REPUBLIQUE DU CAMEROUN

Paix-Travail-Patrie



Institut Africain D’Informatique

Centre d’excellence technologie Paul Biya

BP : 13719 Yaoundé(Cameroun)

Tel (237)242 72 99 57/ 242 72 99 58/ 691 90 21 20

Site web : www.iaicameroun.com

E-mail : contact@iaicameroun.com

REPUBLIC OF CAMEROON

Peace-Work-Fatherland



INFINITY TECHNOLOGY

Tel (237) 659107855

Site web : [www.infinitytechnology.fr](http://www.infinitytechnology.fr)

E-mail infinitech237@gmail.com

THEME : MISE EN PLACE D’UNE APPLICATION WEB DE TELECONSULTATION

**Et**

**SOUS LA SUPERVISION DE :**

**Rédigé Par : Mr TAGNE LOIC FRANCK étudiant à IAI-CAMEROUN**

RAPPORT DE STAGE ACADEMIQUE

**Stage effectué du 8 juillet au 30 septembre 2024**

**En vue de l’obtention du Diplôme de Technicien Supérieur (DTS) option Génie Logiciel**

**Encadrant professionnel**

**Mme : EMELDA TANKEU**

**Encadrant académique**

**Mme : ANEGA YVETTE**

**ANNEE ACADEMIQUE : 2023-2024**

**SOMMAIRE**

**DEDICACE**

A

TOUTE

MA

FAMILLE

**REMERCIEMENT**

Nous Tenons à ne remercier toute personne ayant contribuées de près ou de loin à notre formation, au bon déroulement de notre stage. A la réalisation de ce projet et à sa bonne présentation. Nos remerciement s’adressent à :

* Au représentant Résident de l’institut Africain d’Informatique représentation de Cameroun MR **ARMAND CLAUDE ABANDA** qui œuvre sans relâche pour notre bon encadrement et pour cette opportunité qui nous est offert d’étudier en toute quiétude au Centre d’excellence Technologie Paule Biya, pour avoir faire naitre cet institut.
* Mr **OSTIE JOSPIN Mbita** Messine pour avoir permis que j’effectue le stage dans cette structure.
* Mme **EMELDA TANKEU** notre encadrant professionnel pour son soutien et son suivi sur tous les plan durant notre période de stage.
* Mme **ANEGA YVETTE** notre encadrant académique pour son aide et ses précieux conseil pour le déroulement de notre stage.
* Mes parent, frere, sœur, amis pour leurs encouragements, critiques, réprimandes et leur soutient qui ont contribué à notre évolution.

LISTE DES TABLEAUX

**LISTE DES FIGURES**

**GLOSSAIRE**

**RESUME**

La digitalisation et la numérisation sont au centre du processus de développement des sociétés en particulier le Cameroun, ce sont des concepts indispensables pour l’émergence du Cameroun que l’Etat promeut. Elles jouent un rôle majeur aussi bien dans le développement des entreprises que dans les structures administratives. Afin d’améliorer une consultation fluide, rapide, d’un gain de temps, c’est-à-dire entrer en contact avec un médecin afin d’expose son probleme sans frusquassions, la possibilité de mettre sur pied un site de téléconsultation a fini par voir le jour. Notre thème de travail était axé principalement sur la conception et réalisation d’une application web de téléconsultation, ceci précédé par une analyse basée sur le langage de modélisation UML

**ABSTRACT**

**Digitalization and digitization are at the heart of the development processif of societies, particularly in Cameroon, where these concepts are essential for the country's emergence, which the State promotes. They play a major role in both business development and administrative structures. To improve smooth and quick consultations, saving time, that is, to contact a doctor to present one’s problem without frustrations, the possibility of establishing a teleconsultation website has finally come to fruition. Our work theme was primarily focused on the design and implementation of a web application for teleconsultation, preceded by an analysis based on the UML modeling language.**

**INTRODUCTION GENERAL**

Le mondecontemporain est marqué par une évolution rapide des technologies de l’information et de la communication, qui ont transformé de nombreux aspect de notre vie quotidienne. L’une des avancées les plus significatives est l’utilisation croissante des applications mobiles, qui permettent aux utilisateurs d’accéder à une multitude de services à partir de leurs smartphones. Dans ce contexte, les gouvernements locaux cherchent à moderniser leurs processus administratifs en adoptant des solutions numériques innovantes. L’un des domaines qui bénéficie de cette transformation est la téléconsultation. Les hôpitaux doivent faire faces à des défis tels que les consultations à distances, échange avec les patients a distances. De meme que les patients qui ont du mal à attendre dans les files d’attente dans les hôpitaux pour rencontre les docteurs ou médecin. Le présent rapport se propose d’explorer la conception et la réalisation d’une application web de téléconsultation, dans le cadre d’un stage académique effectué à l’entreprise INFINITY TECHNOLOGIE. L’objectif principal de ce projet est de simplifier et optimiser les consultations et la prise de rendez-vous avec un médecin. Au cours de ce stage nous avons eu l’opportunité de travailler en étroite collaboration avec les responsables de l’entreprise INFINITY TECHNOLOGIE, afin de comprendre les besoins les besoins spécifiques liés à la téléconsultation. Nous avons également réalisé une analyse approfondie des technologies existantes et des meilleures pratiques en matière de développement web. Le rapport présentera les différentes étapes du processus de conception et de réalisation de l’application web, en mettant l’accent sur les choix techniques, les fonctionnalités clés et les défis rencontrés. Nous aborderons également les avantages potentiels de cette solution, tant pour les populations que pour les hôpitaux. Ce rapport vise à fournir une vue d’ensemble complète du travail réalisé lors de ce stage académique, en mettant en évidence l’importance de l’utilisation des technologies web pour améliorer les processus de consultation. Nous espérons que les résultats de ce projet contribueront à moderniser les consultations des hôpitaux, en offrant une solution pratique, efficace et accessible à toute la population.

PARTIE I : PHASE D’INSERTION

Le dossier d’insertion est le tout premier document rédigé durant la période de stage académique. C’est la partie du stage qui présente la structure d’accueil, son fonctionnement, ainsi que l’accueil de l’étudiant en entreprise. Cette phase d’insertion dure généralement deux semaines et permet au stagiaire de se familiariser avec son nouvel environnement.

PLAN

INTRODUCTION

1. ACCUEIL ET INTEGRATION DANS LA STRUCTURE
2. PRESENTATION DE LA STRUCTURE
3. PRESENTATION DU THEME

CONCLUSION

**INTRODUCTION**

Dans le but d’obtenir le Diplôme de Technicien Supérieur (DTS) nous sommes conviés à effectuer un stage académique de 03 mois afin de s’imbiber professionnel et de mettre en pratique les notions acquises au cours de ses deux années de formation. Pour que ces objectifs soient atteints, nous avons (nous avons effectué un stage au sein de la structure …) été accueillis à INFINITY TECHNOLOGY pour mieux implémenter nos connaissances et apprendre d’avantage.

1. **ACCUEIL EN ENTREPRISE ET INTEGRATION**
2. **Accueil**

L’accueil a joué un rôle fondamental lors de notre stage. Dès notre première journée, nous avons été chaleureusement accueillis par L’équipe et introduit à l’environnement de travail dynamique et discipliné de l’entreprise. Les responsables nous ont fourni toutes les informations nécessaires concernant les politiques internes, les horaires de travail et les différentes équipes au sein de l’entreprise.

