# 

**RAPPORT PFE**

**CA-TS**

Etudiants : Lucas Leroy, Auriane Poirier et Margaux Thirion

Tuteurs Entreprise : Damien Teissier et Thomas Deffains

Tuteur École : Lysiane Lopez

[**I. Contexte 2**](#_uh6l9jwxn3t0)

[A. Présentation du projet 2](#_ljvh7rlw427p)

[B. Présentation de l’entreprise 2](#_scmqeogxv1ox)

[C. Contexte et enjeux du projet 2](#_u2apy85f0uyu)

[**II. Travail réalisé 3**](#_84hefb8jyxoz)

[A. Démarche de conception 3](#_jnovt5awz8c6)

[1. Conception Backend 3](#_46x99j6665j2)

[2. Conception Frontend 5](#_49jcugh8q65c)

[B. Résultat 7](#_yamglgqobt8n)

[1. Résultat Intermédiaire 7](#_kv8wbva6visp)

[2. Résultat final 8](#_ezv95uj572uw)

[**III. Problème et solutions 8**](#_g8gni5eq0cb)

[**IV. Gestion du projet 8**](#_ote9cdnrli5a)

[A. Méthodologies utilisées 8](#_3ohnmtmz46ij)

[B. Bonnes pratiques 8](#_x2fsqz4wuxmc)

[**V. Analyse critique 8**](#_b62kc7yzdl48)

[A. Analyse des choix techniques et organisationnels 8](#_ot9laooqk4y2)

[B. Intégration des préoccupations environnementales 8](#_ng6zgit2t45b)

[**VI. Expérience tiré 8**](#_qe3kj5vxfj4n)

[A. Valorisation du travail 8](#_dqdmhji3f7ne)

[B. Compétences techniques et soft skills développées 8](#_l9nz0v54ei9k)

[**VII. Annexes 9**](#_a63o01hf0v47)

[**VIII. Résumés 10**](#_1pv32oz9m1gz)

[A. En français 10](#_kx9t2h79xgn)

[B. En anglais 10](#_yp9qppb79m34)

# Contexte

## Présentation du projet

Ce projet s'inscrit dans le cadre de notre dernière année au département Informatique et Gestion de Polytech Montpellier. Réalisé en collaboration avec le Crédit Agricole Technologies et Services de Montpellier, et notamment nos tuteurs entreprise: Damien Teissier et Thomas Deffains.

D'une durée de deux mois et demi, ce travail a été mené par Lucas Leroy, Auriane Poirier, et Margaux Thirion, répondant à une demande spécifique de l'entreprise. Le projet a bénéficié de l'accompagnement de notre tutrice école, Lysiane Lopez, qui a guidé nos efforts et contribué à orienter nos démarches tout au long du projet.

## Présentation de l’entreprise

Le projet a été réalisé à la demande de Damien Teissier et Thomas Deffains, tous deux développeurs Android chez Crédit Agricole Technologies et Services (CA-TS) à Montpellier. CA-TS joue un rôle crucial au sein du groupe Crédit Agricole, première banque de proximité mutualiste en France, en assurant la conception, la fabrication et la maintenance des systèmes d'information bancaires technologiques pour les 39 Caisses régionales.

Grâce à son expertise technique et son engagement envers l'innovation, CA-TS facilite les opérations bancaires quotidiennes de millions de clients et soutient le développement de solutions numériques sécurisées et performantes. La collaboration avec Damien Teissier et Thomas Deffains, experts dans le domaine du développement Android, a permis de cibler précisément les besoins technologiques et d’adapter notre projet pour répondre efficacement à la demande du projet.

## Contexte et enjeux du projet

Le projet s’inscrit dans le domaine de la gestion des bornes de recharge pour les moyens de transport électrique pour le Crédit Agricole Montpellier, un secteur en plein essor face aux enjeux de mobilité durable et de transition énergétique. Il vise à répondre aux besoins du crédit agricole souhaitant optimiser l’utilisation et la gestion de leurs bornes de recharges électriques tout en facilitant leur gestion par et pour les employés

