Dédicaces

Je dédie ce présent travail

Au seigneur de l'univers,

A mon père qui m'a indiqué la bonne voie en me rappelant que la volonté fait toujours les grands hommes,

A ma mère qui a attendu avec patience les fruits de sa bonne éducation,

A ma sœur qui a été toujours présente avec son soutien moral et son amour,

A tous mes cousins, mes amis et mes professeurs et tous ceux que j'aime.

Ryme

Remerciements

C'EST parce que j'ai beaucoup estimé tous ceux qui m'ont écouté, conseillé, critiqué et encadré que je tiens à leur faire part de toute ma gratitude, et plus particulièrement, je tiens à les remercier à travers ces courtes lignes.

Tout D'abord, je tiens à remercier mon encadrant technique **Monsieur Akrem Ajroudi**, pour m'avoir incité à mener à bien ce travail, pour son aide, son temps passé pour me guider, ses efforts pour m'intégrer dans l'environnement, son dévouement et ses précieux conseils.

Ensuite, mes remerciements vont à mon encadrante pédagogique **Madame Maha Driss** parce qu'elle a accepté de me superviser et suivre les détails de l'avancement de mon travail, ainsi pour son aide et ses conseils dans plusieurs étapes du projet.

Mes remerciements et reconnaissances s'étendent également à toute la famille d'**OPCMA** pour leur accueil et particulièrement **Monsieur Nizar Mertah** pour son aide, ses conseils et son soutien moral incessant.

Je tiens aussi à remercier les différents membres du jury **Madame Ines Elouadi et Monsieur Atef Gharbi,** pour bien vouloir m'accorder de leur temps précieux pour commenter, discuter et juger mon travail.

Enfin, je ne peux pas finir sans exprimer ma profonde gratitude à **tous les professeurs de l'Ecole Nationale Supérieure d'Ingénieur de Tunis** pour leur dévouement et leur assistance tout au long de mes trois années d'étude en cycle ingénieur.

Table des matières

Table des figures

Table des tableaux

Introduction générale

Dans un monde actif et continuellement évolutif, la motivation d'avoir des moyens performants et efficaces de divertissement, de communication et d'échange d'information devient de plus en plus fondamentale. Cette motivation a donné naissance à une révolution favorisant l'accès aux besoins en temps réduit à l'aide d'Internet.

Dans ce cadre, les téléviseurs intelligents apparaissent pour rompre avec nos anciennes idées sur les téléviseurs normaux et donner une autre dimension à cette technologie tout en intégrant de nouveaux apports au téléviseur normal. Grâce à l'émergence de cette nouvelle technologie, les usages des téléviseurs vont être modifiés d'une manière significative puisque ces téléviseurs vont fournir à leurs utilisateurs un accès précieux à toute une panoplie de services divers par exemples : surfer sur Internet avec notre téléviseur connecté, interagir sur les réseaux sociaux, profiter d'albums photo partagés, des vidéo en streaming, consulter nos mails et aussi l'actualité ou la météo.

C'est dans cette optique que se situe notre projet de fin d'études élaboré au cours de la troisième année du cursus de formation du cycle d'ingénieurs .Il vise à développer des services d'une Smart TV pour profiter des fonctionnalités qui n'existent pas dans les téléviseurs normaux. Ce projet a été effectué à **OPCMA** «**Open Consulting & Management**» qui est une entreprise spécialisée dans le développement logiciel et dans le consulting. Les principaux services offerts par **OPCMA** sont : la création, la mise à jour et l'optimisation de sites Web. Son objectif consiste à augmenter la rentabilité des entreprises travaillant dans divers secteurs : financier, commercial et organisationnel.
Parmi les applications réalisées au sein d'**OPCMA**, nous citons notre présente application : la Smart TV. Ce projet nous a été bénéfique vu l'intérêt que nous portons pour les nouvelles technologies et vu que c'était une occasion pour nous pour apprendre de nouveaux environnements de développement tel que le NodeJS.

Le présent rapport se résume en quatre chapitres qui s'enchainent comme suit :

- Le premier chapitre intitulé "Cadre général du projet" est le point de départ. En premier lieu, il donne un aperçu sur l'organisme d'accueil **OPCMA**. En second lieu, il présente une étude de l'existant afin de comprendre l'utilité de la solution proposée. Enfin, il présente les objectifs du projet à réalisé et la méthode de développement à adopter.

- Le deuxième chapitre est dédié à la spécification et à l'analyse des besoins.

- Le troisième chapitre met en exergue l'aspect conceptuel de l'application en présentant les différents scénarii d'utilisation de notre Smart TV par l'intermédiaire de la notation UML.

- Le dernier chapitre spécifie l'environnement technologique de l'application et expose les différentes fonctionnalités du Smart TV réalisée au cours de ce projet de fin d'étude.

Chapitre 1 : Cadre général du projet

Introduction

Dans ce chapitre, nous mettons notre travail dans son cadre général. D'abord, nous exposons le contexte de notre projet. Par la suite, nous présentons l'organisme d'accueil qui est **OPCMA - Open Consulting & Management**, sa vision et ses différentes offres. Enfin, nous décrivons le travail à réaliser et la méthode de développement suivie.

1. Contexte du projet

Le présent projet "développement des services de Smart TV "est réalisé dans le cadre de la préparation du projet de fin d'études présenté en vue de l'obtention du diplôme d'ingénieur en informatique à l'Ecole Nationale Supérieure d'Ingénieurs de Tunis (ENSIT) pour l'année universitaire 2014-2015. Dans ce qui suit, nous présentons l'organisme au sein du quel s'est déroulé notre projet de fin d'études.

2. Organisme d'accueil

Notre projet a été réalisé au sein d'**OPCMA - Open Consulting & Management**. **OPCMA** est fondée en 2008 par 2 ingénieurs Tunisiens issus de grandes écoles d'ingénieurs qui ont cumulé 26 ans d'expériences dans de grands groupes internationaux dans des rôles différents : maître d'œuvre et de conseil et audit [s1].

2.1 Présentation générale d'OPCMA

Cette entreprise est spécialisée en conseil de stratégie et d'innovation. Son positionnement est orienté vers des besoins des entreprises en quête d'accélérer leurs performances financières, commerciales et organisationnelles. Elle propose à ses clients une offre de conseil pour consolider leurs activités et préparer sereinement leur avenir.

Leurs consultants et associés ont fait leurs preuves chez des clients internationaux en pilotant et participant au lancement de projets stratégiques.

Grâce à son expérience, **OPCMA** a développé un savoir faire et une maîtrise des processus d'externalisation[1] et d'offshoring[2] ; elle a mis à la disposition de ses clients un large spectre de compétences techniques en systèmes d'information et systèmes embarqués [s1].

La Figure 1.1 présente le site d'OPCMA accessible via l'URL : http://www.opcma.com/.

Figure 1.1: Page d'accueil du site Web d'OPCMA

2.2 Vision d'OPCMA

La mondialisation induit une concurrence accrue et une complexité des processus au sein de l'entreprise. Le défi est de réussir les projets stratégiques en respectant les budgets, les délais et les normes de qualité.

OPCMA s'engage auprès de ses clients afin de les accompagner à :

[1] L'externalisation : pour une entreprise, c'est l'action d'externaliser, c'est-à-dire de transférer à l'extérieur certaines de ses activités.
[2] L'offshoring : désigne la délocalisation des activités de service ou de production de certaines entreprises vers des pays à bas salaire.

- gérer la complexité croissante des projets et des environnements de déploiement ;
- débloquer les situations difficiles des projets par une approche technique et managériale ;
- accompagner le changement en suscitant l'adhésion des équipes projets et le management.

OPCMA propose également à ses clients de les accompagner dans leurs projets de recherche de croissance rapide, profitable et durable. Ceci en mettant en avant l'intérêt particulier à tirer profit d'une planification flexible. Elle propose les actions suivantes [s1] :

- la définition d'un plan stratégique d'externalisation et d'offshoring ;
- l'identification des risques projets ;
- la proposition d'une organisation adéquate ;
- et l'exécution du plan stratégique qui permet de favoriser la réussite.

2.3 Activités d'OPCMA

OPCMA propose à ses clients une offre de conseil, une offre de management et l'offre de développement offshore pour consolider leurs activités [s1].

L'offre de conseil s'articule autour des axes suivants:

- conseil en stratégie et organisation ;
- conseil en innovation et technologie d'information ;
- conseil en management des projets technologiques ;
- conseil et accompagnement en externalisation et offshoring ;

OPCMA met aussi en avant une forte expertise dans les secteurs :

- l'industrie de l'automobile ;
- l'énergie ;
- l'électronique télécom et grand public ;
- la distribution ;
- et le télécom.

Les domaines d'intervention d'OPCMA sont:

- l'audit de projet ;
- la direction des projets ;
- le pilotage des contrats et des fournisseurs ;
- la préparation CMMI[3];
- la mise en place et le déploiement de stratégie d'externalisation ;
- l'appui au pilotage des projets ;
- le montage d'offre de service : au forfait et en régie ;
- la rédaction des cahiers des charges et des grilles d'évaluation ;
- la mise en place des processus d'externalisation software/hardware ;
- et enfin la gestion des projets au forfait.

Aussi, chaque projet élaboré a une réalité et nécessite un management adapté à son contexte interne et à son environnement externe. Ceci est réalisé grâce à l'offre de management. Pour cela, OPCMA analyse avec soin le contexte de chaque projet et conseille le client du type de management adapté à son projet et apporte des réponses aux différentes questions qui peuvent être posées par le client. Parmi ces questions, nous citons :

- Quels sont les interlocuteurs qui doivent intervenir sur mon projet et quels sont leurs rôles ?
- Quels sont les risques et les grandes problématiques que je peux rencontrer (maturité des produits, pérennités des fournisseurs et des ressources, complexité de l'organisation, importance du changement, etc.) ?
- Quelle est la démarche optimale à appliquer dans mon contexte?
- Comment bien assurer le passage en production?
- Comment mesurer la réussite de mon projet?

OPCMA met également en place un socle de gestion de projet outillé :

- un modèle de planning ;
- un modèle de compte rendus et de tableaux de bord ;

[3] CMMI (Capability Maturity Model Integration) : est l'outil d'une démarche dynamique d'entreprise pour améliorer ces processus.

- un RACI[4]ou matrice de responsabilités et/ou flux de communication inter-organisations ;
- un modèle d'instruction et fréquence de comités de pilotage / directoires ou comités contractuels avec les thèmes ainsi que les objectifs à atteindre ;
- et des modèles d'enquêtes de satisfactions et de recueil d'information.

Et enfin, en se capitalisant sur son expérience et celle de ses collaborateurs, **OPCMA** met à la disposition de ses clients toutes les compétences nécessaires pour concevoir des projets fonctionnels et originaux autour des offres suivantes [s1]:

- développement Java J2EE[5] ;
- informatique embarquée ;
- développement d'applications mobiles intégrées.

3.Étude de l'existant

Dans cette section, nous procédons à une étude détaillée de l'existant permettant de bien comprendre le système actuel et dégager l'importance de l'application à développer.

3.1 Définition d'une Smart TV

Une Smart TV que nous pouvons la traduire en télévision intelligente est un téléviseur connecté à Internet avec lequel nous pouvons accéder à une multitude de fonctionnalités et de services que les téléviseurs réguliers ou normaux n'offrent [s2].

Comme le montre la Figure 1.2, nous avons sur cette télévision tout un catalogue d'applications, comme sur les Smartphones mais bien sûr dédié au format et à l'expérience d'une télévision, un navigateur Internet complet permettant de consulter du contenu texte et même des vidéos via **YouTube[6]** ou **DailyMotion[7]**, et l'accès aux fichiers en branchant une clef USB ou tout autre outil de stockage externe, c'est-à-dire que nous pouvons transférer l'écran de notre ordinateur sur notre téléviseur, et tout ça sans fil ni câble.

[4] RACI : est une méthode qui permet d'identifier les rôles et responsabilités au sein d'un projet.
[5] J2EE : est l'acronyme de Java 2 Entreprise Edition. Cette édition est dédiée à la réalisation d'applications pour entreprises.
[6] YouTube : est un site Web d'hébergement de vidéos sur lequel les utilisateurs peuvent envoyer, regarder et partager des vidéos accessible via l'URL : https://www.youtube.com
[7] Dailymotion : est une entreprise française offrant un service d'hébergement, de partage et de visionnage de vidéo en ligne accessible via l'URL : www.dailymotion.com/fr

La plupart des Smart TV propose aujourd'hui un service de vidéo à la demande qui nous permet de bénéficier d'un large choix de vidéos à la demande, un service de qualité qui sélectionne et classe les contenus les plus plébiscités et les plus utiles, aussi de louer ou même parfois d'acheter des films directement via notre télévision. La Smart TV propose également d'accéder aux réseaux sociaux tout en regardant nos émissions de télé favorites où l'émission est affichée sur les trois-quarts de l'écran et les publications sur le côté.

Là où les Smart TV sont vraiment des télévisions intelligentes c'est dans leur façon de mémoriser notre manière de l'utiliser. Avec cette base de données, notre Smart TV nous proposera du contenu, des programmes ou des applications adaptés à nos goûts et nos habitudes.

Figure 1.2: L'interface d'une Smart TV

3.2 Présentation de l'existant

La télévision fait partie de notre quotidien depuis plus de 50 ans. Elément incontournable dans le foyer, son histoire a été jalonnée par de nombreuses innovations technologiques permettant de rendre ce média toujours plus attractif pour les particuliers.

3.2.1 L'ancienne télévision

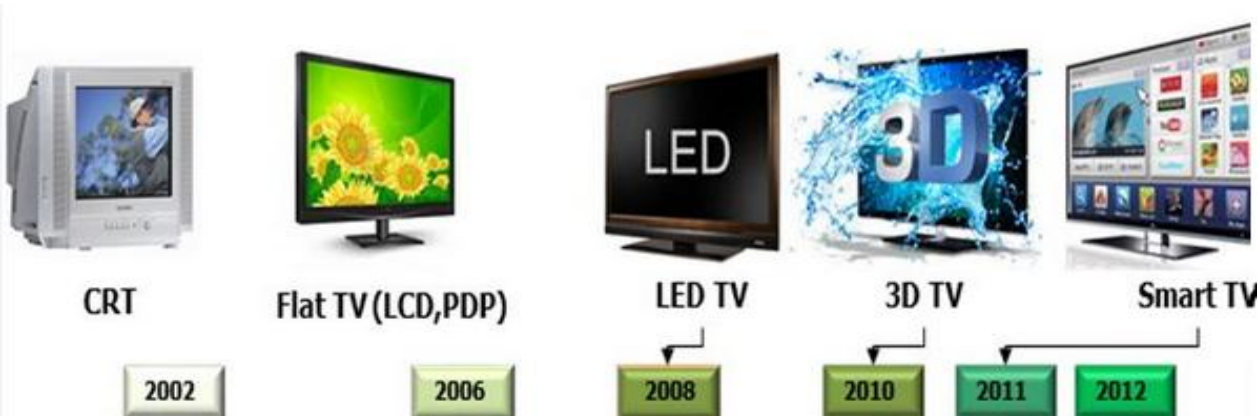
Les moniteurs CRT[8], qui ressemblent à des vieux téléviseurs, ont disparu du marché de la technologie d'aujourd'hui en raison des inconvénients distincts de la CRT, qui comprennent la taille, l'âge, la performance et l'efficacité. Malgré ça, jusqu'à aujourd'hui la plupart des foyers tunisiens sont encore équipés de ce téléviseur vu le coût élevé des téléviseurs modernes pour une grande partie de la population.

Figure 1.3: Une télévision CRT

[8] Les moniteurs CRT **:** sont des tubes cathodiques (cathode ray tube ou en français tube à rayonnement cathodique), équipant les téléviseurs et les moniteurs informatiques traditionnels. A l'intérieur du tube, un canon produit un faisceau d'électrons, lequel balaye une surface plane dans le but de produire une image.

La Figure 1.4 illustre la grande vitesse d'évolution de la télévision, la chaîne de l'évolution est donc :

CRT (TV à tube) > TV plat LCD > TV LED (encore plus plat) > TV 3D > Smart TV.Figure



1.4: L'évolution de la télévision

Nous décrivons ces différents types de téléviseur:

- **CRT :**

La TV cathodique est inventée en 1926 par un Écossais nommé John Baird [s3], les écrans CRT sont lourds avec un poids de 15 à 35 kg rendant ce genre d'écran trop encombrant et quasiment intransportable. Elle n'est pas confortable à regarder, nous pouvons avoir rapidement mal aux yeux ; sa consommation d'énergie est importante, elle est doublée à triplée par rapport à un écran plat.

- **LCD :**

Le LCD (Écran à Cristaux liquides) est une technologie maîtrisée depuis plusieurs années, un téléviseur LCD est caractérisé par une luminosité supérieure à celle du cathodique ; son poids ne dépassant pas les 20kg .Les LCD nous permettent de s'approcher sans fatiguer les yeux, grâce à la stabilité de l'image, nous allons bénéficier d'une excellente luminosité et d'une parfaite netteté. Les écrans LCD consomment moins d'électricité qu'un téléviseur classique.

- **LED :**

Les écrans LED (Light-Emitting Diode) font partie de la famille des écrans LCD avec un design ultra plat et utilisant une technologie de rétroéclairage innovante. La luminosité est mieux gérée que sur les écrans LCD classiques**,** la consommation électrique du téléviseur LED est revue à la baisse de près de 40 % par rapport au téléviseur LCD.

- **3D TV**

C'est sans aucun doute la révolution 2010 dans l'électronique grand-public. La télévision 3D (Trois dimensions) possède une meilleure qualité d'images, affiche rapidement une image décalée sur deux et les lunettes s'occupent de cacher l'image à un œil puis l'autre. Elle possède la plus grande palette de couleur et un angle de vue large, pratique lorsque plusieurs spectateurs regardent l'écran.

- **SMART TV**

Nous remarquons que certains modèles de télévisions comme TV LCD / TV LED intègrent une connectivité réseau, si ces premières TV reliées à Internet proposaient simplement de mettre à jour le logiciel de la TV, désormais les fabricants intègrent aux gammes de Smart TV des passerelles donnant accès à de multiples applications (jeux, réseaux sociaux, communication, service de stockage et de partage de fichiers en ligne, navigateur Internet, etc.), en plus la Smart TV est caractérisé par une ultra Haute Définition offre 4 fois la résolution de celle de la télévision 3D ; et affiche jusqu'à 3 fois de nuances de couleurs par rapport à un téléviseur LED.

3.2.2 Les applications existantes de Smart TV

Sur la Smart TV, nous avons d'un côté la TV classique avec les différentes chaînes, et de l'autre l'interface caractéristique d'une TV connectée à Internet. Celle-ci comprend cinq univers :

- **réseaux** (chaînes YouTube, Skype[9]…),
- **applications** (Twitter[10], Deezer[11]…),

[9] Skype : est un logiciel gratuit qui permet aux utilisateurs de passer des appels téléphoniques et vidéo via Internet.

- **programmes** (Chaînes TV et recommandations),
- **vidéo Club** (bibliothèque de vidéo à la demande),
- **contenus** (photos, vidéos de périphériques externes).

Le portail des applications fonctionne sur le même principe que l'AppStore[12] sur mobile. Nous avons un grand choix d'applications et de jeux que nous pouvons télécharger et installer sur notre Smart TV.

3.3 Critiques de l'existant

Une étape essentielle de tout projet consiste à effectuer une étude complète des outils et des applications actuels pour lesquels nous voulons apporter des solutions afin de détecter les défaillances et les insuffisances auxquelles nous devons remédier.

3.3.1 Critique de l'ancienne télévision

Les anciennes télévisions sont plus lourdes et plus volumineuses que leurs homologues plus contemporaines qui font figure de "poids plume"; elles prennent beaucoup plus d'espace, semblent moins agréable, et sont beaucoup plus difficiles à transporter.

Aussi, la TV cathodique n'est pas confortable à regarder ; sa résolution est très basse, nous pouvons avoir rapidement mal aux yeux. En plus, la consommation d'énergie est très importante. Et le plus important, c'est que ces téléviseurs manquent des dernières innovations technologiques.

3.3.2 Critique des applications existantes

Suite à l'examen de la Smart TV **Samsung** qui a récemment franchi la barre des 1000 applications disponibles et des 10 millions d'applications téléchargées, nous avons identifiés les problèmes suivants :

- Un nombre énorme d'applications qui peut engendrer une surcharge qui provoque à son tour un plantage parce que le débit d'internet utilisé par la majorité des Tunisiens ne dépasse pas 512 Ko, et un risque d'attaques par les virus ce qui nécessite une mise à jour régulière d'antivirus payants.

[10] Twitter : est un réseau social.
[11] Deezer : est un service d'écoute de musique.
[12] L'App Store: est une plateforme de téléchargement d'applications sur les appareils mobiles.

- Parmi les applications existantes, plusieurs peuvent être inutiles pour l'utilisateur ou encore incompréhensibles par un utilisateur non expert ou non habitué de l'utilisation des dernières innovations technologiques.
- Une connexion Internet haut débit est requise pour les fonctions Smart TV alors que le débit Internet en Tunisie est faible de l'ordre de 512KO sinon un haut débit 20Mbps c'est trop cher de l'ordre 50DT/mois.
- Il y a aussi des applications qui sont néfastes pour les enfants, si on leur laisse ces outils, ça peut être un grand danger pour leur développement. Exemple :
  Snapchat [13],Amadou[14], Murmure[15]...
- Les Smart TV ont des prix onéreux, par exemple dans le marché tunisien le modèle TV LG 55 coûte 4200,000 DT et le modèle Samsung UA65HU8500 coûte 8999,000DT [s4].

4. Travail à réaliser

4.1 Apport de notre projet

Nous sommes amenés à concevoir et à réaliser une application permettant le développement des services d'une Smart TV afin de répondre aux besoins du client, en fournissons :

- une interface personnalisée permettant d'accéder à des contenus applicatifs et multimédias selon l'âge de l'utilisateur, son éducation et son intérêt. Cette interface sera modifiée selon le besoin et le profil de l'utilisateur ;
- un accès aux réseaux sociaux avec la possibilité de naviguer sur Internet ;
- une Smart TV à coût raisonnable. Comment peut-on baisser le coût ?
- une sécurité des données des différents utilisateurs ;
- un contrôle parental.

4.2 Objectifs du projet

Nous pouvons résumer les objectifs de notre application en ces 6 points :

- développer des services personnalisés et dynamiques d'une Smart TV ;
- garantir une sécurité des données des différents utilisateurs ;
- l'application développée doit être rapide et performante ;

[13] Snapchat : est une application de messagerie d'image consiste à envoyer des photos embarrassantes ou osées.
[14] Amadou : est une application qui permet la communication avec les fraudeurs.
[15] Murmure : est une application conçue pour répandre des rumeurs et secrets. Il permet aux utilisateurs de poster des photos et du texte anonyme.

- accéder aux différents services à l'aide d'une télécommande ;
- diminuer le coût ;
- assurer un contrôle parental.

4.3 Solution envisagée

La solution adéquate sera de transformer notre téléviseur muet en une télévision intelligente qui propose des contenus personnalisés en développant une application qui nous offre les services de la Smart TV selon les habitudes et le goût des téléspectateurs à coût réduit, environ 700 dinar (prix d'un téléviseur normal et un boitier) au lieu d'acheter une Smart TV de 4 milles dinars. La Figure 1.5 présente la solution envisagée de notre projet.

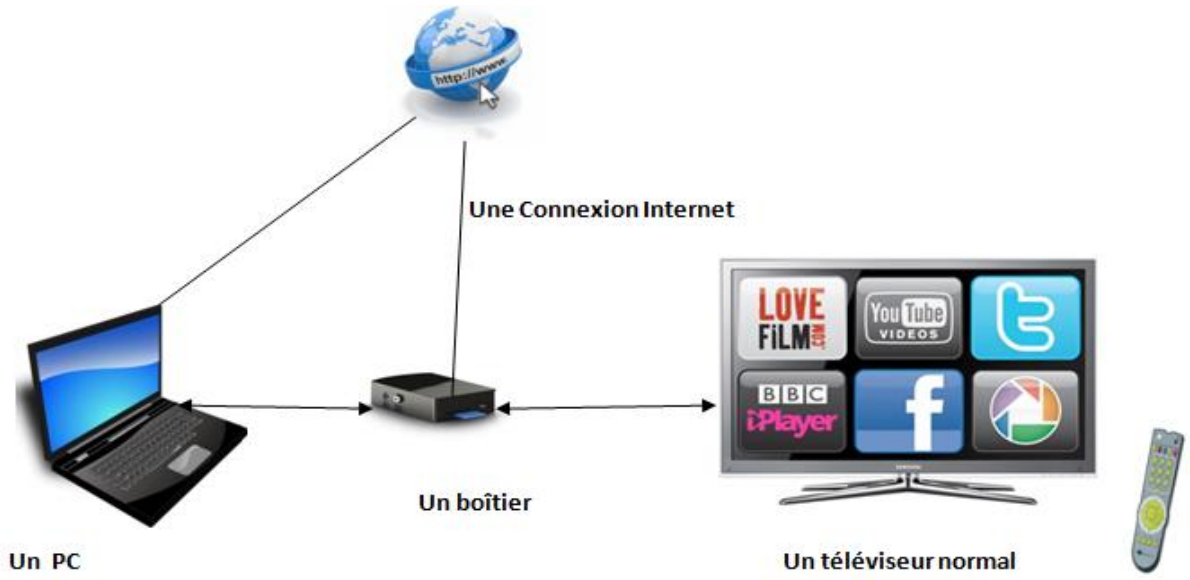


Figure 1.5: Solution envisagée

4.4 Démarche de développement

Pour chaque produit conçu et développé nous choisissons une démarche pour la suivre tout au long du projet que nous appelons cycle de vie.

L'objectif d'un tel découpage est de définir des jalons intermédiaires permettant la validation du développement du logiciel et la vérification de son processus de développement.

