# INFORMATIONS GENERALES

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Candidat** | Nom : | Sesti | Prénom : | Téo |
|  |  |  |  | +41 79 963 04 97 |
| **Lieu de travail :** | Renens | | | |
| **Orientation :** | 88601 Développement d’applications  88602 Informatique d’entreprise  88603 Technique des systèmes | | | |
| **Chef de projet** | Nom : | Fresco | Prénom : | Alain |
| alain.fresco@wavemind.ch | |  +41 79 610 36 43 | |
| **Expert 1** | Nom : | Gilbert | Prénom : | Gruaz |
| gilbert.gruaz@eduvaud.ch | |  +41 21 316 02 69 | |
| **Expert 2** | Nom : | Nicolas | Prénom : | Borboen |
| gilbert.gruaz@eduvaud.ch | | +41 21 693 54 55 | |
| **Période de réalisation :** | 01 mars 2021 - 15 mars 2021 | | | |
| **Horaire de travail :** | Lundi, mardi, mercredi, jeudi, vendredi : 8h-12h 13h-17h35 | | | |
| **Nombre d'heures :** | 88 heures | | | |
| **Planning (en H ou %)** | Analyse : 20 % | | | |
| Implémentation : 40 % | | | |
| Tests : 15 % | | | |
| Documentations : 25 % | | | |

# PROCÉDURE

* Le candidat réalise un travail personnel sur la base d'un cahier des charges reçu le 1er jour.
* Le cahier des charges est approuvé par les deux experts. Il est en outre présenté, commenté et discuté avec le candidat. Par sa signature, le candidat accepte le travail proposé.
* Le candidat a connaissance de la feuille d’évaluation avant de débuter le travail.
* Le candidat est entièrement responsable de la sécurité de ses données.
* En cas de problèmes graves, le candidat avertit au plus vite les deux experts et son CdP.
* Le candidat a la possibilité d’obtenir de l’aide, mais doit le mentionner dans son dossier.
* A la fin du délai imparti pour la réalisation du TPI, le candidat doit transmettre par courrier électronique le dossier de projet aux deux experts et au chef de projet. En parallèle, une copie papier du rapport doit être fournie sans délai en trois exemplaires (L’un des deux experts peut demander à ne recevoir que la version électronique du dossier). Cette dernière doit être en tout point identique à la version électronique.

# TITRE

Chibre Manager

# MATÉRIEL ET LOGICIEL À DISPOSITION

1 Ordinateur Windows

1 License RubyMine

1 Suite Office

# PRÉREQUIS

- Posséder un ordinateur Windows fonctionnel

- Les logiciels nécessaires sont installés

- Maîtrise de l’environnement RubyMine

- Bonnes connaissances de la programmation JavaScript (React) et Ruby

# DESCRIPTIF DU PROJET

L’objectif du projet est de developper “from scratch” une application mobile en utilisant React Native. Cette application doit permettre aux utilisateurs de gérer des parties de jass (chibre). Toutes les données devront être stockées et gérées dans un serveur développé en Ruby on Rails mettant à disposition une API utilisable par l’application mobile.

La liste exhaustive des règles est disponible à l’adresse suivante : [https://fr.wikipedia.org/wiki/Jass#R%C3%A8gles\_du\_chibre](https://fr.wikipedia.org/wiki/Jass" \l "Règles_du_chibre)

Voici les fonctionnalités de base qui doivent figurer dans l’application.

**1. Création d’une partie**

Voici les informations qui doivent être définies lors de la création d’une partie

* Définir le nom des quatre joueurs de la partie
* Définir quels joueurs sont en équipe
* Définir quel joueur commence

**2. Durant la partie**

Lorsqu’une partie est en cours l’utilisateur peut à tout moment avoir accès aux informations suivantes.