1. Intégration

La première semaine s'est révélée fructueuse et très bénéfique pour nous. En effet, nous avons eu le privilège de participer à des sessions de formation qui nous ont permis de mettre à jour nos connaissances et d'acquérir de nouvelles compétences. Ces échanges avec nos collègues et formateurs ont non seulement enrichi notre savoir-faire, mais ont également favorisé un climat de confiance et de collaboration. La deuxième semaine, quant à elle, nous a offert l'opportunité de mieux comprendre et d'analyser notre thème de stage. Nous avons pu cerner les contours de notre projet, en identifiant les enjeux et les objectifs à atteindre. Cette immersion nous a également permis d'observer les dynamiques de travail au sein de l’équipe, ce qui est essentiel pour notre intégration future. Pour améliorer notre intégration, nous avons pris l’initiative d’organiser des rencontres informelles avec nos collègues, afin de renforcer les liens et d’échanger sur nos expériences respectives. Nous avons également sollicité des retours sur notre travail, ce qui nous a permis d’identifier nos points forts et les axes d’amélioration. Ces démarches témoignent de notre volonté de nous intégrer pleinement et de contribuer activement à l’équipe. En somme, ces premières semaines ont été déterminantes pour poser les bases d’une expérience de stage enrichissante et formatric

1. **PRESENTATION DE LA STRUCTURE**
2. **Historique et Mission**

#### Historique

INFINITY TECHNOLOGY SARL est un projet issu d'une série de réflexions du PDG M. OSTIE JOSPIN Mbita, étudiant en génie logiciel, spécialiste du marketing numérique et du référencement, chef de projet informatique certifié de l’United Kingdom Professional Development Academy et jeune entrepreneur informatique. L'entreprise a été fondée en 2023.

1. Mission

Infinity Technology est un fournisseur de solutions technologiques à service complet dédié à offrir aux entreprises et aux particuliers des services et produits informatiques de pointe. Leurs offres complètes comprennent la gestion de projets informatiques, la conception de logos, le montage vidéo, le développement Web, la conception graphique, la formation certifiée, le développement d'IA, la maintenance informatique et la cyber sécurité. Guidé par une mission d'excellence, Infinity Technology s'efforce de fournir des solutions innovantes et sur mesure qui ouvrent des possibilités infinies à leurs clients. Grâce à un engagement envers la qualité, la créativité et le service client, l'entreprise vise à être le partenaire de confiance dans l'exploitation de la puissance de la technologie pour atteindre la croissance, la productivité et le succès.

1. **Situation Géographique**

La société **INFINITY TECHNOLOGY** est située dans la ville de Yaoundé précisément MVOG ADA HYSACAM CARREFOUR. Elle peut être localisée grâce au plan de localisation ci-dessous

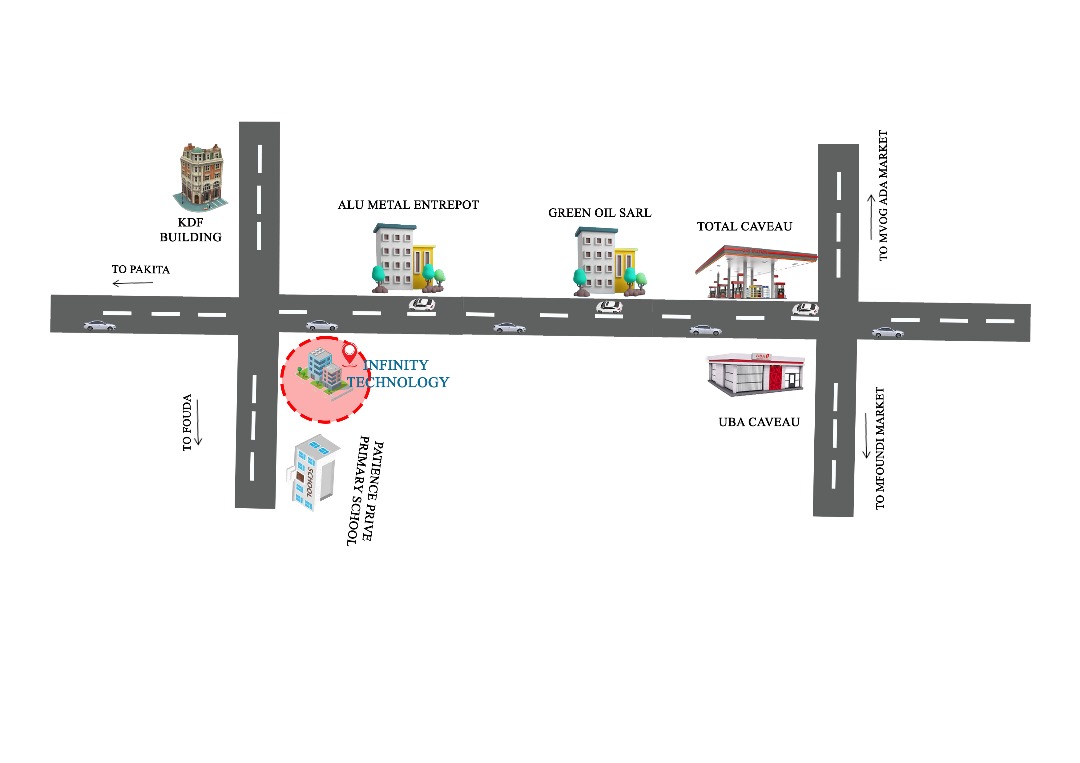


Figure 1:plan de localisation

1. **FORMULAIRE D'IDENTIFICATION DE L'ENTREPRISE**

|  |  |
| --- | --- |
| FORMULAIRE D'IDENTIFICATION | |
| Nom de l'entreprise | TECHNOLOGIE INFINI |
| Directeur général | M. OSTIE JOSPIN Mbita Messine |
| Type d'entreprise | Entreprise informatique |
| Numéro de téléphone | +237 659-107-885 |
| Site web | http://infinitytechnology.fr |
| Langue | FRANÇAIS ANGLAIS |
| Logo |  |

DISPOSITIF MATÉRIEL ET LOGICIEL

1. RESSOURCES MATÉRIELLES

|  |  |  |
| --- | --- | --- |
| RESSOURCES MATÉRIELLES | | |
| N0 | Nom | Description |
| 1 | Imprimante | Ricoh Aficio sp c4010, canon ts5040, hp deskjet 2130 |
| 2 | Serveur | Ordinateur de bureau HP Pentium 5 Intel, processeur 3.0 Ram 16 |
| 3 | ORDINATEUR | DELL, HP, ACCER, LENOVO |
| 4 | SWITCH | LANCON GS-1224 |
| 5 | MODEM | CAMTEL |

1. RESSOURCES LOGICIELLES

|  |  |  |
| --- | --- | --- |
| RESSOURCES LOGICIELLES | | |
| SYSTÈME OPÉRATEUR | LOGICIEL | ANTIVIRUS |
| WINDOWS | ANDROID STUDIOS, NETBEANS, POSTGRESQL, ADOBE POWER AMC, OPERA, CHROME, PHOTOSHOP | WINDOWS SECURITY |
| Linux | COLLECTION, VISUAL PARADIGM, 3D MAX, VISUAL STUDIO CODE, MYSQL | / |
| MAC OS | XCODE, MYSQL, VISUAL STUDIO CODE, MYSQL | CLEANMYMAC X |

CONCLUSION

Notre période d’insertion en entreprise nous a permis de nous accommoder à l’environnement professionnel, aux relations humaines en entreprise et au rythme de travail. Nous nous sommes adaptés assez facilement avec le fonctionnement du système et avons été intégré dans la structure. Il sera question pour nous dans la suite d’entrer dans le vif du sujet avec l’analyse de notre projet.

PARTIE II : PHASE TECHNIQUE

**Résumé :**

La phase technique marque le début de l’étude et de la conception proprement dite de notre sujet d’étude. L’accent sera mis ici sur une analyse bien détaillé de notre projet afin de ressortir ses différentes spécificités et de le mettre en place.

**PLAN**

DOSSIER 1 :L’EXISTANT

DOSSIER 2 : CAHIER DE CHARGE

DOSSIER 3 : DOSSIER D’ANALYSE

DOSSIER 4 : DOSSIER DE CONCEPTION

DOSSIER 5 : DOSSIER DE REALISATION OU DE DEPLOIEMENT

DOSSIER 6 : TEST DES FONCTIONNALITE

DOSSIER 7 : GUIDE D’UTILISATION

DOSSIER 1 : L’EXISTANT

L’existant correspond à la situation ou au système en vigueur à l’heure actuelle. Il s’agit d’un processus qui consiste à décrire et à analyser le système déjà en place.