L'origine de ce découpage provient du constat que les erreurs ont un coût si élevé lorsqu'elles sont détectées tardivement dans le processus de développement.

Le cycle de vie permet de détecter les erreurs au plus tôt et ainsi de maîtriser la qualité du logiciel, les délais de sa réalisation et les coûts associés [s2].

En général, tous les modèles de cycle de vie découpent le projet en plusieurs phases principales :

- **Spécification des besoins**: cette activité consiste à définir la finalité du projet et son intégration dans une stratégie globale.
- **Conception générale**: dans cette activité, il s'agit de la préparation de l'architecture générale du logiciel.
- **Conception détaillée**: elle consiste à définir précisément chaque sous-ensemble du logiciel.
- **Développement**: elle est appelée aussi implémentation ou programmation. Il s'agit d'une traduction des fonctionnalités définies dans la phase de conception en langage de programmation.
- **Tests unitaires**: ils permettent de vérifier individuellement que chaque sous-ensemble du logiciel est implémenté conformément aux normes définies dans la conception.
- **Intégration**: dite aussi tests système, elle consiste à vérifier que le logiciel correspond exactement au cahier des charges du projet en obtenant enfin un manuelle d'utilisation bien détaillé aux utilisateurs.
- **Validation**: c'est-à-dire la validation de conformité de l'application avec les buts spécifiés à la première étape du cycle de vie.

Ces activités peuvent être présentes dans le cycle de vie d'un logiciel selon le modèle choisi par le client et l'équipe de développement.
Il existe différents types de cycles de vie logiciels tels que le modèle en V, en cascade, etc. Ces cycles de vie linéaires possèdent plusieurs limites. En effet, il est irréaliste d'obtenir de l'utilisateur un énoncé complet et cohérent de tous ses besoins. Ce type de cycle est trop rigide et manque donc de flexibilité pour gérer les imprévus. En outre, il ne reflète pas la façon dont le code est développé.
À cause de ces limitations des cycles de vie linéaires, nous adoptons le modèle de cycle de vie par incrément illustré par la Figure 1.6. Dans ce modèle, un seul ensemble de composants est développé à la fois : des incréments viennent s'intégrer à un noyau de logiciel développé au préalable.

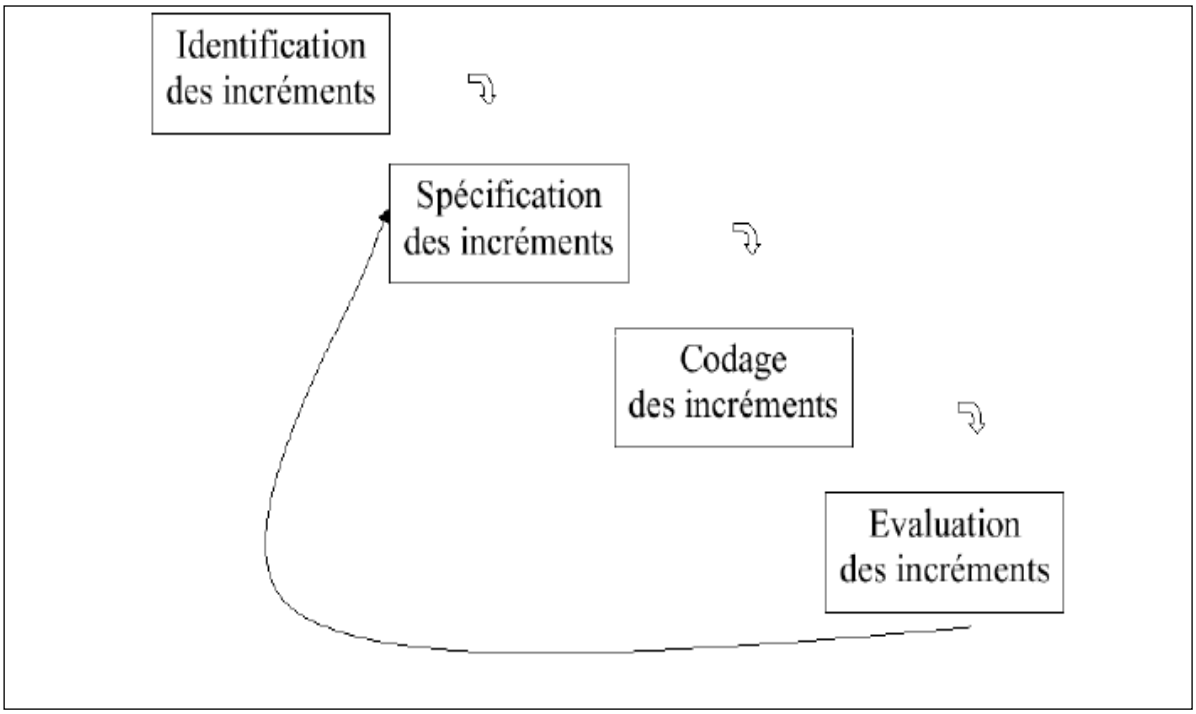


Figure 1.6: Modèle de cycle de vie par incréments

Les avantages du modèle de cycle de vie par incrément sont les suivants [s2] :

- chaque développement est moins complexe ;
- les intégrations sont progressives ;
- il est ainsi possible de livrer et de mettre en service chaque incrément ;
- Il peut y avoir des livraisons et des mises en service après chaque intégration d'incrément ;
- permet d'optimiser le temps et le partage de tâche ;
- diminue l'effort pendant la phase de tests.

Le modèle qui vient d'être décrit est celui utilisé par la plupart des projets actuels, agile ou non. C'est une stratégie par étape qui a connu beaucoup de succès.

Afin de concevoir et développer notre application, nous avons opté pour ce modèle. Ce choix est expliqué par le fait que notre projet est innovant et implique de nouvelles technologies et la plupart des spécifications sont connues à l'avance et vont être sujettes à de faibles évolutions, aussi, ce cycle est le plus efficace avec son principe de travail qui permet d'obtenir rapidement un produit fonctionnel.

4.5 Planning prévisionnel du projet

La clé principale de la réussite d'un projet est un bon planning. En effet, le planning aide à bien subdiviser le travail et séparer les tâches à réaliser, il offre une meilleure estimation et gestion de temps nécessaire pour chaque tâche. De plus, il donne assez de visibilité permettant d'estimer approximativement la date d'achèvement de chaque tâche.

Dans notre projet, nous avons estimé de réaliser notre application dans une durée approximative de 4 mois. Pour faire le planning, nous avons utilisé le diagramme de Gantt. Ce diagramme est un outil de planification des tâches nécessaires pour la réalisation d'un projet quelque soit le secteur d'activité. Il permet de visualiser l'avancement des tâches d'un projet de manière simple et concise, de planifier et suivre les besoins en ressources humaines et matérielles et donc de pouvoir suivre l'avancement du projet [s5].

Le diagramme de Gantt représenté par la Figure 1.7 montre le planning que nous avons opté pour mener à bien la réalisation de notre Smart TV. Nous avons pu définir six grandes étapes :

- Etude préliminaire : consiste à bien délimiter le périmètre du projet et déterminer les objectifs à atteindre dans notre future application en partant de l'existant. Nous avons estimé pour cette étape 15 jours.
- Spécification : nous précisons les principales solutions offertes par notre projet en tenant compte de ses besoins fonctionnels et non fonctionnels durant15 jours.
- Conception : il s'agit de la conception générale et détaillée du système. Nous avons estimé une durée approximative de 20 jours pour cette étape.
- Réalisation : implémenter des programmes, nous avons prévu pour cette étape 40 jours.
- Tests de validation : il s'agit de tester la conformité de l'application avec les buts spécifiés durant une semaine.
- Rédaction du rapport : contient une description détaillée de notre travail. Nous avons estimé pour cette étape 40 jours.

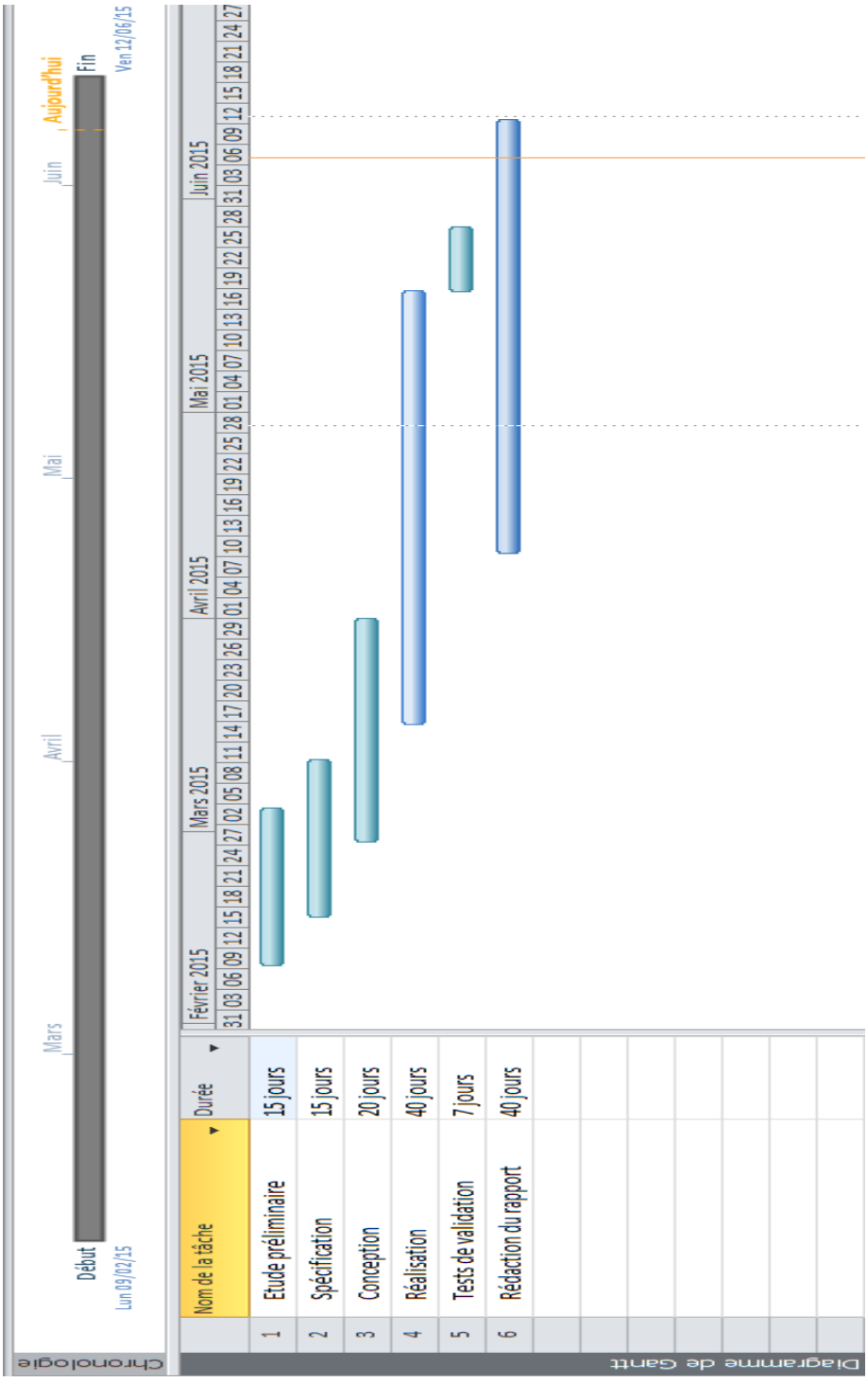


Figure 1.7: Chronogramme prévisionnel

Conclusion

Tout au long de ce chapitre, nous avons donné une présentation claire du cadre de travail. En outre, nous avons projeté l'objectif du système à développer. Pour cela, nous avons commencé par la présentation de l'organisme d'accueil et ses différents produits. Puis, nous avons indiqué l'étude de l'existant et la démarche de développement suivie pour l'élaboration du projet. Enfin, nous avons établit un planning prévisionnel à suivre. L'importance de ce chapitre réside dans l'introduction aux besoins fonctionnels et non fonctionnels qui seront analysés et spécifiés dans le chapitre suivant.

Chapitre 2: Analyse et spécification des besoins

Introduction

La réussite de tout travail dépend de la qualité de son départ. De ce fait, l'étape de spécification constitue le début de cette application. En outre, elle consiste à questionner l'utilisateur afin de dresser la liste de ces besoins qui devront être présents dans l'application puisque l'adéquation de tout travail à réaliser, aux besoins des utilisateurs et aux traitements envisagés au niveau de ses opérations assurera sa réussite et son utilisation future.

Pour ce faire, il y aura au cours de ce chapitre une présentation du sujet à réaliser en définissant les besoins fonctionnels et non fonctionnels et en les spécifiant en utilisant la notation UML.

1. Notation UML

UML est une norme de standardisation des applications développées selon le concept de programmation objet, élaborée en 1994 par Ivar Jacobson pour la société "Rational"[1].

Cette norme basée sur la modélisation et la formalisation de projets informatiques, permet de quantifier les besoins, ressources et relations existantes entres eux et facilite la réutilisation des composants programmés d'une application à l'autre.

UML est l'accomplissement de la fusion des précédents langages de modélisation objet Booch, OMT[16], OOSE[17]. Principalement issu des travaux de Grady Booch, James Rumbaugh et Ivar Jacobson, UML est à présent un standard défini par l'OMG [18][1].

Le choix d'UML a été basé sur ses capacités à exprimer et à élaborer de manière normalisée des modèles objet indépendamment de toute plateforme de réalisation.

UML a été conçu pour permettre la modélisation graphique afin de décrire les besoins, de spécifier et documenter les systèmes ainsi que d'esquisser les architectures logicielles. UML s'articule autour de neuf diagrammes différents divisés en deux catégories:

[16] OMT : Technique d'élaboration d'applications logicielles selon la méthode des objets et de l'encapsulation mise au point par James Rumbaugh de la firme Rational Software.

[17] OOSE : Méthode de développement conçue par Ivar Jacobson de la firme Rational Software se basant sur les techniques de programmation "Objet".

[18] OMG : Groupement international privé, fondé en 1989, définissant des standards relatifs à la programmation objet comme le modèle CORBA.

- **Diagrammes statiques (structurels):** diagramme de classes, d'objets, de composants, de déploiement et diagramme de cas d'utilisation.
- **Diagrammes dynamique (comportementaux):** diagramme d'activités, de séquence, d'états-transitions et diagramme de collaboration.

Les différents éléments représentables par UML sont :

- activité d'un objet/logiciel ;
- acteurs ;
- processus ;
- schéma de base de données ;
- composants logiciels ;
- et réutilisation de composants.

Dans, ce chapitre d'analyse et de spécification des besoins, nous utilisons deux types de diagrammes : un diagramme statique qui est le diagramme de cas d'utilisation qui permet d'éliciter les différents besoins fonctionnels et leurs relations avec les différents acteurs qui interagissent avec notre application, et un diagramme dynamique qui est le diagramme de séquence système qui sert de documentation aux cas d'utilisation élicités.

2. Analyse des besoins

Dans cette partie, nous identifions les besoins fonctionnels et non fonctionnels que doit satisfaire notre application. Nous rappelons qu'il s'agit d'une application qui présente les services d'une Smart TV. Ces besoins ont été définis en répondant au cahier des charges initial élaboré suite à plusieurs réunions de travail avec les responsables à OPCMA.

2.1 Besoins fonctionnels

Les besoins fonctionnels sont les principales fonctionnalités que notre application doit satisfaire. Notre application est une Smart TV qui offre les fonctionnalités suivantes :

- l'inscription des utilisateurs : si un utilisateur n'est pas enregistré, il saisit alors ces informations personnelles et il crée son profil pour accéder à l'application ;
- la personnalisation des services : les services seront personnalisé selon le profil de l'utilisateur ;
- la gestion du contenu de l'application: consultation, mise à jour de services ;
- la gestion des comptes utilisateurs : consultation, suppression, mise à jour.

Les services de cette Smart TV qui sont en outre des fonctionnalités assurées par cette Smart TV sont :

- les prévisions météo;
- les vidéos ;
- la musique ;
- le calendrier ;
- les programmes TV ;
- les jeux ;
- le chat ;
- la navigation sur Internet.

2.2 Besoins non fonctionnels

Les besoins non fonctionnels sont importants car ils agissent de façon indirecte sur le résultat et sur le rendement de l'utilisateur, ce qui fait qu'ils ne doivent pas être négligés.

Les besoins non fonctionnels de notre application que nous avons relevés sont:

- **Performance**: notre Smart TV doit être avant tout performante c'est-à-dire à travers ses fonctionnalités, elle doit répondre à toutes les exigences des usagers d'une manière optimale.
- **Convivialité**: notre Smart TV doit être facile à utiliser. En effet, les interfaces utilisateurs doivent être conviviales c'est-à-dire simples, ergonomiques et adaptées à l'utilisateur.
- **Temps de réponse**: l'application doit être rapide et légère afin de fournir une réponse en un temps raisonnable.
- **Robustesse**: notre Smart TV doit permettre le stockage des informations de tous les utilisateurs inscrits, aussi elle doit assurer une bonne gestion des erreurs.
- **Maintenabilité et extensibilité** : le code de l'application doit être lisible, compréhensible et bien commenté afin d'assurer son évolution et son extensibilité par rapport aux besoins futurs.
- **Sécurité:** la gestion des autorisations d'accès des différents utilisateurs sera par le forçage de mots de passe.

3. Spécification des besoins

Pour tout projet informatique, une bonne spécification des besoins est primordiale. Cette phase permet de décrire le comportement attendu de l'application. Elle a pour objectif de modéliser les besoins de l'application afin de définir le produit à développer dans son contexte d'utilisation.

Pour cette phase de spécification, nous allons utiliser les diagrammes de cas d'utilisation et les diagrammes de séquence système de la notation UML.

Les diagrammes de cas d'utilisation sont des diagrammes utilisés pour donner une vision globale du comportement fonctionnel d'un système logiciel [2]. Ils sont utiles pour des présentations auprès de la direction ou des acteurs d'un projet, mais pour le développement, les cas d'utilisation sont plus appropriés. Un cas d'utilisation représente une unité discrète d'interaction entre un utilisateur (humain ou machine) et un système [3]. Dans un diagramme de cas d'utilisation, les utilisateurs sont appelés acteurs, ils interagissent avec les cas d'utilisation.

Le diagramme de séquence système est une explication détaillée d'un cas d'utilisation, c'est la représentation graphique des interactions entre les acteurs et le système [s6]. Les principales informations contenues dans un diagramme de séquence sont : les messages échangés entre les lignes de vie, présentés dans un ordre chronologique.

3.1 Identifications des acteurs

Un acteur représente l'abstraction d'un rôle joué par des entités externes interagissant directement avec le système à mettre en œuvre : utilisateur, dispositif matériel ou autre système.

L'application que nous allons réaliser est destinée à toute personne intéressée par les services offerts par la Smart TV avec la possibilité de naviguer sur Internet.

Pour notre application, on peut distinguer deux acteurs : utilisateur (adulte, enfant) et un administrateur qui gère les comptes utilisateurs et le contenu de l'application.

3.2 Identification des cas d'utilisation

Un cas d'utilisation représente un ensemble de séquences d'actions réalisées par le système et produisant un résultat observable intéressant pour un acteur particulier [2].

Pour les acteurs identifiés précédemment, nous avons identifié les différents buts qu'ils cherchent à atteindre en utilisant le système.

Le Tableau 2.1 représente l'attribution des cas d'utilisation identifiés pour notre application aux différents acteurs.

Cas d'utilisation	**Acteurs**
S'authentifier	Administrateur
Gérer les comptes	
Gérer le contenu de l'application	
Consulter toutes les fonctionnalités de l'application	
S'authentifier	Utilisateur (adulte)
Consulter les prévisions météo	
Regarder des vidéos	
Ecouter de la musique	
Consulter le calendrier	
Chatter	
Consulter les programmes TV	
Naviguer sur Internet	
Jouer	Utilisateur (enfant)
Regarder des vidéos pour enfant	
Ecouter de la musique pour enfant	
Consulter les programmes TV pour enfant	

Tableau 2.1: Attribution des cas d'utilisation aux acteurs

Par la suite, nous illustrons graphiquement ce tableau par un diagramme de cas d'utilisation global pour chaque acteur.

Nous commençons par présenter le diagramme de cas d'utilisation pour l'administrateur illustré par la Figure 2.1.

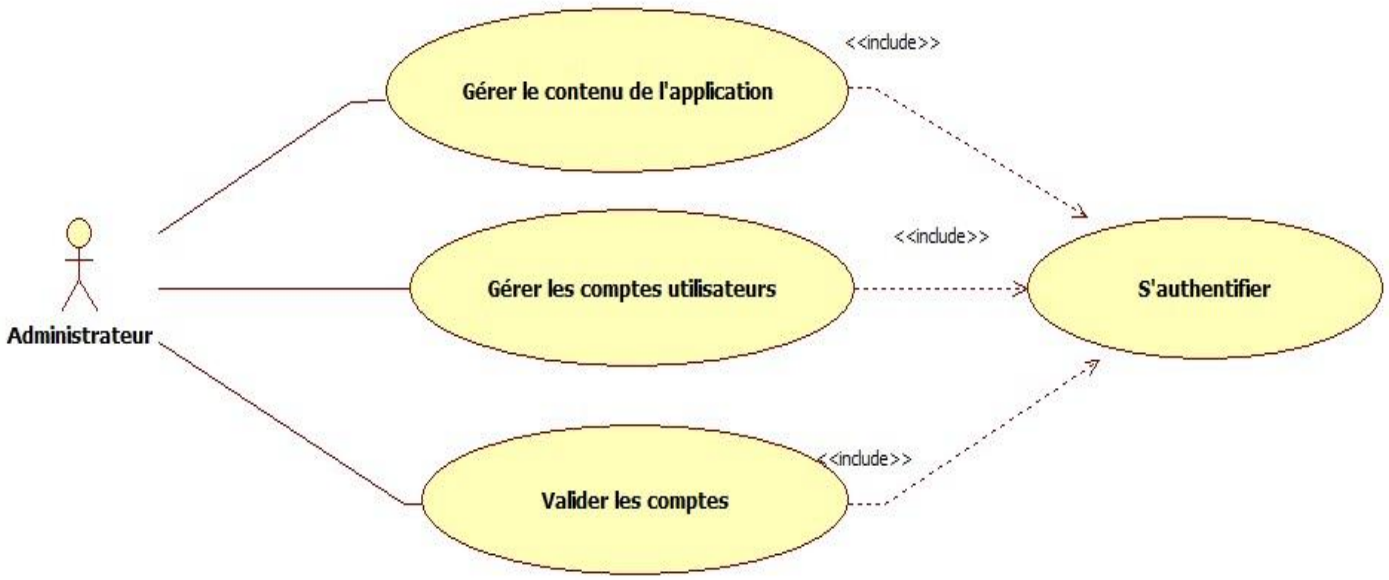


Figure 2.1: Diagramme de cas d'utilisation pour l'acteur administrateur

Puis nous présentons le diagramme de cas d'utilisation pour l'acteur adulte.