* Nombre de point pour chaque équipe
* Quel joueur a annoncé l’atout pour la manche en cours
* Quel est l’atout pour la manche en cours
* Quel joueur a distribué les cartes
* Quelles sont les annonces pour la manche en cours s’il y en a

**3. Gestion des manches**

Chaque partie est constituée d’une série de manches, il faut donc pouvoir créer des manches pour une partie spécifique. Voici les fonctionnalités liées à une manche

* L’application doit nous annoncer quel joueur doit distribuer les cartes
* L’application doit nous annoncer quel joueur doit faire atout. Avec la possibilité d’annoncer si le jouer à chibrer ou non
* L’utilisateur doit pouvoir définir le nombre de point pour une équipe, l’application va automatiquement calculer le nombre de point pour l’équipe adverse.
* L’utilisateur doit pouvoir attribuer des annonces à un joueur.
* Possibilité de saisir les annonces (ci-dessous la liste des annonces possibles)
  + **3 cartes**: 20 points
  + **4 cartes**: 50 points
  + **5 cartes**: 100 points
  + **Carré**
    - **10 / Dames / Roi / As**: 100 points
    - **9**: 150 points
    - **Valet**: 200 points
  + **Mit stöck**: 20 points

Les points attribués ainsi que les annonces d’une manche doivent pouvoir être modifiés durant une partie.

L’application doit automatiquement annoncer à l’utilisateur lorsque une des équipe dépasse le seuil de victoire (1000 ou 1500 points)

**4. Liste des parties**

L’utilsateur doit avoir accès à l’historique de toutes les parties, ainsi que reprendre une partie qui est n’est pas terminée.

**5. Informations générales**

* Pour le backend les données doivent être stockées dans une base de données Postgres.
* Il n’est pas nécessaire de mettre en place un système d’authentification. Toutes les routes de l’api sont ouvertes.

# LIVRABLES

Le candidat est responsable de livrer à son chef de projet et aux deux experts :

1. Rapport de projet
2. Journal de travail
3. Un fichier archive contenant
   1. Le code source de l’application mobile
   2. Le code source du serveur
   3. Le manuel utilisateur
   4. Le guide d’installation
4. Un lien vers l’application héberger en sur heroku
5. Un lien vers le repository github.

# POINTS TECHNIQUES ÉVALUÉS SPÉCIFIQUES AU PROJET

La grille d’évaluation définit les critères généraux selon lesquels le travail du candidat sera évalué (documentation, journal de travail, respect des normes, qualité, …).

En plus de cela, le travail sera évalué sur les 7 points spécifiques suivants (Point A14 à A20) :

1. La qualité du repository Git : messages de commits explicites et lisibles, permettant de retracer l'évolution du code (plusieurs commits par jour, création de branches de fonctionnalités), fichier README.md présentant le projet et son déploiement.
2. Le code suis le principe de DRY (Don’t Repeat Yourself), ce qui implque un code exempt de sections dupliquées et en respectant le style de programmation des langages utilisés (Rubocop et ESLint).
3. Guide d’installation précis et reproductible (ReadMe)
4. Les différentes méthodes HTTP sont implémentées à bon escient en fonction de l'action réalisée sur la ressource indiquée. Les codes de réponse HTTP utilisés permettent aux clients d'avoir une information sur le resultat de leurs requêtes. ( voir: <https://developer.mozilla.org/fr/docs/Web/HTTP/Status>)
5. Le rapport démontre que le candidat a étudié le modèle des données : un diagramme entité-association (ERD) est présent dans le rapport. Le candidat décrit et critique le diagramme et les différentes tables.
6. L'étudiant fourni unte documentation de l'API, qui explique les types de données, les valeurs de retour, les différentes possibilités d'interactions avec l'API. L'utilisation d’outil tel que Swagger est recommandée.
7. Du côté mobile l’application doit gérer proprement les erreurs du serveurs aurant dans le cas où le serveur ne serait pas disponible que dans le cas ou le serveur renderais une erreur. Il faut donc éviter au moins les evenments suivants crash de l’application, WSOD, ecran de chargement infini.

# VALIDATION

|  |  |  |
| --- | --- | --- |
|  | Lu et approuvé le : | Signature : |
| Candidat : |  |  |
| Expert n°1 : |  |  |
| Expert n° 2 : |  |  |
| Chef de projet : |  |  |