# Projet DA50
# - Cahier des charges -

*InvestMate*

Tuteur: M.Galland

*Robert Hakobyan*
*Arthur Aquilano*
*Hugo Pereira*
*Thomas Pierron*
*Adam Bin Azmi*
*Abderrahman Rezki*

# Sommaire

# I) Présentation et Objectifs du projet

## 1. Présentation du projet

A une époque où faire dormir son argent sur un compte en banque est l'idée la plus courante mais également celle qui coûte le plus, on entend parler d'investissements. On conseille d'investir dans l'immobilier, mais avec l'explosion des prêts bancaire cette solution demande de plus en plus de capital, d'acheter de l'or, mais tout le monde ne peut se s'offrir un coffre fort ultra sécurisé, ou encore d'acquérir des actions et des parts d'entreprise en bourse. Cette dernière option, bien que accessible à tout le monde, est un milieu de requin, où chacun profite des faiblesses des débutants pour s'enrichir avec des formations hors de prix ou en utilisant un jargon complexe pour se rendre indispensable.
Partant de ce constat, nous désirons simplifier l'apprentissage et l'investissement en bourse, aussi appelé trading, en concevant puis développant une application de trading algorithmique qui intégrera des fonctionnalités pédagogiques pour former les utilisateurs aux concepts fondamentaux du trading ainsi que l'intégration de robots de trading automatisés, dont leur rôle est d'aider l'utilisateur à faire les meilleurs choix dans ses investissements.

## 2. Portée du projet

Notre projet se divise en deux outils, chacun ayant ses fonctionnalités et une portée propre.

Un site internet qui permet:

- **de découvrir et expliquer** le projet,
- **de décrire et télécharger** l'application de bureau,
- **de contacter l'équipe derrière le projet** afin de faire la demande d'intégrer de nouveaux bots de trading.

Une application bureau qui permet:

- **une formation intégrée** grâce à des modules éducatifs: des cours, des vidéos et des quiz (QCM),
- **du trading automatisé** par l'intégration de bots(robots) de trading qui sont configurables par les utilisateurs et connectés en temps réel aux marchés financiers,
- **l'administration des contenus pédagogiques** par des administrateurs qui créent, modifient, suppriment et finalement publient les cours à l'ensemble des utilisateurs.

# 3. Objectifs

Les deux outils répondent à une liste d'objectifs précises:

Le site internet doit:

- **permettre une compréhension complète** de l'outil (Une page de description),
- **offrir un accès simple et rapide au téléchargement** de l'application bureau (Bouton sur la page d'accueil, et dans la barre des onglets en haut),
- **intégrer un moyen pour contacter l'équipe derrière le projet** pour demander à intégrer de nouveaux algorithmes de trading.

Et l'application de bureau doit:

- **fournir des contenus éducatifs** simples mais complets pour l'apprentissage des du trading, accessibles au sein de l'application. (avec 2 leçons à titre d'exemple intégrées par défaut),
- **standardiser la rédaction des cours** au travers d'une même interface de création des nouvelles des leçons accessible aux administrateurs uniquement,
- **permettre une gestion des cours** par les administrateurs qui pourront créer ou modifier des modules de formation via l'interface dédiée, et finalement les publier,
- **mettre à jour automatiquement les cours accessibles aux utilisateurs** dès le démarrage de l'application où toutes les heures si de nouveaux cours sont disponibles/ont été mis à jour,
- **autoriser et standardiser l'intégration de code python** par les équipes de développement lorsqu'un utilisateur fait la demande d'intégrer un nouveau bot de trading, dans le but de faciliter le déploiement du nouvel algorithme tout en protégeant l'application.

# II) Fonctionnalités

## 1. Site Web

**Description du projet** sur la page d'accueil du site web, pour expliquer la démarche, et pourquoi utiliser notre application.

**Téléchargement de l'application** depuis la page d'accueil ou la barre des pages.

**Guide d'utilisation de l'application** sur une page réservée du site web avec les informations de base pour installer et utiliser l'application.

## 2. Application Desktop

*Le rôle Admin hérite de l'ensemble des permissions du rôle lecteur. Le rôle lecteur ne possède aucune permission exclusive au mode Admin.*

**Système d'authentification** des utilisateurs pour leur attribuer un rôle (Lecteur ou Admin) afin de restreindre les fonctionnalités accessibles et leur portée.

**Chiffrement des données de connexions** MySQL supporte le chiffrement via des fonctions SQL comme `AES_ENCRYPT()` et `AES_DECRYPT()` dans les requêtes notamment pour les mots de passe.