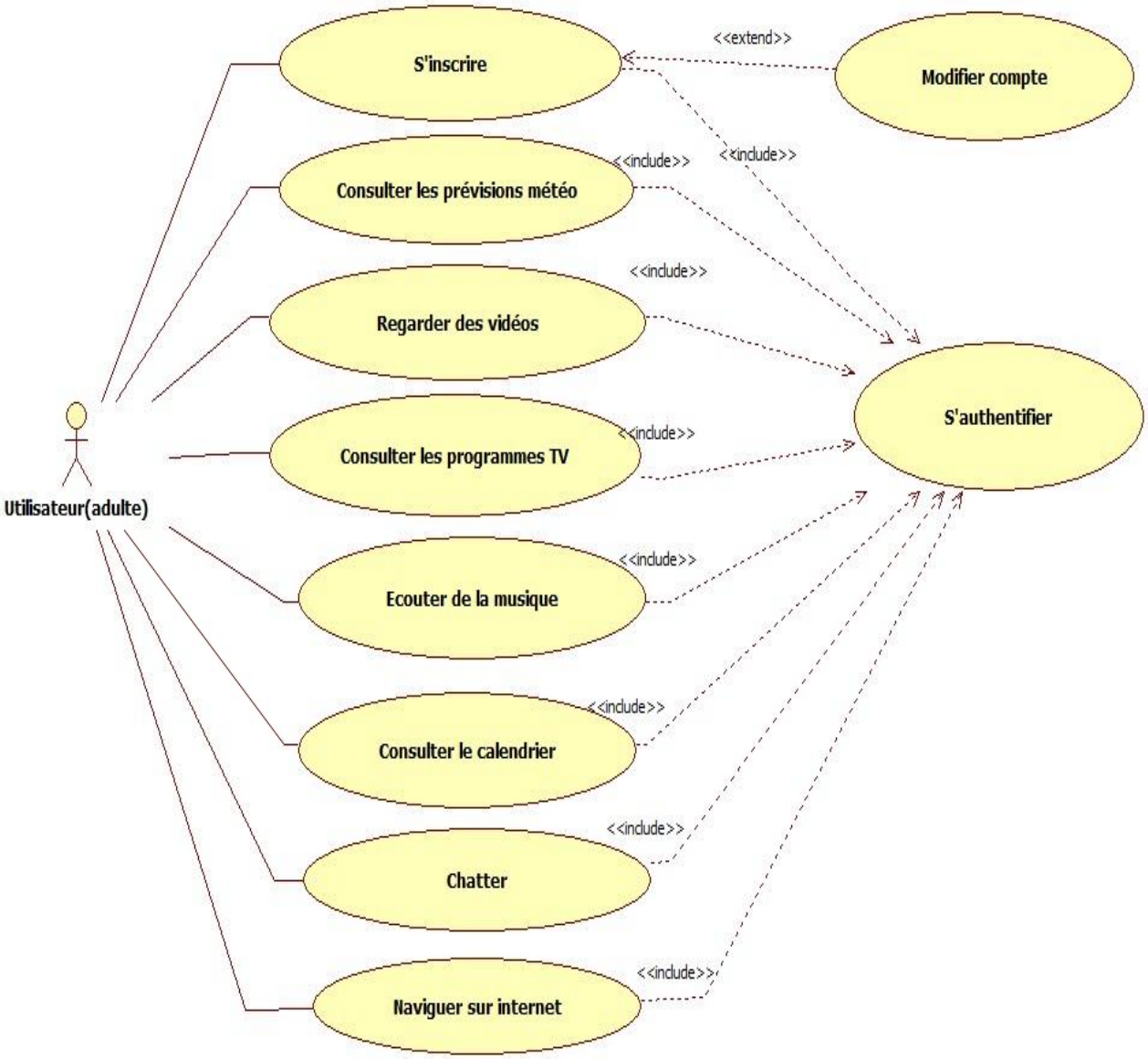


Figure 2.2: Diagramme de cas d'utilisation pour l'acteur adulte

Enfin, nous illustrons le digramme de cas d'utilisation pour l'enfant par la Figure 2.3

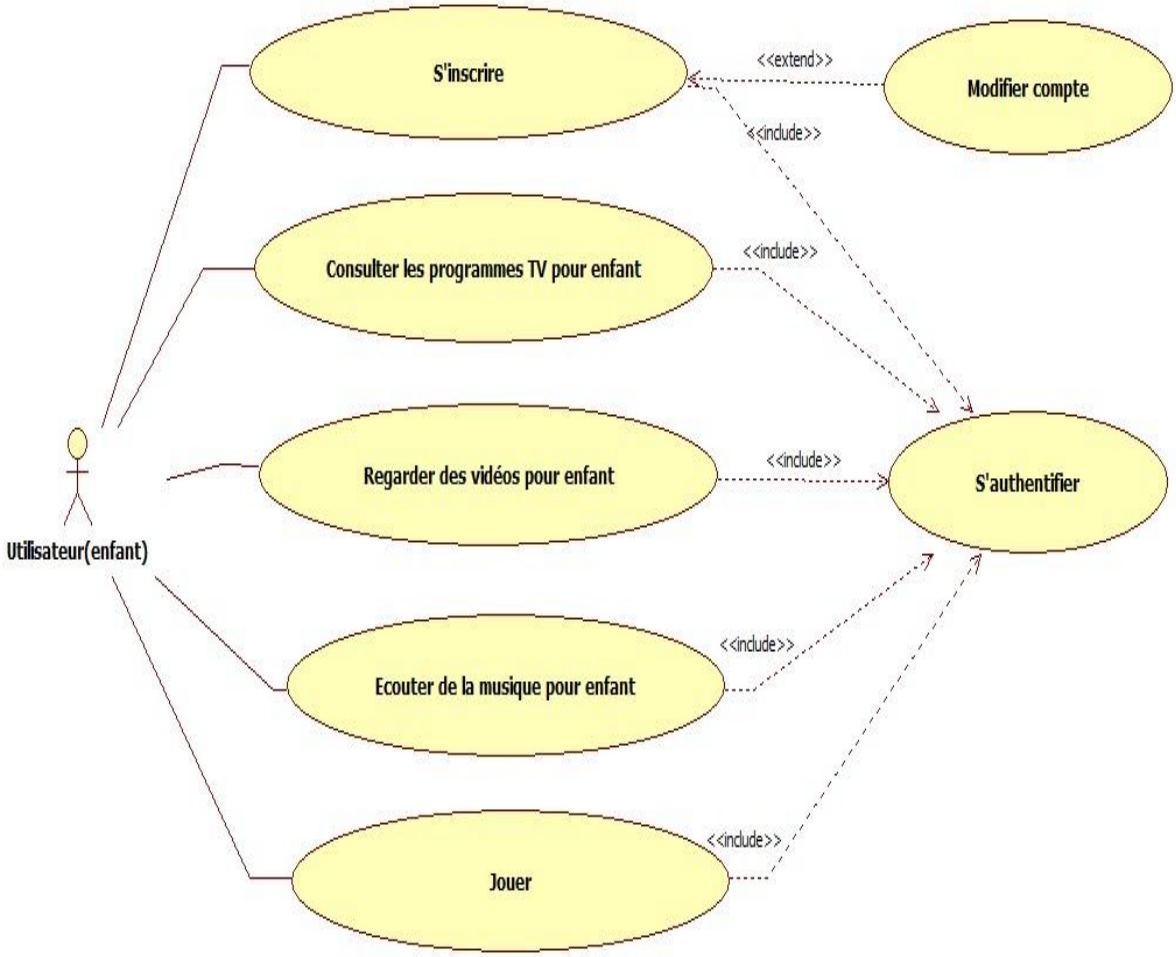


Figure 2.3 : Diagramme de cas d'utilisation pour l'acteur enfant

Les diagrammes représentés par la Figure 2.1, la Figure 2.2 et la Figure 2.3 exposent les différentes fonctionnalités que doit offrir notre application à chacun des acteurs : l'enfant, l'adulte et l'administrateur.

Dans la suite, nous allons présenter les diagrammes de cas d'utilisation détaillés.

4. Raffinement des cas d'utilisation

L'objectif de cette partie est d'accéder à une compréhension plus approfondie des besoins et des exigences et d'en livrer une description facile à entretenir.

Dans cette section, nous allons essayer de représenter le système de point de vue fonctionnel en utilisant principalement des raffinements de cas d'utilisation, des descriptions textuelles de chaque cas d'utilisation et des diagrammes de séquences systèmes.

Il s'agit donc d'associer à chaque cas d'utilisation un nom, un résumé, les acteurs qui y participent, les pré-conditions et des scénarii. Cependant ; il existe trois types de scénarii : les scénarii nominaux ; les scénarii d'exceptions et les scénarii alternatifs. Dans notre description textuelle, nous présentons seulement les scénarii nominaux et alternatifs.

4.1 Cas d'utilisation : « S'authentifier »

Pour ce cas d'utilisation, le système affiche la page de connexion, nous présentons la description textuelle de ce cas d'utilisation.

Nom: s'authentifier.

Résumé : chaque utilisateur doit taper son propre nom d'utilisateur et son mot de passe pour accéder à l'interface qui le concerne. L'administrateur accède au système pour effectuer des modifications, alors que l'utilisateur simple accède aux fonctionnalités offertes par notre application qui s'approprient avec son âge.

Acteurs : administrateur**,** utilisateur (enfant ou adulte).

Pré-conditions: l'utilisateur doit être créé dans la base de données et connaître ses identifiants, il introduit son login et mot de passe.

Scénario nominal :

1. l'utilisateur saisit ses paramètres de connexion (login et mot de passe).

2. le système vérifie les paramètres saisis (champs vides, utilisateur inexistant, etc.).

3. l'utilisateur va accéder à la session qui le concerne.

Scénario alternatif :

2a. le système indique la non-validité des coordonnées de l'utilisateur, ce dernier ne peut pas accéder à sa session.

L'enchaînement reprend à l'étape 1 du scénario nominal, l'utilisateur peut s'authentifier une autre fois, ou il a la possibilité de créer un compte.

3a. le système affiche le menu spécifique pour chaque utilisateur. Si l'utilisateur est un mineur il sera redirigé vers l'interface dédié pour les enfants. Dans le cas contraire, l'utilisateur sera redirigé vers l'interface principale de notre Smart TV.

Post conditions: authentification validée et succès d'accès.

4.2 Cas d'utilisation : «S'inscrire»

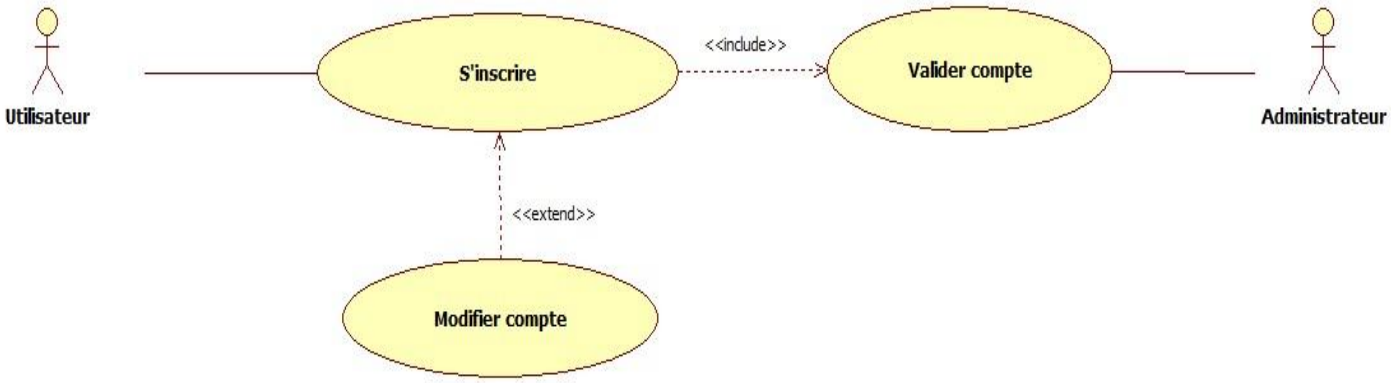


Figure 2.4 : Raffinement du cas d'utilisation «S'inscrire»

Dans ce qui suit, nous présentons la description textuelle de ce cas d'utilisation.

Nom: s'inscrire.

Résumé: chaque utilisateur doit s'inscrire au système pour profiter des fonctionnalités de notre application selon son âge.

Acteurs: utilisateur (enfant ou adulte), administrateur.

Pré-conditions: Interface d'inscription active.

Scénario nominal :

1. l'utilisateur demande au système l'ajout d'un nouveau compte.
2. l'utilisateur saisit ses coordonnées : nom, âge, e-mail, pays, identifiant et mot de passe.
3. le système vérifie les paramètres saisis (champs vides, âge non numérique, utilisateur existant, etc.).
4. l'utilisateur valide l'ajout du compte.
5. l'administrateur valide le compte.
6. l'application enregistre les nouvelles données saisies.

Scénario alternatif :

2a le système affiche un message d'échec d'ajout d'un utilisateur qui indique que les champs sont erronés ou manquants ou l'utilisateur à ajouter existe déjà dans l'application.

L'enchaînement reprend à l'étape 1 du scénario nominal ; l'utilisateur peut renouveler la saisie comme il peut l'abandonner.

Post conditions : Le système crée le nouveau compte et l'utilisateur est enregistré.

4.3 Cas d'utilisation : « Gestion des comptes utilisateurs »

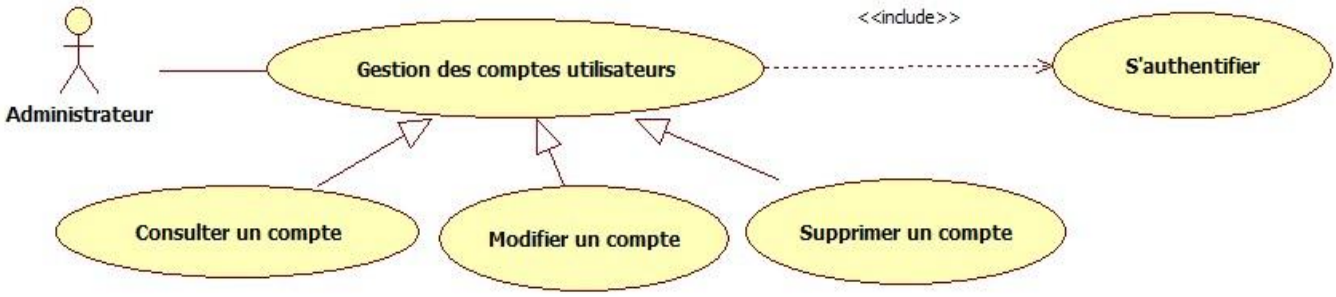


Figure 2.5: Raffinement du cas d'utilisation «Gestion des comptes utilisateurs»

Ce cas d'utilisation permet essentiellement à l'administrateur de gérer les comptes utilisateurs (consulter, modifer ou supprimer).

a. Cas d'utilisation : « Consulter un compte»

L'administrateur a le droit de consulter un compte utilisateur.

Ci-dessous la description textuelle ce cas d'utilisation.

Nom: consulter un compte.

Résumé: consultation des informations sur un compte.

Acteurs: administrateur.

Pré-conditions: administrateur authentifié.

Scénario nominal :

1. le système affiche la liste des utilisateurs.
2. l'administrateur sélectionne un compte utilisateur a consulté.
3. le système affiche le compte.

Post-condition: le compte utilisateur est consulté.

b. Cas d'utilisation : « Modifier un compte»

L'administrateur a le droit de modifier un utilisateur.

Ci-dessous la description textuelle du cas d'utilisation "modifier un compte".

Nom: modifier un compte.

Résumé: faire enregistrer les modifications du compte, afin de mettre à jour les données.

Acteurs : administrateur.

Pré-conditions : l'administrateur s'est authentifié correctement à l'application.

Le compte à modifier existe déjà.

Scénario nominal :

1. le système affiche la liste des utilisateurs dans laquelle l'administrateur peut choisir l'utilisateur à modifier.
2. un formulaire se charge après la sélection d'un utilisateur contenant ces anciennes valeurs des champs à modifier.
3. modifier le compte.
4. valider la modification.
5. le système enregistre les données modifiées et affiche un message indiquant la confirmation de modification.

Enchaînements alternatifs :

4a. le système affiche un message d'échec de modification en indiquant les champs erronés.

L'enchaînement reprend à l'étape 3 du scénario nominal, l'administrateur peut renouveler la saisie comme il peut l'abandonner.

Post conditions: un utilisateur est modifié avec succès.

c. Cas d'utilisation : « Supprimer un compte»

L'administrateur a le droit de supprimer un utilisateur.

La description textuelle de ce cas d'utilisation est la suivante :

Nom: supprimer un compte.

Résumé: supprimer les données relatives à un utilisateur.

Acteurs: administrateur.

Pré-conditions: l'administrateur s'est authentifié correctement à l'application.

Le compte à supprimer existe déjà.

Scénario nominal :

1. le système affiche la liste des utilisateurs.
2. l'administrateur sélectionne un ou plusieurs utilisateurs à supprimer.

3. l’administrateur valide la suppression de l'utilisateur.
4. l'application affiche un message de confirmation de la suppression.
5. l’administrateur confirme la suppression.
6. l’application effectue la surpression.

Post conditions : Un utilisateur est supprimé avec succès.

4.4 Cas d'utilisation : « Consulter les prévisions météo»

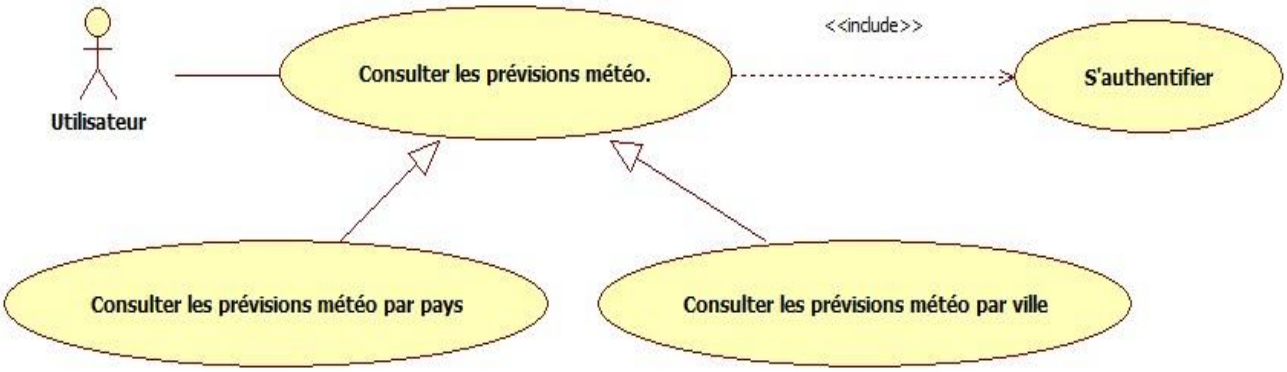


Figure 2.6: Cas d’utilisation «Consulter les prévisions météo»

La description textuelle du scénario est la suivante :

Nom : consulter les prévisions météo.

Résumé: consultation des prévisions météo par pays et par ville.

Acteurs : utilisateur.

Pré conditions : l’utilisateur doit être authentifié.

Nous devons disposer d’une connexion Internet.

Scénario nominal :

1. l’utilisateur demande au système de lui retourner des prévisions météo.
2. l’utilisateur sélectionne un pays puis une ville.
3. le système affiche la météo de la ville sélectionné.

Post-condition : le service de prévisions météo est consulté.

4.5 Cas d'utilisation : « Regarder les vidéos»

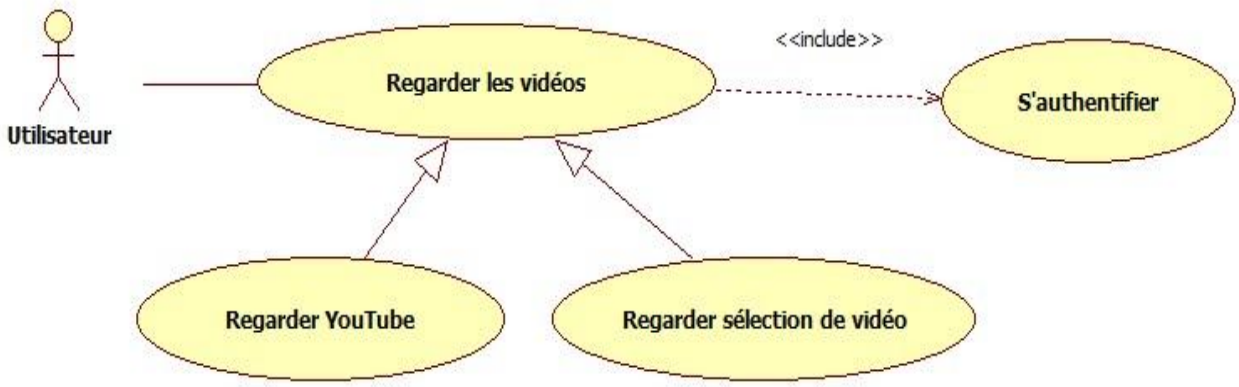


Figure 2.7: Raffinement du cas d'utilisation «Regarder les vidéos»

Ce cas d'utilisation permet essentiellement à l'utilisateur de regarder des vidéos soit en cherchant une vidéo ou une chaîne via YouTube ou en sélectionnant une vidéo parmi une sélection offerte par l'application.

a. Cas d'utilisation : « Regarder YouTube»

La description textuelle de ce cas d'utilisation est la suivante :

Nom : regarder YouTube.

Résumé: choisir de regarder en ligne des vidéos ou des chaînes via YouTube.

Acteurs: utilisateur.

Pré-conditions: l'utilisateur doit être authentifié.

Nous devons disposer d'une connexion Internet.

Scénario nominal :

1. l'utilisateur accède au service "vidéo".
2. l'utilisateur cherche une vidéo ou une chaîne via YouTube.
3. le système affiche la vidéo ou la chaîne choisie.

Post-condition : la vidéo ou la chaîne choisie est regardée.

b.Cas d'utilisation : « Regarder une sélection de vidéos»

Ci-après la description textuelle de ce cas d'utilisation :

Nom : regarder la sélection de vidéo.

Résumé: consultation des vidéos contenues dans notre application.

Acteurs: utilisateur.

Pré-conditions: utilisateur authentifié.

Scénario nominal:

1. l'utilisateur demande au système de lui afficher les vidéos.
2. l'utilisateur sélectionne une vidéo contenue dans le système.
3. le système affiche la vidéo.

Post-condition : la vidéo choisie est regardée.

5. Diagrammes des séquences

Vu la diversité des diagrammes de séquence, nous présentons dans cette section quelques exemples qui reflètent le fonctionnement de l'application. Nous nous concentrons sur l'inscription, l'authentification et la gestion des utilisateurs.

5.1 Diagramme de séquence : « S'inscrire »

Description du scénario : pour bien profiter des privilèges dédiés aux utilisateurs, un visiteur doit d'abord entamer la phase d'inscription avec succès et pour cela il faut qu'il passe par l'ensemble des séquences suivantes que nous avons présentées dans la Figure 2.8:

- le visiteur demande le formulaire d'inscription ;
- le formulaire s'affiche ;
- le visiteur rempli le formulaire ;
- une vérification de l'existence d'utilisateur dans la base se lance ;
- si le visiteur existe déjà un message s'affiche ;
- si c'est un nouveau visiteur l'inscription sera validé.

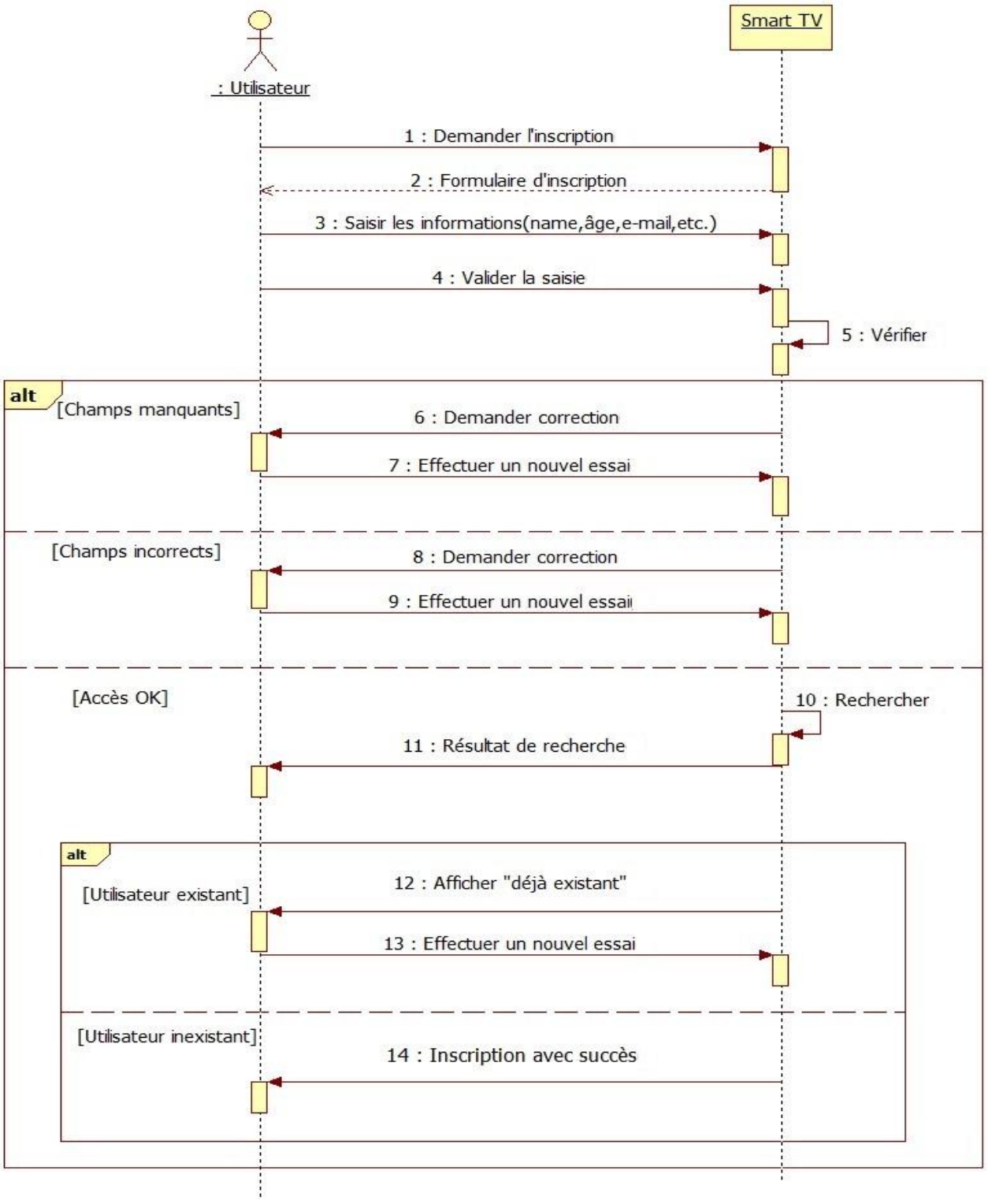


Figure 2.8: Diagramme de séquence « S'inscrire »

5.2 Diagramme de séquence : « S'authentifier »

Description du scénario : l'utilisateur lance l'application, le système lui affiche un formulaire d'authentification où il saisit son login et son mot de passe. Puis, le système vérifie leurs validités et affiche le menu spécifique pour chaque utilisateur. Si l'utilisateur est un mineur il sera redirigé vers l'interface dédié pour les enfants, dans le cas contraire l'utilisateur sera redirigé vers l'interface principale de notre Smart TV.

Le diagramme de séquence concernant l'authentification est illustré par la Figure 2.9.