**PLAN**

INTRODUCTION

1. PRESENTATION DU THEME
2. DESCRIPTION DE L’EXISTANT
3. CRITIQUE ET LIMITE DE L’EXISTANT
4. PROBLEMATIQUE
5. PROPOSITION DE LA SOLUTION

CONCLUSION

INTRODUCTION

Pour la bonne réalisation d’un projet informatique, il est nécessaire de faire une étude au préalable du système existant. De ce fait, apres avoir ressorti les spécificités du projet, il sera question pour nous d’effectuer une description précis et détaillée du système global de gestion d’une téléconsultation. Nous procèderons donc à la description de l’existant, la critique de l’existant, une problématique, la proposition de solution et une capture des besoins fonctionnels de ladite solution.

.

1. PRESENTATION DU THEME

Notre projet d’implémentation d’une plateforme de téléconsultation vise à développer une solution innovante pour le processus de consultation en ligne.

L’objectif principal de ce projet est de concevoir une plateforme permettant aux malades d’effectuer leurs consultations depuis leurs domicile a parti de leur smartphone. Cette initiative s’inscrit dans une démarche de modernisation des services médicaux, en offrant aux malades une alternative aux procédures traditionnelles de consultation ; souvent stressant et pénible.

Notre application mettra l’accent sur la simplicité d’utilisation, en offrant une interface conviviale.

Une fois la phase de conception terminée, nous procéderons au développement et à la mise en œuvre de l’application en utilisant les meilleures pratiques de développent web.

Nous évaluerons également l’efficacité et l’acceptation de l’application par les utilisateurs finaux, en recueillant leurs retours et en effectuant des tests utilisateurs. Cette évaluation nous permettra d’apporter les ajustements nécessaires et de proposer des recommandations pour améliorer l’expérience utilisateur et assurer le déploiement a plus grande échelle de l’application.

En résumé, notre projet vise à développer une application web de téléconsultation. En offrant une solution pratique, sécurisée et moderne, nous aspirons à simplifier et à optimiser le processus de consultation, contribuant ainsi à la modernisation des services médicaux

1. DESCRIPTION DE L’EXISTANT
2. CRITIQUES ET LIMITES DE L’EXISTANT
3. PROBLEMATIQUE
4. PROPOSITION DES SOLUTIONS

CONCLUSION

INTRODUCTION

1. CONTEXTE ET JUSTIFICATION DU PROJET
2. LES OBJECTIFS DU PROJET
3. BENEFICIAIRES ET CIBLES
4. EXPRESSION DES BESOINS DES UTILISATEURS
5. ESTIMATION DU COUT DU PROJET
6. PLANIFICATION DU PROJET
7. CONTRAINTES DU PROJET
8. LES LIVRABLES

CONCLUSION

DOSSIER II : CAHIER DE CHARGE

Le cahier de charge est un document essentiel élaboré par le responsable du projet en collaboration avec le commanditaire. Il sert de référence tout au long du projet et définit de manière détaillée les fonctionnalités de l’application à développer, tout en mettant l’accent sur les contraintes, les ressources nécessaires et les livrables attendus. Il constitue un guide précis qui permet de garantir la conformité et la réussite du projet dans le respect des attentes et des exigences de toutes les parties prenantes.

**PLAN :**

INTRODUCTION

Le cahier des charges est un document qui donne les directives relatives aux produits à livrer, les conditions de leur livraison et les spécifications techniques des livrables. Il est établi par le maître d’œuvre en accord avec le maître d’ouvrage et présente un ensemble d’étapes d’orientations et de procédures. Notamment il permet de définir et de comprendre le travail à réaliser. En effet, il précise les attentes du client sur le nouveau système d’information, les détails à respecter ainsi que les contraintes sur les outils à utiliser. Le cahier des charges permet d’organiser, de lister, de cadrer l’ensemble d’un projet avec précision, il doit être compréhensible par le profane. La présente section définit le cahier des charges pour la mise en place d’une plateforme de téléconsultation qui s’articulera autour des points suivants : tout d’abord le contexte et les objectifs ; par la suite nous indiquerons la cible visée et les bénéfices de notre travail ; nous continuerons notre exposé par la mise en exergue des besoins des utilisateurs (fonctionnalités de l’application) ; ensuite, par la planification du travail et l’estimation du coût du projet seront présentées ; nous terminerons enfin cette section par l’évaluation des contraintes du projet et la présentation des livrables.

1. CONTEXTE ET JUSTIFICATION DU PROJET
2. Contexte du Projet

La consultation peut se définir comme étant la rencontre entre un professionnel de la santé, généralement un médecin, pour discuter des symptômes, des préoccupations médicales, et pour obtenir un diagnostic et des recommandations de traitement. Avec l’avènement des TIC et d’internet, une autre forme de consultation a vu le jour : il s’agit de la **téléconsultation.**  Cette dernière s’effectue essentiellement en ligne. Cependant de nombreuses plateformes ont été mises sur pied pour faciliter les consultations dans notre société ; car nombreux sont ceux qui ont des difficultés à se déplacé pour se rendre dans un hôpital.

Il sera question pour nous de concevoir un système dans lequel chaque patient aura la possibilité se faire consulter meme en étant dans son domicile et recevoir son sa liste d’ordonnance. Suite à cela, les patients pourront contacter des médecins pour avoir une consultation à distance via un smartphone ou un ordinateur.

1. Justification du Projet

L’intérêt de notre étude est de permettre aux malades d’etre en contact avec des médecins avec qui ils pourront échanger facilement échanger. La mise sur pied de notre plateforme permettra :

* Faciliter la consultation
* Faciliter la prise de rendez-vous
* Sensibiliser la population
* Limiter les dépenses de transport

1. OBJECTIF DU PROJET
2. Objectif Général

L’objectif général ici est de mettre sur pied une plateforme de téléconsultation pour permettre au personne malades ou personne ayant besoin d’un suivi médical d’entrer en contact avec des professionnel de la sante afin de se faire consulter sans quitter leurs domicile.

1. Objectif Spécifiques

Nous avons constaté dans notre société qu’il y’a des personnes qui vivent dans des zones reculées et qui ont du mal à se rendre dans un hôpital; d’autre part les malades sont confronté à de longue file d’attente. Nous avons alors trouvé nécessaire de mettre sur pied une plateforme sur laquelle:

* Le malade pourra entre en contact avec un médecin pour se faire consulter
* Le malade pourra voir les médecins disponibles dans la plateforme afin de les contacter
* Les médecins pourrons effectuer des annonces questions de sensibilisé les populations
* Le patient pourra gérer ses rendez-vous (prendre, modifier ou supprimer)
* Noté un médecin
* Le médecin pourra suivre un patient
* Le médecin pourra gérer des rendez-vous (annuler, refuser et modifier)
* L’administrateur pourra gérer les comptes
* L’administrateur pourra Supprimer un utilisateur
* L’administrateur pourra Supprimer une annonce

1. BÉNEFICIAIRES ET CIBLES

Notre projet ne s’adresse principalement qu’à toute personne qui vivant dans des zones reculées, personne âgée et des personnes ayant besoin d’un suivi médical particulier. Vue la difficulté que ses personne rencontre pour se rendre dans un hôpital, nous avons pensé un système pour faire des consultations sans se déplacé.

