Documentation

## **1. Introduction**

### 1.1 Objectif du document

### 1.2 Contexte du projet

### 1.3 Portée du projet

## **2. Description du système**

### 2.1 Vue d'ensemble du système

### 2.2 Diagramme d'architecture du système

### 2.3 Description détaillée des composants du système

## **3. Spécifications du système**

### 3.1 Spécifications fonctionnelles

### 3.2 Spécifications non fonctionnelles (sécurité, performance, etc.)

## **4. Plan de mise en œuvre**

### 4.1 Calendrier du projet

### 4.2 Ressources nécessaires

### 4.3 Risques et atténuation

## **5. Test et validation**

### 5.1 Stratégie de test

### 5.2 Plan de validation

## **6. Maintenance et support**

### 6.1 Plan de maintenance

### 6.2 Support technique

## **7. Conclusion**

### 7.1 Résumé

### 7.2 Prochaines étapes

## Introduction

### 1.1 Objectif du document

L'objectif de ce document est de fournir une description détaillée du projet de contrôle des panneaux d'affichage. Il décrit le contexte, les spécifications, le plan de mise en œuvre, les stratégies de test et de validation, et le plan de maintenance et de support pour le système. Le document servira de référence pour toutes les parties prenantes du projet, y compris les équipes de développement, de test, de gestion de projet, ainsi que les utilisateurs finaux.

### 1.2 Contexte du projet

Le projet intervient dans un contexte plus global de modernisation. La partie concernée par ce projet est le système d'affichage qui vise à informer le public. Actuellement, le système envoie automatiquement trois informations deux fois par jour : le débit entrant, le débit sortant et le niveau de l'eau. L'objectif est de reproduire ce fonctionnement avec une communication GPRS / 4G aussi sécurisée que possible, tout en permettant l'envoi d'images.

### 1.3 Portée du projet

Le projet comprend le développement de l'application web, des applications embarquées, la configuration du serveur, la mise en place de mesures de sécurité appropriées, le test du système et le déploiement de l'application sur le serveur et les appareils Asus TinkerBoard. Le système sera capable de basculer entre trois modes d'affichage : affichage d'image, affichage de données et boucle d'affichage. Le système sera également capable de planifier l'envoi de données et d'envoyer des images pour des messages de communication personnalisés ou pour l'affichage de médias.

### **2. Description du système**

#### 2.1 Vue d'ensemble du système

Le système est conçu pour fournir une solution d'affichage robuste et sécurisée pour un barrage. Il se compose de plusieurs composants clés qui interagissent pour fournir des fonctionnalités d'affichage dynamique et sécurisé. Ces composants incluent des panneaux d'affichage, un serveur dédié, une application web, un réseau VPN et plusieurs mesures de sécurité.

#### 2.2 Diagramme d'architecture du système

**A diagram of a computer

Description automatically generated with low confidence**

#### Description détaillée des composants du système

* **Application web** : L'application web est conçue en utilisant la pile technologique MERN qui se compose de MongoDB, Express.js, React.js et Node.js.
  + **MongoDB** : C'est le système de gestion de base de données utilisé. Il est responsable du stockage des instructions destinées aux panneaux d'affichage ainsi que des identifiants et des mots de passe des utilisateurs.
  + **Express.js** : Il s'agit d'un Framework de serveur web basé sur Node.js, utilisé pour construire le backend de l'application web.
  + **React.js** : C'est une bibliothèque JavaScript pour la construction d'interfaces utilisateur. Il est utilisé pour développer le frontend de l'application web, offrant une expérience utilisateur interactive.
  + **Node.js** : C'est un environnement d'exécution côté serveur pour JavaScript. Il est utilisé pour exécuter l'application web.
* **Serveur** : Le serveur du système tourne sur le système d'exploitation Debian. Ce système d'exploitation a été choisi en raison de sa robustesse, de sa fiabilité et de son excellente sécurité. Le serveur héberge la base de données MongoDB ainsi que l'application web.
* **VPN** : Le système utilise un réseau VPN pour sécuriser la communication entre le serveur et les panneaux d'affichage. Cela assure que les informations sensibles ne sont pas exposées et que le système est à l'abri des attaques externes.
* **Panneaux d'affichage** : Il y a deux panneaux d'affichage qui sont équipés d'un routeur 4G et d'une Asus TinkerBoard. Ces panneaux affichent des informations sur le barrage, notamment le débit entrant, le débit sortant et le niveau de l'eau, ainsi que des messages de communication personnalisés. Les panneaux d'affichage sont pilotés par des TinkerBoard qui exécutent des applications développées à l'aide du Framework Electron.js.
* Application embarquée : Cette application, développée sous Electronjs, est chargée de gérer l'affichage des informations sur les panneaux. Electron.js permet de construire des applications de bureau robustes en utilisant des technologies web standard comme JavaScript, HTML et CSS.
* **Communication** : La communication entre l'application web et les panneaux d'affichage est réalisée en utilisant le protocole WebSocket Secure (WSS). Le WSS offre une communication bidirectionnelle en temps réel entre le client et le serveur sur une connexion sécurisée. La sécurisation de la connexion est garantie par l'emploi d'un réseau VPN.
* **Mesures de sécurité** : Diverses mesures de sécurité sont mises en place pour assurer la protection des données et la stabilité du système :
  + **Pare-feu UFW** : Il est utilisé pour limiter l'accès au serveur, en permettant uniquement les communications entrantes sur le port 443.
  + **Certificat SSL** : Il est utilisé pour crypter la connexion entre le serveur et les clients, assurant ainsi la confidentialité et l'intégrité des données échangées.
  + **Reverse Proxy Nginx** : Il est configuré avec SSL/TLS pour fournir une couche supplémentaire de sécurité, notamment en bloquant l'accès par IP non autorisées.
  + **Authentification par jeton d'accès** : Elle est mise en œuvre pour sécuriser la communication entre les panneaux d'affichage et le serveur.
* **Planification :** Un outil de planification, Cron, est utilisé pour programmer l'envoi de données aux panneaux d'affichage à des moments précis, permettant ainsi une diffusion régulière et fiable des informations.