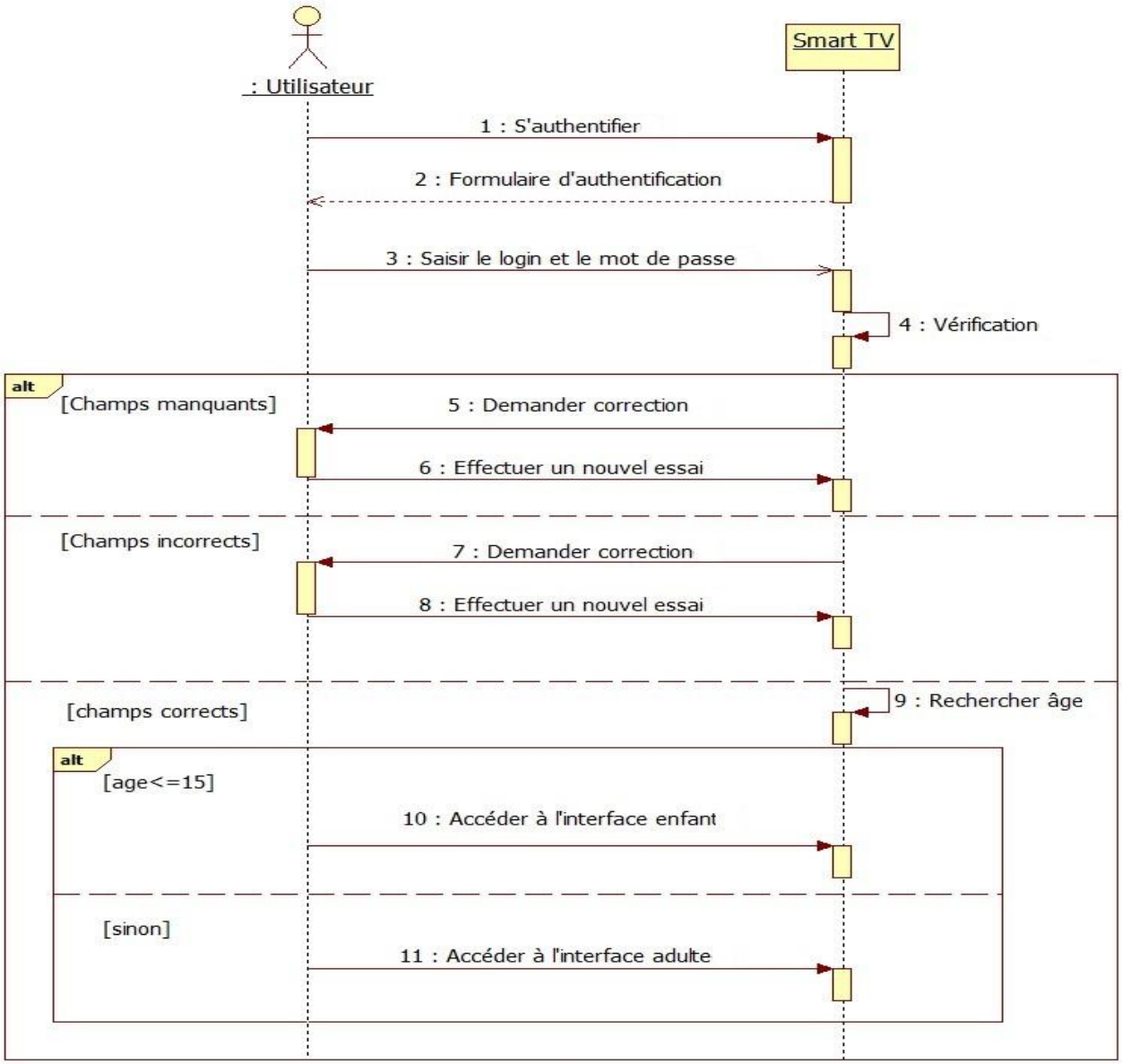


Figure 2.9 : Diagramme de séquence « S'authentifier »

5.3 Diagramme de séquence : « Gérer les utilisateurs »

Description du scénario : dès que la phase d'authentification est effectuée avec succès, le système affiche à l'administrateur l'interface de gestion. Quand l'administrateur choisit de gérer les utilisateurs la liste d'utilisateurs s'affiche dans un tableau ainsi que les différentes actions qu'il peut effectuer. Ce dernier choisit l'action à exécuter. S'il s'agit d'une consultation, l'administrateur doit sélectionner un utilisateur de la liste. Un tableau qui contient les données de l'utilisateur sélectionné est affiché. S'il s'agit d'une suppression il faut sélectionner un utilisateur de la liste pour le supprimer, le système averti l'administrateur de l'opération de suppression, ce dernier confirme la suppression, l'opération de suppression se termine avec succès. Et s'il s'agit d'une modification, l'administrateur doit sélectionner un utilisateur de la liste, un formulaire qui contient les données de l'utilisateur sélectionné est affiché et l'administrateur peut modifier ce qu'il veut. Après avoir terminé la modification, le système effectue une vérification sur les champs modifiés et un message d'erreur s'affiche si l'une des contraintes de saisie est violée, sinon comme dernière étape, la confirmation et l'enregistrement des informations. La Figure 2.10 illustre le diagramme de séquence qui résume cette partie.

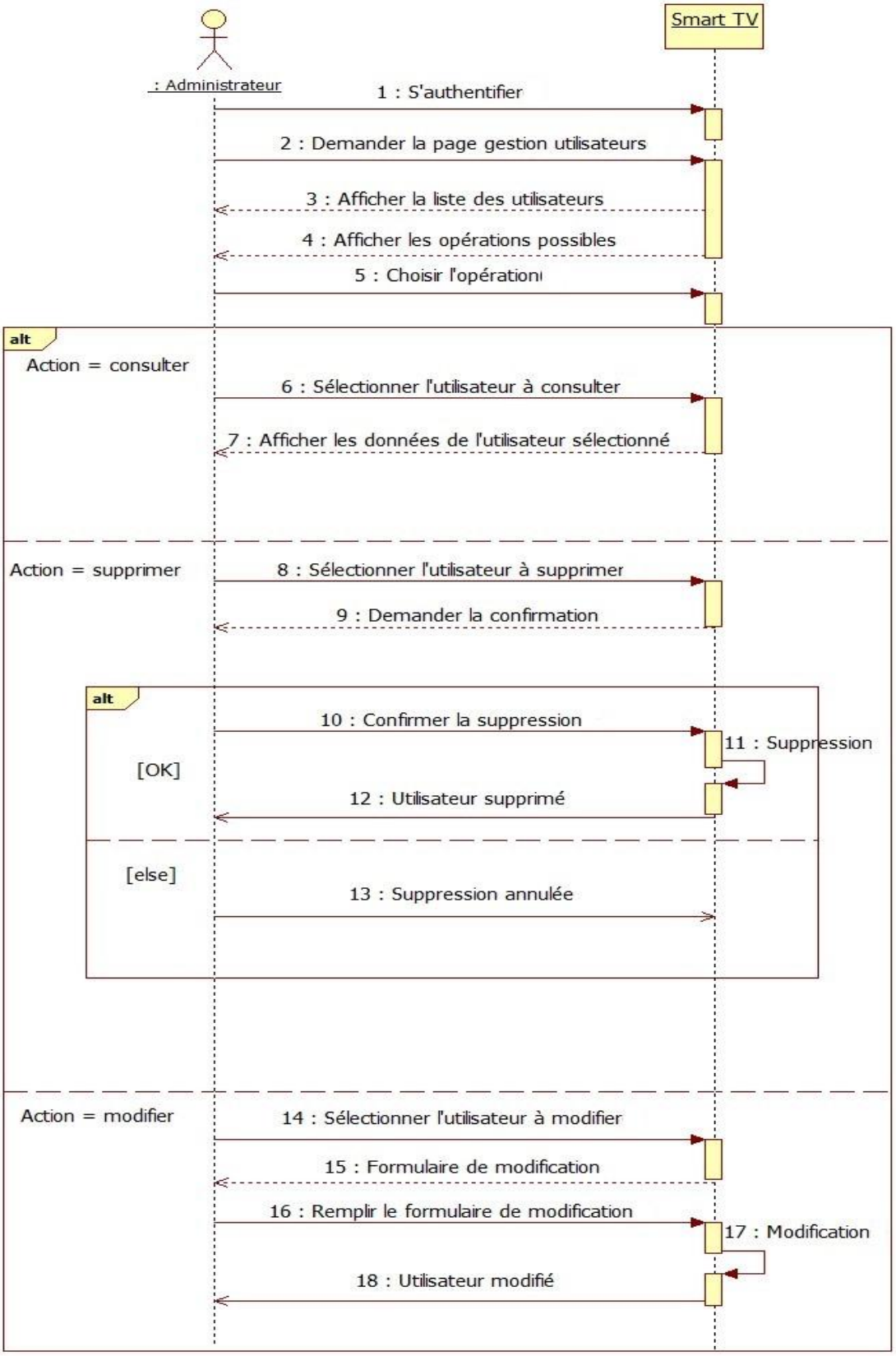


Figure 2.10 : Diagramme de séquence « Gérer les utilisateurs »

Conclusion

Dans ce chapitre, nous avons identifié et spécifié les fonctionnalités que doit offrir l'application à réaliser. A présent nous sommes prêts à passer à l'étape de conception dans le chapitre suivant.

Chapitre 3: Conception

Introduction

Dans ce chapitre nous allons élaborer une étude conceptuelle détaillée de notre application Smart TV. C'est une phase primordiale et déterminante pour produire une application de haute qualité puisqu'elle prépare à l'implémentation. Elle permet de décider comment satisfaire les besoins dégagés dans la phase de spécification.

En premier lieu, nous présentons l'architecture de l'application. En second lieu, nous allons décrire les différents scénarios à l'aide de diagrammes UML.

1. Conception architecturale du système

L'architecture joue un rôle important dans la conception du logiciel ; elle représente le point de pivot autour duquel s'articulent tous les composants de l'application à développer. L'adoption d'une bonne architecture facilite la compréhension des besoins et des fonctions.

1.1 Vue physique

Les choix architecturaux d'une application sont décisifs dès lors qu'ils interviennent sur les performances, l'évolutivité, le temps de développement, et bien sûr le coût. Aujourd'hui nous parlons d'une séparation des applications en différentes niveaux, et nous parlons alors d'applications multi niveaux.

Notre application est distribuée sur trois niveaux : client, serveur Web et le SGBD. C'est pour cela, nous adoptons l'architecture à trois niveaux appelées aussi architecture 3-tiers qui est représentée par la Figure 3.1.Généralement cette architecture est partagée entre :

1. Un client qui assure la gestion de l'affichage avec un simple navigateur Web.
2. Un serveur d'applications ou middleware ou encore serveur intermédiaire, qui gère la logique d'application.
3. Un serveur de données où est stockée la base de données.

Figure 3.1: Architecture 3-tiers

Les avantages de l'architecture 3-tiers sont principalement [s7] :

- les clients sont légers ;
- les requêtes clients vers le serveur sont d'une plus grande flexibilité que dans celles de l'architecture 2-tiers, cette flexibilité permet à une entreprise d'envisager dans le cadre d'une architecture 3- tiers une grande souplesse pour l'introduction de toutes nouvelles technologies ;
- d'un point de vue développement, la séparation qui existe entre le client, le serveur et le SGBD permet une spécialisation des développeurs sur chaque tiers de l'architecture ;
- les modifications sur les applications ne posent pas le problème de mise à jour ;
- une plus grande sécurité pour chaque service ;
- l'amélioration des performances (les tâches sont partagées).

1.2 Vue logique

Le modèle de conception qui présente la vue logique de notre application est le MVC (Model View Controller) qui est un modèle très commun pour développer des applications munies d'une interface graphique manipulant des données persistantes.

C'est une architecture et une méthode de conception qui stipule la division des fonctions nécessaires de l'application en trois parties:

- **Modèle** : représente le comportement de l'application : traitements des données et interactions avec la base de données. Il décrit les données manipulées par l'application et définit les méthodes d'accès.
- **Vue :** correspond à l'interface avec laquelle l'utilisateur interagit. Les résultats renvoyés par le modèle sont dénués de toute présentation mais sont présentés par les vues.

Elle peut être conçue en html, ou tout autre langage de présentation. La vue n'effectue aucun traitement, elle se contente d'afficher les résultats des traitements effectués par le modèle, et de permettre à l'utilisateur d'interagir avec elles.

- **Contrôleur :** prend en charge la gestion des évènements de synchronisation pour mettre à jour la vue ou le modèle. Il n'effectue aucun traitement, ne modifie aucune donnée, il analyse la requête du client et se contente d'appeler le modèle adéquat et de renvoyer la vue correspondante à la demande [s8].

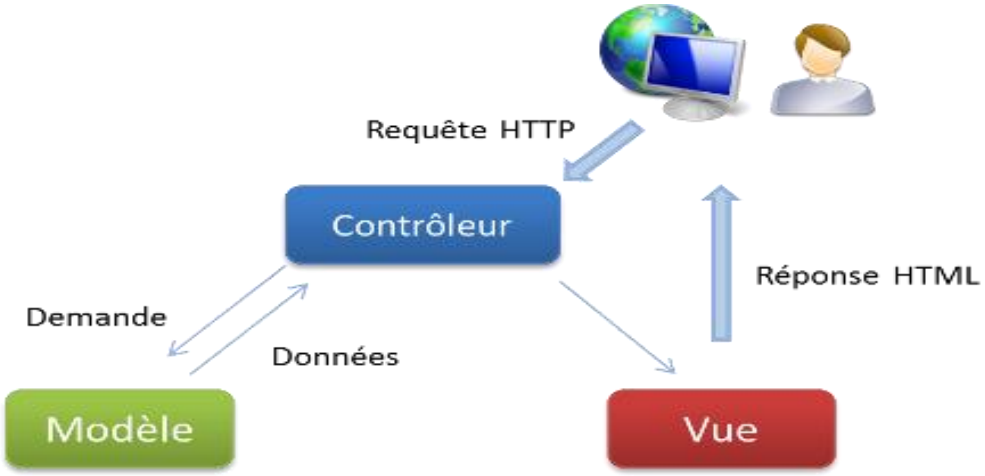


Figure 3.2 Le modèle MVC

Sur la Figure 3.2, un internaute se connecte sur un site Web. Le contrôleur est alors appelé pour traiter la demande de l'internaute. Pour cela, le contrôleur utilise le modèle s'il y a des données à manipuler, il renvoi ensuite les informations nécessaires à la vue qui est affichée au client.

Nous avons choisi de travailler avec l'architecture MVC car elle permet de bien séparer la logique de la présentation. La vue n'aura aucune logique d'imbriquer. Aussi, étant donné que tout est très bien séparé, il est très facile d'ajouter et de modifier du code sans que le reste ne s'effondre.

2. Conception détaillée

2.1 Vue statique

Pour la modélisation statique de notre système nous allons présenter le diagramme de composants.

2.1.1 Diagramme de composants

Un diagramme de composants propose une vision statique de l'organisation du système du point de vue des éléments logiciels comme les modules (fichiers sources, bibliothèques, exécutables), des données (fichiers, bases de données) ou encore d'éléments de configuration (paramètres, scripts, fichiers de commandes). Ce diagramme permet de mettre aussi en évidence les dépendances entre les composants [s9].

En se basant sur le modèle d'architecture MVC, le diagramme de composants représenté par la Figure 3.3 montre les différents types de composants de notre application.

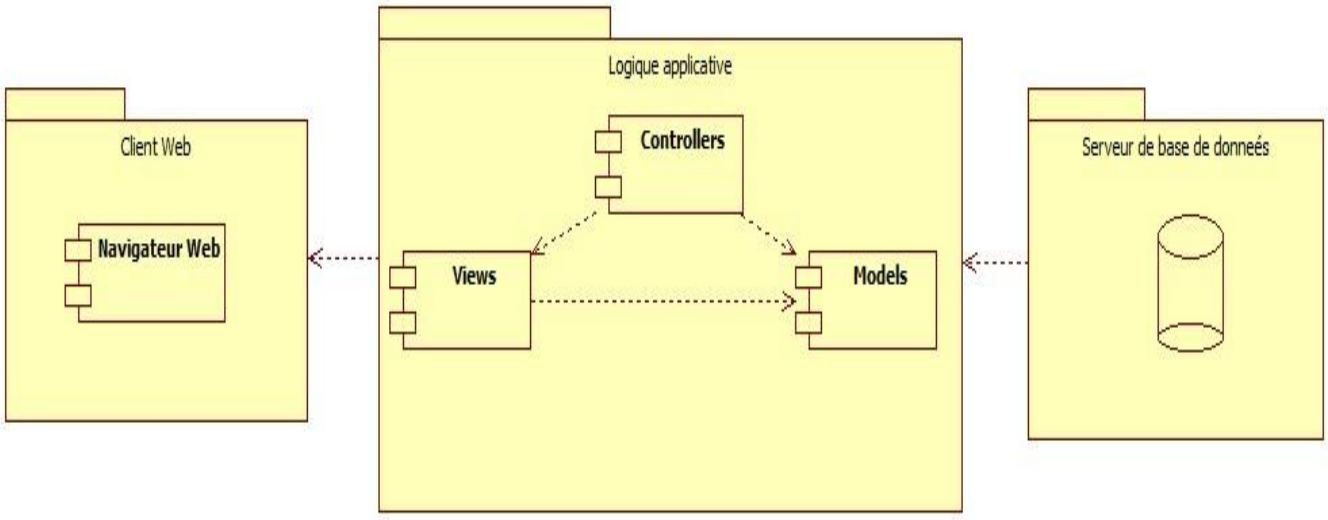


Figure 3.3: Diagramme de composants de notre application Smart TV

Ce diagramme, nous explique comment interagissent les différents composants dans notre application, nous avons :

- le client : représente le navigateur qui communique avec le serveur.
- la logique applicative : correspond à la partie fonctionnelle de l'application, elle est responsable de la gestion des sessions utilisateurs et la gestion du contenu de l'application, elle contient trois élément : modèle, contrôleur et vue.
- le serveur base de données : représente la partie centrale qui gère la base de données et ses accès.

2.2 Vue dynamique

Dans cette partie, nous allons utiliser deux types de diagrammes : les diagrammes d'activités et les diagrammes de séquence représentant les interactions entre objets.

Un diagramme d'activité permet de modéliser un processus interactif global ou partiel pour un système donné .C'est un diagramme, associé à un objet particulier ou à un ensemble d'objets, qui illustre les flux entre les activités et les actions. Il est recommandable pour exprimer une dimension temporelle sur une partie du modèle, à partir des diagrammes de cas d'utilisation. C'est une représentation proche de l'organigramme ; la description d'un cas d'utilisation par un diagramme d'activités correspond à sa traduction algorithmique. Une activité est l'exécution d'une partie du cas d'utilisation, elle est représentée par un rectangle aux bords arrondis. [s10].

Les diagrammes de séquence représentant les interactions entre objets sont particulièrement utiles au concepteur pour représenter graphiquement ces décisions d'allocations des responsabilités. Chaque diagramme va représenter un ensemble d'objets collaborant dans le cadre d'un scénario d'exécution du système. Les objets communiquent en s'envoyant des messages qui invoquent des opérations sur les objets récepteurs. Il est ainsi possible de suivre visuellement les interactions dynamiques entre objets, et les traitements réalisés par chacun d'eux. Par rapport aux diagrammes de séquences système, nous remplaçons le système, vu comme une boîte noire, par un ensemble d'objets en collaboration. Ces objets sont des dialogues, des contrôles et des entités [3].

2.2.1 Scénario d'authentification

Pour accéder à notre application, la vue d'authentification s'affiche, l'utilisateur doit entrer son login et son mot de passe. Après la vérification des champs saisis le contrôleur interroge la base de données pour charger les privilèges accordés à l'utilisateur. Il accède alors automatiquement à son interface d'accueil. Si l'utilisateur est un mineur le contrôleur le redirige vers la vue dédié pour les enfants, dans le cas contraire le contrôleur le redirige vers la vue principale de notre Smart TV.

Le processus d'authentification peut être résumé dans le diagramme d'activités présenté par la Figure 3.4.

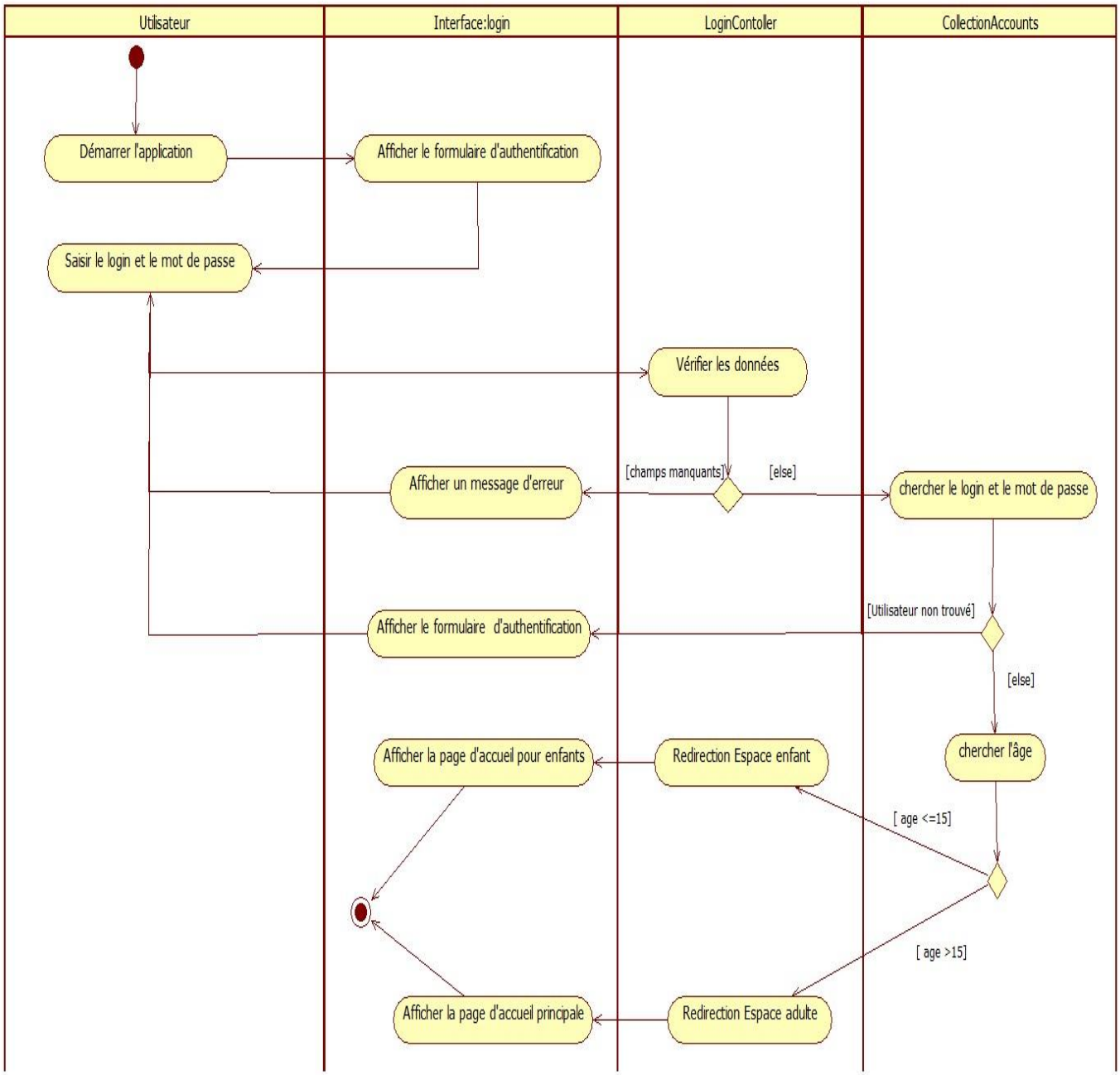


Figure 3.4: Diagramme d'activités du scénario « S'authentifier »

Le scénario d'authentification cité ci-dessus est décrit plus précisément par le diagramme de séquence objets de la Figure 3.5 :

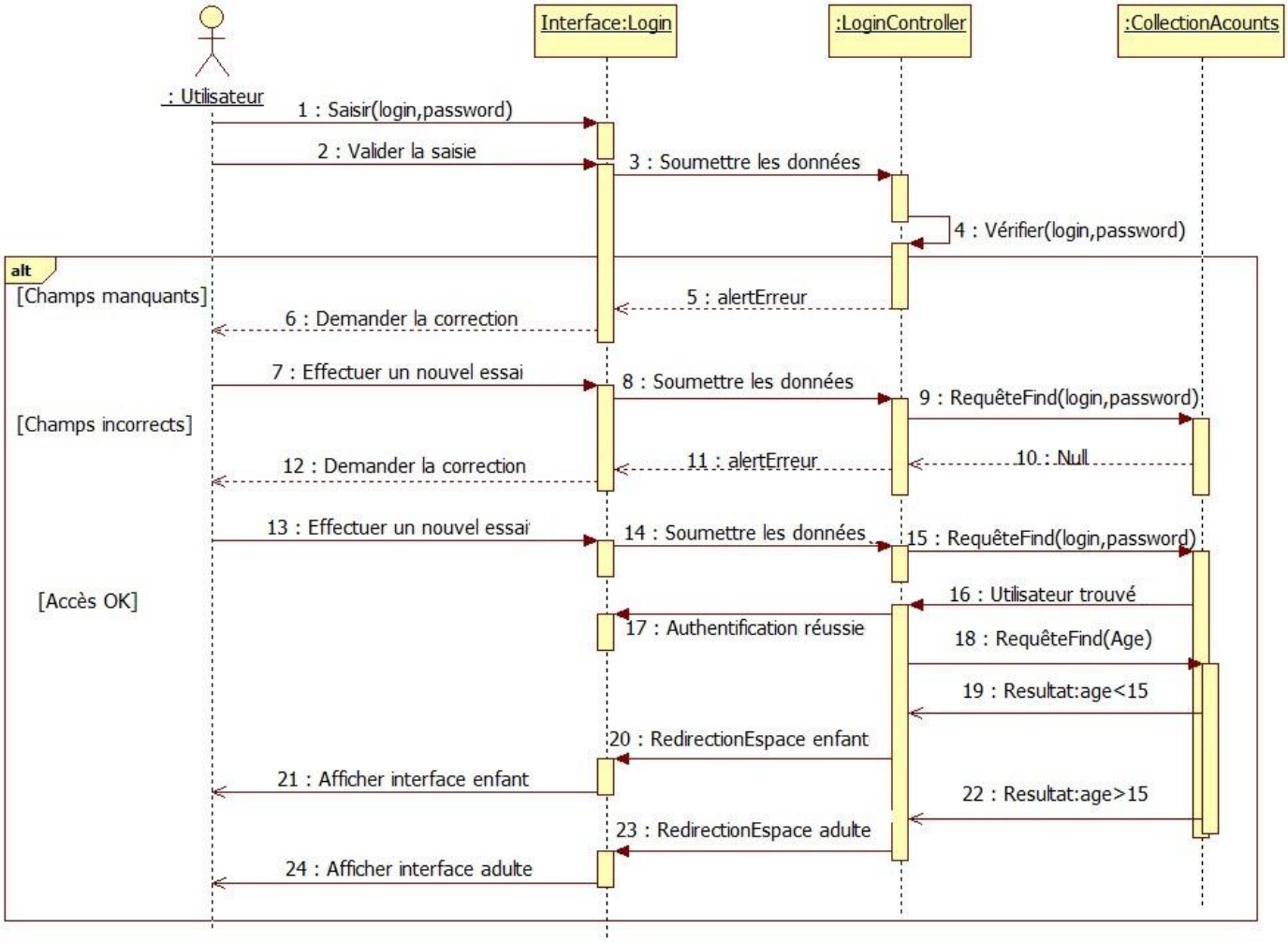


Figure 3.5 : Diagramme de séquence objets du scénario « S'authentifier »

Ce diagramme de séquence décrit la communication et la propagation des messages entre les composants de l'application. L'acteur interagit d'une façon directe avec la vue, en lui envoyant une requête de connexion. La vue soumit l'action au contrôleur qui vérifie les champs, ce dernier retourne le résultat de la requête et la transmettra à la vue où elle sera affichée et lisible par l'utilisateur.