1. EXPRESSION DES BESOINS DES UTILISATEURS
2. Les acteurs du système

Un acteur est toute personne qui interagi avec le système. En ce qui concerne notre système, les acteurs sont les suivants :

* Administrateur : charger de la gestion globale du système. Ils ont accès à des fonctionnalités administratives avancées leur permettant de gérer et de contrôler les activités du système ainsi que de sa maintenance.
* Médecin : charger de prendre soin des malades
* Malade :
* Visiteur :

1. Les besoins fonctionnels

* Pour l’Administrateur
* Supprimer un utilisateur
* Supprimer une annonce
* Gérer les comptes des médecins
* S’authentifier
* Pour le Médecin
* Faire une annonce
* S’authentifier
* Suivre un patient
* Contacte un patient
* Gérer un rendez-vous
* Supprimer son compte
* Modifier son mot de passe
* Pour le Patient
* Prendre un rendez-vous
* S’authentifier
* Se faire consulter
* Noté un médecin
* Gérer son rendez-vous
* Supprimer un compte
* Payer une commission
* Modifier son mot de passe
* Passe un appel vidéo
* Pour le visiteur
* Visiter la plateforme
* Créer un compte

1. Les Besoins non fonctionnels

Comme besoin non fonctionnel de notre plateforme nous pouvons citer :

* Performance : l’application doit etre performante, c’est-à-dire qu’elle doit etre rapide, réactive et capable de gérer un grand nombre d’utilisateurs et de transactions simultanément.
* Sécurisé : la sécurité des données personnelles des utilisateurs est primordiale. L’application doit mettre en œuvre des mesures de sécurité telles que le cryptage des données, l’authentification sécurisé et la protection contre les attaques.
* Disponibilité: la plateforme doit etre disponible sept jour sur sept (7j/7) afin que les malades puisse effectuer leurs consultation peut import le jour.
* Convivialité : l’interface utilisateur doit etre convivial et facile à utiliser meme pour des utilisateurs novices en technologie.
* Fiabilité : la plateforme doit etre fiable. Les erreurs et les problèmes doivent etre gérer de manière adéquate.
* Evolutivité : la plateforme doit etre évolutives pour pouvoir gérer une augmentation du nombre d’utilisateur et des fonctionnalités.
* Confidentialité : les informations des utilisateurs y compris leurs données personnelles doivent etre garantie.

1. ESTIMATION ET COUT DU PROJET
2. Ressources Logicielles

|  |  |  |
| --- | --- | --- |
| Logiciels | Rôles | Prix (CFA) |
| vscode | Permet de développe les applications web | gratuit |
| Wamp server | Permet la mise en place d’un server local | gratuit |
| Gant Projet | Permet la gestion du temps et des ressources pour une bonne planification du projet |  |
| Navigateur (Google chrome, Microsoft Edge) | Logiciel pour ouvrir mon application web | gratuit |
| MySQL | Système de gestion de base de données | gratuit |
| Windows 10 | Système d’exploitation |  |
| Microsoft office 2013 | Logiciel de traitement de texte |  |
| Sybase power AMC | Logiciel de Modélisation |  |
|  |  |  |

1. Ressources Matérielles

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| désignation | quantité | caracteristique | Cout unitaire  (FCFA) | Cout total |
| ordinateur | 01 |  |  |  |
| Clé USB | 01 |  |  |  |
| Support papier | 01 |  |  |  |
| modem | 01 |  |  |  |
| CD-ROM |  |  |  |  |
| TOTAL | | | |  |

1. Ressources Humaines

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Fonction | Quantité | Description | Cout /j (en FCFA) | jour | Total (en FCFA) |
| Analyste concepteur | 01 | Effectuant une analyse approfondie à l’aide d’outils afin de prévoir le futur système |  | 30 |  |
| Développeur | 01 | Il procèdera à l’implémentation de la solution à l’aide des directives de l’analyste. |  |  |  |
| Testeur | 01 | Effectue des tests unitaires afin de valider la solution obtenue |  |  |  |
| Formateur | 02 | S’occupent de la formation des futurs utilisateurs du logiciel |  |  |  |
| TOTAL | | | | |  |

1. Ressources Financières

|  |  |
| --- | --- |
| Désignation | MONTANT (En FCFA) |
| Ressources logicielles |  |
| Ressources Matérielles |  |
| Ressources Humaines |  |
| Imprévus (Fourchette de 20% sur le total) |  |
| TOTAL |  |

1. PLANIFICATION DU PROJET

Cette étape nous permettra de représenter la répartition des taches sur des intervalles de temps bien définis. Pour cette planification, nous ferons recours au logiciel Gantt Project. Nous obtiendrons donc un chronogramme de Gantt qui représentera visuellement l’état d’avancement de nos différentes tâches.

1. Déroulement

Le diagramme de Gantt est un outil qui permet sur un même plan de faire apparaitre les taches d’un projet leur ordonnancement et dépendance ainsi que le suivie de l’avancement et les ressources affectées.

1. Planification des taches

|  |  |  |
| --- | --- | --- |
|  | taches | durée |
| Intégration | Insertion en entreprise | 10 |
| Collecte de données et élaboration de l’existant | 2 |
| Elaboration du cahier de charge | 3 |
| Développement | Analyse | 15 |
| Analyse et conception | 10 |
| Codage | 40 |
| Déploiement | Tests unitaire et d’intégration | 2 |
| validation | 4 |
| déploiement | / |
| Exploitation | / | Indéterminé |

1. Diagramme de Gantt
2. CONTRAINTES DU PROJET

Les contraintes font références non pas aux obstacles mais plutôt aux exigences qui nous ont été donnés par notre encadreur pour le travail que nous devons fournir. Pour ce faire nous pouvons les classer de la manière suivante :

* Contrainte de cout : la réalisation de notre projet nécessitera des dépendes en ressources humaines, matériels et logiciels pour un montant de ??
* Contrainte de délais : ce projet devra etre réalisé sur une période de trois (3) mois allant du 08juillet au 30 septembre 2024
* Contrainte de qualité : sécurité, robustesse et fiabilité
* L’expérience utilisateur : l’application doit offrir une expérience utilisateur intuitive, conviviale et attrayante.

1. LES LIVRABLES

Apres avoir développé et clôturer notre travail nous devons fournir des biens livrables qui sont :

• Un CD : dans le quel sera gravé le setup de l’application crée ainsi que le code source de celle-ci et dans une moindre mesure le guide d’utilisation puisque celui-ci figurera dans notre bien livrable que nous citerons par la suite.

• Un rapport : le rapport contient l’analyse complète du système que nous avons développé avec des diagrammes liés au langage de modélisation qui a été utilisé ; Nous retrouverons aussi dans le rapport un dossier d’insertion qui parle de l’entreprise et du milieu professionnel dans lequel nous nous présenterons le langage de programmation utilisé ainsi que les outils de développement, l’architecture de l’application et le script de la base de données sans toutefois oublier le guide d’utilisation qui montrera à l’utilisateur comment utiliser l’application.

CONCLUSION

En somme, le cahier des charges nous a permis de représenter les attentes du maitre d’ouvrage, définir les différents besoins de l’utilisateur, spécifier les contraintes du projet, ressortir les ressources nécessaires à la réalisation du projet, enfin, d’organiser les tâches. Cette partie primordiale terminée, il est question pour nous de passer à présent à la partie suivante de notre travail, qui repose sur une analyse plus détaillée de notre projet.

DOSSIER III : DOSSIER D’ANALYSE

Le dossier d’analyse occupe une place primordiale dans la réalisation d’un projet. Il constitue une étape essentielle ou sont présentées de manière claire et structurée plusieurs composantes fondamentales. Il comprend notamment l’étude de l’existant, une analyse critique de cette situation, la problématique identifiée, la proposition de solution, le choix de la méthode de modélisation et enfin la modélisation de la solution envisagée. Ce dossier d’analyse permet d’avoir une vision approfondie du projet, de mettre en lumière les enjeux et les solutions envisagées, tout en fournissant une base solide pour la suite du processus de réalisation

PLAN

INTRODUCTION

1. METHODOLOGIE
2. CAPTURE DES BESOINS FONCTIONNELS

CONCLUSION

INTRODUCTION

L’analyse joue un rôle crucial dans la réussite d’un projet. Car une analyse insuffisante ne peut conduire à des résultats satisfaisants. Cette étude se concentre spécifiquement sur les spécifications fonctionnelles de l’application, afin de clarifier les fonctionnalités qu’elle doit offrir aux utilisateurs. Les résultats de cette analyse ne sont pas dépendants d’une technologie spécifique. Cependant, il permet de délimiter le périmètre de l’application à concevoir, en identifiant les besoins des futurs utilisateurs. Dans cette optique, il est essentiel de choisir une méthode qui facile le suivie du développement de notre application. C’est pourquoi nous avons opté pour la méthode 2TUP du langage UML. Ainsi, notre objectif est de présenter de manière détaillée cette méthode d’analyse, de justifier notre choix en fonction de notre domaine d’étude ; et d’utiliser les protocoles et normes de cette méthode pour analyser le problème spécifique qui se pose.