**3. Spécifications du système**3.1 Spécifications fonctionnelles

* Authentification : L'application doit permettre à un utilisateur de se connecter en utilisant des identifiants sécurisés.
* Configuration de l’affichage : Une fois connecté, l'utilisateur doit pouvoir configurer l'affichage des deux panneaux. Les options de configuration devraient inclure les données fournies par le barrage (débits et cote) ainsi que la possibilité d'ajouter des images pour des communications ou des médias.
* Envoi de données : L'application doit être capable d'envoyer les informations configurées aux deux panneaux via une connexion sécurisée.

3.2 Spécifications non fonctionnelles

* Performance : Bien que l'application ne soit destinée qu'à un utilisateur simultané maximum, elle doit être optimisée pour minimiser l'utilisation des données et éviter des temps de réponse excessifs. Cela est particulièrement important compte tenu du fait que la communication se fait via la 4G.
* Sécurité : L'application doit être hautement sécurisée pour prévenir toute attaque potentielle. Les mesures de sécurité devraient inclure l'utilisation d'un VPN, l'obtention d'un certificat SSL d'une autorité de certification, l'utilisation d'un pare-feu, et d'autres mesures appropriées.
* Maintenabilité : L'application doit être conçue de manière à faciliter la maintenance à long terme.

4. Plan de mise en œuvre

* 1. Calendrier du projet
* Du 20 mars au 19 mai : Définition des spécifications détaillées, conception de l'architecture du système et modélisation des bases de données.
* Du 20 mai au 30 juin : Développement de l'application web et des interfaces de la base de données MongoDB.
* Du 1 juillet au 20 juillet : Développement de l'application Electron.js pour les Tinkerboards.
* Du 21 juillet au 31 Juillet : Intégration de toutes les composantes du système, tests et ajustements.
* Du 1er Août au 14 août : Tests de sécurité, optimisations et déploiement du VPN, pare-feu et autres mesures de sécurité.
* À partir du 15 Août : Tests finaux, ajustements et déploiement.
  1. Ressources nécessaires
* Une machine serveur pour héberger l'application web et la base de données MongoDB.
* Deux Tinkerboards pour gérer les affichages.
* Deux routeurs 4G pour la communication avec les panneaux d'affichage.
* Des outils de développement pour la stack MERN, Electron.js et les autres technologies impliquées.
  1. Risques et atténuation
* Retards dans le développement : Un planning détaillé et une gestion efficace du temps peuvent aider à éviter ce risque. Il serait également utile de commencer par les tâches les plus critiques.
* Problèmes de sécurité : Des tests de sécurité réguliers, l'utilisation de meilleures pratiques de sécurité et la mise en place de mesures de sécurité robustes (VPN, pare-feu, etc.) peuvent aider à atténuer ce risque.
* Problèmes de performance : Des tests de performance réguliers et une optimisation continue peuvent aider à prévenir les problèmes de performance.
* Difficultés techniques avec les Tinkerboards ou les routeurs 4G : Avoir des équipements de rechange à disposition et obtenir de l'aide technique en cas de besoin peut aider à atténuer ce risque.