2.2.2 Scénario d'inscription

La phase d'inscription est indispensable pour bénéficier de la totalité des services de notre application Smart TV.

Pour cela l'utilisateur doit passer par l'ensemble des étapes que nous allons simplifier par le diagramme d'activité présenté par la Figure 3.6:

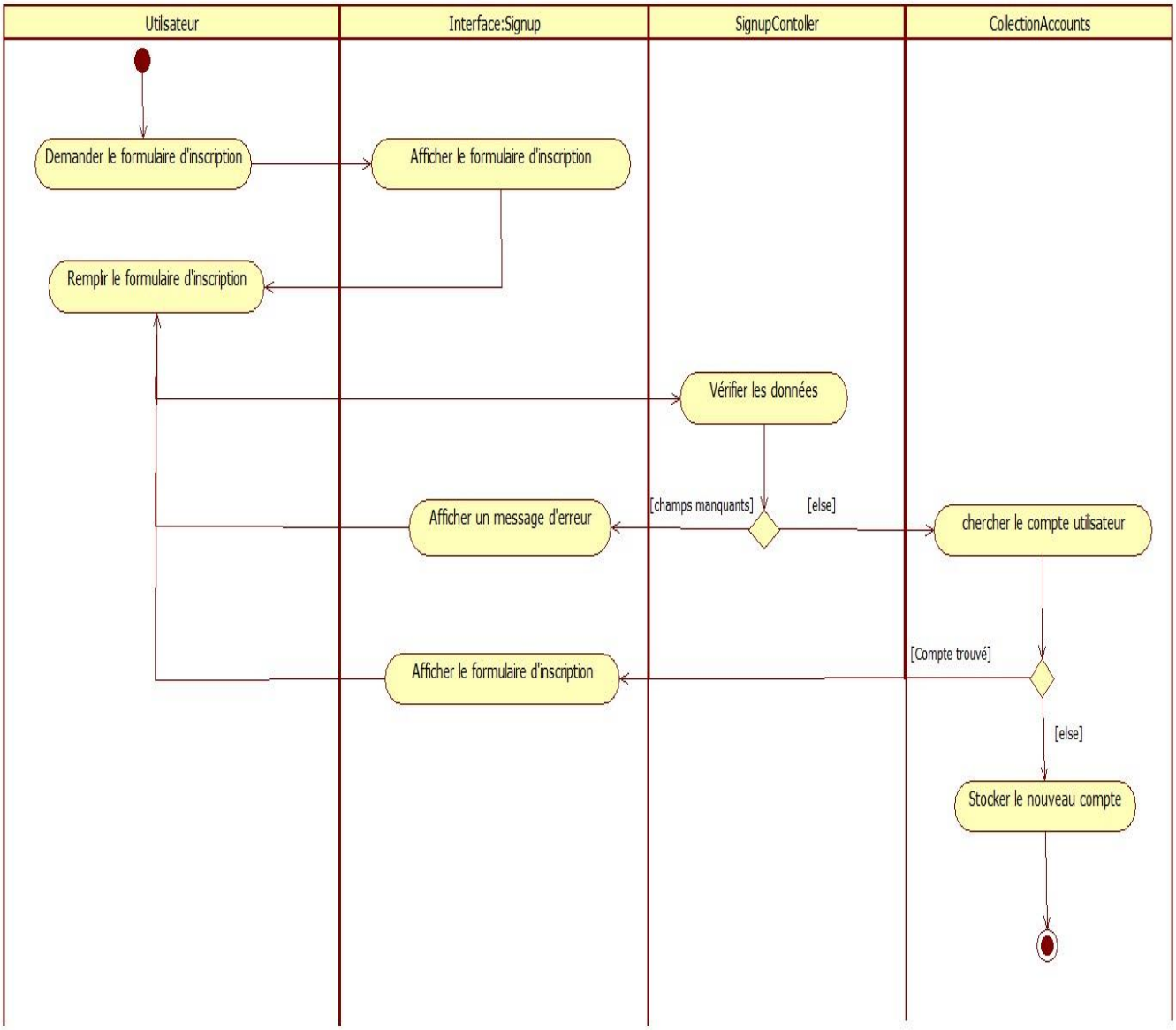


Figure 3.6 : Diagramme d'activité du scénario « S'inscrire »

Ce scénario est décrit suivant les étapes suivantes :

- le visiteur demande la vue d'inscription ;
- le formulaire d'inscription s'affiche ;
- le visiteur remplit les champs demandé dans le formulaire ;
- la vue transmet l'opération au contrôleur qui vérifie les données entrées ;
- si les donnés sont invalides un message d'erreur s'affiche dans la vue d'inscription ;

- sinon si après la recherche du compte utilisateur dans notre base de donnés nous trouvons que ce visiteur existe déjà, un message d'erreur s'affiche dans la vue d'inscription ;
- sinon l'inscription se termine avec succès et l'utilisateur sera stocké dans une collection nommé accounts.

Le scénario « S'inscrire » établi dans le paragraphe précédant est modélisé par le digramme de séquence objets de la Figure 3.7.

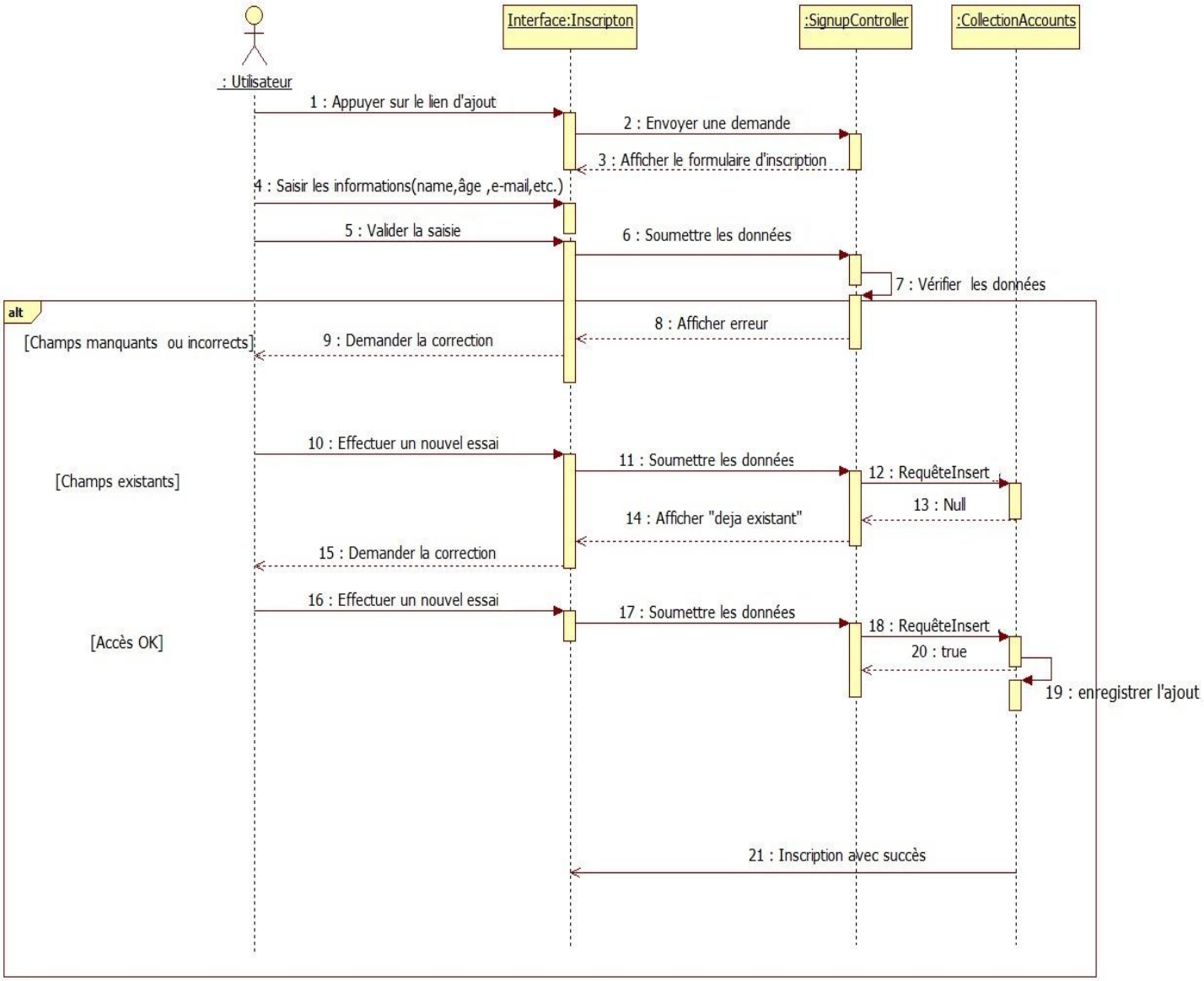


Figure 3.7 : Diagramme de séquence objets du scénario « S'inscrire »

2.2.3 Scénario de la suppression d'un utilisateur

L'administrateur doit assurer la fiabilité de ses services proposés, ainsi que leur audimat et rentabilité. Parmi ces services nous pouvons citer : la gestion des comptes utilisateurs, cette gestion inclut les tâches de manipulation des données qui sont : la consultation, la modification et la suppression. Pour supprimer un utilisateur, l'administrateur doit sélectionner un utilisateur de la liste, le contrôleur « Global.js » averti l'administrateur de l'opération de suppression, si ce dernier valide la suppression, l'utilisateur en question se disparait définitivement de la collection « Accounts ».

La Figure 3.8 illustre le diagramme d'activité qui résume cette partie.

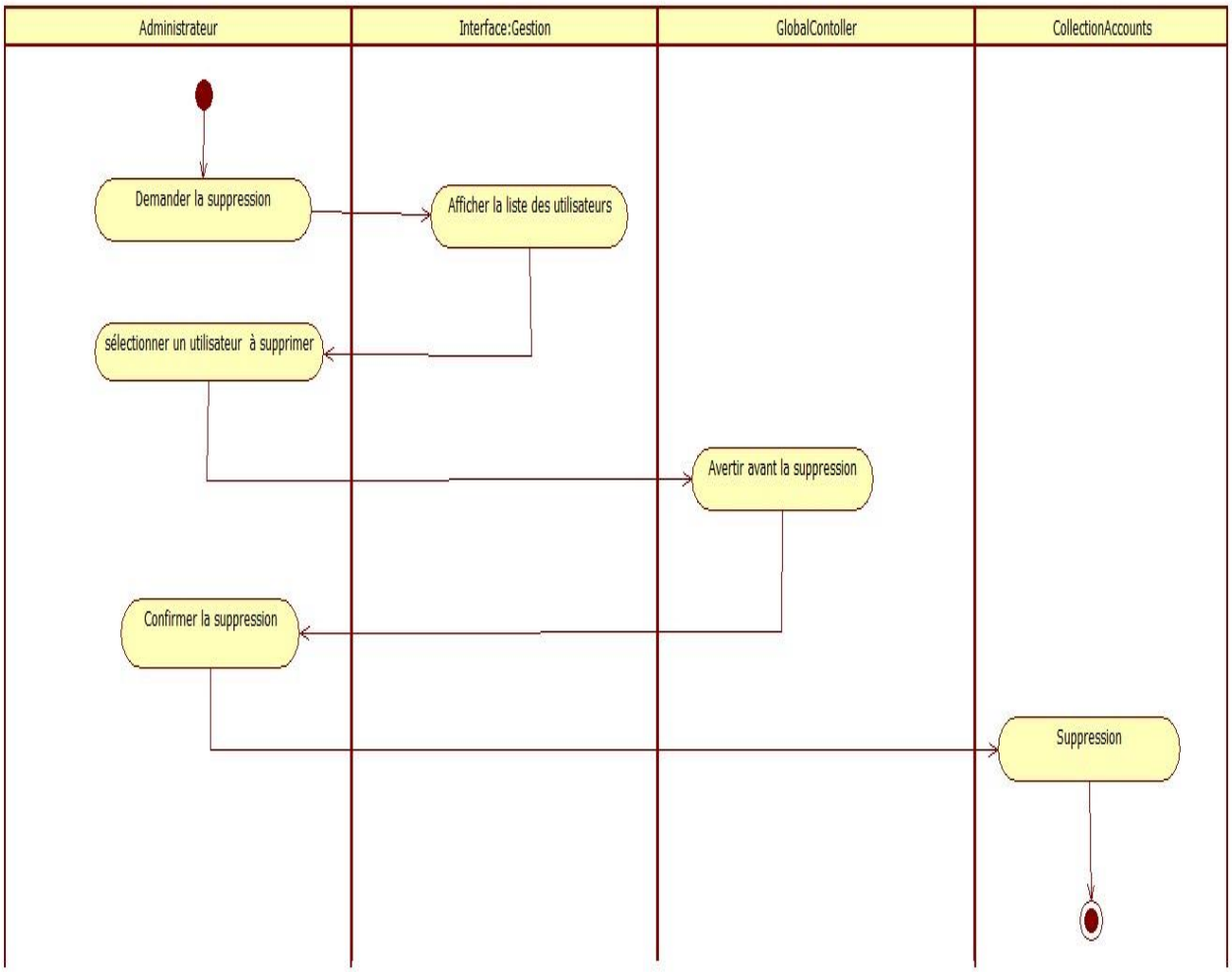


Figure 3.8: Diagramme d'activité du scénario « Supprimer un utilisateur »

2.2.4 Scénario de la modification d'un utilisateur

Après avoir décidé de mettre à jour un compte utilisateur, l'administrateur effectue une demande de modification. Il doit sélectionner un utilisateur de la liste. Le contrôleur transmet la demande de modification, une fois le formulaire de mise à jour apparait, l'administrateur saisit les nouvelles données puis il valide la modification pour que la base de données enregistre les informations modifiées.

La Figure 3.9 illustre le diagramme d'activité qui résume ce scenario.

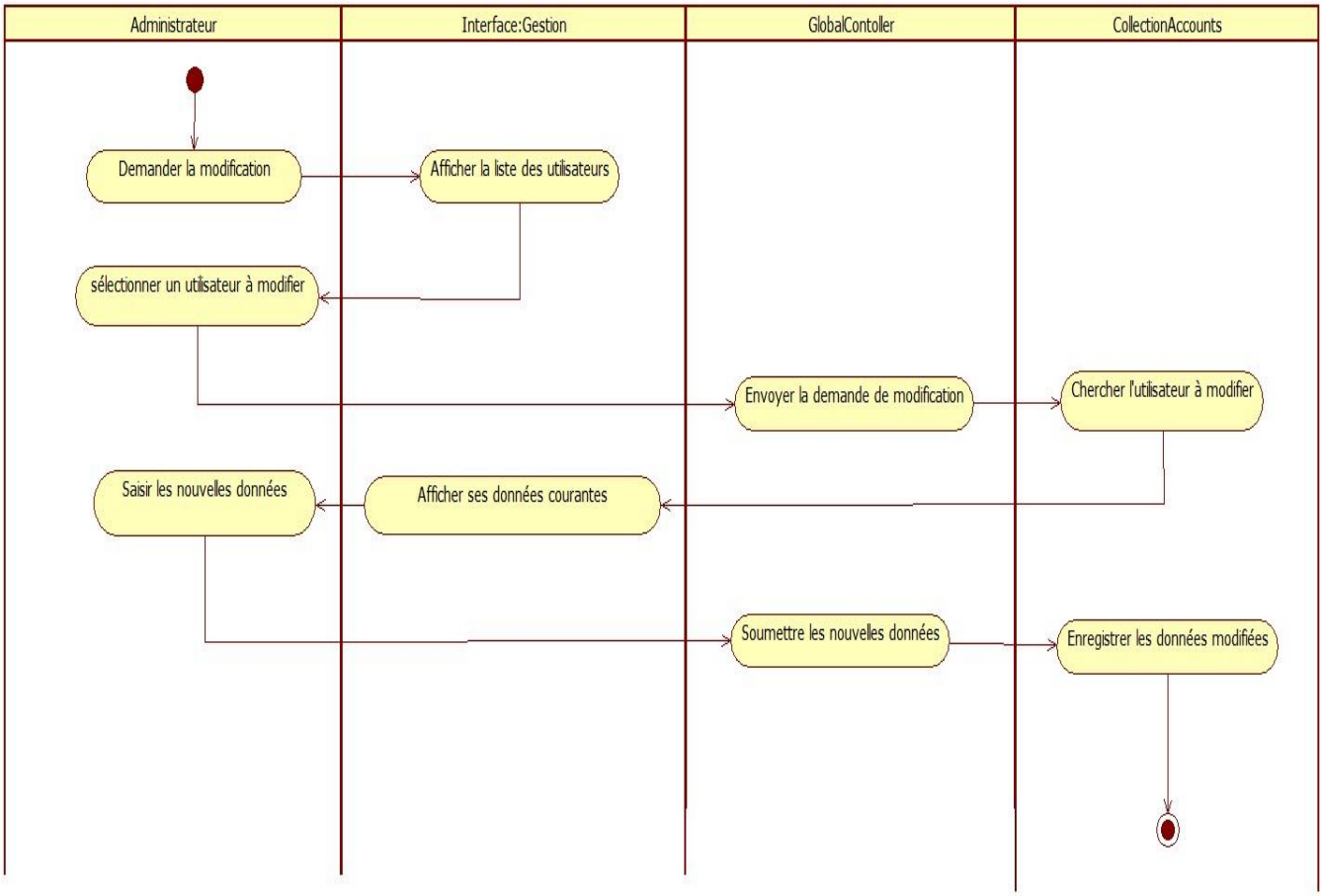


Figure 3.9 : Diagramme d'activité du scénario « Modifier un utilisateur »

Conclusion

Au cours de ce chapitre, nous avons présenté l'architecture de l'application, ensuite nous avons établi les diagrammes d'activités et les diagrammes de séquence représentant les interactions entre objets. Nous allons entamer dans le chapitre suivant la phase de réalisation dans laquelle nous allons mettre en évidence notre étude conceptuelle et technique.

Chapitre 4: Réalisation

Introduction

La réalisation vient couronner le travail de l'étude préalable et de l'étude conceptuelle. Elle présente le dernier volet de ce rapport, elle sera consacrée à l'aspect implémentation et à l'exposition du travail réalisé dans laquelle on s'assure que le système est prêt pour être exploité par les utilisateurs finaux. Nous commençons par présenter l'environnement matériel et logiciel. Par la suite nous élaborons une présentation des différentes interfaces créées.

1. Environnement de développement

Dans cette section, nous allons présenter l'environnement de travail sur les côtes matériels et logiciels utilisé lors du développement de l'application.

1.1 Environnement matériel

Pour la réalisation de notre projet, nous avons utilisé:

- un ordinateur portable disposant les caractéristiques suivantes :
  - Modèle : DELL ;
  - Système d'exploitation : Windows 7 ;
  - Mémoire : 4 Go ;
  - Processeur : Intel(R) Core(TM) i3-2348M CPU @ 2.30GHz 2.30 GHz ;
  - Disque dur : 500 Go ;
  - Graphique : intel(R) HD Graphics Family ,1632 Mo partagée.
- Un Raspberry Pi Type B:

C'est un PC miniature destiné à fonctionner sous linux. Il est développé par la Foundation Raspberry en 2011 et destiné à favoriser et encourager l'apprentissage de l'informatique et du développement en proposant sur le marché une machine simple, universelle, peu gourmande en énergie et très attractive en terme de prix [s11].

Grâce à ses caractéristiques et à son processeur, il nous a permis de faire fonctionner la distribution Linux de notre choix installé sur une carte SD. Il se révèle très pratique pour toutes les applications relié à un téléviseur.

Les caractéristiques techniques de ce mini PC Raspberry Pi Type B sont représentées dans la Figure 4.1.

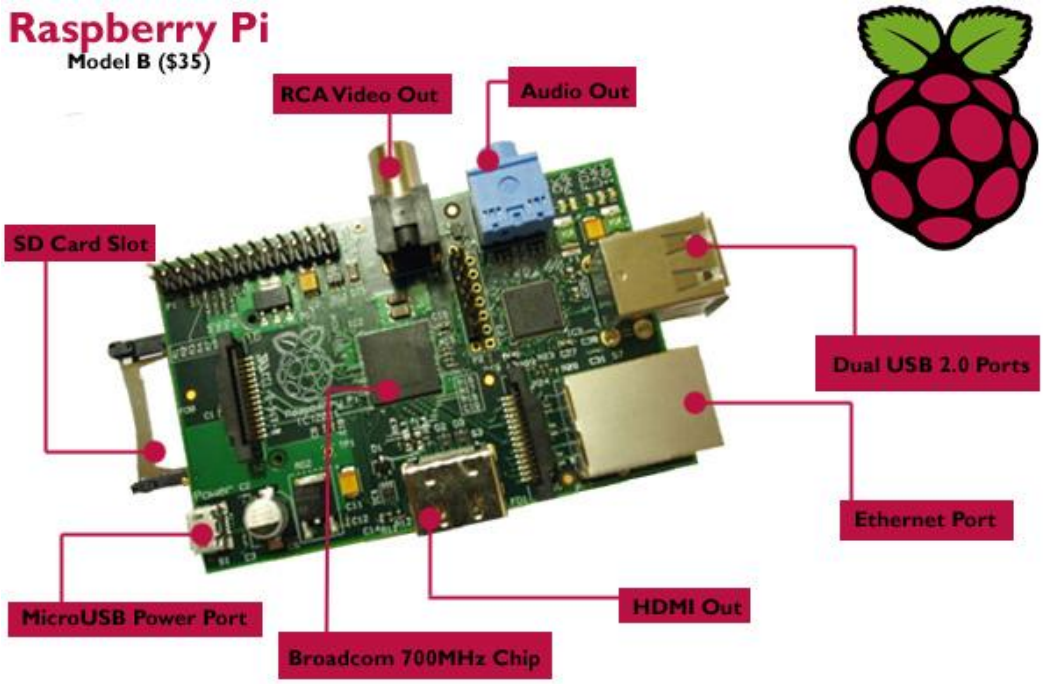


Figure 4.1 : Architecture de Raspberry pi model B.

Ce nano-ordinateur est équipé des entrées/sorties suivantes :

- deux ports USB ;
- un port RJ45 ;
- un lecteur de cartes SD ;
- un port HDMI ;
- une sortie audio ;
- une sortie RCA.

Le Raspberry type B est mieux équipé que le A puisqu'il dispose d'un second port USB, d'un port réseau RJ45 et de 512 Mo de mémoire au lieu de 256 Mo sur le A.

Nous détaillons dans ce qui suit la liste du matériel nécessaire pour pouvoir configurer le RaspberryPi et mettre en place notre solution :

- Un support d'installation du système

Le Rapsberry Pi ne dispose pas de disque dur. A la place, nous allons installer le système sur une carte mémoire SD de 8Go.

- Un PC

Tournant de préférence sous un système Gnu/Linux, qui possède un lecteur de carte SD pour installer le système d'exploitation du Raspberry Pi sur la carte SD.

- Un clavier et une souris USB

Un clavier et une souris USB sont indispensables pour l'installation. Ensuite, ils ne seront plus nécessaires.

- Un câble micro-USB

Nécessaire pour le branchement entre le PC et le RaspberryPi.

- Un câble HDMI ou vidéo RCA

Dans notre cas nous avons utilisé un câble HDMI, pour brancher le RaspberryPi au téléviseur.

- Un câble Ethernet

Un câble Ethernet RJ45 pour connecter le RaspberryPi au réseau.

- Un ecran de visualisation

Un écran disposant d'une prise HDMI, ce qui est le cas de la plupart des télévisions récentes, il est indispensables pour l'installation et l'implémentation de notre projet.

La configuration du RaspberryPi est détaillée dans l'annexe.

1.2 Environnement logiciel

Cette section sera consacrée à la présentation des outils logiciels et des langages que nous avons utilisés pour mettre en œuvre notre application.

Les technologies, logiciels, éditeurs et Framework mis en œuvre pour élaborer l'application sont les suivants :

- **JS (Java Script)**

Est un langage de scripts qui est incorporé aux balises Html. Il permet d'agrémenter la présentation et l'interactivité des pages Web ainsi que l'ajout des contrôles sur les champs des formulaires des applications Web [s12].

