Titre du Document

socsport Documentation

Choix sujet

"SocSport" est une application innovante visant ˆ faciliter la vie active et sociale. Elle permet aux utilisateurs dÛorganiser facilement des matchs sportifs avec amis, coll•gues et passionnŽs. En favorisant un mode de vie sain, renfor•ant les liens sociaux, facilitant lÛorganisation dÛŽvŽnements sportifs, crŽant des communautŽs et offrant des opportunitŽs commerciales, SocSport se positionne comme bien plus quÛune simple application, intŽgrant Žgalement des aspects technologiques modernes pour une expŽrience utilisateur optimale.

Concurrence

Des applications comme SportyHQ, OpenSports, Beballer et Strava peuvent •tre utilisŽes pour organiser des ŽvŽnements sportifs et rassembler des communautŽs autour du sport.

SocSport se distingue de la concurrence en offrant une expŽrience unique et conviviale. En effet, notre application permet une rŽservation gratuite de terrains avec lÒajout facile dÒŽvŽnements sportifs personnalisŽs. Gr‰ce ˆ une technologie de localisation avancŽe, les utilisateurs peuvent trouver rapidement les terrains les plus proches. LÒinterface intuitive, les fonctionnalitŽs sociales avancŽes et les retours positifs des utilisateurs renforcent notre engagement envers une communautŽ sportive dynamique. Optez pour SocSport pour une expŽrience inŽgalŽe dans lÒorganisation dÒactivitŽs sportives.

Mod•les

Installation de SocSport

0. PrŽrequis

Assurez-vous dÕavoir Python installŽ sur votre machine avec une version compatible (Python 3.12 ou supŽrieur recommandŽ). Vous pouvez vŽrifier si Python est installŽ en exŽcutant `python --version` dans votre terminal. Dans le cas oÙ Python nÕest pas installŽ, vous pouvez le faire en suivant les instructions sur le site officiel de Python : Installation de Python.

1. Installation de Flask

Vous pouvez installer Flask en lan•ant votre terminal de commande en tant qu'administrateur puis en entrant cette ligne de commande :

```
pip install Flask
```

2. Installation de Node.js

Si vous n'avez pas Node.js installŽ sur votre syst•me, vous pouvez dans un premier temps le tŽlŽcharger sur le site officiel : Installation de Node.js.

3. Clonage du projet

Maintenant que Node.js et Flask sont installŽs, vous pouvez cloner le dŽp™t Github de SocSport sur votre machine locale en utilisant la commande suivante :

```
git clone https://github.com/mfrj22/socsport.git
```

4. Installation des librairies Python

Pour notre projet, nous avons utilisŽ diffŽrentes biblioth•ques Python :

¥ SQLAlchemy permet dÕintŽgrer une base de donnŽes SQL dans notre application Flask.

```
pip install Flask-SQLAlchemy
```

¥ PyMySQL permet dÕassurer la connectivitŽ et la communication avec notre base de donnŽes MySQL.

```
pip install pymysql
```

¥ Cryptography est utilisŽ pour renforcer la sŽcuritŽ.

```
pip install cryptography
```

¥ Geopy g•re les opŽrations de gŽolocalisation (calcul de distances entre des points gŽographiques).

```
pip install geopy
```

¥ PyTest permet dÕexŽcuter des tests unitaires.

```
pip install pytest
```

5. Installation des modules React

Pour le bon fonctionnement de notre application, nous avons ajoutŽ des modules React :

```
npm install
npm install react-router-dom
npm install react-slick slick-carousel
```

6. Lancement de l'Application

Vous êtes maintenant prêt à lancer l'application ! Utilisez la commande suivante pour démarrer le serveur de développement :

```
npm start
```

Pour lancer l'API Flask, il vous faudra lancer un terminal depuis la racine du projet puis lancer cette ligne de commande :

```
python app.py
```

7. Diagramme de classe

```
@startuml model

class Ville {
Ê + id: Integer
Ê + nom: String
Ê + code_postal: Integer
Ê + departement: String
}

class Terrain {
Ê + id: Integer
Ê + nom: String
Ê + adresse: String
Ê + latitude: Float
Ê + longitude: Float
Ê + ville_id: Integer
Ê + ville: Ville
}

class Evenement {
Ê + id: Integer
Ê + nom: String
Ê + date: Date
Ê + heure_debut: Time
Ê + heure_fin: Time
Ê + terrain_id: Integer
Ê + terrain: Terrain
}

Ville "1" -- "*" Terrain : has
Terrain "1" -- "*" Evenement : has

@enduml
```

8. Diagramme de sŽquence

```
@startuml diagsec

actor User
participant App
participant Vue
participant Controller
participant Terrain
participant Evenement
participant Database

User -> App: Send request to create event
App -> Vue: Handle user input
Vue -> Controller: Request to create event
Controller -> Terrain: Get terrain details
Terrain --> Controller: Terrain details
Controller -> Evenement: Create new event
Evenement --> Controller: New event created
Controller -> Database: Save event to database
Database --> Controller: Event saved
Controller --> Vue: Event created successfully
Vue --> App: Notify user about event creation

@enduml
```

9. Diagramme entitŽ/association

```
@startuml diagEA

entity "Ville" {
Ê + id: Integer
Ê --
Ê + nom: String
Ê + code_postal: Integer
Ê + departement: String
}

entity "Terrain" {
Ê + id: Integer
Ê --
Ê + nom: String
Ê + adresse: String
Ê + latitude: Float
Ê + longitude: Float
}

entity "Evenement" {
Ê + id: Integer
Ê --
Ê + nom: String
Ê + date: Date
Ê + heure_debut: Time
Ê + heure_fin: Time
}

Ville --o{ Terrain : "has"
Terrain --o{ Evenement : "has"

@enduml
```