1. METHODOLOGIE

L’analyse est une étape fondamentale dans la conception d’un logiciel. C’est la base de tous travaux de réalisation de système d’information. Un système d’information est un système organisé de ressources, de personnes, et de structures qui évoluent dans une organisation et dont le comportement coordonné vise à atteindre un but commun. Plusieurs méthodes de langages ont été développées pour faciliter et normaliser l’analyse et la conception des systèmes d’information parmi lesquels nous avons principalement UML et MERISE.

MERISE (Méthode d’Etude et de Réalisation Informatique pour les Systèmes d’Entreprise) est une méthode d’analyse, de conception structurelle et de réalisation des systèmes d’informations très utiles notamment dans les entreprises françaises. Elle est basée sur la séparation des données et des traitements à effectuer en plusieurs modèles conceptuels et physique. Son but principal est d’arriver à concevoir un système d’information (SI). C’est une méthode systémique d’analyse et de conception de SI qui propose de considérer le système réel selon deux points de vue : une vue statique (données) et une vue dynamique (traitement).

UML (Unifie Modeling Language) quant à lui, est un langage de modélisation des systèmes standards, qui utilise des diagrammes pour présenter chaque aspect d’un système en s’appuyant sur la notion d’orienté objet qui es un véritable atout pour ce langage. UML propose donc une approche différente de celle de MERISE en ce sens qu’il associe les données aux traitements. En effet, avec UML centraliser les données d’un type de traitement associé permet de limiter les points de maintenance dans le code et faciliter l’accès à l’information en cas d’évolution de logiciel. De plus, UML décrit la dynamique du système d’information comme un ensemble d’opérations attachée aux objets du système.

1. Analyse comparative

UML et merise ont des approches différentes. Tout d’abord, MERISE prône une séparation des données et des traitements tandis qu’avec UML, les données sont associées à leur traitement. Le tableau ci-dessus illustre une comparaison entre ces deux méthodes.

|  |  |
| --- | --- |
| **MERISE** | **UML** |
| Méthode d’étude et de Réalisation Informatique pour les Systèmes  d’Entreprises. | Unifie Modeling Language (Langage de modélisation Unifie). |
| MERISE est une méthode systémique d’analyse et de conception de systèmes d’information. C’est-à-dire qu’elle utilise une approche systémique. | UML est un langage de modélisation objet à qui il faut associer un processus (2TUP, RUP, XUP) pour en faire une méthode. |
| MERISE propose de considérer le système réel selon deux points de vue :   * **Une vue statique (données)** * **Une vue dynamique (traitements)** Avec la méthode MERISE ? nous avons une étude séparée des données et des traitements. | UML propose une approche différente en ce sens qu’il associe les données et les traitements.  Avec UML, centraliser les données d’un type et les traitements associés permet de limiter les points de maintenance dans le code et faciliter l’accès à l’information en cas d’évolution. |
| Approche systémique. | Approche objet. |
| Démarche **« Bottom Up »** : De la base de données vers le code. | Démarche « **Top Down** » : Du modèle vers la base de données. |

1. Choix de la démarche d’analyse

Commençons d’abord par définir ce que c’est qu’un modèle. Un **modèle** est donc une représentation simplifiée d’une réalité, il permet de capturer des aspects pertinents pour un objectif défini à priori.

**UML (**Unified Modeling Language**)** est un langage de modélisation orienté objet graphique basé sur des pictogrammes. Il est apparu dans le monde du génie logiciel pour résoudre les problèmes de contraintes qui survenaient à différentes phases de conception d’un logiciel. Il en résulte que de nombreuses méthodes de développement ou d’analyse de logiciel ont vu le jour chacune plus ou moins adaptée ou spécialisée à une démarche particulière. Celles-ci ayant été développées indépendamment les unes des autres, elles sont souvent partiellement redondantes ou incompatibles lorsqu’elles font appel à des notations ou des terminologies différentes. UML est donc l’accomplissement de précédents langages de modélisation que nous allons lister ici :

* **OMT** (Object Modeling Technic) de **James Rumbaugh,** c’est une méthode d’analyse et de conception orienté objet qui insistait sur l’analyse des systèmes logiciels.
* **OOD** (Object Oriented Design) de **Grady Booch**, c’était une méthode de conception orienté objet qui mettait un accent sur le design et la construction des systèmes logiciels.
* **OOSE** (Object Oriented Software Engineering) d’**Ivar Jacobson** elle se concentrait au business engineering et à l’analyse des exigences.

UML est donc un langage qui permet de définir des modèles sans toutefois définir leurs

procéssus d’élaboration.

UML à partir de sa version 1.3 propose neuf (09) diagrammes tandis qu’il en existe quatorze (14) depuis UML 2.3. Les 14 diagrammes UML sont dépendants hiérarchiquement et se complètent de façon à permettre la modélisation d’un projet tout au long de son cycle de vie.

Ils sont regroupés en trois grandes vues :

* **Diagrammes structurels ou statiques**
  + Diagramme de classes : il représente les classes intervenant dans le système ;
  + Diagramme d’objets : Représente les instances de classes (objets) utilisées dans le système ;
  + Diagramme de composants : Montrer les composants du système d’un point de vue physique ;
  + Diagramme de déploiement : Il sert à représenter les éléments matériels

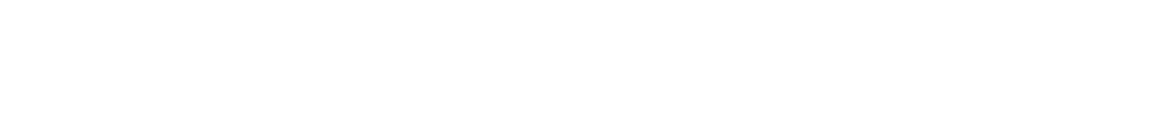
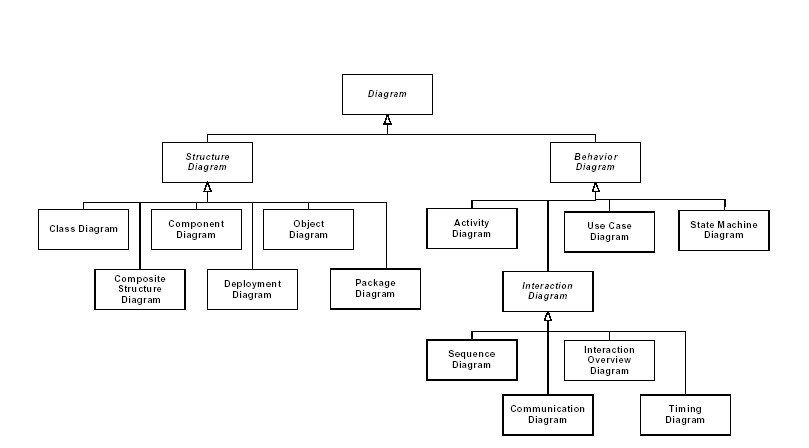
(ordinateurs, périphériques, réseaux, système de stockage …) et la manière dont les composants du système sont répartis sur ces éléments matériels et interagissent entre eux.

* + Diagramme de paquetages : ce diagramme sert à représenter les dépendances entre les différents packages utilisés dans le système ;
  + Diagramme de structure composite : Depuis UML 2.x permet de décrire les relations entre composants d’une classe ;
  + Diagramme de profils : Depuis UML 2.2 permet de spécialiser, de personnaliser pour un domaine particulier un métamodèle de référence d’UML.

* **Diagrammes comportementaux**
  + Diagramme des cas d’utilisation : Il permet d’identifier toutes les fonctionnalités que doit fournir le système ;
  + Diagramme état-transition : Permet de décrire sous forme de machine à états finis le comportement du système ou de ses composants ;
  + Diagramme d’activité : Il décrit sous forme de flux ou d’enchainement d’activités le comportement du système ou de ses composants.
* **Diagramme d’interaction ou dynamiques**
  + Diagramme de séquence : il représente séquentiellement le déroulement des traitements et des interactions entre les éléments du système et/ou de ses acteurs ;
  + Diagramme de communication : Disponible depuis UML 2.2, c’est une représentation simplifiée d’un diagramme de séquence qui se concentre sur les échanges de messages entre objet ;
  + Diagramme global d’interaction : Depuis UML 2.x, ce diagramme permet de décrire les enchainements possibles entre les scénarios préalablement identifiés sous forme de diagrammes de séquences.

