BTS SERVICES INFORMATIQUES AUX ORGANISATIONS	**Session 2021**

PPE – ANDROID STUDIO

Durée : 4h * 3	**Auteur : NGO**

Le lycée Jean ROSTAND ouvre ses portes au public le premier samedi du mois de mars chaque année.

Accueil des familles par les enseignants, les élèves, les étudiants et les différents services de l'établissement.

A l'occasion de cette journée, venez découvrir l'ensemble de nos **formations lycée** (Seconde, Première et Terminale) et Post-bacs (Classe de Préparation aux Concours de la Santé et du Social, BTS et DCG), visiter nos salles spécifiques (laboratoires, salles informatiques...), notre **internat** et assister à nos **conférences** sur le cycle Terminal.

Contexte du projet

Les potentiels futurs étudiants de la section BTS SIO du Lycée Jean Rostand saisissent actuellement sur feuille leur nom, prénom, adresse mail, leur Bac, leur établissement scolaire, leur choix d'option (SLAM/SISR/NSP) lors de leur visite. Venir aux **portes ouvertes** ou pendant le **salon de l'étudiant** est un gage de motivation et cette motivation est récompensée lors des sélections des dossiers.

Afin de moderniser, promouvoir et informatiser cette saisie, vous êtes chargé individuellement de réaliser une **application mobile Android** sous **Android Studio** en langage **Java / hybride IONIC**, destiné aux visiteurs lors des portes ouvertes ou pendant le salon de l'étudiant.

Tous les **champs sont obligatoires** et devront faire l'objet d'un contrôle, la base de données devra être externalisé. Une étude Merise est requise pour la réalisation du projet et cette étude doit figurer dans le livrable.

Les **enseignants** pourront **consulter les candidats** qui sont venus et inscrits pendant les portes ouvertes ou le salon de l'étudiant selon une **année** donnée.

L'enseignant pourra **trier les candidats** selon leur nom, section ou établissement scolaire.

Un **tableau statistique** du nombre de candidats inscrits, du nombre de candidat en fonction de la section et de l'établissement scolaire devra être effectué.

Un rappel du RGPD ainsi le nom de l'auteur et l'année de réalisation devront être présentes dans l'application.

Organisation

Vous travaillerez en méthode agile avec le framework scrum et en utilisant l'outil trello https://trello.com/ (inviter ngobtssio@gmail.com)

• Les cérémonies principales

Le scrum propose différentes cérémonies (réunion) pour rythmer l'ensemble des sprints (itérations)

Un sprint est constitué de :

- **un sprint planning** :
  - ouverture du sprint
  - l'équipe démarre un sprint en planifiant le travail à réaliser.
  - elle définie un objectif de sprint
  - elle définie le plan pour atteindre cet objectif
- **un sprint review** :
  - fermeture du sprint
  - l'équipe fait une revue du parcours réalisé pendant le sprint
  - l'équipe invite les utilisateurs/clients à faire des feedbacks sur le produit avec les dernières évolutions « done »
- **un sprint rétrospectif** :
  - après la review
  - l'équipe fait un point ensemble pour chercher des axes d'amélioration
- **un daily scrum**
  - l'équipe de développement se réunit pour s'harmoniser ensemble : qu'est-ce qui a été fait la dernière fois, ce qui a été fait aujourd'hui, y a t il une alerte
  - chaque matin

Il y aura **3 sprints** de 3h30 (3 sessions en tout), un sprint review à la fin de chaque session (20min) avec le client et un sprint rétrospective après la review à la fin de chaque session (10min)

Au début de chaque sprint n+1 et n+2, un daily scrum de 15/20min est attendu.

Vous travaillerez en **binôme** et comme vous êtes en concurrence, innover et montrez ce que vous savez faire, la meilleure application sera exploitée par la section. Vous avez 3 **sessions de 4 heures** pour réaliser ce projet. Vous devrez rendre le livrable de l'application à la fin des deux sessions.

Vous avez en **annexe** des éléments d'aide pour votre réalisation.

En résumé vous devez réaliser :

- Une modélisation BDD
- Une base de données externe et les tables correspondantes
- Page de présentation de l'application
- Les classes et méthodes
- un formulaire pour saisir les coordonnées des visiteurs/candidats
- une insertion dans la BDD
- une connexion en tant qu'enseignant
- un affichage des candidats selon l'année
- affichage des statistiques selon l'année
- information RGPD, auteurs et année de réalisation
- les tests et le livrable

OBJECTIFS PEDAGOGIQUES

- Travailler en équipe avec une véritable répartition et des outils de versionning
- Analyser des besoins précis, une demande déjà structurée
- Appliquer une démarche et rendre compte
- Créer une application
- Respecter des contraintes

CONNAISSANCES UTILISEES

- méthodes agiles : scrum
- Créer une application exploitant une base de données
- Modifier un schéma de données et l'implanter dans une base de données
- Créer une application mobile
- Appliquer des normes de développement
- Valider et documenter une application
- Rédiger une documentation
- Utiliser des outils de versionning et de travail en équipe

