**Université Hassan I**

**Faculté des Sciences et Techniques**

**Département Informatique**

**Mémoire de Projet de Fin d’études**

Présenté en vue de l’obtention du Cycle master en Sciences et Techniques option réseaux et systèmes informatiques (RSI)

Sujet : Affichage Dynamique

Réalisé par : Encadré par :

M. LAFRAOUZI Mouhssine Prof.

Soutenu le xx juin xx devant les membres de jury :

Prof. XX ……………………………………………Professeur à la FST Settat

Prof. XX ……………………………………………Professeur à la FST Settat

Prof. XX ……………………………………………Professeur à la FST Settat



Dédicace

À MES CHERS PARENTS

Je dédie ce modeste travail et mes profondes gratitudes à mes chers parents pour l'éducation qu'ils m'ont prodiguée, avec tous les moyens et au prix de tous les sacrifices qu'ils ont consentis à mon égard, pour le sens du devoir qu'ils m'ont enseigné depuis mon enfance.

À MA FAMILLE PROCHE

J'espère sincèrement que ce travail est une façon de témoigner ma profonde gratitude envers ma famille pour leur amour et leur soutien inconditionnel tout au long de ces années. Aucun mot ne peut véritablement exprimer toute l'appréciation que je ressens à leur égard.

À MES AMIS

Je tiens à exprimer ma profonde gratitude à mes amis pour tous les moments inoubliables que nous avons partagés ensemble et pour leur aide et leur soutien. Je leur suis reconnaissant pour leur présence à mes côtés et pour les instants précieux que nous avons partagés.

À MES PROFESSEURS

Pour votre encadrement, vos conseils, vos critiques, et vos jugements, je vous remercie sincèrement. Votre soutien a été inestimable.

Remerciement

Ce mémoire n'aurait pas pu être réalisé sans le soutien inestimable de nombreuses personnes que je tiens à remercier sincèrement pour leur aide tout au long de mon parcours.

Je tiens à exprimer ma profonde gratitude au Professeur Sara, Chef de la filière, pour son engagement exemplaire envers ses étudiants. Sa bienveillance et son dévouement ont créé un environnement d'apprentissage stimulant et propice à notre développement

Je remercie également chaleureusement les membres du jury, dont les conseils avisés et l'expertise ont enrichi la qualité de mon mémoire. Leur présence et leur soutien sont une véritable reconnaissance de mon travail, et je leur en suis profondément reconnaissant.

Je souhaite exprimer toute ma reconnaissance à ELKADDARI Hamza et NUIH Omar pour leur soutien indéfectible. Leur disponibilité et leurs conseils précieux ont été d'une aide inestimable tout au long de mon parcours académique.

Un grand merci également à tous mes professeurs de la filière RSI pour leur enseignement de qualité et leur accompagnement attentif. Leurs cours ont été essentiels pour acquérir des connaissances solides et développer mes compétences.

Je n'oublie pas non plus mes camarades de promotion, avec qui j'ai partagé des moments précieux et enrichissants. Leur amitié et leur soutien ont été une source d'inspiration tout au long de cette expérience.

Résumé

Abstract

Liste des abréviations

|  |  |
| --- | --- |
| Abréviation | Signification |
| BDD | Base de données |
| CMS |  |
| XMR |  |
|  |  |

Liste des tableaux

[Tableau 1. 1](#_Toc189732621)

Liste des figures

[Figure 1 : image 1 1](#_Toc189732627)

Sommaire

[Dédicace I](#_Toc191159306)

[Remerciement I](#_Toc191159307)

[Résumé II](#_Toc191159308)

[Abstract II](#_Toc191159309)

[Liste des abréviations II](#_Toc191159310)

[Liste des tableaux II](#_Toc191159311)

[Liste des figures II](#_Toc191159312)

[Sommaire II](#_Toc191159313)

[Introduction générale 1](#_Toc191159314)

[Chapitre I : Organisme d’accueil 2](#_Toc191159315)

[Chapitre II : Xibo 2](#_Toc191159316)

[Introduction 2](#_Toc191159317)

[I. Architecture 2](#_Toc191159318)

[II. Flux de Communication 4](#_Toc191159319)

[Conclusion 5](#_Toc191159320)

[Chapitre III : 1](#_Toc191159321)

[Analyse et spécification des besoins 1](#_Toc191159322)

