Cahier Technique - TempoTrain

1. Contexte du Projet

L'agence **TempoTrain** propose des voyages temporels sécurisés et programmés. L'objectif de ce projet est de développer une application permettant aux **managers** et **conducteurs** de gérer efficacement les trajets, en optimisant :

- La **gestion du planning des conducteurs**
- Le **suivi des billets et des places disponibles**

2. Planification revue du projet v1 (Diagramme de Gantt)

Un **Gantt** a été créé pour suivre la progression des tâches, attribuées aux membres de l'équipe.

NUMÉRO	TITRE DE LA TÂCHE	PROPRIÉTAIRE DE LA TÂCHE	DATE DE DÉBUT	DATE LIMITE	DURÉE	TÂCHE TERMINÉE (EN %)
1.1	Maquette	Emir	06/12/24	08/12/24	2	100 %
1.2	ReadMe	Semih.C	29/11/24	30/11/24	1	100 %
1.3	Diagramme d'utilisation	Semih.S	06/02/25	07/02/25	1	100 %
1.4	Diagramme de Classe Métier	Emir	06/02/25	07/02/25	1	80 %
1.5	Diagramme de GANTT	Semih.S	13/12/2024	07/02/25	54	80 %
2.1	Modélisation de la Base de Données	Semih.S				
2.2	Développement des Classes Modèle	Emir				
2.3	Création des Vues FXML	Semih.C				
2.4	Connexion Backend	Emir				
3.1	Coder l'interface login	Semih.S				
3.2	Gérer l'ajout des utilisateurs	Semih.C				
3.3	Gérer l'ajout des trajets	Emir				
3.4	Gérer l'ajout des billets	Semih.C				
3.5	Gérer l'ajout des arrêts	Semih.S				
4	Tests Unitaires	Emir				

Outil utilisé : Google Sheets (Gantt coloré selon l'affectation des tâches).

3. Contraintes Techniques

Langages & Frameworks :

- **Backend** : Java **22**
- **Frontend** : JavaFX 22.0.1
- **Base de données** : MariaDB Client 3.4.1

Outils de gestion et conception :

- **JUnit** : 5.10.2
- **Maven Compiler Plugin** : 3.13.0
- **JavaFX Maven Plugin** : 0.0.8
- **IntelliJ** → Éditeur de code
- **GitHub** → Versioning & Collaboration (https://github.com/)
- **Excalidraw** → Conception des diagrammes (https://excalidraw.com/)
- **Figma** → Conception du design (https://www.figma.com/fr-fr/)
- **Linear** → Gestion des taches (https://linear.app/)

4. Modèle Relationnel

- `utilisateur(id: int(3), nom: varchar(30), mail: varchar(100),prenom: varchar(30), mdp: varchar(256), role: enum('MANAGER','CONDUCTEUR'))`
  - `clé primaire: id`
  - `clé étrangère: ___`

- `trajet(id: int(3), heureDepart: datetime, heureArrivee: datetime, anneeDestination: int(4), placesTotale: int(2), placesDisponibles: int(2), statut: enum('REALISE','EN_COUR','A_REALISE'), conducteur: int(3), arretDepart: int(3), arretArrivee: int(3))`
  - `clé primaire: id`
  - `clé étrangère: conducteur REFERENCES utilisateur(id)`
  - `clé étrangère: arretDepart REFERENCES arret(id)`
  - `clé étrangère: arretArrivee REFERENCES arret(id)`

- `billet(id: int(3), nom: varchar(30), trajet: int(30))`
  - `clé primaire: id`
  - `clé étrangère: trajet REFERENCES trajet(id)`

- `arret(id: int(3), localisation: varchar(100), trajet: int(30))`
  - `clé primaire: id`
  - `clé étrangère: trajet REFERENCES trajet(id)`

5. Organisation de votre code

Packages et leurs responsabilités

- Package principale : fr.treiinquille

- Modele

- Exemples :
  - `Utilisateur.java`
  - `Trajet.java`
  - `Billet.java`
  - `Arret.java`

- Vue

- Exemples :
  - `ajoutUtilisateur-view.fxml`
  - `ajoutTrajet-view.fxml`
  - `ajoutBillet-view.fxml`
  - `ajoutArret-view.fxml`

- Controleur

- Exemples :
  - `AjoutUtilisateurController`
  - `AjoutTrajetController`
  - `AjoutBilletController`
  - `AjoutArretController`