**Enjeux du projet :**

* **Optimisation de la gestion des ressources de transport électrique :**Il s'agit de centraliser et simplifier la gestion des bornes de recharges de l’entreprise. L’application doit permettre une utilisation efficace des bornes tout en maximisant leur disponibilité.
* **Transition écologique et responsabilité sociétale :**Ce projet s'inscrit dans les objectifs du Crédit Agricole de promouvoir une mobilité durable et de contribuer à la réduction de notre empreinte carbone. En facilitant l’utilisation de moyens de transport électriques et en sensibilisant les utilisateurs à des pratiques écoresponsables
* **Amélioration de l'expérience utilisateur :**L'application doit offrir une interface intuitive et des fonctionnalités adaptées pour répondre aux attentes des employés. Une adoption rapide et un usage régulier par les utilisateurs finaux sont essentiels pour garantir le succès du projet.
* **Fiabilité et disponibilité des services :**Il est crucial de fournir une solution performante pour assurer la continuité des services, notamment en termes de réservation, d’évaluation de la disponibilité des bornes, et de visualisation de l’état des bornes.

**Contraintes du projet :**

* **Contraintes techniques :** Gestion des bornes, incluant les localisations des bornes sur le parking, les historiques d’utilisation, et les états et utilisations des bornes. Développement en Kotlin Multiplatform (iOS et Android) pour garantir l’accessibilité à tous les employés.
* **Contraintes temporelles :** Respect des délais de livraison pour aligner le rendu de l’application avec les objectifs stratégiques de l’entreprise.

# Travail réalisé

## Démarche de conception

Pour ce projet, il était essentiel de mettre en place une architecture robuste comprenant un backend pour gérer le stockage des informations et la logique des données, ainsi qu’un frontend destiné à communiquer les informations aux utilisateurs.

### Conception Backend

**Choix initial de la base de données :**

Initialement, nous avions opté pour une solution SQL en utilisant PostgreSQL, reconnue pour sa fiabilité et ses performances dans la gestion des données complexes. Cependant, nous avons rencontré des difficultés lors du déploiement de notre solution. La première étape consistait à initialiser la base de données puis de déployer notre solution. Ce processus devait permettre à notre équipe de trois personnes de travailler simultanément sans la contrainte de devoir cloner en local le backend.

Face aux défis de déploiement de PostgreSQL et aux limitations des options de déploiement gratuit disponibles sur le marché, nous avons dû reconsidérer notre choix. Après une évaluation approfondie, nous avons opté pour MongoDB. Cette technologie est largement utilisée dans l'industrie pour sa flexibilité, sa facilité de prise en main et sa capacité à être déployée automatiquement de manière gratuite.

**Migration vers MongoDB :**

La transition vers MongoDB s'est avérée être une solution pragmatique face aux obstacles techniques rencontrés avec PostgreSQL. MongoDB offre non seulement une gestion simplifiée et dynamique des schémas de données, mais aussi une intégration aisée avec les environnements de déploiement gratuits, facilitant ainsi la mise en œuvre et l'accès à distance par toute l'équipe.

Cette décision a marqué un tournant dans la conception de notre backend, nous permettant de maintenir l'efficacité tout en simplifiant la gestion technique du projet.

**Choix de la technologie du back :**

Dans la conception de notre application mobile, le choix de la technologie backend revêtait une importance cruciale pour assurer la performance et la maintenabilité du système. Après mûre réflexion, nous avons opté pour Java en tant que langage de programmation, en raison de plusieurs facteurs clés qui alignaient cette technologie avec les besoins de notre projet.

Java, avec sa robustesse et son écosystème étendu, nous offrait la stabilité et la flexibilité nécessaires pour développer un backend performant. L'un des avantages significatifs de l'utilisation de Java pour notre backend était la compatibilité avec Gradle, un système de gestion et d'automatisation de build puissant. Gradle supporte spécifiquement des dépendances liées à MongoDB, ce qui simplifiait considérablement l'intégration et la gestion de notre base de données nouvellement adoptée, MongoDB.

De plus, cette décision nous permettait de mettre en pratique dans un contexte professionnel les compétences et les connaissances acquises lors de notre formation en Informatique et Gestion à Polytech Montpellier. Utiliser Java et Gradle nous a non seulement aidés à appliquer les théories apprises en cours, mais a également renforcé notre compréhension pratique de ces outils dans des situations réelles de développement de logiciels.