[Introduction : 1](#_Toc191159323)

[III. Contexte Générale : 1](#_Toc191159324)

[IV. Cahier des charges 2](#_Toc191159325)

[V. Les Besoins Fonctionnelles 2](#_Toc191159326)

[VI. Les Besoins non Fonctionnelles 4](#_Toc191159327)

[Chapitre IV : Etat de l’art 1](#_Toc191159328)

[Introduction 1](#_Toc191159329)

[I. Les solutions CMS 1](#_Toc191159330)

[II. Les solution players 2](#_Toc191159331)

[III. Comptabilité des players avec les CMS 2](#_Toc191159332)

[IV. Technologies et Protocoles de Communication 2](#_Toc191159333)

[V. Frameworks et Outils de Développement 3](#_Toc191159334)

[Conclusion 3](#_Toc191159335)

[Chapitre V : Management de projet 4](#_Toc191159336)

[Chapitre VI : Conception et modélisation 4](#_Toc191159337)

[Chapitre VII : Outils 4](#_Toc191159338)

[Chapitre VIII : Réalisation 5](#_Toc191159339)

[VI. Titre 5](#_Toc191159340)

[Table des matières 5](#_Toc191159341)

[Bibliographie & Sitographie 5](#_Toc191159342)

[Annexes 6](#_Toc191159343)

Introduction générale

L’affichage dynamique est devenu un pilier essentiel dans les environnements modernes, que ce soit dans les entreprises, les institutions ou les espaces publics. Il permet de diffuser des contenus multimédias de manière ciblée, interactive et en temps réel, offrant ainsi une expérience engageante et adaptée aux besoins des utilisateurs. Xibo, une plateforme open source de gestion de contenus numériques, se positionne comme une solution clé pour répondre à ces exigences. Cependant, pour exploiter pleinement son potentiel, une personnalisation approfondie est souvent nécessaire afin de l’adapter aux spécificités de chaque contexte d’utilisation.

Dans ce cadre, notre projet se concentre sur la personnalisation de Xibo et le développement de trois lecteurs (players) dédiés, Windows, Android et BROWSER. Ces lecteurs permettront une diffusion optimale des contenus sur différents types de dispositifs, tout en offrant une expérience utilisateur fluide et adaptée aux exigences techniques et fonctionnelles des environnements cibles.

En outre, pour garantir une mise en œuvre efficace et scalable de cette solution, nous avons intégré des technologies modernes de déploiement et d’orchestration. Ces technologies permettent de simplifier les processus de déploiement, d’assurer une haute disponibilité et de faciliter la gestion des ressources dans des environnements complexes. Cette approche contribue à rendre la solution plus robuste et adaptable aux besoins évolutifs des utilisateurs.

Ce rapport détaille les différentes étapes de ce projet, depuis l’analyse des besoins jusqu’à la réalisation technique et le déploiement. Il est structuré en chapitres comme suite :

**CHAPITRE 1 : Organisme d’accueil**

**CHAPITRE 3 : Analyse et spécification des besoins**

**CHAPITRE 4 : Etat de l’art**

**CHAPITRE 5 : Xibo**

**CHAPITRE 6 : Management de projet**

: Organisme d’accueil

:

Analyse et spécification des besoins

Introduction

Ce chapitre présente l’analyse et la spécification des besoins pour le développement d’une solution SaaS destinée à la gestion et à la diffusion de contenus numériques. L’objectif est de proposer une alternative locale et économique aux solutions existantes sur le marché.

1. Contexte Générale

La société ***DBM Accent*** souhaite renforcer son indépendance technologi-que et optimiser ses coûts en développant sa propre solution logicielle. Ce projet vise à concevoir un produit **SaaS 100% marocain (Made in Morocco)** intégrant un **CMS personnalisé** ainsi que des **players dédiés**. Cette solution répondra aux besoins spécifiques des clients en leur offrant une alternative locale, accessible et performante.

1. Cahier des charges

L’objectif est de développer une solution logicielle proche de **Xibo** en termes de fonctionnalités, tout en apportant des améliorations et des adaptations spécifiques aux exigences locales.