- **NodeJS**

Est une plateforme logicielle libre et événementielle en JavaScript orientée vers les applications réseau qui doivent pouvoir monter en charge. Il est basé sur le moteur Google V8qui s'exécute coté serveur. NodeJS contient une bibliothèque de serveur HTTP intégrée, ce

qui rend possible de faire tourner un serveur web sans avoir besoin d'un logiciel externe comme Apache, et permettant de mieux contrôler la façon dont le serveur web fonctionne.
NodeJS est de plus en plus populaire comme plateforme serveur, elle est utilisée par LinkedIn, Microsoft , Yahoo, etc [s13].

- **HTML 5**

(HyperText Markup Language 5) est un système qui formalise l'écriture d'un document avec des balises de formatage indiquant la façon dont doit être présenté le document et les liens qu'il établit avec d'autres documents, HTML5 est la dernière révision majeure d'HTML (format de données conçu pour représenter les pages web). Cette version est en développement en 2013 .Dans le langage courant, HTML5 désigne souvent un ensemble de technologies Web (HTML5, CSS3 et JavaScript) permettant notamment le développement d'applications [s12].

- **CSS3**

(Cascading Style Sheets) qu'on peut traduire par feuilles de style en cascade) a été créé pour le web : il sert principalement à gérer l'aspect visuel d'une page Web.
En d'autres termes, le langage CSS gère la présentation d'une page web : la couleur du texte, l'utilisation de telle ou telle police, les images ou couleurs d'arrière-plan, les espaces et marges des différents éléments, le positionnement des éléments, etc. Le CSS n'existe pas par lui-même. Il s'applique forcément à un autre document : il ne peut être utilisé qu'en association avec des langages de balisage tels que HTML et XHTML [s12].

- **Jade**

Est un moteur de Template créé en 2009 pour NodeJS, inspiré de Haml Mais en beaucoup plus complet, il est basé sur JavaScript. Simple à prendre en main, puissant et modulaire, nous pouvons utiliser Jade avec PHP, Scala, Ruby, python, Java.

De la même manière que les préprocesseurs CSS, il est possible de faire des inclusions de fichiers très rapidement pour éviter la répétition de blocks, c'est ce qui fait réellement la puissance de Jade [s14].

- **JQuery**

Est une bibliothèque JavaScript rapide et riche en fonctionnalités [s15]. Elle fournit une couche d'abstraction générique pour les scripts Web classiques et elle permet de gérer des évènements et d'ajouter des animations dans l'application Web. Cette librairie a pour but de simplifier les commandes communes du JavaScript. Légère, simple à coder et gratuite, elle se trouve la plus adéquate à notre cas.

- **Twitter Bootstrap**

Est une collection d'outils utiles à la création de sites Web et applications Web. C'est un ensemble qui contient des codes HTML et CSS, des formulaires, boutons, outıs ɑe navigation et autres éléments interactifs, ainsi que des extensions JavaScript en option[s14]. C'est l'un des projets les plus populaires sur la plate-forme de gestion de développement GitHub.

- **MongoDB**

Est un système de base de données dans la mouvance NoSQL. Il est orienté documents. Son nom vient de Humongous qui veut dire énorme ou immense. Son objectif est donc de pouvoir gérer de grandes quantités de données. Le moteur de base de données facilite l'extension si bien que l'on pourra supporter l'accroissement de la quantité de données par l'ajout de machines. Dans MongoDB, l'information est modélisée sur un document au format JSON codée binaire appelé BSON les coulisses [s15].BSON étend le modèle JSON de fournir des types de données supplémentaires et à être efficace pour le codage et le décodage dans les différentes langues.

- **JSON**

(JavaScript Object Notation – Notation Objet issue de JavaScript) est un format léger d'échange de données. Il est facile à lire ou à écrire pour des humains. Il est aisément analysable par des machines. JSON est un format texte complètement indépendant de tout langage. Ces propriétés font de JSON un langage d'échange de données idéal [s15].

Un document JSON ne comprend que deux éléments structurels :

- des ensembles de paires nom / valeur,

- des listes ordonnées de valeurs.

- **NPM**

(Node Package Manager) est le gestionnaire de paquet de NodeJS, apparu avec NodeJS en 2009, il permet d'installer très simplement les contributions de la communauté (applications, outils, Framework…). Il facilite la création, le partage et l'installation de modules [s14].

- **Express**

Est un Framework développé par TJ Holowaychuk. Il est utilisé pour ajouter à NodeJS ce qu'il faut pour créer une application Web (au niveau de la récupération des requêtes Web et de la création des réponses notamment).Express s'appuie sur un autre module appelé Connect (un Framework de serveur HTTP qui regroupe un ensemble de middleware[19]). Il utilise par défaut le moteur de Template Jade mais d'autres sont également disponibles. Express est construit au-dessus de Connect. En fait, Express n'est rien d'autre que le module Connect auquel on a ajouté certaines fonctionnalités comme les routes et les vues [s14].
Connect est un module qui permet à d'autres modules appelés middlewares de communiquer entre eux. Il est fourni avec plus de 18 middlewares de base, et les développeurs peuvent bien entendu en proposer d'autres via NPM. Les middlewares livrés avec Connect fournissent chacun des micro-fonctionnalités. Express ne fait qu'ajouter les routes et les vues ceci est montré par la Figure 4.2.

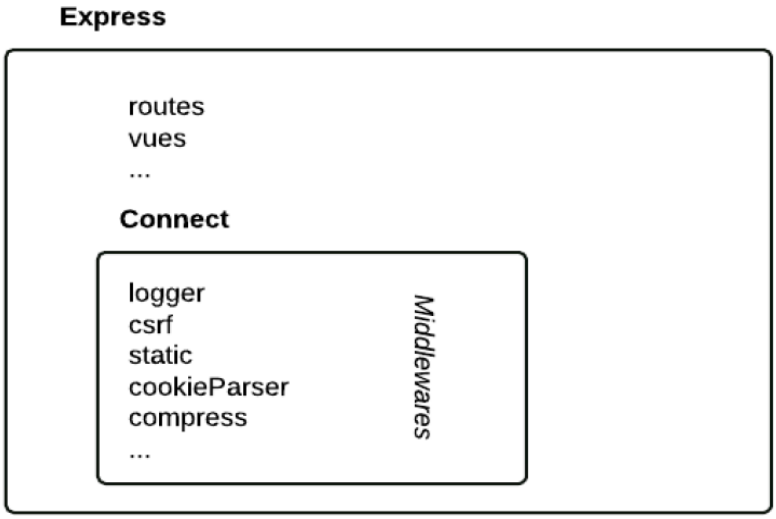


Figure 4.2 : Structuration du framwork Express

[19] Un middleware : est un logiciel qui permet à différentes applications d'échanger et d'interopérer.

- **Raspbian**


Est un système d'exploitation libre et gratuit fondé sur GNU/Linux/Debian et optimisé pour fonctionner sur un Raspberry Pi, Raspbian est un mot-valise formé par la fusion des mots “Raspberry Pi” et “Debian”.Il s'agit d'une modification de Debian spécifiquement adaptée pour les puces de type ARMv6 [s16].

- **Chromium**


Est un navigateur web Open Source sur lequel se base Google Chrome, compatible Windows, Linux et Mac OS [s17]. Le logo de Chromium est identique à celui de Google Chrome à l'exception de sa couleur uniformément bleue.

1.3 Choix des outils de développement

1.3.1NodeJS

NodeJS possède différents avantages qui le rendent avantageux dans certains cas d’utilisation [s13]:

- NodeJS semble être très rapide, ça tient très bien la monté en charge ;
- pas de compilation donc simplification des tests et de la mise en œuvre ;
- multiplateforme: déploiement fortement facilité par le moteur V8 dont les sources en C peuvent être compilées pour tous types de système d’exploitation ;
- La vitesse de l'interpréteur Javascript est en constante progression V8 (JavaScript engine) ;
- open Source, sous licence MIT ;
- pas besoin de serveur web HTTP comme Apache pour développer l'application ;
- fournit une architecture événementielle et non-bloquante.

À titre comparatif, voici les principales différences entre NodeJS et Apache + PHP résumées dans le Tableau 4.1.

	Apache+PHP	**NodeJS**
La technologie de mise en œuvre	C	le V8 de Google (moteur JavaScript) ; C++
Non bloquante	Non	oui
Opération asynchrone	Non	oui
Exigence de mémoire par connexion client	15-30MB	1-2KB
Programmation orienté	Objet	événement
Serveur web	Apache	Pas besoin de serveur web

Tableau 4.1: Comparaison des caractéristiques entre Apache+PHP et NodeJS

NodeJS trouve parfaitement sa place pour notre application. En effet, il est très puissant lorsqu'il s'agit de transit de données de points à points. Il permet, grâce à sa rapidité, de mettre en place des applications temps réel .Par contre, celui-ci n'est clairement pas adapté pour des traitements de données a cause de sa nature interprétée.

1.3.2 MongoDB

C'est un système Open Source de gestion de base de données orienté documents appartenant à la stratégie de stockage NoSQL (Not Only SQL).Parmi ses avantages principaux, nous pouvons citer [s15]:

- fiabilité ;
- stockage orienté document ;
- support complet des indexes ;
- stockage des objets volumineux binaires (vidéos, photos, …) ;
- mise en œuvre simple ;
- manipuler des données rapidement malgré une charge et un volume important ;

- Flexibilité (exemple des documents) ;
- schémas dynamique : les données ne sont pas obligées de respecter une structure. ;
- haute scalabilité ;

Beaucoup de concepts de MySQL ont des analogues proches dans MongoDB. Le Tableau 4.2 donne un aperçu des concepts communs dans chaque système.

MySQL	**MongoDB**
Table	Collection
Rangée	Document ou *BSON* documents
Colonne	Terrain
Clé primaire	Champ-id
Rejoint	Les documents incorporés, reliant
Index	Index

Tableau4.2: Les concepts de MySQL et leurs analogues dans MongoDB

À titre comparatif, voici les principales différences entre MySQL et MongoDB présentées par le Tableau 4.3:

	MySQL	**MongoDB**
Schéma dynamique	Aucun	Oui
Facile pour les programmeurs	Aucun	Oui
Transactions complexes	Aucun	Oui

Mises à jour champ	Oui	Oui
Performance	Bonne	mauvaise

Tableau 2.3: Comparaison des caractéristiques entre MySQL et MongoDB

Nous adoptons MongoDB car il permet de construire des applications plus rapidement, gérer les types de données très diverses, et gérer des applications plus efficacement à l'échelle. Les données sont stockées dans un format du type JSON, Binary JSON, très efficace pour diffuser de l'information sur les Backend et très facilement manipulable par les frontend via Javascript.

2. Implémentation

2.1 Architecture du projet réalisé

En se basant sur les technologies citées dans la section précédente, nous élaborons l'architecture physique de notre application qui est montré au niveau de la Figure 4.3.

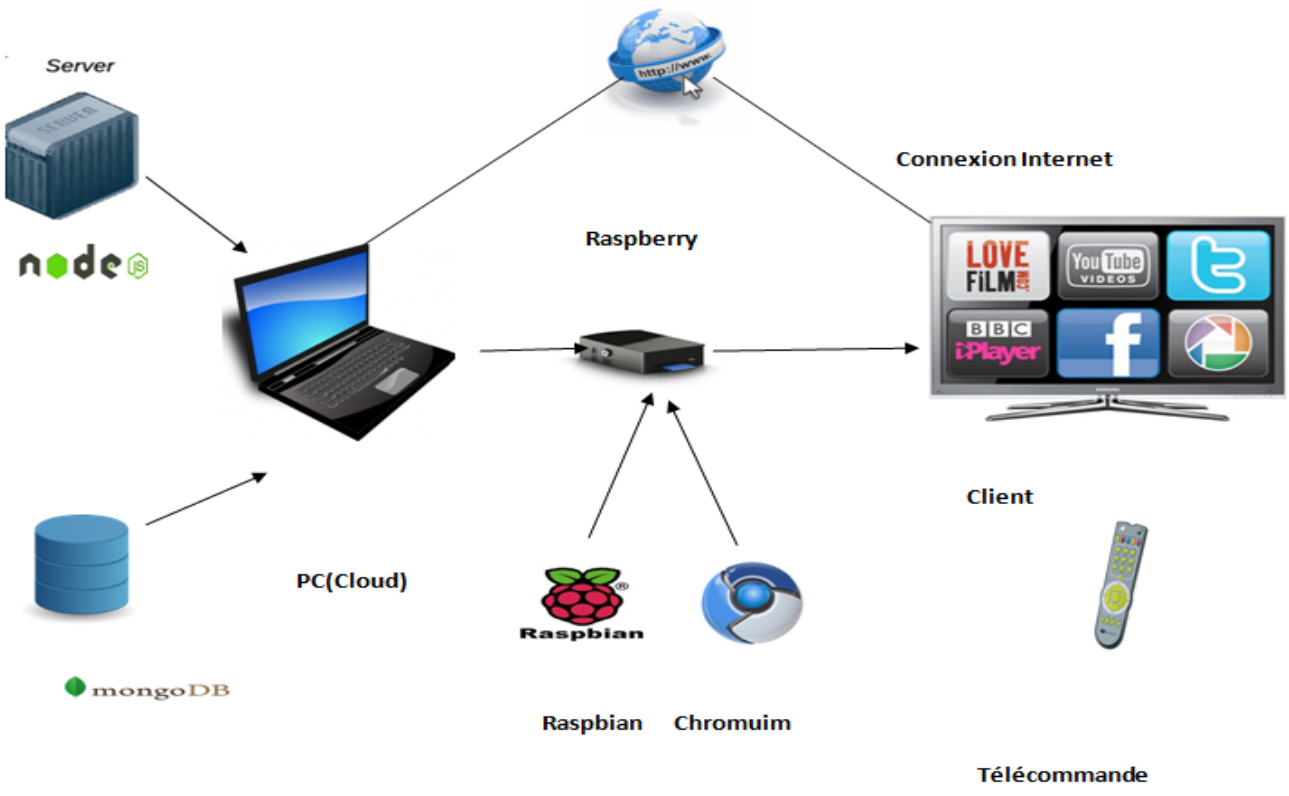


Figure 4.3: L'architecture générale de notre application Smart TV

Cette architecture est composée d'un serveur web NodeJS et un serveur de base de données MongoDB inclus dans le PC qui représente notre Cloud, puisqu'il sauvegarde tous nos services et notre environnement de développement. Ce PC est lié à un Rasberry avec un câble USB. Nous avons installé dans la carte SD du Rasberry le système d'exploitation GNU/Linux "Raspbian" et le navigateur Web "Chromium". Le Rasberry est branché en le raccordant via un port HDMI au téléviseur connecté à Internet.

Pour profiter des fonctionnalités de notre Smart TV, le client demande l'accès à un service, une requête sera envoyée vers le PC, le Rasberry apporte le service souhaité du Cloud et l'affiche sur le téléviseur.

2.2 Les interfaces de notre application

Les interfaces conçues ne sont que les fruits d'une longue chaîne de travail ; ce n'est qu'une simple image dont l'utilisateur ignore les arrières plans.

Nous naviguerons à travers les différentes interfaces de notre application Smart TV afin de mettre en évidence les objectifs atteints en termes de fonctionnalités fixées dans le cahier des charges.

Vu le nombre important des fonctionnalités qu'offre notre application, nous présentons uniquement les interfaces les plus importantes. Le lancement de l'application commence par l'interface d'authentification. La Figure 4.4 illustre l'interface d'authentification.

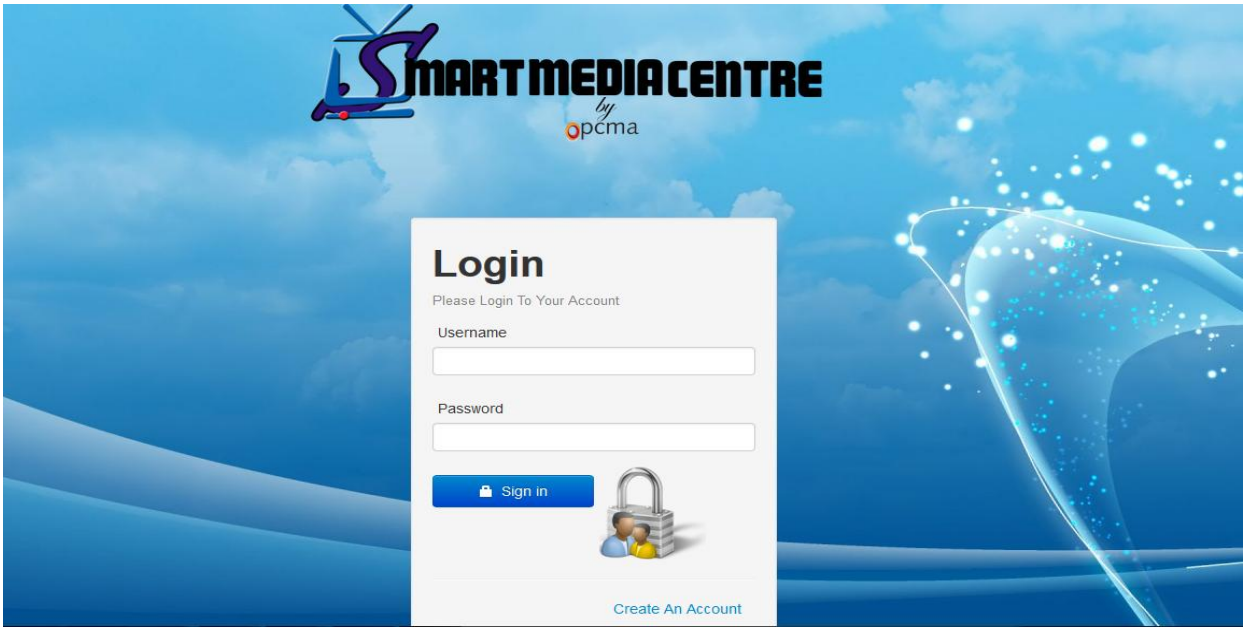


Figure 4.4: Interface d'authentification du Smart TV

L'authentification est un certificat de sécurité du système, et à partir d'elle chaque utilisateur authentifié peut accéder à notre application.

L'application vérifie l'existence de ce compte dans sa base des données de contenu. Si l'utilisateur est identifié dans la base, il accède à l'application selon son mode d'accès fixé par l'administrateur .Tant que le nom utilisateur ou/et le mot de passe ne sont pas valides, l'utilisateur reste bloqué, le système prévient l'utilisateur comme ainsi :

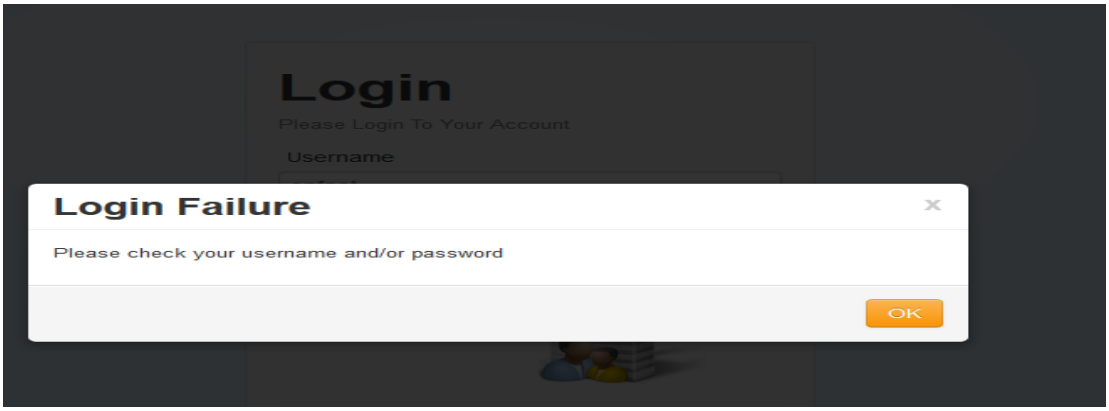


Figure 4.5: Message d'erreur

Si l'utilisateur ne possède pas un compte il doit s'inscrire pour bénéficier des fonctionnalités de l'application. L'interface d'authentification permet aussi aux utilisateurs de créer des nouveaux comptes en appuyant sur « Create an account».

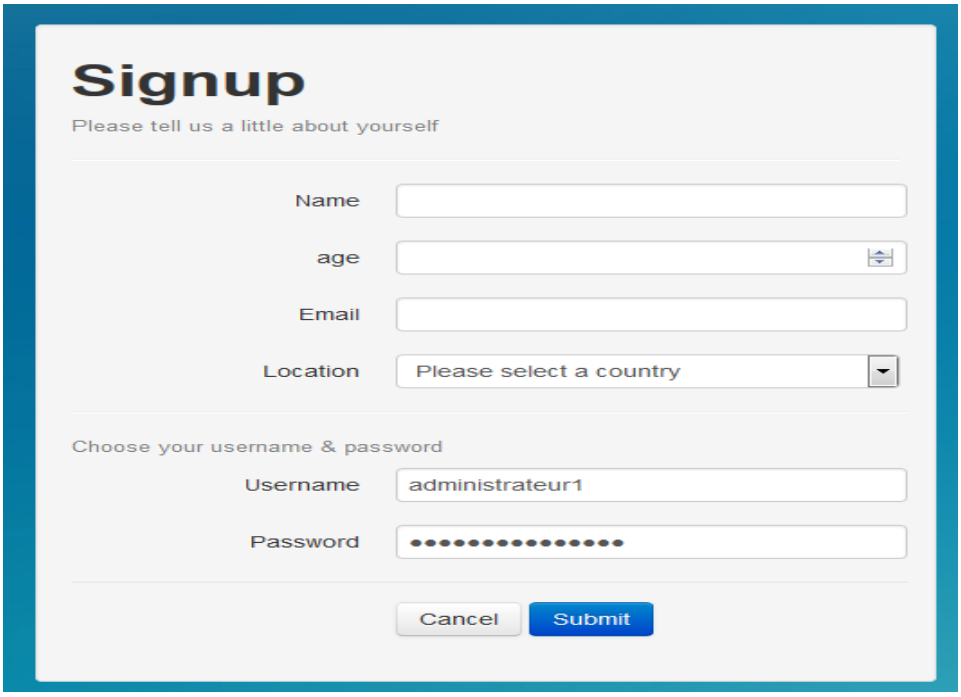


Figure 4.6: Interface d'inscription

La nécessité de l'identification permet non seulement de sécuriser l'accès mais en plus de charger l'interface spécifique à chaque utilisateur. En effet, si l'utilisateur est âgé de moins 15 ans il sera redirigé vers l'interface dédié pour les enfants comme le montre la Figure 4.7.

Figure 4.7: Interface spécifique aux enfants

Cet espace est consacré aux applications pour les enfants, il est équipée de plusieurs fonctionnalités : dessin animé, jeux, musiques…

La Figure 4.8 illustre le service « TV programs for kids».

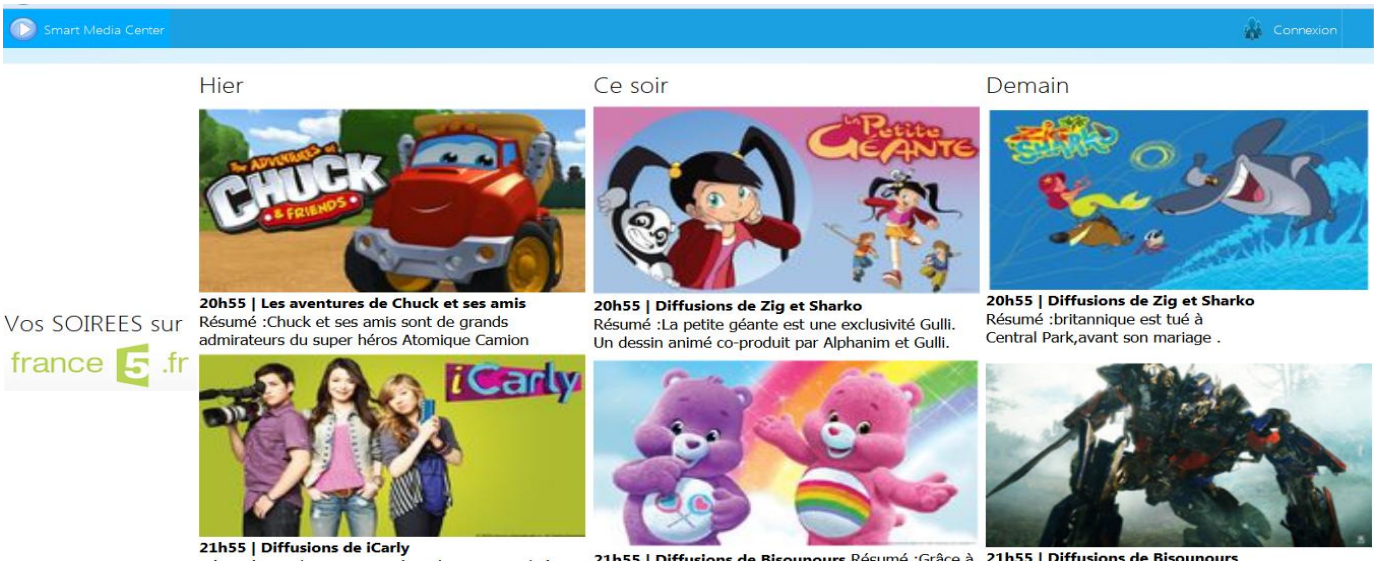


Figure 4.8: Interface de « TV programs for kids»

Nous avons également deux catégories de jeux : jeux éducatifs et jeux de divertissement, ils sont illustrés respectivement par la Figure 4.9 et la Figure 4.10.