**Choix de la conception du back :**

Pour la structure de notre backend, nous avons adopté une variante de l'architecture Modèle-Vue-Contrôleur (MVC), spécifiquement adaptée pour répondre aux exigences d'une application backend. Cette approche est communément désignée sous le terme de modèle MVC ou parfois référencée comme une architecture en trois couches, comprenant :

* **Modèle** : Représente la couche de données de l'application. Dans notre cas, il correspond aux objets et mécanismes de gestion de la base de données MongoDB, manipulés à travers des classes de modèles qui reflètent les structures de données et les logiques métiers.
* **Repository** : Agit comme une couche d'abstraction entre le modèle et la source de données. Les repositories facilitent les interactions avec la base de données pour récupérer, stocker et mettre à jour les données nécessaires.
* **Service** : Constitue la logique métier de l'application. Les services définissent les règles métiers et les opérations complexes en utilisant les données fournies par les repositories. Cette couche s'assure que les demandes du frontend sont traitées efficacement et correctement.
* **Contrôleur** : Sert de lien entre le frontend et le backend, gérant les interactions utilisateur. Il reçoit les requêtes du frontend, les dirige vers les services appropriés pour traitement, et renvoie les réponses adéquates.

Cette structuration permet non seulement une séparation claire et une gestion efficace des responsabilités dans notre backend, mais elle assure aussi une maintenabilité et une évolutivité optimales du système.

**Déploiement :**

L'une des étapes finales de notre projet consistait à déployer notre application pour permettre une utilisation en conditions réelles. Pour ce faire, nous avons choisi d'utiliser Render, une plateforme de déploiement d'applications offrant un service gratuit, ce qui représentait un avantage significatif pour notre projet.

Render se distingue par sa simplicité d'utilisation, permettant une mise en œuvre rapide et efficace sans nécessiter de connaissances approfondies en configuration de serveurs. Cette accessibilité nous a permis de concentrer nos efforts sur le développement de l'application plutôt que sur les détails techniques du déploiement.

Cependant, nous avons rencontré certains défis avec Render, notamment en ce qui concerne les performances. Nous avons observé une latence prolongée dans les réponses aux requêtes HTTP, ce qui pourrait impacter l'expérience utilisateur. Bien que cet inconvénient ne compromette pas la fonctionnalité globale de l'application, il est crucial de considérer cet aspect pour les futures itérations ou pour un déploiement à plus grande échelle.

### Conception Frontend

Pour la conception du frontend de notre application, la technologie à utiliser nous a été imposée. Nous devions employer Kotlin Multiplatform Mobile (KMM) avec Jetpack, un choix dicté par nos tuteurs qui utilisent cette technologie dans leur cadre professionnel.

**Choix de Kotlin Multiplatform Mobile :**

Le choix de Kotlin Multiplatform par CATS (Crédit Agricole Technologies et Services) se justifie pour plusieurs raisons. Tout d'abord, Kotlin est le langage officiellement soutenu par Google pour le développement Android, et il est également bien adapté au développement iOS, offrant ainsi une approche cross-platform efficace. Cela permet aux développeurs spécialisés en Android ou iOS de travailler avec une base de code commune, réduisant le besoin de double développement pour les deux plateformes.

**Avantages de Kotlin Multiplatform :**

Kotlin Multiplatform suit l'architecture basique de la plupart des projets mobiles, ce qui simplifie la formation pour les développeurs déjà familiarisés avec le développement Android ou iOS. La logique réseau et la structure du code sont similaires entre les plateformes, ce qui facilite la maintenance et l'évolution de l'application. De plus, Kotlin permet d'intégrer des fonctionnalités natives de Swift ou Android grâce à l'utilisation de bibliothèques dédiées, offrant ainsi la possibilité de développer des fonctionnalités spécifiques à une plateforme de manière simple.

Un autre avantage significatif de Kotlin Multiplatform est son soutien par JetBrains, une entreprise reconnue pour son engagement envers le développement d'outils robustes pour les développeurs. Ce soutien garantit que Kotlin continuera à évoluer et à s'adapter aux nouvelles exigences du développement mobile. De plus, de nombreux grands acteurs du secteur technologique adoptent Kotlin, ce qui confirme la viabilité et la pertinence de cette technologie pour des applications professionnelles et à grande échelle.