1. **Objectifs du projet**

* Concevoir un **CMS personnalisé** pour la gestion et la diffusion de contenus.
* Développer des **players dédiés** compatibles avec la plateforme.
* Assurer une **expérience utilisateur optimisée** et intuitive.
* Offrir une **solution économique et accessible** aux entreprises marocaines.
* Garantir une **sécurité et une stabilité accrues** du système.

1. **Contraintes techniques**

* Développement basé sur des **technologies open-source**.
* Compatibilité avec plusieurs **systèmes d’exploitation** (Windows, Android).
* Architecture **scalable et modulaire** pour faciliter l’évolution de la solution.
* Sécurisation des données et mise en conformité avec les **normes locales et internationales**.

1. Les Besoins Fonctionnelles
2. Xibo CMS
3. Gestion des contenus :

* Ajout, modification et suppression de médias (vidéos, images, textes, PDF, etc.).
* Création de playlists pour organiser les contenus.

1. Gestion des écrans et players :

* Ajout et configuration des écrans et players.
* Attribution de contenus ou de playlists à des écrans spécifiques.
* Surveillance de l'état des players (en ligne/hors ligne, statut de lecture).

1. Synchronisation :

* Synchronisation de contenus sur plusieurs écrans (ex : afficher la même vidéo sur 10 écrans simultanément).
* Gestion des mises à jour en temps réel via le XMR (Xibo Message Relay).

1. Modèles de disposition (Layouts) :

* Création de modèles personnalisés pour organiser les contenus sur un écran (zones pour vidéos, images, textes, etc.).
* Utilisation de widgets pour ajouter des fonctionnalités dynamiques (horloges, météo, flux RSS).

1. Rapports et statistiques :

* Génération de rapports sur l'utilisation des contenus et des écrans.
* Suivi des performances des players (temps de lecture, erreurs, etc.).
* Gestion des utilisateurs et permissions :
* Création de comptes utilisateurs avec des permissions spécifiques (admin, éditeur ...).
* Gestion des accès aux contenus et aux écrans.

1. Xibo player Windows et Android
2. **Lecture de médias**

* Support pour les formats multimédias courants : **Vidéos**, **Images**, **Textes**, **PDF** ...
* Lecture en boucle des playlists.

1. **Synchronisation avec le CMS**

* Récupération des contenus et des playlists depuis le **serveur Xibo**.
* Mises à jour automatiques des contenus via le **XMR (Xibo Message Relay)**.
* Support pour les mises à jour en temps réel (changements de contenu, synchronisation multi-écrans).

1. **Gestion des écrans**

* Affichage de contenus différents ou synchronisés sur plusieurs écrans.
* Configuration des résolutions et orientations d'écran.

1. **Planification**

* Affichage des contenus selon un calendrier défini dans le **Xibo CMS**.
* Basculer entre différents contenus en fonction de l'heure ou de déclencheurs spécifiques (ex : événements en direct).

1. **Surveillance et rapports**

* Envoi de rapports sur l'état du player et des écrans au **serveur Xibo** (Erreurs, Statut de lecture, Statut de connexion …)

1. **Mode hors ligne**

* Cache des contenus pour une lecture hors ligne.
* Synchronisation des rapports et des mises à jour une fois reconnecté au serveur.

1. **Personnalisation**

Configuration des paramètres du player :

Volume, démarrage automatique, boucle de lecture, etc.

Support pour les **modèles de layouts** créés dans le **Xibo CMS** :Zones pour vidéos, images, textes, widgets, etc.

1. **Optimisation**

* ***Windows*** :

Optimisation pour les systèmes d'exploitation Windows (10, 11).

Support pour les écrans haute résolution et les configurations multi-écrans.

* ***Android*** :

Optimisation pour les appareils mobiles (tablettes, smartphones, écrans Android).

1. Les Besoins non Fonctionnelles

* Performance : Temps de réponse rapide et optimisation des ressources.
* Fiabilité : Haute disponibilité et tolérance aux pannes.
* Sécurité : Protection des données utilisateurs et communications chiffrées.
* Ergonomie : Interface intuitive et expérience utilisateur fluide.

: Etat de l’art

Introduction

La création d’un player interactif avec un CMS nécessite une analyse des technologies existantes pour garantir flexibilité, synchronisation et performance. Cette étude explore les CMS, protocoles de communication et frameworks de développement pour proposer une solution robuste et adaptée aux besoins modernes de Digital Signage.

