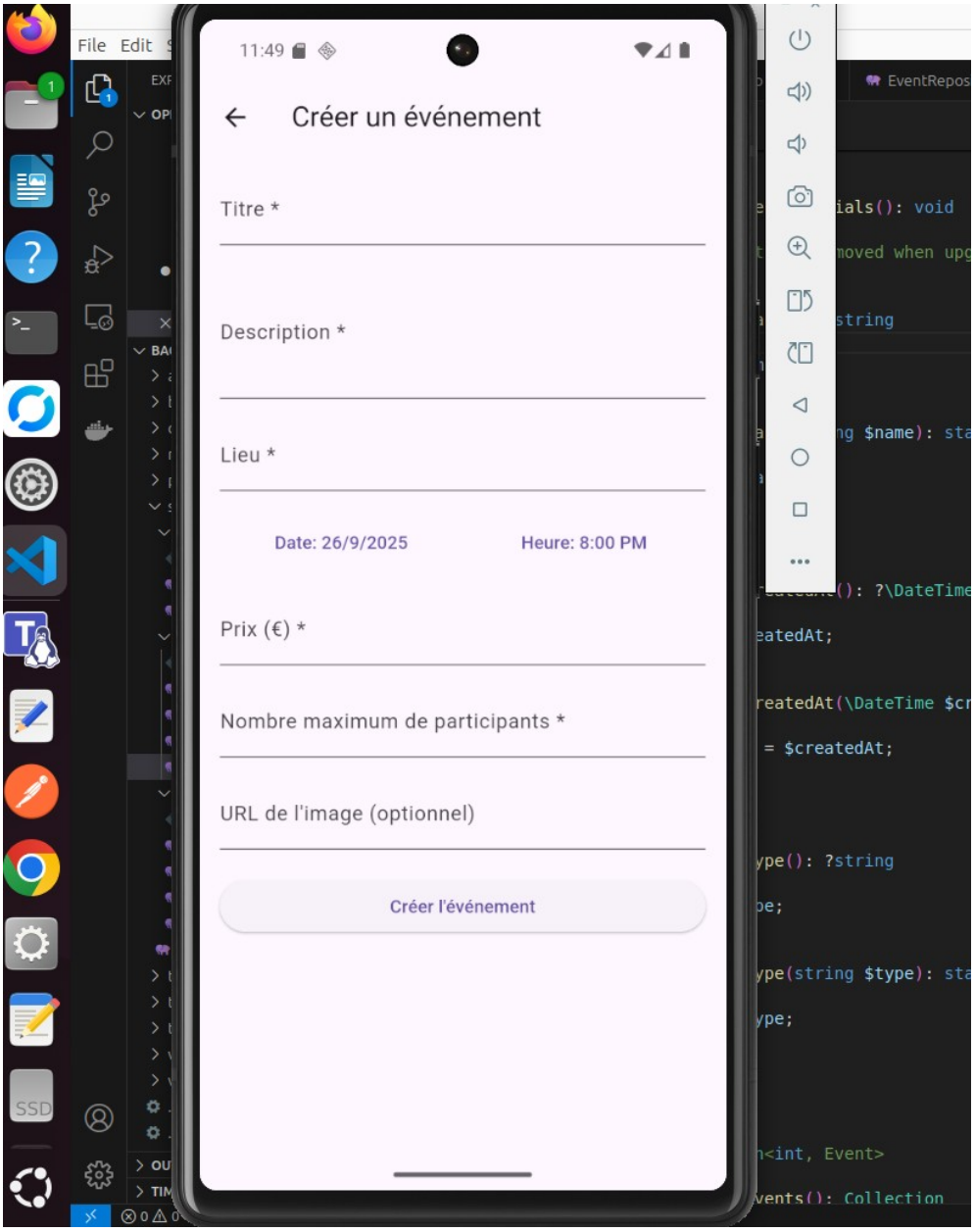


→ **DASHBOARD ORGANISATEUR**



Ici, je souligne que la croix matérialisée en rouge signifie que l'évenement n'existe plus.

→ **CRÉATION D'ÉVÉNEMENT (Organisateur)**



Conclusion

Cette documentation couvre l'ensemble du processus d'installation, de configuration et d'utilisation de l'application de gestion d'événements. L'architecture modulaire permet une maintenance aisée et des extensions futures.

Webographie

- https://docs.flutter.dev/get-started/install/linux/android#add-flutter-to-your-path
- https://symfony.com/download
- https://docs.docker.com/desktop/setup/install/linux/ubuntu/
- Postman
- PHP