**Organisation du Projet avec Kotlin Multiplatform :**

Notre projet est structuré en trois parties principales conformément à l'architecture Kotlin Multiplatform :

* **iOSApp** et **AndroidApp** : Ces modules contiennent des éléments spécifiques à chaque plateforme. Cela inclut l'intégration de bibliothèques natives et de fonctionnalités spécifiques qui ne sont pas partagées entre les plateformes.
* **SharedApp** : Ce module centralise toute la logique commune, y compris la gestion des données, la logique métier, et les interactions réseau. La réutilisation de ce code commun entre les plateformes iOS et Android minimise les efforts de développement et maintient la cohérence fonctionnelle à travers les différentes versions de l'application.

**Choix de la conception :**

Pour notre application mobile, nous avons opté pour l'architecture MVVM (Model-View-View Model), qui est particulièrement adaptée à Kotlin et aux environnements de développement modernes comme Jetpack Compose. Ce choix offre plusieurs avantages significatifs :

**1. Séparation des préoccupations**

L'un des principaux avantages de l'architecture MVVM est la claire séparation des préoccupations qu'elle propose :

* **Modèle (Model)** : Gère les données et la logique métier.
* **Vue (View)** : Présente les données (l'interface utilisateur) et déclenche des événements au modèle de vue.
* **Modèle de Vue (ViewModel)** : Agit comme un intermédiaire entre le modèle et la vue, gérant la logique de présentation et les états de l'interface utilisateur.

Cette séparation facilite la maintenance et le test des différentes composantes de l'application, puisque chaque partie peut être développée et testée indépendamment des autres.

**2. Facilitation des tests**

Avec MVVM, il est plus facile de tester les composantes de l'application car le ViewModel ne dépend pas directement de la vue. Cela permet de tester la logique de présentation sans nécessiter une interface utilisateur, ce qui peut considérablement accélérer le développement et garantir une plus grande qualité du code.

**3. Réutilisation du code**

Le ViewModel dans MVVM peut souvent être réutilisé entre les plateformes lorsqu'il est combiné avec des solutions comme Kotlin Multiplatform Mobile. Cela réduit le besoin de dupliquer la logique de présentation pour Android et iOS, minimisant ainsi les efforts de développement et les risques d'erreurs.

## Résultat

### Résultat Intermédiaire

Tout au long de ce projet, nous avons mis en place un système de suivi rigoureux pour assurer la transparence et la continuité dans nos avancements. Ce système comprenait des rapports préliminaires, hebdomadaires, et un rapport final.

**1. Rapport Préliminaire**

Dès le début du projet, nous avons établi les fondations de notre planification :

* **Gantt prévisionnel (voir Annexe 1)** : Ce diagramme de Gantt initial a servi à définir les étapes clés du projet, les délais associés, et la répartition des tâches. Il a été un outil essentiel pour visualiser la progression du projet et pour anticiper les besoins en ressources et en temps.
* **Lettre de mission** : Ce document a formalisé l'objectif du projet, les attentes des tuteurs, et les responsabilités de chaque membre de l'équipe. Elle a servi de référence tout au long du projet pour aligner nos efforts avec les objectifs fixés.

**2. Rapports Hebdomadaires**

Pour maintenir une dynamique de projet efficace et réactive, nous avons instauré des réunions hebdomadaires chaque vendredi de 14h à 15h. Ces réunions étaient l'occasion de:

* **Présenter les minutes** : Documenter les discussions, les décisions prises, et les actions à entreprendre, assurant ainsi la continuité entre les différentes sessions de travail.
* **Présenter les travaux accomplis durant la semaine** : Chaque membre de l'équipe partageait ses progrès, permettant ainsi une évaluation continue de l'avancement du projet et l'ajustement rapide des plans en fonction des résultats obtenus et des défis rencontrés.

**3. Rapport Final**

Le point culminant de notre projet a été la présentation du produit final :

**Rendu final de l'application** : Cette étape a consisté à démontrer le fonctionnement de l'application, à valider les fonctionnalités développées par rapport aux exigences initiales, et à recevoir les retours des tuteurs et autres parties prenantes. Le produit final reflétait l'intégration des différentes composantes développées et testées tout au long du projet.