1. Les solutions CMS

Les **Content Management Systems (CMS)** sont au cœur des systèmes de Digital Signage, permettant de gérer, organiser et diffuser des contenus multimédias sur plusieurs écrans.

|  |  |  |
| --- | --- | --- |
| ***Solution***  ***Critère*** | ***Xibo*** | ***Screenly OSE*** |
| ***Type*** | *Open source* | *Open source* |
| ***Coût*** | *Gratuit* | *Gratuit* |
| ***Flexibilité*** | *Élevée* | *Moyenne* |
| ***Synchronisation*** | *Via XMR* | *Limitée* |
| ***Facilité*** | *Technique* | *Technique* |
| ***Communauté*** | *Grande* | *Moyenne* |

Tableau 1 : Comparaison des solutions CMS

1. Les solutions players

Les players sont des applications ou des logiciels qui permettent d'afficher du contenu sur des écrans. Ils fonctionnent comme des lecteurs multimédias spécialisés, capables de recevoir, interpréter et afficher les playlists et les médias configurés via la plateforme Xibo.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ***Player*** | ***Plateforme*** | ***Prix*** | ***Licence*** | ***Fonctionnalités*** | ***Limites*** |
| ***Xibo Player*** | *Windows* | *Gratuit* | *Open Source (AGPLv3)* | *Supporte le CMS Xibo, mise en cache* | *Nécessite un serveur Xibo* |
| ***Screenly OSE*** | *Windows/Linux* | *Gratuit* | *Open Source (MIT)* | *Affichage web, vidéos, images* | *Nécessite configuration manuelle* |
| ***Concerto*** | *Windows/Linux* | *Gratuit* | *Open Source (GPLv3)* | *Gestion multi-écrans, médias en réseau* | *Interface plus technique* |
| ***Xibo Player (Android)*** | *Android* | *Payant* | *Licence commerciale* | *Supporte le CMS Xibo, gestion avancée* | *Licence obligatoire après essai* |
| ***PiSignage*** | *Android/Linux* | *Gratuit* | *Open Source* | *Compatible Raspberry Pi, support web* | *Certaines fonctionnalités payantes* |

Tableau 2 : Les solutions Players

1. **Compatibilité des players avec les CMS**

La compatibilité entre les players et les CMS est essentielle pour assurer une gestion fluide et centralisée des contenus diffusés. Chaque player est généralement optimisé pour un CMS spécifique, limitant ainsi les possibilités d'interopérabilité.

|  |  |  |
| --- | --- | --- |
| ***Player*** | ***Xibo CMS*** | ***Screenly*** |
| ***Xibo Player (Windows)*** | *✅ Oui* | *❌ Non* |
| ***Xibo Player (Android)*** | *✅ Oui* | *❌ Non* |
| ***Screenly Player*** | *❌ Non* | *✅ Oui* |

Tableau 3: Compatibilité des players avec les CMS

1. **Technologies et Protocoles de Communication**

Les technologies et protocoles de communication jouent un rôle clé dans la synchronisation et la transmission des contenus entre les CMS et les players.

|  |  |  |  |
| --- | --- | --- | --- |
| ***Technologie / Protocole*** | **Type** | **Utilisation** | **Avantages** |
| ***HTTP/HTTPS*** | *Protocole Web* | *Communication entre le CMS et le Player* | *- Simple*  *- sécurisé (HTTPS)* |
| ***WebSocket*** | *Temps réel* | *Envoi de mises à jour instantanées* | *- Bidirectionnel*  *- faible latence* |
| ***FTP/SFTP*** | *Fichier (Push)* | *Transfert de médias (images, vidéos)* | *- Fiable pour de gros fichiers* |
| ***REST API*** | *API Web* | *Gestion centralisée des Players* | *- Flexible*  *- supporte JSON/XML* |
| ***Art-Net*** | *Protocole DMX (Réseau)* | *Contrôle de l'éclairage numérique (scènes, LED)* | *- Très rapide*  *- faible latence pour l'affichage* |
| ***SNMP*** | *Protocole de supervision réseau* | *Monitoring et contrôle des Players à distance* | *- Surveillance détaillée des Players (température, état)* |