Diagramme de temps : Depuis UML 2.3, permet de décrire les variations d’une donnée à cour du temps.

Nous pouvons illustrer la hiérarchie des diagrammes d’UML comme suit :



*:*

Source

:

https://fr.wikipedia.org/wiki/Fichier:Uml\_hierarchie\_des\_diagrammes.png

Dans le cadre de ce projet, nous travaillerons avec les spécifications de la version 1.4 sortie en 2003 qui compte 9 diagrammes repartis sur 2 vues.

* **Les vues statiques :**
  + Diagrammes de cas d’utilisation ;
  + Diagrammes d’objets ;
  + Diagrammes de classes ;
  + Diagrammes de composants ;

▪ Diagrammes de déploiement.

* **Les vues dynamiques :**
  + Diagrammes de collaboration ;
  + Diagrammes de séquence ;
  + Diagrammes d’état-transitions ;
  + Diagrammes d’activité.

Dans le cadre de ce projet, nous travaillerons avec les spécifications de la version 2.3 qui compte 14 diagrammes repartis sur 03 vues.

L’utilisation d’UML dans le cadre d’un projet nécessite l’association de ce langage à un processus. En effet, malgré ses nombreux atouts, UML ne propose pas de démarche à suivre dans l’élaboration d’un projet, de la compréhension des besoins utilisateurs à la production du logiciel ; UML reste une norme. Pour cette phase d’analyse, nous allons donc joindre à UML un processus unifié de traitement.

1. L’Etude comparative entre les méthodes

Il existe plusieurs processus pour accompagner le langage UML et chacun de ces processus à ses propres spécificités. Le tableau ci-dessous présente quelques-uns des processus unifiés utilisant UML avec leurs spécificités.

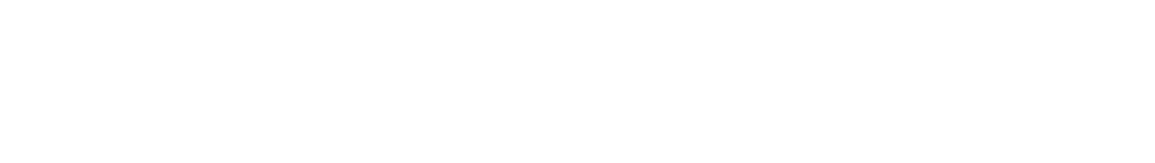
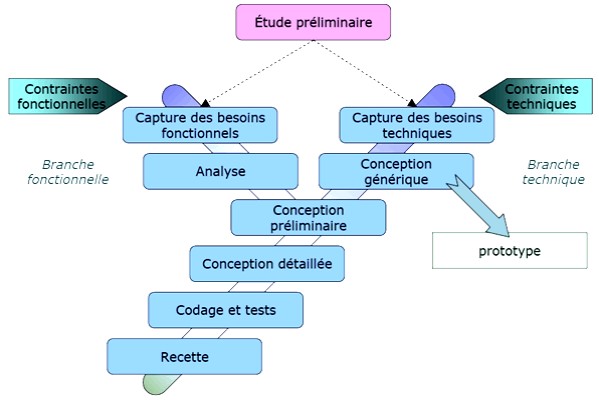
|  |  |
| --- | --- |
| Processus | Description |
| AUP | Agile Unified Process (AUP) est une variante simplifiée d’UP qui intègre des pratiques agiles comme le développement piloté par les tests (TDD). Elle reprend ainsi les phases séquentielles, et les jalons de fin de phase du processus unifié, mais renforce l’approche agile en préconisant une mise à jour du plan en fin de chaque phase, la mise en œuvre d’une version potentiellement publiable à la fin de chaque itération, et l’application des principes du manifeste agile. |
| EssUP | Essential Unified Process (EssUP) cherche à alléger le processus unifié qui bien qu’itératif et incrémental est souvent perçu comme trop lourd et trop formalisé. EssUP adopte pour cela le principe de la séparation des préoccupations. Plutôt qu’un processus monolithique. EssUP défini un ensemble de pratiques indépendantes qui peuvent être librement combinées pour former un processus adapté au contexte. |
| 2TUP | 2TUP est un processus unifié qui a pour but d’apporter une réponse aux contraintes de changement fonctionnelles et techniques qui s’imposent aux systèmes d’information, il propose un cycle de développement qui dissocie les aspects techniques des aspects fonctionnels. |
| RUP | Dérivé d’UP et commercialisé par IBM en 1998, RUP est l’une des plus célèbres implémentations de la méthode UP permettant de donner un cadre au développement logiciel. RUP est un exemple de formalisation d’un processus fondé sur le cycle de vie en spirale. |

1. Choix de la méthode

En ce qui concerne la méthode que nous allons utiliser il s’agit de celle directement rattachée au langage de modélisation UML à savoir la méthode **2TUP** qui signifie **2 Track**

**Unified Process.** C’est un cycle de développement en Y qui dissocie les aspects techniques des aspects fonctionnels qui correspondent aux deux chemins utilisés pour le développement des systèmes informatiques. Tout processus UP (Unified Process) unifié répond aux caractéristiques suivantes : il est itératif et incrémental. Ainsi donc les avantages que nous présente la méthode 2TUP sont :

* Elle permet de rendre les systèmes extensibles ;
* Elle permet l’ajout des modules ;
* Elle permet la flexibilité dans l’analyse d’un projet ;
* Elle est facile à la conception et la compréhension ;
* Elle permet d’avoir une vue préalable de comment se présentera l’application.



Source

:

https://www.memoireonline.com/05/13/7195/m\_Mise

-

en

-

place

-

dune

-

application

-

webmapping

-

de

-

geolocalisation

-

des

-

points

-

dintert

-

d

e

-

la

-

vill6.html

Décrivons la forme Y de la méthode 2TUP :

* **La brache gauche (fonctionnelle)** comporte :
  + **La capture des besoins fonctionnels :** qui produit un modelé des besoins focalisé sur le métier des utilisateurs. Elle qualifie au plus tôt le risque de produire un système inadapté aux utilisateurs. De son côté, la maitrise d’œuvre consolide les spécifications et en vérifie la cohérence et l’exhaustivité ;
  + **L’analyse :** qui consiste à étudier précisément la spécification fonctionnelle de manière à obtenir une idée de ce que va réaliser le système en termes de métier.

Les résultats de l’analyse ne dépendent d’aucune technologie particulière.

* **La branche droite (architecture technique) comporte :**
  + **La capture des besoins techniques :** qui recense toutes les contraintes et les choix dimensionnant la conception du système. Les outils et les matériels sélectionnés ainsi que la prise en compte de contraintes d’intégration avec l’existant conditionnent généralement des prérequis d’architecture technique ;
  + **Les architectures matérielles et logicielles :** constituant de la conception générique, qui définit ensuite les composants nécessaires à la construction de l’architecture technique. Cette conception est complètement indépendante des aspects fonctionnels. Elle a pour objectif d’uniformiser et de réutiliser les mêmes mécanismes pour tout un système. L’architecture technique construit le squelette du système informatique et écarte la plupart des risques de niveau technique. L’importance de sa réussite est telle qu’il est conseillé de réaliser un prototype pour assurer sa validité.
* **La branche du milieu (conception) comporte :**
  + **La conception préliminaire :** qui représente une étape délicate, car elle intègre le modèle d’analyse dans l’architecture technique de manière à tracer la cartographie des composants du système à développer ;
  + **La conception détaillée :** qui étudie ensuite comment réaliser chaque composant, l’étape de codage, qui produit ces composants et teste au fur et à mesure les unités de code réalisées ; l’étape de recette, qui consiste enfin à valider les fonctions du système développé ;
  + **Codage, tests et recentrage :** qui renvoie à la phase de programmation des fonctionnalités au fur et à mesure ; et aussi, à la validation des fonctions du système développé.