### Résultat final

*Description fonctionnelle de la solution, tests réalisés, et la livraison de la solution au demandeur)*

# Problème et solutions

# Gestion du projet

## Méthodologies utilisées

## Bonnes pratiques

*Bonnes pratiques de résolution de problèmes et de communication*

# Analyse critique

## Analyse des choix techniques et organisationnels

*Reflexions sur ce qui pourrait être amélioré ou fait différemment*

## Intégration des préoccupations environnementales

*Comment l’écoconception et la RSE ont été intégrées dans le projet*

# Expérience tiré

## Valorisation du travail

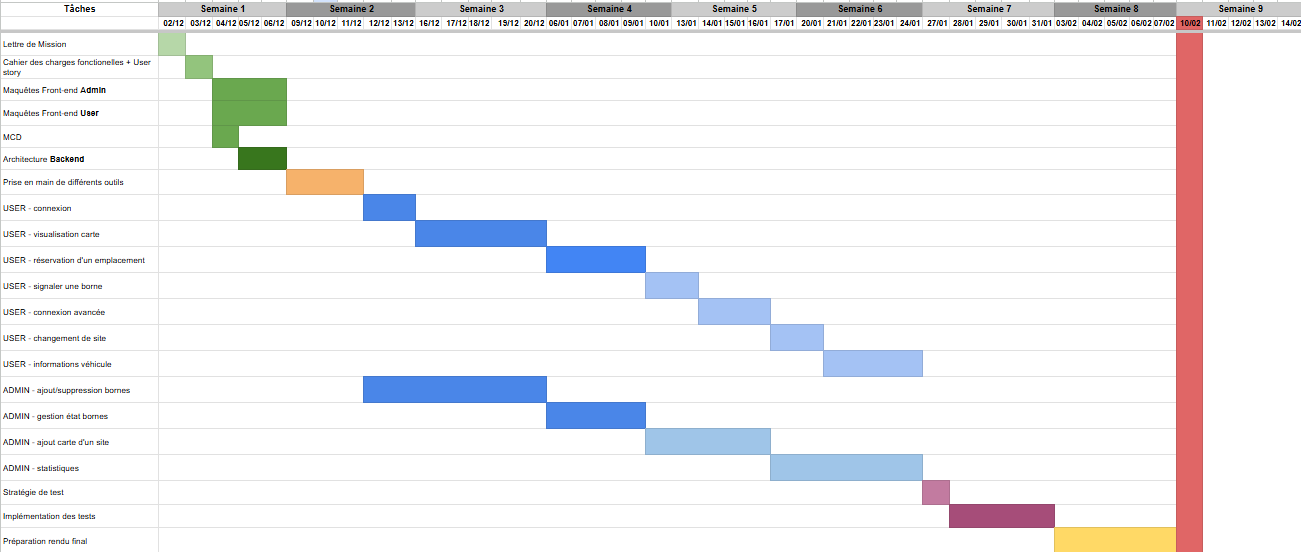
*Valorisation du travail (Tes apports au demandeur, contribution à la résolution du problème « métier », et tes apprentissages)*

