

Projet Portes-ouvertes

Présentation du projet

L'école de commerce **ComSupEco** organise une journée **portes-ouvertes** une fois par an afin de faire découvrir l'école et ses enseignements.

Une fois les portes ouvertes réalisées, le secrétariat envoie à chaque visiteur un dossier d'inscription.

Pour cela le directeur veut développer une application exécutable sur *smartphone* capable de noter les informations sur les visiteurs le jour des portes-ouvertes.

Le directeur souhaite une première version de l'application permettant :

- aux professeurs de noter des informations sur les visiteurs qu'ils ont reçu,
- au directeur d'avoir la liste des visiteurs
- **PROJET A RENDRE AU PLUS TARD LE 17 décembre 2021**
- Le projet sera mémorisé sur un dépôt **Git**

Sprints

Le directeur souhaite dans un premier temps un prototype à base de [SQL Lite](#) pour l'enregistrement des données.

Base de données

Une [ébauche de classe](#) concernant l'accès à la base de données [SQL Lite](#) a été développé grâce à la classe [SQLiteOpenHelper](#) : la classe [VisiteurSQLLite.java](#)

En vous appuyant sur cette [ébauche](#) et les informations de la partie [Ressources](#) contenues dans cette page vous devez réaliser un **prototype opérationnel** sous le nom de projet **ProjetPortesOuvertes**.

1- Réalisez la page d'accueil

La page d'accueil est dotée de 2 **Boutons** :

- un bouton **Visiteur** : qui permettra la saisie et l'enregistrement des informations sur un visiteur
- un bouton **Directeur** : qui permettra la visualisation de la liste des visiteurs par le Directeur

Contraintes

- Nom du contrôleur : `ControleurAccueil.java`
- Nom du Layout : `accueil.xml`

Fonctionnement

- Le **Clic** sur le bouton **Visiteur** permettra de démarrer le contrôleur [ControleurVisiteur1](#)
- Le **Clic** sur le bouton **Directeur** permettra de démarrer le contrôleur [ControleurDirecteur1](#)

2- ControleurVisiteur1

La vue **visiteur1** permet de saisir le nom et le prénom du visiteur.

Contraintes

- Nom du contrôleur : ControleurVisiteur1.java
- Nom du Layout : visiteur1.xml

Fonctionnement

- Deux champs **EditText** permettent de saisir le nom et le prénom. Les champs doivent être vides avant toute nouvelle saisie.
- Le **Clic** sur le bouton **Suivant**
 - doit contrôler que les champs *nom et prénom* ont été remplis
 - si les champs sont remplis il faudra démarrer le contrôleur [ControleurVisiteur2](#)

3- ControleurVisiteur2

La vue **visiteur2** permet de saisir des informations sur l'orgine scolaire du visiteur.

Contraintes

- Nom du contrôleur : ControleurVisiteur2.java
- Nom du Layout : visiteur2.xml

Fonctionnement

- Un champ **ComboDialog** permet de choisir l'établissement d'origine de l'élève dont voici la liste exclusive :
 - Bac général
 - Bac technologique
 - Bac professionnel
 - Université
 - Autre
- Un champ **EditText** permettent de saisir le nom de l'établissement. Dans le cas **Autre** il devra indiquer de saisir **Autre à préciser**.
- Le **Clic** sur le bouton **Suivant**
 - doit contrôler que les champs ont été correctement remplis
 - Pour les **Bacs** : le nom de l'établissement
 - Pour **Autre** des précisions sur le parcours du visiteur.
 - si les champs sont correctement remplis
 - il faudra démarrer le contrôleur [ControleurVisiteur3](#)
 - Le [ControleurVisiteur2](#) doit être arrêté

4- ControleurVisiteur3

1. La vue **visiteur3** permet de saisir des informations sur la spécialisation souhaitée.
2. Il ajoutera les informations saisie dans la base de données.

Contraintes

- Nom du contrôleur : ControleurVisiteur3.java
- Nom du Layout : visiteur3.xml

Fonctionnement

- Un champ **ComboDialog** permet de choisir la spécialisation souhaitée :
 - Commerce international
 - Commerce internet
 - Finance

- Le **Clic** sur le bouton **Terminer**
 - doit contrôler que la spécialisation a été renseignée
 - si le champ est renseigné :
 - l'application ajoute les informations dans la base de données
 - le contrôleur **doit être arrêté**, ce qui devrait provoquer automatiquement l'affichage de [controleurvisiteur1](#)

5- ControleurDirecteur1

Vous réaliserez une [Activity](#) capable d'afficher les données contenues dans la base [porteouverte.db](#).

Ce **ListView** s'appuiera sur un **ArrayAdapter personnalisé**. Il doit permettre pour chaque ligne :

- L'affichage de toutes les informations d'un visiteur [porteouverte.db](#).

contraintes

- La liste doit être triée par ordre alphabétique




6- Avis du Directeur

Le directeur souhaite réaliser un premier avis directement depuis la liste des visiteurs.

Pour cela il doit pouvoir visualiser **toutes** les informations concernant chaque *visiteur*.

Attribut avis

L'attribut **avis** de la table **visiteur** offre trois possibilités selon sa valeur :

valeur	Signification	Image
0	aucun avis donné	
1	avis favorable	
2	avis défavorable	

Mémorisation de l'avis

Le directeur doit pouvoir donner un avis sur chacun des visiteurs.

Vous programmerez une solution qui permet au directeur de donner un avis sur un visiteur depuis le **ListView**.

Affichage de l'avis

Pour visualiser l'avis émis par le directeur vous afficherez une image à côté de chaque visiteur.

Vous pouvez vous inspirer du dessin proposé par le Directeur



Pour afficher une image il faut :

- Copier cette image dans **res\mipmap**
- Déclarer un composant **<ImageView ... />**

Exemple de code

```
ImageView imageView = (ImageView) convertView.findViewById(R.id.image);  
  
imageView.setImageResource(R.mipmap.monImage);
```

7- Filtrage

Le directeur de l'école souhaiterait pouvoir réaliser un filtrage des données de la manière suivante :

- Selon l'origine scolaire des élèves
- Selon la spécialité choisie

Pour cela vous utiliserez un composant (*widget*) [Spinner](#).

8 - Contact

le directeur veut pouvoir contacter par téléphone les visiteurs. Cette information devra être disponible sur la fiche visiteur.

- Il faut donc prévoir un champ téléphone dans la base de données
- Prévoir la saisie du numéro de téléphone au moment de la visite
- Permettre d'accéder à la numérotation depuis la fiche visiteur (Directeur) pour réaliser un appel (en utilisant l'appli du téléphone du portable) en cliquant sur le champ contenant le numéro de téléphone.

PorteOuverteSQLite.java

La classe **PorteOuverteSQLite** est issue de la classe [SQLiteOpenHelper](#). Elle permet :

- de créer une base de données : ici **Visiteur.db**
- de créer la/les tables de la base de données : ici la table **Visiteur**
- d'accéder aux données en lecture, ajout, mise à jour et de supprimer les données d'une table.