1) EXPRESSION DU BESOIN

Le produit demandé consiste en plusieurs applications web. Il s'agit d'un jeu géolocalisé dans l'environnement réel urbain.

On distinguera
- l'application web pour les joueurs, dédiée mobile,
- l'application web pour le gestionnaire du jeu, dédiée desktop

Principe du jeu :

Les joueurs doivent s'approprier des lieux sur la carte, pour ensuite gagner des points chaque fois qu'un autre joueur passe dans ou à proximité d'un des lieux qui leur appartient. Il s'agit en sorte d'une conquête de territoires type « Monopoly ». Plus on « possède » de lieux, plus on a de chance de gagner des points.

Dans le détail :

Les joueurs partent tous avec un même capital de points (30 points)

Chaque lieu a une « valeur » en points qu'il faut dépenser pour pouvoir l'acquérir. La valeur peut changer selon les lieux (en fonction de leur potentiel en gain de points etc.).

La carte doit afficher les lieux qui n'appartiennent à personne ainsi que les lieux déjà conquis par d'autres joueurs (l'affichage doit être différencié). Mais concernant les lieux « libres » (n'appartenant à personne), le joueur ne voit que ceux dont le prix d'acquisition (en points) est inférieur ou égal aux points dont il dispose : ainsi le joueur doit progresser dans le temps pour découvrir de nouveaux lieux disponibles.

De plus le joueur ne peut voir que les lieux qui sont dans un rayon de 30km de sa position.
Exception : un joueur peut voir tous les lieux qu'il possède, peu importe leur emplacement dans le monde.

Un clic sur un des lieux marqués sur la carte permet de savoir quel est son prix d'acquisition (dans le cas d'un lieu libre) ou de savoir qui est le joueur qui en est propriétaire (dans le cas contraire). Ces informations seront présentées dans une info bulle. Dans le deux cas il serait utile de pouvoir avoir également photo du lieu et adresse.

Pour prendre possession d'un lieu, un joueur doit se trouver dans un périmètre de 50m du lieu en question et appuyer sur un bouton dédié sur l'application. (puis confirmer qu'il accepte de payer les points nécessaires).

Un joueur peut libérer un lieu qu'il possède, sans avoir besoin de s'y rendre. Il est alors re-crédité d'un montant de points égal à 75% du prix d'acquisition initial du lieu.

Chaque fois qu'un joueur se trouve dans un périmètre de 50m d'un lieu possédé par un autre joueur, le joueur propriétaire du lieu gagne un point. Note : Le joueur qui visite le lieu ne perds aucun point (c'est-à-dire : les points ne sont pas prélevés à l'un pour donner à l'autre).

En temps réel le joueur doit pouvoir connaitre son classement dans le jeu (celui qui a le plus de points est classé premier).

Gestionnaire du jeu :

Il est le « maître du jeu ». Il dispose d'une application web desktop (page web dynamique) qui lui permet les actions suivantes :

- Créer de nouveaux lieux dans le jeu (position, valeur en points …)
- Supprimer un lieu
- Modifier la valeur en points d'un lieu
- Bannir un joueur du jeu

Il doit avoir également pouvoir consulter des statiques

- Classement en temps réel des joueurs
- Nombre de lieux libres
- Nombre de lieux occupés par chaque joueur
- Statistiques (graphes) sur la prises de possession des lieux dans le temps

2) ENVIRONNEMENT TECHNIQUE

Le projet est en environnement 100% full web.

Pour l'application utilisateur joueur :
Il n'est pas question ici de développer un application native Android ou IOS ou WindowsPhone : Le principe d'une WebApp a été retenu. Cela signifie une application web basée sur HTML + Javascript + JQuery Mobile ou MaterializeJS ou … pour une ergonomie optimisée mobile (look & feel smartphone, design responsive). L'intérêt est ainsi de pouvoir fonctionner sur toutes les plateformes.

L'application backoffice administrateur du jeu sera en web HTML/CSS version desktop.

Le serveur sera constitué de scripts PHP :

- services web pour l'application joueur
- pages web dynamiques pour l'application administrateur du jeu

Ces scripts seront exécutés sous Apache.
Les données seront stockées dans une base de données sur un serveur MySQL.

On utilisera au maximum les possibilités d'Ajax pour fluidifier les échanges client/serveur et optimiser l'ergonomie (pas de rechargements de l'écran du joueur !)

3) LIVRABLES /TRAVAIL DEMANDE

ANALYSE FONCTIONNELLE / SPECIFICATION

1/ fournir un **diagramme des use cases de l'application**

2/ fournir **la liste des scénarios** (liste la plus exhaustive possible), pour chaque use case.

- sous forme de texte avec juste un nom (titre de scenario) et un indicateur de succès (OK) ou échec (KO)
- *le but est de déterminer tous les cas possibles, notamment tous les cas "particuliers" qui vont demander un algorithme ou des vérifications ou des traitements supplémentaires. (exemple : que se passe t il si plusieurs personnes signalent le même incident ? => il ne faut pas le récréer plusieurs fois dans la bdd)*
- *ex de proposition de nomenclature :*
    - *"UC n° 1 - Scenario n° 1 – utilisateur créé avec succès OK"*
    - *"UC n° 1 - Scenario n° 2 – utilisateur non créé car login manquant - KO"*

3/ Décrire les scénarios les plus complexes (seulement ceux là) sous forme de **diagramme de séquence.** Les diagrammes de séquence feront notamment apparaitre les entités : IHM (interface graphique) client / logique coté client / la base de données coté client / la logique coté serveur / la base de données coté serveur.

4/ Proposer un **découpage du projet en "lots"**, accompagné d'un planning prévisionnel.
Chaque lot regroupant un ou plusieurs use cases (ou un ou plusieurs scénarios de use cases) : en d'autres termes, indiquer dans quel ordre (par quel bout) vous pensez aborder le développement du produit.

5/ proposer une première conception (diagramme de classes UML ou MCD Merise) pour la **base de données**

Pour la fin du projet

1/ une application en état de marche, déployée par FTP sur un serveur d'accès public
2/ le code source avec l'arborescence des fichiers, sous forme de fichier zip (ou équivalent)

Soutenance orale

Le projet se conclut par une soutenance orale illustrée d'une présentation PowerPoint durant laquelle seront présentés et argumentés

- les choix techniques
- les choix fonctionnels (comment avez-vous décidé de traiter les cas particuliers)
- les moyens mis en œuvre pour les tests
- un planning prévisionnel et réel, avec analyse des écarts
- votre organisation de travail en binôme (partage du code source, travail collaboratif, outils et méthodologie)
- les éventuelles améliorations que vous avez apporté au besoin initial
- les problèmes rencontrés et solutions adoptées pour y faire face

Durée de la soutenance orale = 45 minutes (30 min de présentation + démo , 15 minutes de questions réponses)