**Liste de l'ensemble des cours** disponibles pour l'utilisateur Lecteur

**Système de recherche avec filtres** des cours disponibles pour l'utilisateur Lecteur

**Lecture des cours** par les utilisateurs avec le rôle Lecteur

**Participation aux quiz** par les utilisateurs avec le rôle Lecteur

**Liste des cours non publiés** pour les utilisateurs Admin

**Interface de création des cours selon un modèle** similaire

**Ecriture et modification des cours** par les utilisateurs avec le rôles Admin

**Suppression et publication des cours** par les utilisateurs avec le rôles Admin

**Utilisation des algorithmes de trading** par les utilisateurs avec le rôle Lecteur.

**Intégration et mise à jour des bots de trading** par les membres de l'équipe de développement.

## 3. Base de données

**Gestion des comptes utilisateurs et leur rôle**, qui devront se connecter à chaque fois qu'il relance l'application permettant de ne pas avoir à gérer une session complexe mais en gardant la sécurité de l'application.

**Création (resp. suppression) de nouvelles instances** lorsqu'un nouveau cours est créé (sans être publié)  (resp. supprimé)

**Mise à jour des cours** des données des tables lorsqu'un cours et modifié

**Accès à des l'ensemble des utilisateurs** d'un cours lorsque que l'Admin le publie

**Actualisation automatique** des cours accessibles aux Lecteurs au démarrage de l'application ou toutes les heures.

# III) Exigences techniques

Les outils utilisent les technologies suivantes:

## 1. Site internet

- **Vite, React.js et Tailwind CSS** pour le frontend du site web. Vite assure une construction rapide et optimisée du projet, React.js fournit des composants dynamiques et interactifs, et Tailwind CSS propose un style basé sur des utilitaires pour un design moderne et réactif.
- Un environnement de serveur local tel que **MAMP ou XAMPP** pour développer et tester le site web créé.

## 2. Application (FrontEnd)

- **Java et JavaFX** pour le développement de l'interface graphique et des fonctionnalités de base de l'application. Le choix de faire une application et non une application web et pour deux raisons principales. La première pour pouvoir bénéficier d'une connexion hors ligne pour les cours. La deuxième est pour gérer la sécurité nous même sans dépendre du web.
- **Python** (*utilisé au travers de méthodes Java*) pour l'intégration des bots de trading automatisés. *Indiquer les librairies à utiliser et tester*

## 3. Base de données (Backend)

- **J2EE** pour la gestion des utilisateurs, de l'authentification et des cours.
- **Hibernates** pour la connexion à la base de données, la gestion des informations utilisateur et les cours.
- **Système de gestion de base de données relationnelle** sous MySQL pour le stockage des informations utilisateurs, des cours, des configurations des bots de trading et des données

## 4. Sécurité

- **Système d'authentification** par nom d'utilisateur et mot de passe (8 caractères dans la table ASCII), protocole HTTPS, limitation des tentatives de connexion et journalisation des accès, déconnexion automatique après inactivité, génération de tokens JWT après connexion, vérifiés à chaque requête.
- **Restriction des permissions et accès aux données** selon deux rôles (Lecteur et Admin qui en hérite).
- **Suivi des logs et audit** des activités. Des historiques de connexion, des cours suivis, du temps de connexion.
- **Gestion des permissions** : Les utilisateurs doivent avoir des niveaux de permissions distincts (par exemple, admins, créateurs de bots, utilisateurs standards).

# IV) Contraintes

1. Performance
    - Gestion de plusieurs connexions d'utilisateurs en simultanées.
2. Compatibilité
    - L'application est compatible avec les principaux systèmes d'exploitation (Windows, macOS, Linux).
3. Evolutivité
    - L'infrastructure est capable d'intégrer de nouvelles fonctionnalités à moyen et long terme sans impacter l'expérience utilisateur.

# V) Échéancier et livrables

## 1. Estimation des coûts en temps

Nous basons l'estimation de la durée d'une tâche du projet en "PM". Ce terme désigne la durée d'une tâche en mois pour une seule personne qui travaille à temps plein (soit 35h par semaine de 5 jours). Ensuite nous pourrons rapporter la durée de ces dites tâches au temps que nous allouons au projet chaque semaine.