Objectif des sprints (4h) à présenter lors des revues de sprint (sprint review)

Sprint 1 – Mardi 16 mars

- Product backlog
- Trello détaillé
- MCD/MPD/Diagramme de classe
- Création de la BDD externe
- Cnx à la BDD via l'application
- Formulaire d'authentification
- Connexion et authentification administrateur, le mot de passe doit être crypté
- Formulaire d'inscription
- Activité de garde / présentation / logo

➔ Sprint review (5min en fin de sprint)

Sprint 2 – Mardi 23 mars

- Daily sprint (5/10min)
- Mise à jour Trello sprint 2
- Insertion des données du candidat dans la BDD (un candidat est unique, seul son année et/ou son lieu d'inscription change)
- Confirmation de l'insertion des données du candidat
- Affichage des candidats par année et/ou par portes ouvertes/salon de l'étudiant (tri possible par nom et/ou par section d'origine)

➔ Sprint review (5min en fin de sprint)

Sprint 3 – Mardi 30 mars

- Daily sprint (5/10min)
- Mise à jour Trello sprint 3
- Archivage d'une session de candidat (par année)
- Modification du mot de passe administrateur
- Livrable
- APK

➔ Sprint review (5min en fin de sprint)

Annexes

1. Raccourcis Clavier Android Studio

Voici les raccourcis clavier sur Android Studio pour augmenter votre rapidité de développement.

1. Aller sur la classe ou l'élément : Cmd/Ctrl+click sur l'élément
2. Indenter le code et revoir les imports : Cmd/Ctrl+Alt+L
3. Renommer une variable : Shift+F6
4. Auto-Générateur de getter/setter, constructeur, méthodes surchargées : Cmd/Ctrl+N
5. **Auto-complétion : Cmd/Ctrl+Space**
6. **Correction automatique** : se placer sur la ligne, **Alt+Enter**

Remarque : Il est possible de voir tous les raccourcis à partir d'Android Studio > Preferences > Keymap.

2. Tutoriels IONIC

https://ionicframework.com/

https://ionicframework.com/docs

https://firebase.google.com/docs/firestore

https://drissas.com/ionic-firestore/

3. Tutoriels Android / MYSQL

https://www.youtube.com/watch?v=PKd8CEXXyLk

https://www.youtube.com/watch?v=n5AeP-fqT30

https://www.youtube.com/watch?v=8Kyq69u9iqU

4. Tutoriels Android

Cf PPE

5. Passer du MCD au diagramme de classe avec win'design

Menu → Modèle → >Générer le diagramme de classe (UML)

6. Many to one

Règle de gestion : un salarié appartient à une et une seule catégorie.

```
public class Salarie {

private String matricule ;

private String nom;

private Categorie laCategorie ;


// constructeurs et accesseurs modificateurs

public Categorie getLaCategorie() {

return laCategorie;

}

public void setLaCategorie(Categorie laCategorie) {

this.laCategorie = laCategorie;

}


public class TestSalarie {

public static void main(String[] args) {

Salarie unSalarie1 = new Salarie("Dupont", 25000);

Categorie laCateg = new Categorie("DJV","Developpeur Java) ;

unSalarie1.setCategorie(laCateg) ;

System.out.println("Le salarié se nommant" + unSalarie1.getNom()
+ " a pour catégorie " +
unSalarie1.getlaCategorie().getLibelle());
```

7. One to Many

Régle de gestion : un salarié peut être responsable de plusieurs projets → la classe Salarie contiendra une propriété qui est une **liste de projets**.

```
public class Salarie {

private String nom ;

private double salaire ;

private ArrayList<Projet> listeProjets ;

// getter de la liste de projets

public ArrayList<Projet> getListeProjets() {

return listeProjets;

}

//setter de la liste de projet

public void setListeProjets(ArrayList<Projet> pListeProjets) {

this.listeProjets = pListeProjets;

}

// méthode permettant d'ajouter un seul projet à la liste.

public void addProjet(Projet unProjet){

if (this.listeProjets == null){

this.listeProjets = new ArrayList<Projet>();

}

listeProjets.add(unProjet);

}

// autres constructeurs et getters setters

}
```

```
public class TestSalarie {



public static void main(String[] args) {

Salarie unSalarie1 = new Salarie("Dupont", 25000);

Projet projet1 = new Projet("Application de gestion des
interventions");

Projet projet2 = new Projet("Application vente");

//Dupont est responsable des 2 projets

unSalarie1.addProjet(projet1);

unSalarie1.addProjet(projet2);

for(int i = 0; i < unSalarie1.getListeProjets().size(); i++){

System.out.println("Le salarie est responsable du projet " +
unSalarie1.getListeProjets().get(i).getIntitule());

}
```