Tableau 4 : Technologie et Protocoles de communication

1. **Frameworks et Outils de Développement**

Les solutions de Digital Signage reposent sur divers langages et frameworks en fonction de leurs objectifs et plateformes cibles. Ces choix technologiques influencent directement la flexibilité, la performance et l’évolutivité des solutions de Digital Signage.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Solution*** | ***Type*** | ***Langage / Framework*** | ***Base de données*** | ***Compatibilité*** |
| ***Xibo CMS*** | *CMS* | *PHP (Laravel) + Angular* | *MySQL* | *Windows, Android* |
| ***Xibo Player*** | *Player* | *C# (.NET) / Java (Android)* | *N/A* | *Windows, Android* |
| ***DigitalSignage.com*** | *CMS* | *JavaScript (Meteor.js)* | *MongoDB* | *Windows, Android* |
| ***DigitalSignage*** | *Player* | *JavaScript (HTML5 Player)* | *N/A* | *Windows, Android* |
| ***Scala Digital Signage*** | *CMS* | *Java + Scala* | *PostgreSQL* | *Windows, Android* |
| ***Scala Player*** | *Player* | *Java* | *N/A* | *Windows, Android* |
| ***OnSign TV*** | *CMS* | *JavaScript + Web-based* | *Cloud* | *Windows, Android* |
| ***OnSign Player*** | *Player* | *Java (Android) / C# (.NET)* | *N/A* | *Windows, Android* |
| ***DSPlay*** | *CMS* | *Java (Spring Boot)* | *PostgreSQL* | *Android* |
| ***DSPlay Player*** | *Player* | *Java (Android)* | *N/A* | *Android* |

Tableau 5 : comparative des langages et frameworks utilisés par les CMS et les Players

**Conclusion**

L’analyse des solutions de Digital Signage montre que le Xibo CMS avec Xibo Player est une solution de Digital Signage optimale grâce à sa flexibilité, son faible coût, sa compatibilité multi-plateforme et son statut open source soutenu par une large communauté.

: Xibo

Introduction

Xibo est une solution **open source** de **gestion d'affichage numérique** (Digital Signage) conçue pour être flexible et puissante pour gérer les contenus, les écrans, les planifications…



Figure 2 : Logo de Xibo

1. Architecture

L'architecture de Xibo repose sur 4 **composants principaux** qui lui permet d’interagissent pour fournir une solution de Digital Signage complète **;** **Serveur Xibo (*CMS*), Clients Xibo (Players), *XMR (Xibo Managed Replay)* et les écrans d'affichage.**

1. CMS

C'est le cerveau du système. Il gère tout le contenu, les utilisateurs, les écrans et les planifications.

1. Fonctions principales :

* Stockage et gestion des médias (images, vidéos, flux *RSS*, etc.).
* Création des modèles d'affichage (layouts).
* Planification du contenu (quand et quoi afficher).
* Surveillance des écrans et des players.
* Gestion des utilisateurs et des permissions.

1. Technologie de développement

* Backend : PHP
* Frontend : HTML , CSS , JS , jQuery, Bootstrap

1. **Clients Xibo**

Ce sont les applications qui s'exécutent sur les appareils connectés aux écrans. Ils récupèrent le contenu du serveur et l'affichent.

1. **Types de players** et Technologies de développements

* **Xibo Windows Player: C# .Net**
* **Xibo Android Player** : Java
* **Xibo Linux Player: C++**
* **Xibo Web Player**: JS, jQuery, HTML, CSS

1. **Fonctions principales** :

* Se connecter au serveur pour récupérer les layouts et les médias.
* Afficher le contenu sur les écrans selon la planification.
* Envoyer des rapports au serveur (statut, erreurs, etc.).

1. **XMR (Xibo Managed Relay)**

Le XMR facilite la communication en temps réel entre le serveur Xibo et les players. Il agit comme un relais de messages pour :

* Transmettre des instructions du serveur aux players.
* Collecter des informations des players et les rapporter au serveur.

1. **Écrans d'affichage**

Ce sont les dispositifs physiques où le contenu est affiché (téléviseurs, moniteurs, panneaux LED, etc.).

Les écrans sont connectés aux appareils exécutant le player Xibo (PC, Android, etc.) et Le player contrôle ce qui est affiché sur l'écran.



Figure 3: écran d'affichage LED

1. Flux de Communication
2. CMS -> XMR -> Player -> écran
3. **Instructions du serveur vers les players**

* Le serveur Xibo envoie des instructions au XMR via des protocoles de communication.
* Le XMR reçoit ces instructions et les transmet aux players concernés.
* Les players se connectent au XMR pour écouter les instructions en temps réel.