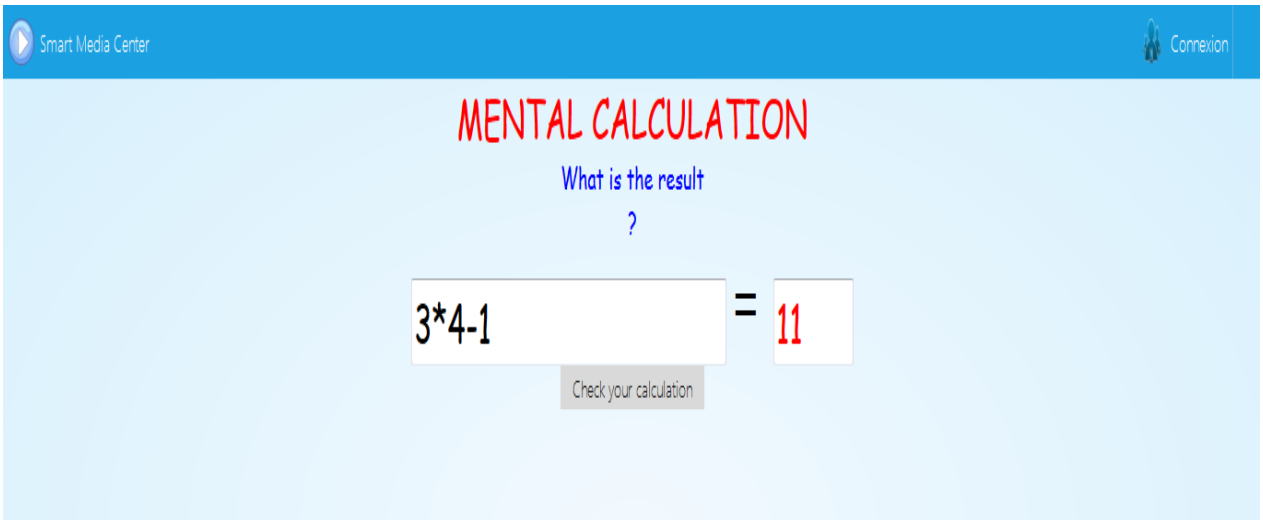


Figure 4.9: Interface d'un jeu éducatif

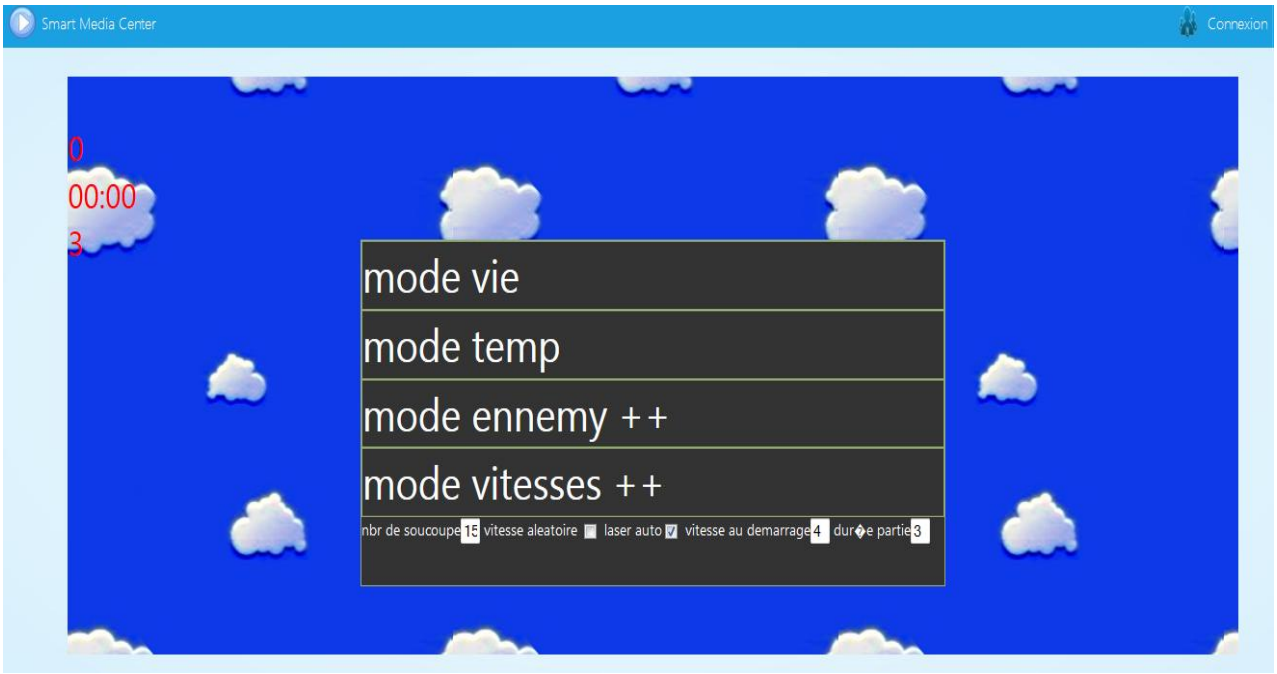


Figure 4.10: Interface d'un jeu de divertissement

Dans le cas contraire l'utilisateur sera redirigé vers l'interface principale de notre Smart TV illustrée par la Figure 4.11.

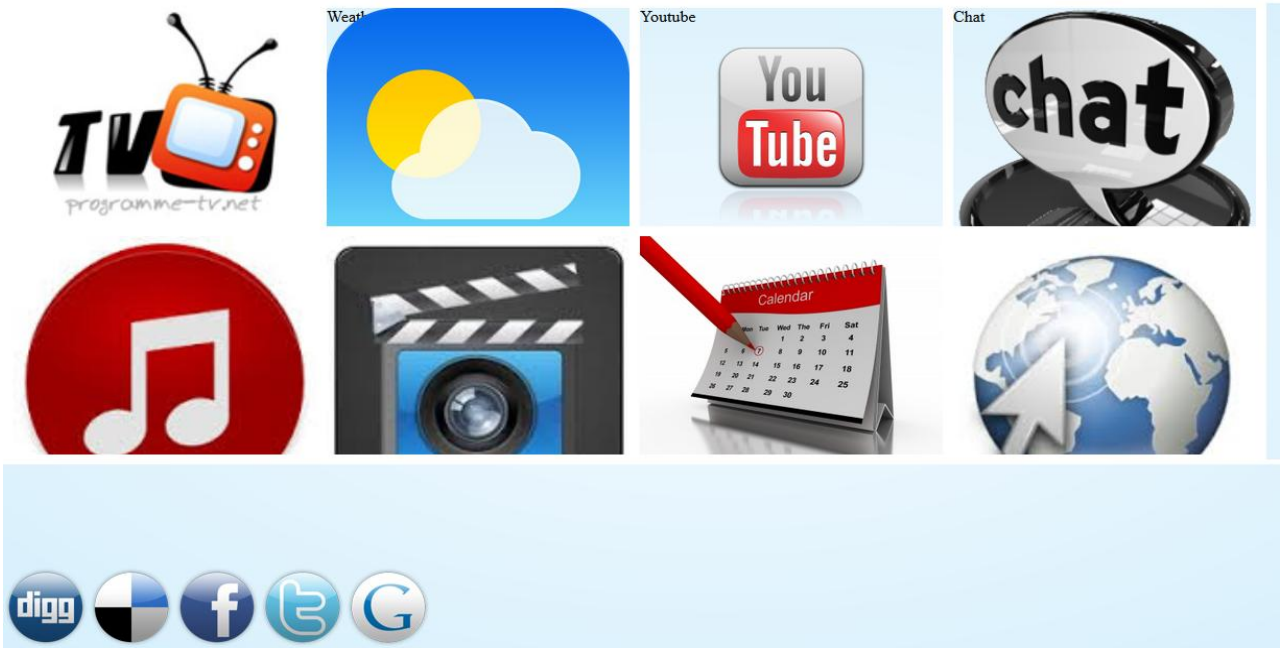


Figure 4.11: Interface principale du Smart TV

Parmi les fonctionnalités de cette interface, nous trouvons la météo. Lorsque l'utilisateur clique sur le menu « Weather» et entre sa localisation actuelle, une fenêtre affiche la météo. La Figure 4.12 présente le retour du service Météo.

Figure 4.12: Service météo (Web Service Yahoo)

Si l'utilisateur clique sur le menu « chat» et entre son pseudo, une fenêtre de discussion s'affiche.

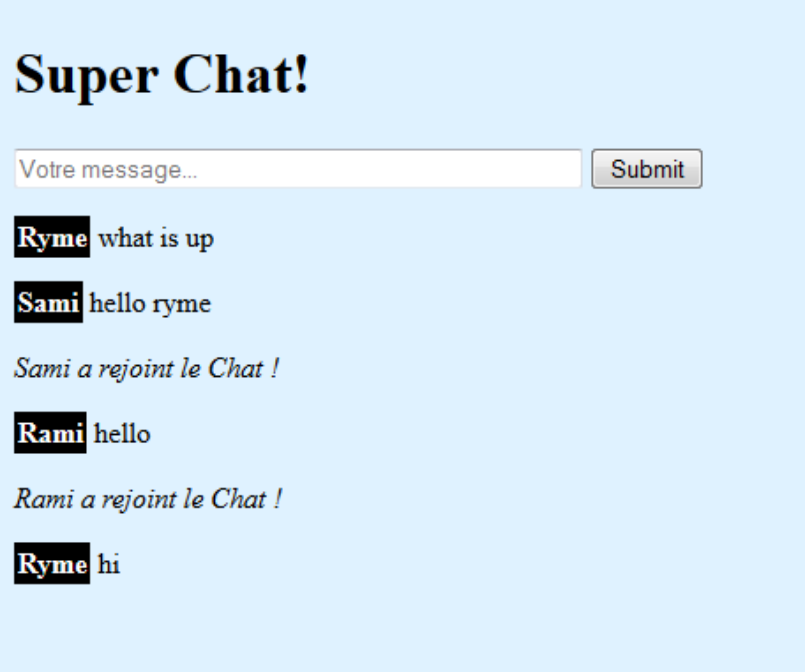


Figure 4.13: Interface de chat

L'utilisateur peut aussi consulter le calendrier.

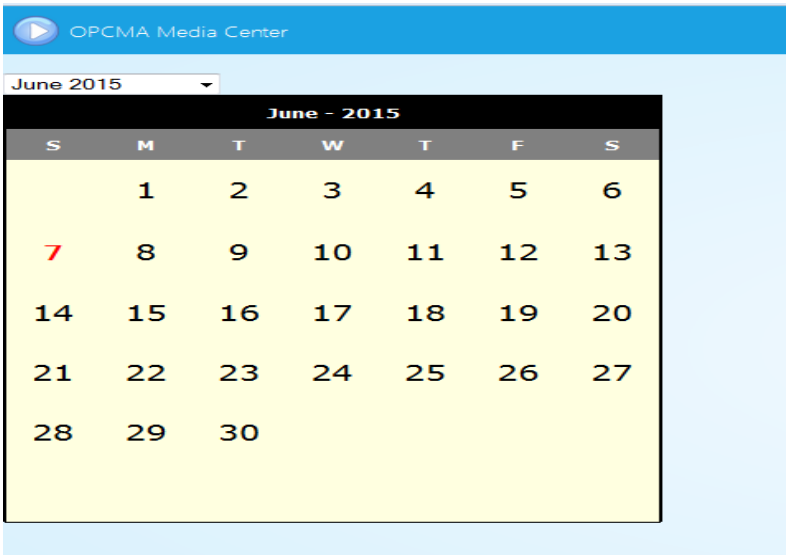


Figure 4.14: Interface de calendrier

L'utilisateur peut également surfer sur Internet depuis son téléviseur en cliquant sur « browser » comme le montre la Figure 4.15.

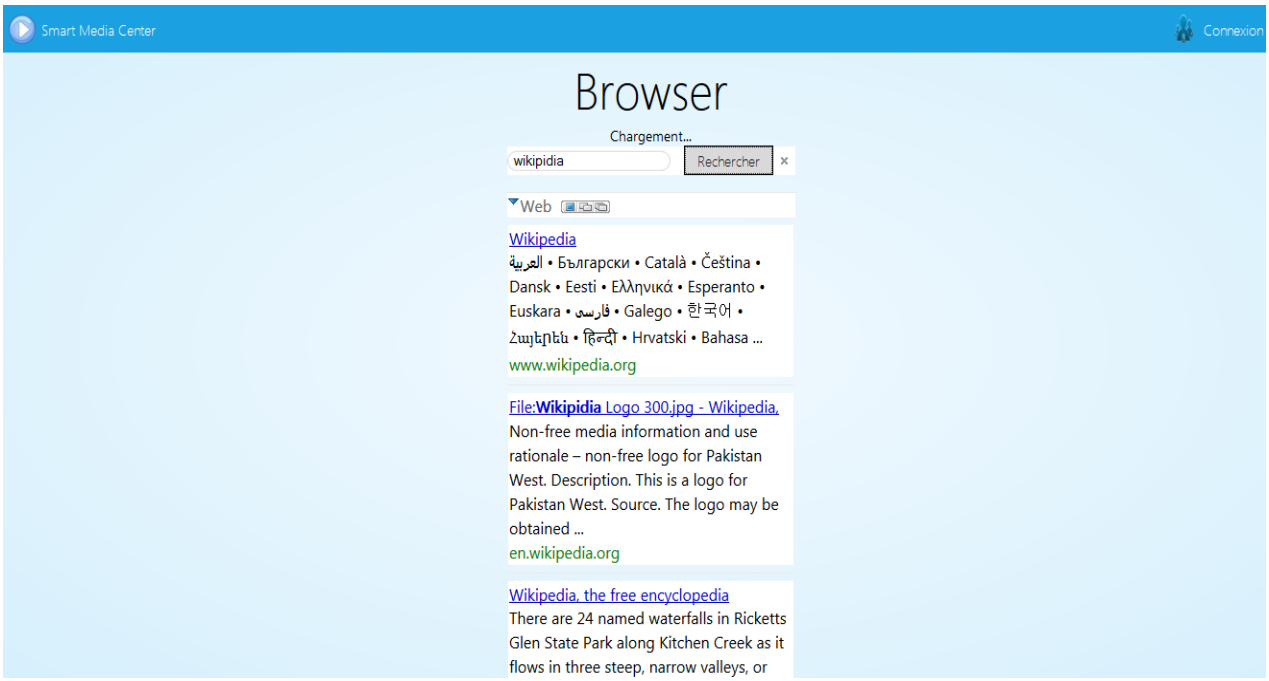


Figure 4.15: Interface du navigateur

Il peut aussi choisir de regarder les vidéos et les chaînes YouTube.

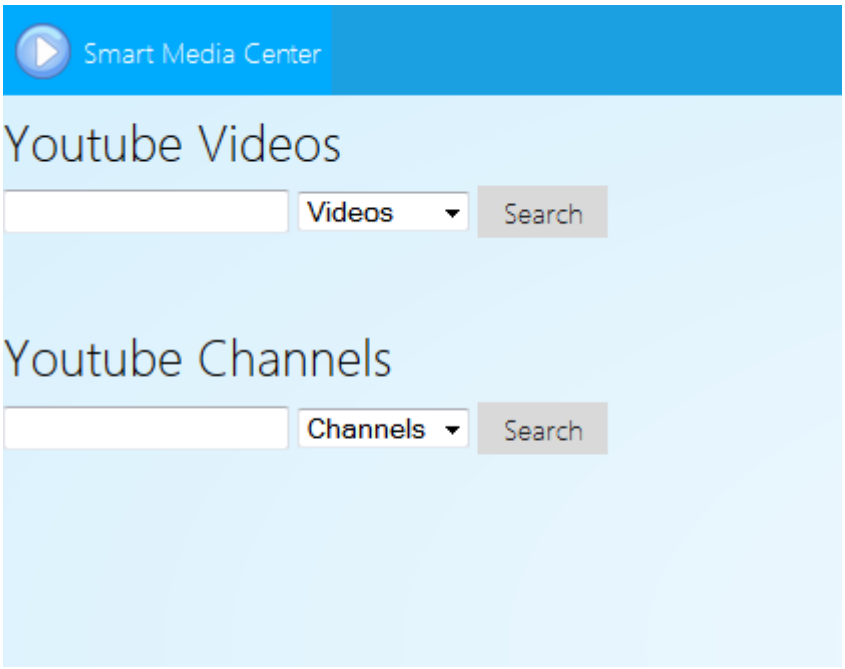


Figure 4.16: Interface de recherche des vidéos sur YouTube

L'administrateur doit assurer la fiabilité de notre application. Parmi ces services nous pouvons citer : la gestion des comptes utilisateurs, la gestion du contenu de l'application et la consultation de toutes les fonctionnalités de l'application.

Figure 4.17: Interface de gestion

Si l'administrateur souhaite gérer les utilisateurs, le système lui affiche la liste des utilisateurs inscrits et les actions qu'il peut faire. La Figure 4.18 illustre l'interface de cette partie.

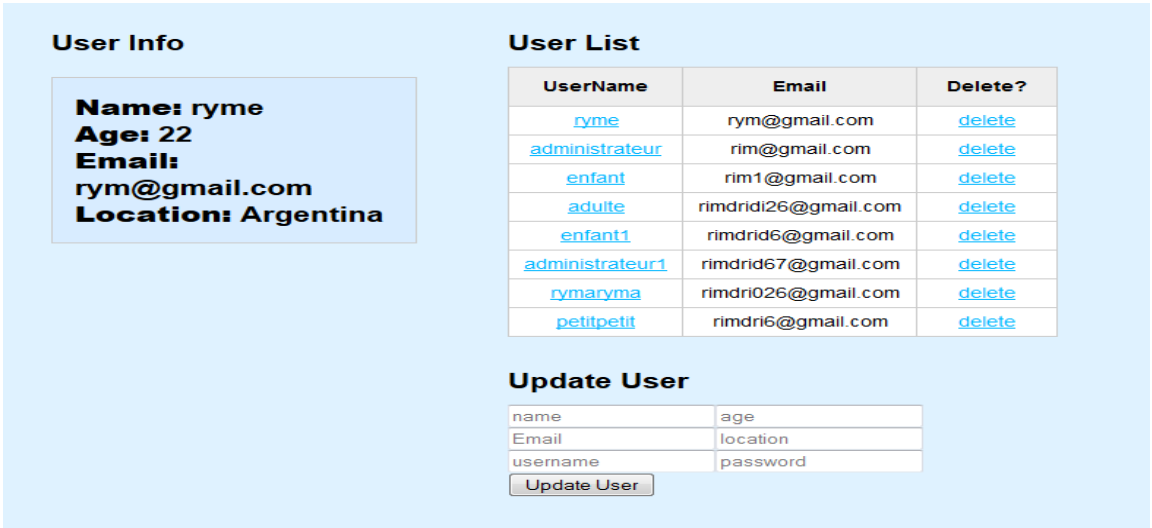


Figure 4.18: Interface de la gestion des utilisateurs

Si l'administrateur choisit de supprimer un des utilisateurs, il faut choisir un de la liste. Après la sélection d'un utilisateur et en appuyant sur le bouton supprimer un message de confirmation s'affiche, si l'administrateur appuie dessus, l'utilisateur sélectionné va être supprimé. La Figure 4.19 illustre l'interface de suppression.

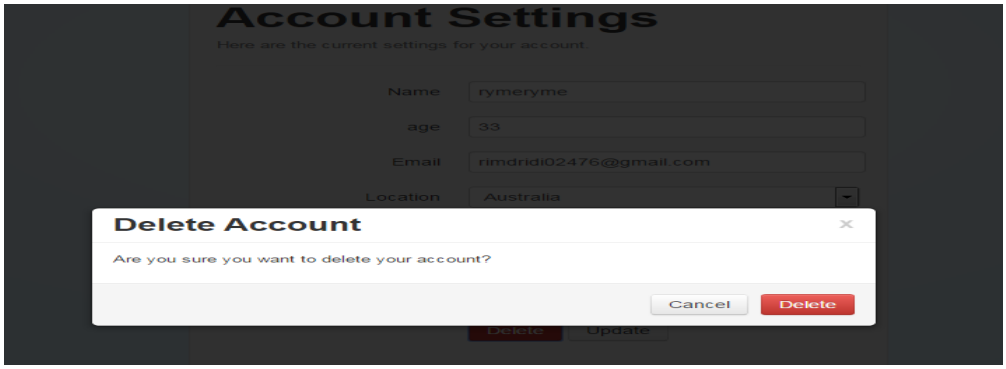


Figure 0.19: Interface de suppression d'un utilisateur

Après avoir développé notre application et configurer le Raspberry, il nous reste qu'à brancher le téléviseur pour transformer notre poste ordinaire en Smart TV.

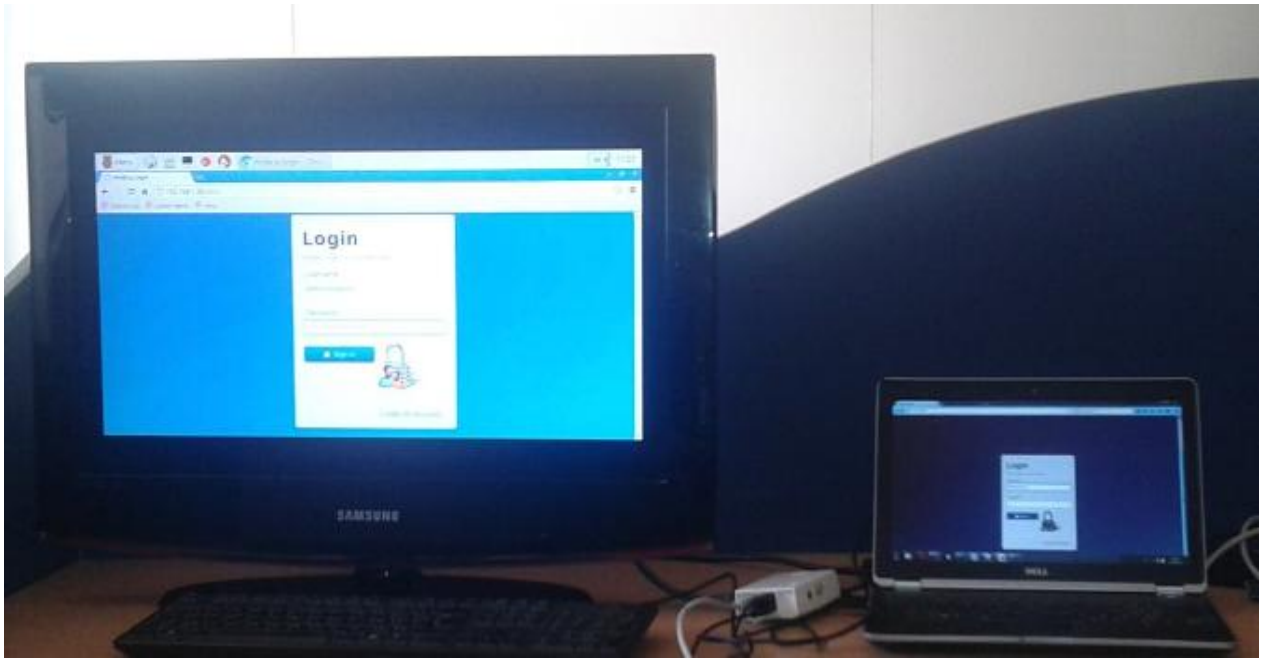


Figure 0.20: Mise en place du Smart TV sur le téléviseur

Lorsque nous démarrons la télévision, l'interface de la Smart TV s'affiche après avoir s'authentifié avec succès.

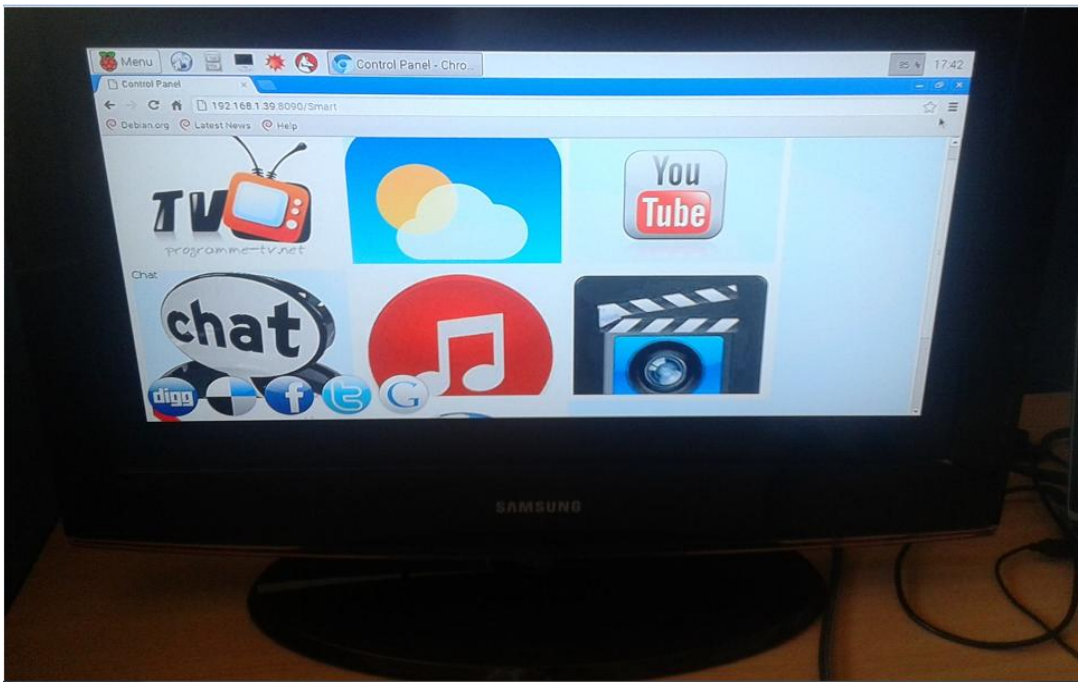


Figure 4.21: La Smart TV installée sur le téléviseur

Conclusion

Au cours de ce chapitre, nous avons présenté en détails la projection de la conception du plan théorique sur le plan pratique. Nous avons décrit les plates-formes matérielles et logicielles sur et avec lesquelles nous avons construit notre application, en justifiant nos choix technologiques, puis, nous avons décrit l'architecture de l'application. Enfin, nous avons présenté les interfaces les plus significatives de notre application.

Conclusion et perspectives

Dans un monde continuellement évolutif, la motivation d'avoir des moyens performants interactifs donne naissance à une révolution favorisant le développement des solutions dite « intelligente ».