Notre projet comporte 7 grands jalons:

1. **Organisation du projet**
2. **L'analyse et la conception des fonctionnalités**
3. **Prototypage du site web et de l'application et définition de la charte graphique**
4. **Développement du site web**
5. **Mise en place de la base de données**
6. **Développement du frontend de l'application**
7. **Développement du backend de l'application**

Voyons avec un peu plus de détail les tâches à réaliser dans chaque jalons:
*Méthode de calcul: 1 PM = 1 mois = 4.27 semaine(5 jours) = 21.35 jours => 1 jour = 0.0468PM ~ 0.05PM*

**Organisation du projet:**

- Choix des outils (IDE, Version JDK, Frameworks, etc.) → 1 jour / 0.05PM
- Mise en place des outils (Repository Github, Channel de discussions, etc.) → 1 jour / 0.05PM
- Définir les jalons de développement et les deadlines → 1 jour / 0.05PM
- Réaliser le diagramme de Gantt → 1 jours / 0.05PM
- Rédaction du cahier des charges → 1 semaines / 0.25PM
- Amélioration du cahier des charges → 1 semaines / 0.25PM
- Réaliser le livrable: "Pitch vidéo de 2 minutes" → 3 jours / 0.15PM
- Journal de bord du temps de travail effectué → 1 jour / 0.05PM
- Préparation de la soutenance → 2 jours / 0.10PM
- Réunions avec les acteurs du projet → 1 jour / 0.05PM
- Préparation du livrable: "Code source" → 1 jour / 0.05PM

Durée totale: 4 semaine et 2 jours -> 1.10PM

**L'analyse et la conception des fonctionnalités:**

- Lister l'ensemble des fonctionnalités du projet → 2 jours / 0.10PM
- Conception d'un diagramme de cas d'utilisation → 1 jours / 0.05 PM
- Conception d'un diagramme de classes → 1 jours / 0.05 PM

Durée totale: 4 jours -> 0.20PM

**Développement du site web:**

- Coder le contenu du site → 1 semaine / 0.25PM

Durée totale: 1 semaine -> 0.25PM

**Mise en place de la base de données:**

- Conception de l'architecture de la base de données (à partir du diagramme de classe) → 1 jours / 0.05PM
- Règles de sécurité de la base de données → 1 jours / 0.05PM
- Création de la base de données → 1 jours / 0.05PM

Durée totale: 3 jours -> 0.15PM

**Développement du frontend de l'application:**

- Coder la page principale de l'application → 2 jours / 0.10PM
- Coder la page de liste de l'ensemble des cours disponibles –> 1 jours / 0.05PM
- Coder le menu de connexion → 1 jours / 0.05PM
- Coder la page de création de nouveaux cours → 3 jours / 0.15PM
- Coder la page de création des quiz → 3 jours / 0.15PM

Durée totale: 2 semaines / 0.50PM

**Développement du backend de l'application:**

- Coder le squelette de l'application → 1 jour / 0.05PM
- Coder le système de recherche et de filtres → 2 jours / 0.10PM
- Coder le système d'authentification et de token pour une session hors ligne → 1 semaine / 0.25 PM
- Coder le système de sauvegarde des cours → 2 jours / 0.10PM
- Coder le système de chargement et d'affichage des cours sauvegardés → 3 jours / 0.15PM
- Coder le système de publication des cours → 1 jours / 0.05PM
- Coder le système de sauvegarde des quiz → 2 jours / 0.10PM
- Coder le système de chargement et d'affichage des quiz sauvegardés → 3 jours / 0.15PM
- Coder le système de publication des quiz → 1 jours / 0.05PM
- Coder le système de sauvegarde des résultats des cours utilisateurs → 2 jours / 0.10PM
- Coder le système d'actualisation (manuel et automatique) des cours disponibles → 2 jours / 0.10PM
- Implémenter un algorithme de trading automatique → 1 semaine / 0.25PM

Durée totale: 5 semaine et 4 jours / 1,45PM

Ainsi on obtient une estimation de la durée du projet, en considérant un ingénieur junior à temps plein et si le projet ne rencontre aucun problème, d'environ **3.75PM, soit 3 mois et 3 semaines**.

**Adaptation de l'échéancier à notre équipe**

Nous avons estimé une charge de travail pour un ingénieur junior à temps plein de **3,75 PM** . Cependant, dans notre contexte de l'UE DA50, chaque membre a une disponibilité de 10 heures par semaine, pour un total de 150 heures sur 15 semaines, entre septembre et janvier, donc selon cette échelle, un membre de l'équipe travaillant selon le temps à consacrer pour l'UE, compléterait une tâche estimée à **1 PM en 3,75 mois**.

En reprenant les estimations initiales, il est clair qu'une seule personne terminerait l'intégralité du projet en environ **14 mois**, à condition qu'aucun contretemps ne survienne. Donc en intégrant une marge supplémentaire de 10 % pour gérer les aléas potentiels, cette durée atteindrait environ **15,5 mois**.

