Cahier des Charges application SolarisScanner

1. Présentation Générale

1.1. Contexte

Le cinéma Solaris souhaite développer une application mobile permettant la validation/le compostage des réservations clients, lors de leur arrivée au cinéma. Cette application visera exclusivement les appareils Android. Elle interagira avec la partie Back-end du site web du cinéma (**cinemasolaris.fr**), qui gèrera l'interaction directe avec la base de données MySQL.

1.2. Objectif

L'application doit permettre aux agents du cinéma de **vérifier l'intégrité des réservations client** via un **scan de QR code** (que le client aura reçu au moment de sa réservation), et de les valider si tout est en ordre, permettant ainsi au client d'accéder à la séance. Son usage devra être restreint aux employés du cinéma, nécessitant donc la mise en place d'une authentification fiable.

1.3. Fonctionnalités principales

- Scan de QR Code de réservation.
- Connexion sécurisée via une API qui contactera le back-end du site cinemasolaris.fr
- Vérification et mise à jour de la réservation via cette même API.
- Affichage des résultats du scan, en fonction du degré de validité du QR code soumis.

2. Spécifications Fonctionnelles

L'application sera épurée et comportera 3 pages :

- Page de connexion (point d'entrée initial de l'application)
- Page de Scan (point d'entrée de l'application une fois l'utilisateur authentifié)
- Page de résultat

2.1. Page de connexion

- Champs : Login et Mot de passe, ces identifiants seront identiques à ceux requis pour la connexion sur le site web cinemasolaris.fr.
- Envoi des informations via **requête HTTP POST** au backend Laravel.
- Mot de passe chiffré avant envoi.
- Le serveur vérifiera l'existence de l'utilisateur et ses droits.
- Retour : **Token d'authentification** à utiliser pour les requêtes suivantes et comme témoin de connexion.

Lorsque la connexion d'un utilisateur est réussi, le token d'authentification sera stockée dans la mémoire sécurisée de l'appareil, évitant d'avoir à se reconnecter à chaque réouverture de l'application, **dans un laps de temps précis à déterminer.**

2.2 Page de Scan

- Interface de caméra épurée avec un **bouton central** pour activer le scan.
- Deux options activables/désactivables envisagées :
  - Activation du flash.
  - **Activation de l'autoscan** (scan en continu sans appui sur le bouton central).
- Redirection vers la page de résultat lorsqu'un scan est effectué, avec la « valeur » du QR code scannée.

2.3 Page de Résultat du Scan

- Envoi d'une requête HTTP POST avec :
- Token d'authentification dans le header de requête.
  - **ID de réservation** (« valeur » du scan transmise par la page de scan).
  - Date/heure du scan.
- Le serveur devra effectuer les vérifications d'intégrité de la séance, en tenant compte des cas d'erreurs suivants :
  - La réservation existe-t-elle ?
  - Est-elle toujours valide ?
  - La séance est-elle proche du moment du scan ?
  - La réservation a-t-elle déjà été scannée ?
- Le serveur renvoie les informations suivantes si la réservation est valide (ou un message d'erreur adaptée à l'erreur rencontrée) :
  - Titre du film.
  - Numéro de salle (pour pouvoir orienter les clients dans le cinéma).
  - Nombre de places réservées (pour vérifier que le nombre de personnes correspond aux nombre de sièges réservés).
- Affichage des informations ou d'un message d'erreur en cas de problème.

3. Spécifications Techniques

3.1. Technologies utilisées

- Application Mobile : .NET MAUI (C#)
- Back-end côté serveur : Laravel (PHP)
- Communication Application/Serveur : API REST
- Sécurisation de l'API : Laravel Sanctum
- Base de données : MySQL (héritée de cinemasolaris.fr)
- **Scan de QR Code** : Bibliothèque .NET MAUI **BarcodeScanning.**

3.2. Sécurité

- Authentification via identifiants et Bearer **Token**.
- Chiffrement du mot de passe avant envoi.
- Contrôle des droits d'accès et/ou intégrité du token sur Laravel.

4. Validation et Tests

- Tests via Postman pour assurer le bon fonctionnement des requêtes API.
- Tests UI sur appareil android pour vérifier l'ergonomie et la fluidité de l'application.