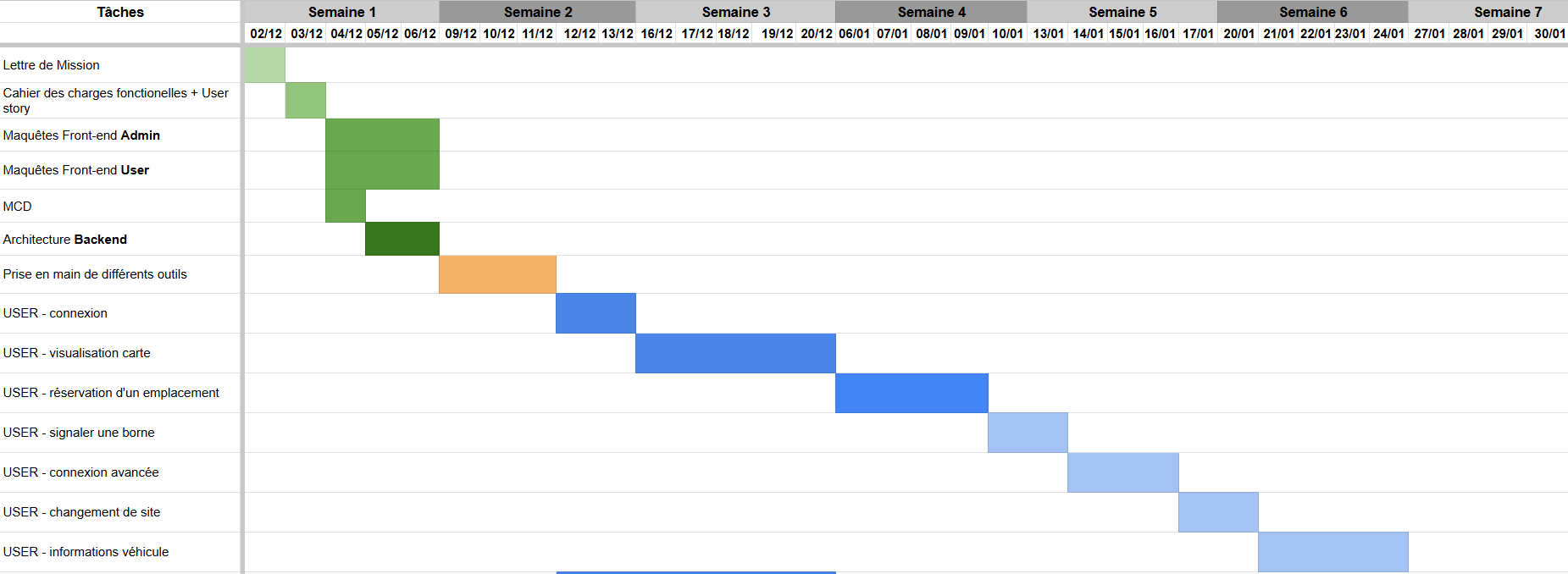
## Compétences techniques et soft skills développées

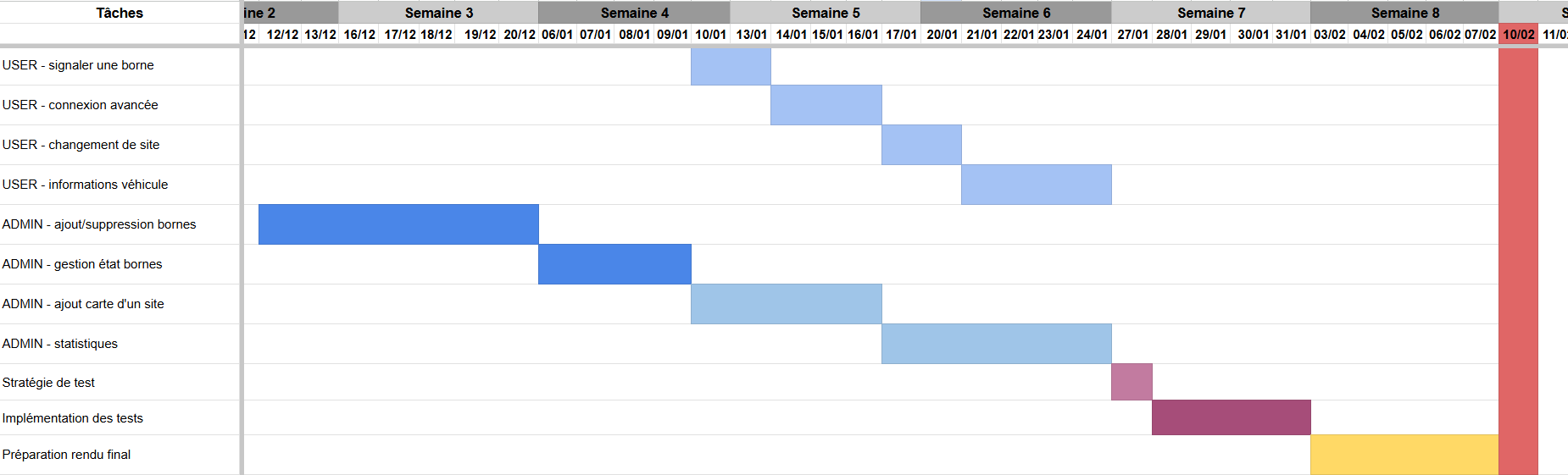
*Compétences techniques et soft skills développées*

# Annexes

***Annexe 1: Diagramme de Gantt prévisionnel***

****

****

****

# Résumés

## En français

## En anglais