A cet effet, notre travail a porté sur la réalisation d'une application Web de développement des services de Smart TV qui permet de répondre aux préoccupations des utilisateurs et leur offre les services attendus.

Ce projet nous a permis d'appliquer nos connaissances théoriques acquises lors de nos études et de prendre ainsi conscience des priorités dans le monde de travail. En effet, il nous a permis d'approfondir les connaissances que nous avons acquises tout au long de notre formation et de nous nous familiariser avec la conduite du projet informatique de façon méthodique et organisée. L'intégration dans l'équipe technique d'OPCMA nous a été bénéfique aussi bien sur le plan scientifique que sur le plan humain.

Ce projet nous a fournit aussi un grand apport sur le plan technique, En effet, nous avons appris à mieux manipuler des outils de programmation telle que le Framework Express, Twitter Bootstrap et Jquery, ainsi que l'environnement de programmation NodeJS. Nous avons également eu l'opportunité de mieux assimiler l'approche objet en utilisant la notation UML dans notre propre démarche de développement.

Nous sommes arrivés à développer toutes les fonctionnalités du système dans les temps.

L'intégration a été réalisée avec succès, c'est-à-dire que l'application a été installée sur le téléviseur et elle fonctionne correctement.

Bien que les principaux objectifs de notre projet soient atteints, l'application que nous avons développé pourrait être enrichie par d'autres fonctionnalités avancées et des améliorations peuvent être envisagées pour l'enrichir, tel que le développement d'une application permettant de communiquer en visioconférence avec nos contacts à l'aide d'une caméra de communication, mettre un catalogue des derniers films, accéder à des Webradios,etc.

Ainsi, comme perspective principale pour l'état actuel de ce projet, nous comptons intégrer un module qui nous permet de se débarrasser de la télécommande en la remplaçant par un Smartphone ou une tablette.

Afin de commercialiser notre solution actuelle, nous avons besoin d'un décodeur qui remplace le Raspberry et qui permet de retrouver sur notre téléviseur l'ensemble des contenus

nécessaire pour le fonctionnement de notre application que nous avons stockés avec le Cloud au lieu d'un PC.

Nous souhaitons, enfin, que ce modeste travail apporte satisfaction aux membres du jury et à toute personne intéressée de près ou de loin.

Bibliographie

[1] A. Madani. «Génie logicielUML : UnifiedModelingLanguage».

[2] CAPUOZZO, Olivier. « Cas d'utilisation, une introduction », France, Editions CERTA, 2004.

[3] CHARROUX, Benoît, OSMANI, Aomar et THIERRY-MIEG, Yann. « Diagrammes de cas d'utilisation », France, Pearson Education France – UML2 3e édition, 2010.

[4] Rapport de stage de fin d'études 2009/2010.

Netographie

[s1] http://www.opcma.com/ [Consulté le 02/2015].

[s2] http://www.telehd.com/191-technologie-televiseur-smart-tv-cest-quoi [Consulté le 02/2015].

[s3] http://www.histoire-pour-tous.fr/inventions/744-invention-de-la-television.html [Consulté le 03/2015].

[s4] http://www.lg.com/fr/accessoires-tv [Consulté le 03/2015].

[s5] http://www.free-logistics.com/fr/gestion-de-projet/diagramme-de-gantt-3/ [Consulté le 03/2015].

[s6] http://laurent-audibert.developpez.com/Cours-UML/?page=introduction-modelisation-objet [Consulté le 03/2015].

[s7] http://cedric.babault.free.fr/rapport/node4.html [Consulté le 04/2015].

[s8] http://fr.wikipedia.org/wiki/Modèle-vue-contrôleur [Consulté le 04/2015].

[s9] http://fr.wikipedia.org/wiki/Diagramme_de_composants [Consulté le 04/2014].

[s10] http://fr.wikipedia.org/wiki/Diagramme_d%27activité [Consulté le 04/2014].

[s11] http://fr.wikipedia.org/wiki/Raspberry_Pi [Consulté le 05/2015].

[s12] http://www.alsacreations.com/tutoriels/ [Consulté le 05/2015].

[s13] https://NodeJS.org/[Consulté le 05/2015].

[s14] http://www.andreagrandi.it/2013/02/24/[Consulté le 05/2015].

[s15] https://www.mongodb.com/fr [Consulté le 06/2015].

[s16] http://raspbian-france.fr/ [Consulté le 06/2015].

[s17] http://www.santadenn.com/et-pourquoi-pas-chromium/ [Consulté le 06/2015]

Acronymes

Acronymes	Description
API	**A**pplication **P**rogramming **I**nterface
CRT	**C**athode **R**ay **T**ube
CRUD	**C**reate, **R**ead, **U**pdate, **D**elete
CSS	**C**ascading **S**tyle **S**heets
HD	**H**aute **D**éfinition
HDMI	**H**igh **D**efinition **M**ultimedia Interface
HTML	**H**yper**T**ext **M**ark-Up **L**anguage
HTTP	**H**ypertext **T**ransfer **P**rotocol
IU	**I**nterface **U**tilisateur
JDBC	**J**ava **D**ataBase **C**onnectivity
LCD	**L**iquid **C**rystal **d**isplay
LED	**L**ight-**E**mitting **D**iode
MVC	**M**odel **V**iew **C**ontroller
SD	**S**ecure **D**igital
SGBD	**S**ystème de **G**estion de **B**ases de **D**onnées
NoSQL	**N**ot **O**nly **S**tructured **Q**uery **L**anguage
UC	**U**ses **C**ase
UML	**U**nified **M**odeling **L**anguage
USB	**U**niversal **S**erial **B**us
XHTML	**E**xtensible **H**yperText **M**arkup **L**anguage

Annexe

1. Configuration de Raspberry Pi

1.1 Installation du système d'exploitation :

Nous allons montrer l'installation du système d'exploitation sur la carte SD du Rapsberry Pi depuis un PC, puis la configuration de base du système. Nous commençons par télécharger l'image de Raspbian à partir du site officiel de Raspberry Pi, dans notre cas, ca sera la wheezy-raspbian, puis décompressons l'archive zip téléchargé.
Il existe un utilitaire proposé sur le site raspberrypi.org, nous le téléchargeons et installons le Win32DiskImager.
Connectons notre carte SD (8Go) sur notre PC et faire un clique droit sur l'icône Win32DiskImager.exe et choisissons "Exécuter en tant qu'administrateur".

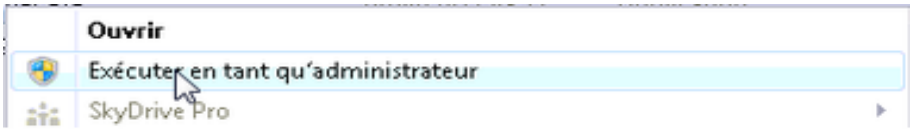


Figure A.1: choisir "Exécuter en tant qu'administrateur"

Une nouvelle fenêtre s'ouvre comme le montre la Figure A.2, cliquons sur le lecteur carte SD dans la colonne de gauche ; copions-collons ensuite l'image Wheezy-Raspbian.img présente dans le fichier zip précédent puis cliquons sur Ouvrir.

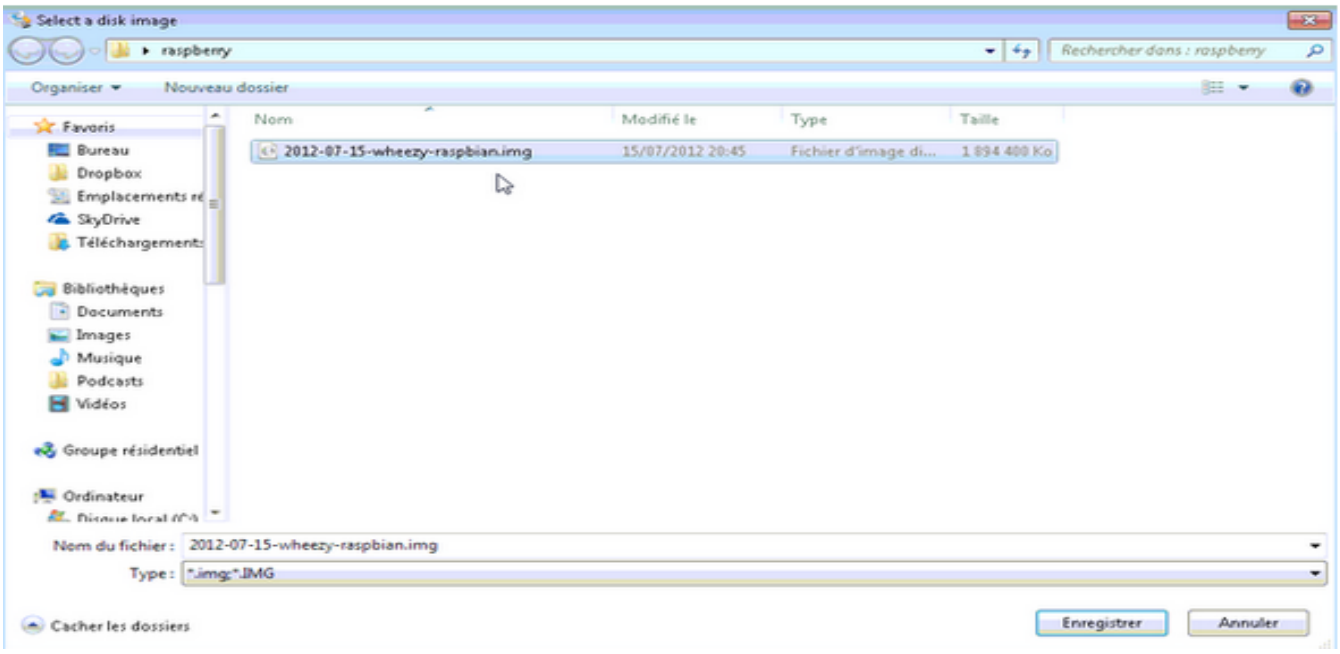


Figure A.2: l'image Wheezy-Raspbian.img

Cliquons enfin sur la fonction Write pour installer Raspbian puis sur Yes.

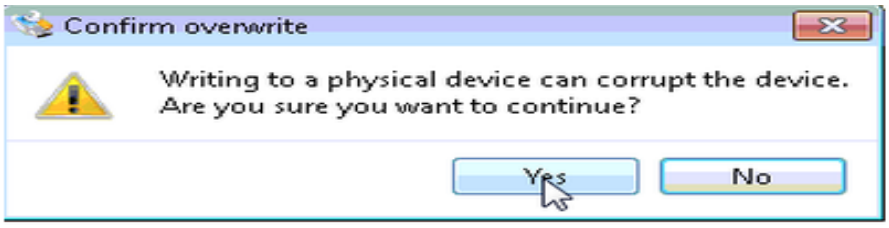


Figure A.3: Installer Raspbian

A la fin de l'installation, nous insérons la carte SD sur le connecteur du Raspberry Pi et nous branchons le clavier et la souris sur les ports USB, le câble HDMI sur l'écran, le câble Ethernet au routeur, et enfin, l'alimentation. Le système devrait démarrer de lui-même.

Figure A.4: Raspberry Pi et carte SD

Figure A.5: Branchement des différents équipements

Lorsque le Raspberry Pi aura achevé son démarrage, l'écran de configuration s'affiche

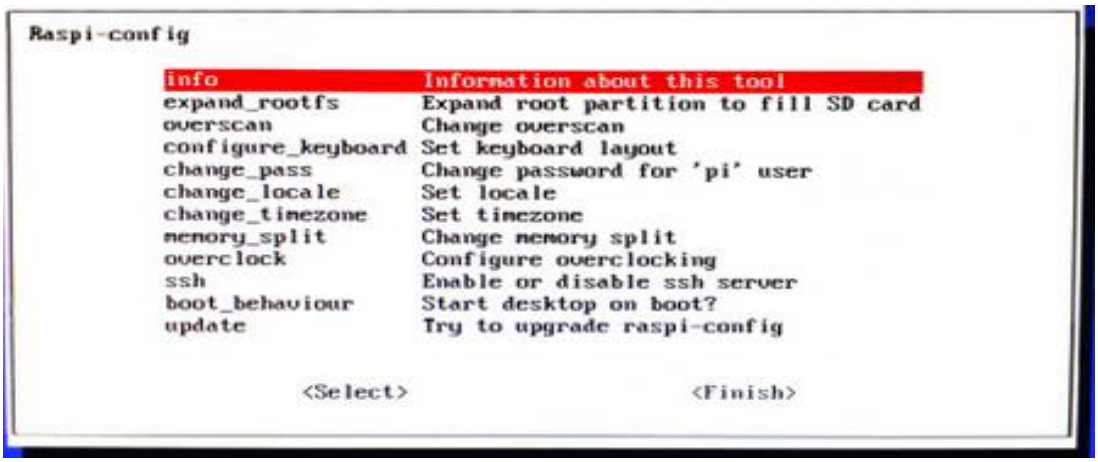


Figure A.6: Ecran de configuration

Quelques options peuvent être intéressantes à changer :

- **expand_rootfs :** permet d'utiliser tout l'espace de notre carte SD.
- **configure_keyboard :** permet de mettre le clavier en français.
- **change_pass :** modifier le mot de passe de l'utilisateur 'pi' (par défaut : raspberry).
- **change_locale :** changer la langue du système.
- **change_timezone :** changer l'heure locale.
- **ssh :** activer le SSH.
- **boot_behavior :** permet d'atterrir directement sur le bureau, pas sur une console.
- **Update :** permet de faire la mise à jour de ce menu via Internet.

Sélectionnons les options qui nous intéressent, puis sélectionnons « Finish » et laissons le Raspberry redémarrer.

Notre Raspberry Pi est maintenant installé et configuré, la Figure A.6 montre le bureau du Raspberyy Pi.

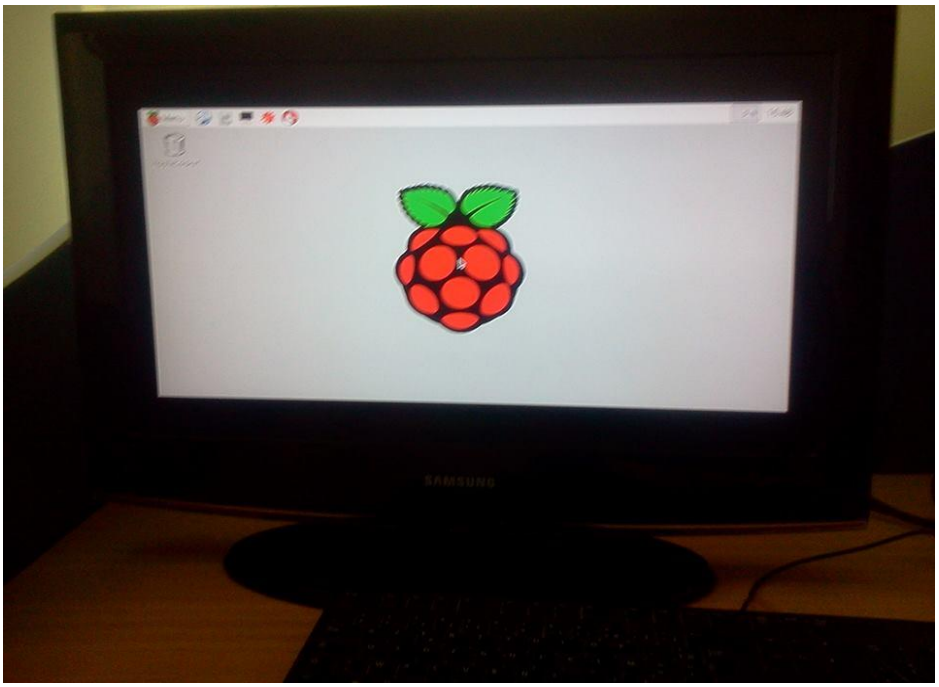


Figure A.7: Bureau du Raspberyy Pi

1.2 Installation du navigateur sous Raspbian

Commençons par ouvrir LX Terminal, et tapons:

sudo apt-get update

Cette ligne aura pour but de rafraichir la liste des paquets disponible, une fois ceci terminé tapons :

sudo apt-get install chromium

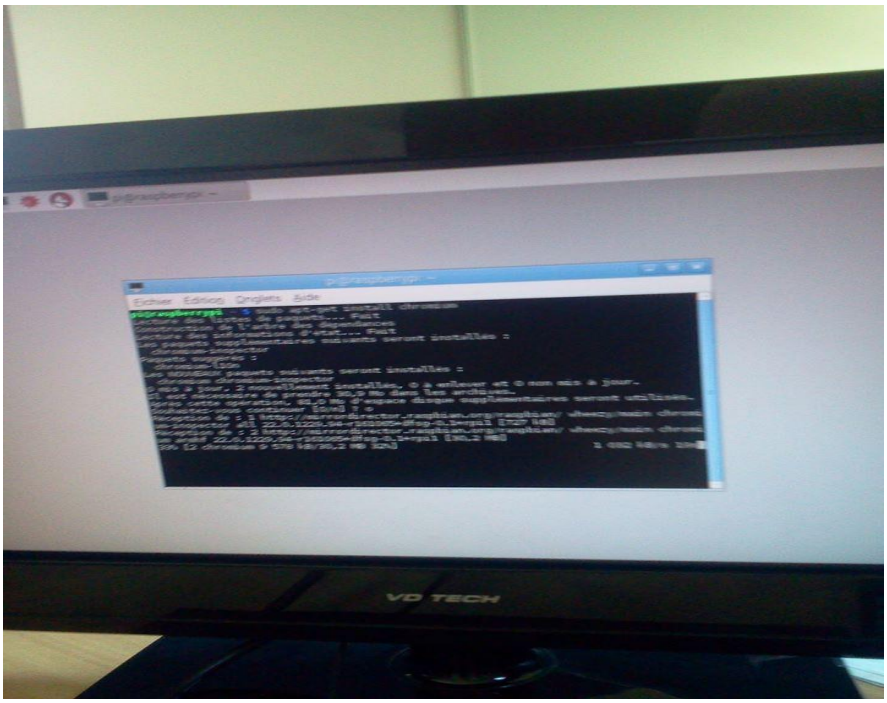


Figure A.8: Installer chromium

Chromium est maintenant installé.

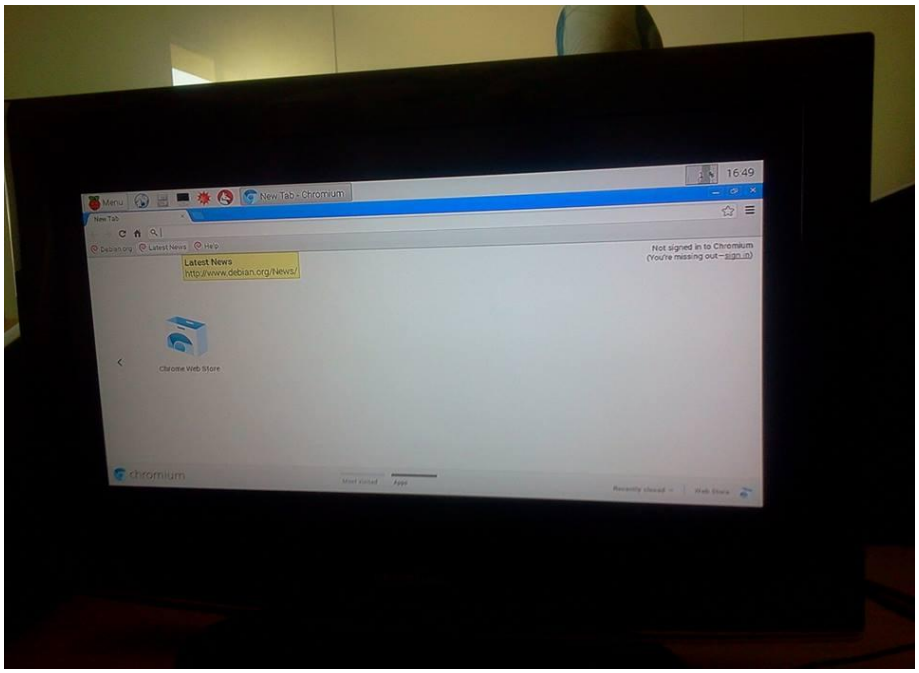


Figure A.9: Chromium est installé

Pour lancer notre application automatiquement lors du démarrage de la télévision, sans passer par le bureau de Raspberry, nous tapons ce script:

openbox &openbox_pid=$!

/usr/bin/chromium-browser &chromium_pid=$!

$chromium_pid

$openbox_pid

1.3 configurer une adresse IP statique

Identifions-nous avec le login « pi » et le mot de passe « raspberry» (par défaut).

Figure A.10: Identification de Raspberry

Tapons dans le terminal :

sudo nano /etc/network/interfaces

Puis

iface eth0 inet static

address 192.168.1.x

netmask 255.255.255.0

network 192.168.1.0

broadcast 192.168.1.255

gateway 192.168.1.1

Ce script permet de configurer le Raspberry pour l'attribuer une adresse IP fixe afin de le connecté au PC avec une adresse inchangeable.

1.4 Intégrer notre application

Apres avoir développé notre application et configurer le Raspberry, il nous reste qu'a démarrer le téléviseur. La page d'authentification s'affiche.

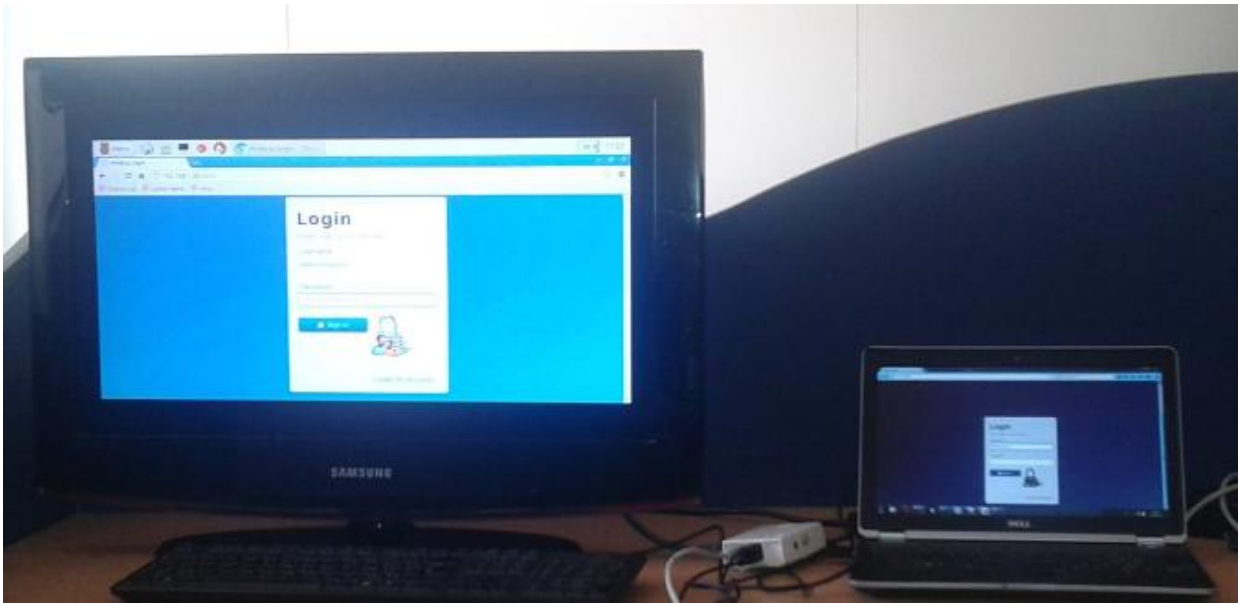


Figure A.11: Page d'authentification

L'interface de la Smart TV s'affiche après l'authentification avec succès.

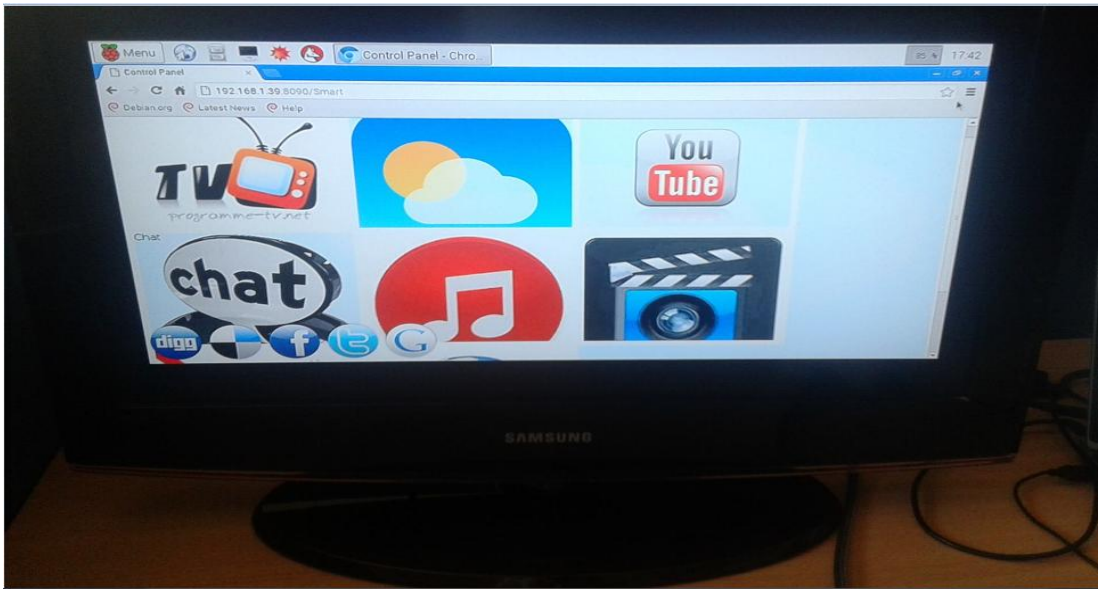


Figure A.12: La Smart TV installée sur le téléviseur