1. **Types d'instructions**

* **Mises à jour de contenu** : Ajouter, modifier ou supprimer des éléments multimédias (vidéos, images, textes).
* **Synchronisation** : Coordonner plusieurs écrans pour afficher le même contenu simultanément.
* **Commandes de contrôle** : Démarrer, arrêter ou redémarrer un player.

1. Player -> XMR -> CMS
2. **Rapports des players vers le serveur**

* Les **players** envoient régulièrement des rapports au **XMR** sur leur état et celui des écrans.
* Exemple de rapport : "Player 1 est en cours de lecture, écran 1 est allumé, écran 2 est éteint."
* Le **XMR** collecte ces rapports et les transmet au **serveur Xibo**.

1. **Types de rapports**

* **État des players** : Statut de lecture, erreurs, version du logiciel.
* **État des écrans** : Allumé/éteint, résolution, connectivité.
* **Statistiques d'utilisation** : Temps de lecture, interactions utilisateur.

1. Protocoles de communication

HTTP/HTTPS : Utilisé pour les requêtes standards (mises à jour de contenu, rapports).

WebSocket : Utilisé pour la communication en temps réel (synchronisation, notifications).

API REST : Le serveur Xibo expose des API REST pour interagir avec le XMR et les players.

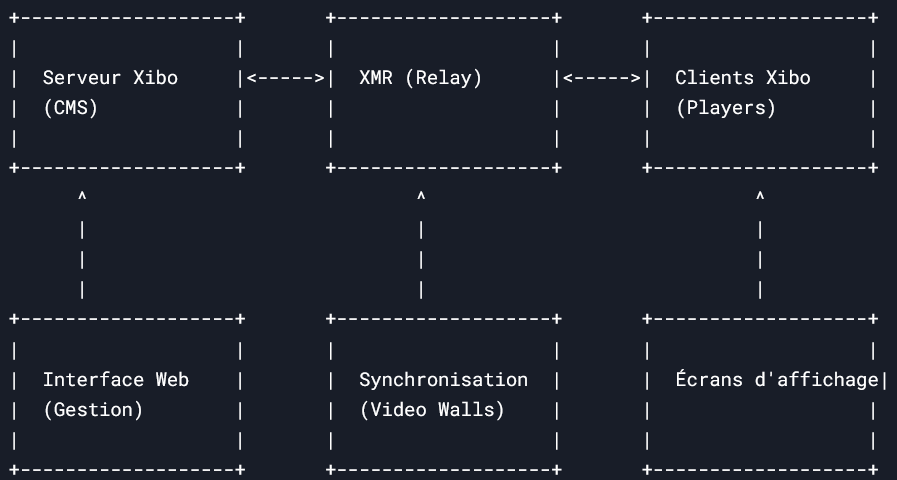


Figure 4 : Communication CMS <--> XMR <--> Players

Conclusion

**Xibo CMS** est le cerveau du système, il stocke le contenu, planifie les playlists et gère les utilisateurs.

**Xibo XMR** est le système nerveux, il assure la communication rapide et efficace entre Xibo CMS et les players

: Management de projet

1. Outils et technologies
2. CMS
3. Backend : PHP

PHP est un langage de programmation côté serveur largement utilisé pour développer des applications web dynamiques. Il est souvent utilisé dans les CMS comme WordPress, Joomla ou Drupal pour gérer la logique métier et interagir avec la base de données.

1. Frontend : Angular

Angular est un framework JavaScript open source développé par Google. Il est utilisé pour créer des interfaces utilisateur dynamiques et réactives (Single Page Applications). Il permet de structurer le code frontend et de gérer les interactions utilisateur de manière efficace.

1. Players (Lecteurs multimédias ou applications)
2. Windows : C# .Net

C# est un langage de programmation orienté objet développé par Microsoft. Il est souvent utilisé avec le framework .NET pour créer des applications Windows robustes et performantes, y compris des lecteurs multimédias ou des logiciels métiers.

1. Android : Java

Java est un langage de programmation polyvalent et orienté objet, largement utilisé pour développer des applications Android. Il permet de créer des applications natives pour les appareils mobiles fonctionnant sous le système d'exploitation Android.