**Impact de la composition de l'équipe**

Notre équipe compte six ingénieurs débutants. Ainsi, en répartissant équitablement les tâches, la durée estimée du projet passe à environ **2 mois et 3 semaines**.

En tenant compte de notre démarrage le **20 septembre**, nous prévoyons de finaliser le projet vers le **15 décembre**.

## 2 . Les livrables

Les livrables suivants sont requis pour assurer le bon déroulement et la validation du projet :

- **Vidéo de type "pitch commercial"** : Une vidéo de présentation visant à exposer les objectifs, les fonctionnalités principales et la valeur ajoutée du projet, à remettre pour le 12 novembre.
- **Cahier des charges** : Document détaillant les exigences fonctionnelles et techniques du projet, à soumettre pour validation avant le 25 octobre.
- **Rapport de projet** : Document complet expliquant le déroulement du projet, les choix techniques, les méthodologies utilisées, les résultats obtenus, et les analyses. Ce rapport est attendu pour le 10 janvier.
- **Rapport du temps de travail effectif** : Un suivi individuel du temps consacré par chaque membre du groupe aux différentes tâches, afin de comparer l'effort investi à à la durée du projet initialement estimée. À soumettre pour le 10 janvier.
- **Code source de l'application** : L'ensemble du code source, correctement structuré et documenté, pour assurer la pérennité et la compréhension du projet. À remettre pour le 10 janvier.
- **Application et base de données associée** : Version finale de l'application, du site web, prête à être utilisée, ainsi qu'une base de données configurée et opérationnelle. Date de remise : 10 janvier.
- **Présentation finale avec support** : Une présentation formelle du projet accompagnée d'un support visuel (ex : diaporama), détaillant les fonctionnalités clés et les points techniques importants, pour la démonstration finale le 10 janvier.

**Description des livrables**

Chaque livrable comprendra :

- **Type de document** : Précisé pour chaque livrable (ex : vidéo, document Word, fichier PDF, code source).
- **Preuve de son existence** : Tous les livrables seront disponibles dans le dépôt de projet GitHub.

**Réunions de suivi avec le tuteur de projet**

Nous anticipons également la tenue de 2 à 3 réunions de suivi avec le tuteur du projet, M. Galland. La présence de l'ensemble du groupe est requise pour ces réunions, d'une durée estimée entre 1 et 2 heures chacune.

# VI) Budget prévisionnel

Le détail de l'estimation du budget nécessaire à la complétion du projet:

## 1. Coûts de Personnel

Pour un Ingénieur Junior IT à Temps Plein :

- **Taux Horaire Brut Facturé** : 45 €
- **Durée Totale du Projet** : 4 mois (arrondi)
- **Calculs** :
    - Heures par Mois (basé sur 35 h/semaine, 4.27 semaine/mois) : environ 150 heures
    - Heures Totales pour 4 Mois : 600 heures (4 x 150 heures)
    - **Coût du Personnel** = 600 heures x 45 €/heure = 27 000 €

## 2. Coûts des Logiciels et Outils

- **Suite JetBrains (IntelliJ)** : Utilisée pour le développement Java.
  Estimation à environ 60€ par mois par utilisateur. Pour le projet de 4 mois:
    - **Coût** : 240€
- **Gestion de Base de Données et Outils Backend** :
    - **MySQL** :Nous optons pour la version entreprise pour plus de sécurité, de scalabilité et de support.
    - La version entreprise coûte environ 5000€ par an par serveur.
    - **Coût** : 1670€ (1 serveur 4 mois)
- **Bibliothèques d'Intégration Python** :
    - **Bibliothèques standards** : choix de n'utiliser que des bibliothèques gratuites.
- **Outils de Gestion de Projet et de Collaboration (Github)** : Estimation à 20 € par utilisateur par mois pour le service entreprise.
  Nous avons donc un utilisateur (l'ingénieur sur le projet):
    - **Coût** 80€

## 3. Matériel et Infrastructure

- **Machines de développement pour l'ingénieur junior** : Ordinateur pour le développement entre 1 200 à 1 500 €.
    - **Coût** : 1200 € - 1 500 € (si nécessaire).
- **Environnement de Test (Serveur Local)** : Utilisation de MAMP ou XAMPP, avec les versions gratuites suffisantes pour les tests.
    - **Coût** : 0 €
- **Hébergement** : Pour l'utilisation d'une base de données, de l'authentification de connexion et des données des cours, un serveur cloud dédié (AWS, Azure, ou

Google Cloud) est envisagé.
Estimation à 50-100 € par mois pour une instance cloud standard capable de gérer les données en temps réel pour l'accès des utilisateurs et les bots de trading.