1. Justification du choix de la méthode

Nous avons porté notre choix sur le langage de modélisation UML associé au processus de développement logiciel 2TUP grâce à certains critères :

* UML est basé sur l’approche objet ;
* UML est un langage centré sur les besoins des utilisateurs ;
* UML associe les données et les traitements ;
* 2TUP facilite la modélisation des systèmes complexes destinés à subir plusieurs évolutions ;
* 2TUP est un processus basé sur l’approche objet et est construit sur UML ;
* 2TUP fait partie de la famille des processus unifiés, par conséquent il possède toutes les caractéristiques de ces derniers (incrémental, itératif, piloté par les risques, orienté composant, orienté utilisateur) ;
* 2TUP offre un cycle de développement en Y qui dissocie les aspects techniques des aspects fonctionnels.

1. CAPTURE DES BESOINS FONCTIONNELS

Dans le cadre de la capture des besoins fonctionnels de notre système, nous établirons un ensemble de diagramme UML. En utilisant la version 2.3 d’UML, nous présenterons les diagrammes de cas d’utilisation (USE CASE), de séquence et de communication.

1. Diagramme de cas d’utilisation

Le diagramme de cas d’utilisation permet de structurer les besoins des utilisateurs et les objectifs correspondants d’un système. Les objectifs principaux des diagrammes de cas d’utilisation sont :

* Fournir une vue de haut niveau de ce que fait le système
* Identifier les utilisateurs du système
* Déterminer les secteurs nécessitant des interfaces d’homme-machine

1. Formalisme

Les diagrammes de cas d’utilisation sont représentés par une ellipse contenant le nom du cas d’utilisation. Un acteur et un cas d’utilisation sont mis en relation par une association présenter par une ligne.

Trois types de relation sont pris en charges par la norme UML et son graphiquement représenté par des types particulier de ses relations. Les relations indiquent que les cas d’utilisations sources présentent les mêmes conditions d’exécution que le cas d’utilisation issu. Une relation simple entre un acteur et un cas d’utilisateur est un trait simple.

|  |  |  |
| --- | --- | --- |
| Composant | Description | Représentation Graphique |
| Le système | Représenté par un cadre rectangulaire, il contient toutes les fonctionnalités de l’application. | Nom système |
| L’Acteur | Il représente un élément externe qui interagit avec le système. Cet élément peut être un utilisateur ou un tiers (autre ordinateur, autre programme, base de données). |  |
| Le cas d’utilisation | Un use case (cas d’utilisation) représente une fonctionnalité du système. Cette fonctionnalité est définie par une action déclenchant un ou plusieurs déroulements possibles. |  |
| L’association | Les acteurs et les cas d’utilisation sont reliés par des associations. Ainsi que les acteurs du système entre eux. |  |

Dans un diagramme de cas d’utilisation, il existe plusieurs types de relation ou dépendance entre les différents cas d’utilisation.

|  |  |  |
| --- | --- | --- |
| Relations | Description | Représentation graphique |
| L’inclusion | Un cas d’utilisation A inclus un cas d’utilisation B signifie que pour effectuer le cas d’utilisation A, il faut obligatoirement effectuer le cas d’utilisation B. |  |
| L’extension | Un cas d’utilisation B étend un cas d’utilisation A si pendant que l’on effectue le cas A, on a la possibilité d’effectuer B indépendamment de A. |  |
| La généralisation | La relation d’héritage permet de dire qu’un cas d’utilisation est une généralisation d’autres cas d’utilisation qui sont des spécialisations. Elle est représentée par une ligne donc l’extrémité est en forme de triangle. |  |

Comme l’illustre le tableau précédent, entre cas d’utilisation il existe trois types de relation. Entre les acteurs du use case, il peut exister une unique relation : la généralisation. Celle-ci est utilisée lorsqu’ un acteur peut effectuer les mêmes cas d’utilisation qu’un autre.

1. **Exemple de diagramme de cas d’utilisation

Liste des acteurs du système

* Administrateur
* Médecin
* Patient
* Visiteur

Pour une meilleure compréhension du diagramme, celui-ci va etre subdivisé en sous diagramme en fonction des acteurs du système.

|  |  |
| --- | --- |
| Acteur | Cas d’utilisation |
| Administrateur | **S’authentifier**  **Gérer les utilisateurs**   * Enregistrer les médecins * Supprimer les utilisateurs * supprimer une002 annonce |
| Médecin | **S’authentifier**  **Gérer les rendez-vous**   * Valider un rendez-vous * Rejeter un rendez-vous * Modifier un rendez-vous   **Gérer compte**   * Supprimer compte * Modifier mot de passe * Changer de photo   Suivre un patient  Faire une annonce |
| Patient | **S’authentifier**  **Gérer les rendez-vous**   * Prendre un rendez-vous * Annuler un rendez-vous * Modifier un rendez-vous   **Gérer compte**   * Supprimer compte * Modifier mot de passe * Changer de photo   Noté un médecin  Payer une consultation |
| Visiteur | Visiter l’accueil  Créer un compte |
| Fournisseur de service de paiement | Traiter les paiements  Effectuer les transactions |

* Diagramme de cas d’utilisation global



* Cas d’utilisation spécifique « cas de l’administrateur »



* Cas d’utilisation spécifique « cas du médecin »



* Cas d’utilisation spécifique « cas du patient »



1. Description textuelle de quelque cas d’utilisation

Un cas d’utilisation décrit un ensemble de scénario. Un scénario décrit une exécution particulière d’un cas d’utilisation du début à la fin. Il correspond à une sélection d’enchainement du cas d’utilisation.

On peut distinguer plusieurs types de scénarios

* Nominaux : ils réalisent les post conditions du cas d’utilisation, d’une façon naturelle et fréquente ;
* Alternatifs: ils remplissent les post conditions du cas d’utilisation, mais en empruntant des voies détournés ou rares ;
* Description : ne réalisent pas les post conditions du cas d’utilisation.

Il faut signaler que tous les scénarios possibles ne peuvent etre énuméré et décrit du fait qu’ils en existent beaucoup. C’est pour cette raison que nous allons faire une description des scénarios les plus pertinents.

Chaque cas d’utilisation est associé à une série d’action représentant la fonctionnalité voulue ainsi que les stratégies à utiliser dans l’alternative ou la validation échoue ou les erreurs se reproduisent. Un cas d’utilisation est donc une abstraction de plusieurs chemins d’exécution. Une instance d’un cas d’utilisation est un scénario. C’est un chemin particulier pris lors de l’exécution d’un cas d’utilisation.

* Description textuelle du cas d’utilisation s’authentifier

|  |
| --- |
| **Titre** : s’authentifier |
| **Acteurs** : utilisateurs |
| **Présupposé** : le système fonctionne correctement |
| **Précondition** : le site doit etre lancée et l’utilisateur doit avoir un compte |
| **Objectif :** permettreà l’utilisateur d’accéder à son interface |
| **Scénario Nominal :**   1. L’utilisateur clique sur le bouton se connecter 2. Le système affiche le formulaire de connexion à l’utilisateur 3. L’utilisateur rempli et soumet le formulaire 4. Le système vérifie si les champs du formulaire sont bien remplis 5. Le système envoie une requête d’authentification à la base de données 6. Une vérification est faite dans la base de données et une réponse est renvoyée au système 7. Une session de travail est créé et l’utilisateur es redirigé sur sa page |
| **Scénario Alternatif :**  4-a A l’étape 4, si le formulaire est mal rempli, le système notifie l’utilisateur et on retourne à l’étape 2.  6-a A l’étape 6, si l’utilisateur n’est pas trouvé dans la base de données, le système notifie l’utilisateur et on revient à l’étape 2 |
| **Post condition échec** : l’utilisateur n’accède pas à l’espace utilisateur |