1. Alternative : MAUI (Multi-platform App UI)

MAUI est un framework de développement d'applications multiplateform-es créé par Microsoft. Il permet de développer des applications pour Windows et Android (ainsi que iOS et macOS) en utilisant un seul codebase en C#. Il est une évolution de Xamarin. Forms et simplifie la création d'interfaces utilisateur communes pour plusieurs plateformes.

1. BDD (Base de Données)

Une base de données est un système organisé pour stocker, gérer et récupérer des données de manière structurée.

1. MySQL

MySQL est un système de gestion de base de données relationnelle (SGBDR) open source. Il utilise le langage SQL (Structured Query Language) pour gérer et manipuler les données. Il est largement utilisé dans les applications web pour stocker des informations structurées, comme les utilisateurs, les articles, les commandes, etc.

 : Conception et modélisation

Recherche XIBO architecture

CMS installation et configuration

Players

XMR

Etat de l’art

Management de projet

Réalisation

: Outils

PHP

MySQL

C#

.net

**MAUI**

OpenVpn

Docker

: Réalisation

****Table des matières****

le **sommaire** est un résumé du contenu, tandis que la **table des matières** est un outil de navigation détaillé.

Bibliographie & Sitographie

|  |  |
| --- | --- |
| Lien | Objet |
| <https://xibosignage.com/manual-tempel/en/index.html> | Manual d’utilisation Xibo |
| https://community.xibo.org.uk/c/guides/11 | Xibo Community |
| https://github.com/xibosignage | Xibo Digital Signage (Srouce Code) |
| https://xibosignage.com/pricing | Priceing Xibo |
| https://github.com/xibosignage/xibo-docker.git | Repo Xibo Docker |

Annexes

Structure du Projet

Dossier Stats

Fichier StatManager.cs : Ce fichier contient la classe StatManager qui gère les statistiques de "Proof of Play". Il inclut des méthodes pour initialiser la base de données, démarrer et arrêter le gestionnaire de statistiques, enregistrer des événements de début et de fin de mise en page et de widget, et envoyer des statistiques.

Dossier Logic

Fichier Schedule.cs : Ce fichier contient la classe Schedule qui gère la planification des affichages. Il inclut des méthodes pour initialiser les composants, gérer les événements de changement de planification, et contrôler les agents et les threads associés à la planification.

Fichier RequiredFiles.cs : Ce fichier contient la classe RequiredFiles qui gère les fichiers nécessaires pour l'affichage. Il inclut des méthodes pour évaluer et ajouter des fichiers requis, marquer les fichiers comme complets, et rapporter l'inventaire des fichiers requis.

Dossier Rendering

Fichier WebMedia.cs : Ce fichier contient la classe abstraite WebMedia qui représente un média web. Il inclut des méthodes pour configurer le média web, vérifier si le HTML est prêt, et rafraîchir le cache local.

Dossier XmdsAgents

Fichier ScheduleAndFilesAgent.cs : Ce fichier contient la classe ScheduleAndFilesAgent qui gère la récupération des fichiers requis et la planification depuis le serveur XMDS. Il inclut des méthodes pour exécuter le thread de récupération, gérer les fichiers requis, et traiter les listes de purge.

Dossier XmdsAgents

Fichier ScheduleManager.cs : Ce fichier contient la classe ScheduleManager qui contrôle la planification en cours. Il inclut des méthodes pour exécuter la planification, charger la planification à partir de fichiers, et valider et résoudre les éléments de planification.

Fonctionnalités Clés

Gestion des Statistiques : Collecte et enregistrement des statistiques de "Proof of Play" pour les mises en page et les widgets.

Planification : Gestion des horaires d'affichage, y compris les événements de changement de planification et les actions de superposition.

Gestion des Fichiers Requis : Téléchargement et gestion des fichiers nécessaires pour l'affichage, y compris la vérification de l'intégrité des fichiers et la gestion de l'espace disque.

Affichage Web : Gestion des médias web, y compris la configuration des packages HTML et la mise à jour du cache local.

Communication avec le Serveur XMDS : Récupération des fichiers requis et des horaires de planification depuis le serveur XMDS.

Cette structure permet de gérer efficacement les affichages numériques, en assurant la collecte des statistiques, la gestion des horaires, et la disponibilité des fichiers nécessaires.