- **Coût pour 4 mois** : ~400 €

## 4. Coûts de Sécurité et de Conformité

- **Authentification et contrôle d'accès** : Les bibliothèques standard (Java, Python) couvrent la plupart des fonctionnalités de contrôle d'accès et sont gratuites.

## 5. Fonds de Contingence

Tout comme nous avons estimé à hauteur de +10% la durée du projet en cas de problèmes dans le développement, nous estimons qu'une hausse de 10 % du budget total est nécessaire pour couvrir les coûts imprévus, tels que des abonnements supplémentaires à des outils, des pannes matérielles, ou des risques anticipés mais inévitables. Ce fonds de contingence nous garantit d'avoir l'ensemble des ressources à disposition pour respecter l'ensemble des deadlines du projet.

## 6. Tableau du budget prévisionnel

| **Catégorie** | **Coût estimé** (€) |
|---|---|
| Personnel (Ingénieur IT) | 27 000€ |
| Intellij (JetBrain) | 240€ |
| Database & Backend (MySQL) | 1 670€ |
| Github | 80€ |
| Ordinateur de développement | 1 500€ |
| Hébergement | 400€ |
| Fonds de contingence | 3089 € |
| **Budget total estimé** | **33 980€** |

Nous estimons donc que le projet demande un investissement financier de 34 000€ pour une durée de 4 mois et avec un seul employé travaillant sur le projet.

Ce projet est né et restera la propriété d’InvestMate.

L'utilisateur peut investir librement le montant de son choix, permettant à ses fonds d'être gérés par les bots, ou de les conserver dans son portefeuille (fonds non investis ou retirés). Tant qu'il maintient un investissement d'au moins 100 euros sur la plateforme (sur une période d'une semaine minimum au cours du dernier mois), il a accès aux cours éducatifs.

Cependant lors de la création du compte, le client peut avoir accès au cours pendant deux semaines, sans pour autant avoir besoin d’investir son argent.

Quant à l’origine de nos bénéfices, ils seront de 25% sur les bénéfices générés par les robots de trading au moment où l’utilisateur décide de retirer son investissement vers son portefeuille. Si les bots ne sont pas positifs, alors aucun pourcentage ne sera retiré à l'utilisateur quand il voudra retirer son argent de la partie investissement à portefeuille.

De plus, 2 euros seront facturés lors de chaque dépôt d’argent de l’utilisateur.

# VII) Analyse des risques

## 1. Risques Technologiques

| Risque | Probabilité | Solutions |
|---|---|---|
| Incompatibilité des technologies entre les membres de l'équipe | Très peu probable | - Mêmes IDE, à version égale,<br>- Même version de JDK |
| Obsolescence d'une technologie | Probable | - Veille technologique<br>- Prévoir de nouvelle technologie au cas où une devient obsolète |
| Défauts de performance | Très probable | - Optimiser le code<br>Simplifier les processus |

## 2. Risques Humains

| Risque | Probabilité | Solutions |
|---|---|---|
| Manque de compétences | Très probable | - Prendre 2 semaines pour faire une montée en compétences |
| Mauvaise communication au sein de l'équipe | Très peu probable | - Organiser des réunions hebdomadaires pour échanger sur le projet |
| Absence non planifiée | Probable | - Répartir les tâches du membre à l'équipe tant qu'il n'est pas revenu. |
| Membre qui quitte le projet | Probable | - Répartir ses tâches aux autres membres<br>- Négocier les attentes clients pour la première deadline |

## 3. Risques Environnementaux

| Risque | Probabilité | Solutions |
|---|---|---|
| Pandémies ou crises sanitaires | Très peu probable | - Organiser les réunion en distancielle |
| Catastrophes naturelles (tempête, violent orage, tremblement de terre, …) | Probable | - Négocier les attentes clients pour la première deadline |

## 4. Risques Matériels

| Risque | Probabilité | Solutions |
|---|---|---|
| Pannes matérielles d'un membre du projet | Probable | - Utiliser des outils comme GitHub pour sauvegarder son travail |
| Pannes des serveurs | Très peu probable | - Prévoir des interfaces où rediriger l'utilisateur pour l'informer de la situation |
| Perte des serveurs (incendie, etc.) | Très peu probable | - Avoir des archives sur deux autres appareils |