* Description textuelle de cas d’utilisation créer compte

|  |
| --- |
| **Titre** : s’inscrire |
| **Acteur : visiteurs** |
| **objectif** : permettre à un visiteur de créer un compte |
| **Présupposé** : le système fonctionne correctement |
| **Scénario nominal :**   1. L’utilisateur clique sur le bouton s’inscrire 2. Le système envoie le formulaire d’inscription 3. Le patient rempli le formulaire et le soumet au système 4. Le système vérifie les champs du formulaire 5. Le système soumet les informations dans la base de données 6. Le SGBD enregistre les informations reçues par le système 7. Un message de validation de création du compte est envoyé au système 8. Le patient est redirigé vers la page d’utilisateur |
| **Scénario alternatif :**  5-a A l’étape 5 si le formulaire est mal rempli le système notifie le visiteur et on le renvoie à l’étape 4 |
| **Post condition succès** : le compte du patient est crée |
| **Post condition échec** : le compte du patient n’a pas été crée |

* Description textuelle du cas d’utilisation cas de la consultation

|  |
| --- |
| Titre : se consulter |
| Objectif : permettreau malade d’effectuer une consultation |
| Acteur : patient |
| Présupposé : le système fonctionne correctement |
| **Scénario nominal :**   1. L’utilisateur clique sur le bouton se consulter 2. Le système affiche le formulaire de consultation 3. L’utilisateur rempli et soumet le formulaire 4. Le système vérifie si les champs du formulaire sont bien remplis 5. Le système stocke les données du patient dans la base de données 6. Le système affiche un formulaire de prise de rendez-vous et le formulaire de payement 7. Le patient effectue le payement, remplie et soumet le formulaire de prise de rendez vous 8. Le système vérifie le formulaire de prise de rendez vous 9. Le rendez-vous est stocké dans la base de données 10. Une vérification est faite dans la base de données s’il y’a pas une redondance de date. 11. Le système notifie le patient de la prise en compte de son formulaire et d’un retour du médecin dans de bref délai. |
| **Scénario alternatif** :  4-a A l’étape 4, si le formulaire est mal rempli, le système notifie l’utilisateur et on retourne à l’étape 2.  8-a A l’étape 8 si le formulaire est mal rempli le système notifie le patient et on retourne à l’étape 7-b  10a A l’étape 10 s’il y’a redondance de date, le système notifie le patient et on retourne à l’étape 7-b |
| **Post condition succès :**  Le patient parvient à envoyer sa demande de consultation et obtient un rendez-vous |
| **Post condition échec :**  le patient ne parvient pas à envoyer sa consultation et n’obtient pas de rendez-vous. |

* Description textuelle du cas d’utilisation payer une consultation

|  |
| --- |
| **Titre** : payer une consultation |
| **objectif** : permet au patient de payer la consultation |
| **Acteur** : patient |
| **Présupposé :** le système fonctionne normalement |
| **Scénario nominal :**   1. L’utilisateur clique sur se consulter 2. Le système envoie les différents modes de payement 3. L’utilisateur choisi le mode de payement 4. L’utilisateur confirme sa transaction 5. Le fournisseur de service de paiement traite la transaction et confirme la réussite du payement 6. Le patient accède à la page de consultation |
| **Scénario alternatif :**  A l’étape 4 si la confirmation n’est pas faite le système renvoie le patient à l’étape 3 |
| **Post condition succès :** le patient peut effectuer une consultation ou converse avec un médecin |
| **Post condition échec :** le patient ne pourra pas bénéficier d’une consultation |

1. Diagramme de séquence

Les diagrammes de séquences sont des représentations graphiques des interactions entre les acteurs et le système selon un ordre chronologique dans la formulation UML. Le diagramme de séquence permet de d’écrire les interactions entre les objets au sein d’un diagramme de cas d’utilisation.

1. Formalisme

Le formalisme de séquence énumère les objets horizontalement et le temps verticalement ; il modélise l’exécutable des différents modèles en fonctions du temps. Dans ce diagramme les objets et les acteurs sont énumérée en colonnes avec leur ligne de vie verticale indiquant la durée de vie de l’objet.

|  |  |  |
| --- | --- | --- |
| Composant | Description | Représentation |
| Objet | Les objets sont les instances de classe et sont rangés horizontalement. |  |
| Acteur | Personne qui interagit ou communique avec le système ou les objets. |  |
| Ligne de vie | La ligne de vie identifie l’existence de l’objet par rapport au temps. |  |
| Activation | Les activations sont modélisées par des boites rectangulaires sur la ligne de vie. Ils permettent d’indiquer que l’objet effectue une action |  |
| Message synchrone | C’est un message qui lorsqu’il est envoyé bloque l’activité de l’expéditeur jusqu’au signal de prise en compte ou à la réponse du destinataire. Il est symbolisé par une flèche pleine. |  |
| Message asynchrone | Lorsqu’un message est asynchrone, l’expéditeur continue son activité que le message soit parvenue et pris en compte ou non par le destinataire. |  |
| Message réflexif | Représente une activité interne à l’objet. |  |
| Message de retour | C’est le message représentant la réponse à un message synchrone. |  |
| L’opérateur alt | Opérateur conditionnel, il permet de spécifier les différentes alternatives dans les zones délimitées par les pointillés. |  |

1. Exemple de diagramme de séquence
2. Quelque diagramme de séquence du système étudié

* Diagramme de séquence s’authentifier



* Diagramme de séquence créer un compte



* Diagramme de séquence se faire consulter



* Diagramme de séquence payer une consultation

1. Diagramme de communication

Le diagramme de communication est un diagramme d’interaction. C’est la représentation simplifiée du diagramme de séquence se concentrant sur les échanges des messages entre les objets.

1. Composant du diagramme de communication

|  |  |  |
| --- | --- | --- |
| Composant | Description | Représentation graphique |
| Objet | Symbolisé par un rectangle, l’objet représente un élément communiquant avec d’autres objets et exécutant un ordre dans le système. |  |
| Lien | Il permet de montrer la relation entre les objets au sein du système. Cet élément permet de diriger le flux de message. |  |
| message | Représenté par une flèche indiquant le sens du message. Les nombres présents donnent l’ordre dans lequel les messages sont envoyés. |  |

1. Exemple de diagramme de communication
2. Diagramme de communication créer un compte pour s’inscrire



1. Diagramme d’activité

Les diagrammes d’activité sont particulièrement adaptés à la description des cas d’utilisation. Plus précisément, ils viennent illustrer et consolider la description textuelle des cas d’utilisation

1. Formalisme

L’usage général des diagrammes d’activité permet de faire apparaitre les flots de traitement induit par le processus interne par rapport aux évènements externes. Les constitutifs d’un diagramme d’activité sont :

|  |  |  |
| --- | --- | --- |
| Formalisme | Description | Représentation graphique |
| L’activité | L’état d’activité marque une action faite par un objet. Il est représenté par un rectangle aux coins arrondis. |  |
| La transition | Quand un état d’activité est accompli, le traitement passe à un autre état d’activité. Les transitions (modélisées par des flèches.) sont utilisées pour marquer ce passage. |  |
| Les couloirs | Le diagramme d’activité fait intervenir les acteurs de chaque activité. Chaque activité sera placée dans une colonne (couloir) qui correspond à l’acteur. Les objets sont énumérés au-dessus de la colonne, et les barres verticales séparent les colonnes pour former les couloirs d’activités. |  |
| L’état initial | Il marque le point d’entrée la première activité. Il est représenté come dans le diagramme d’état par un cercle plein |  |
| Etat final de fin d’activité | Elément symbolisant l’étape finale de l’activité. Il représente l’achèvement de tous les flux d’un procédé. On le représente par un cercle noir avec un contour. |  |
| L’état final de fin de flux | Il représente la fin d’un schéma de procédé spécifique. |  |

1. Exemple de diagramme d’activité



1. Quelque diagramme d’activités du système étudié

* Diagramme d’activité d’authentification



* Diagramme d’activité d’inscription



CONCLUSION

Le dossier d’analyse à jouer un rôle crucial en nous fournissant une vision détaillée du système à concevoir. En utilisant le langage de modélisation et le processus associé 2TUP, nous allons maintenant passer au dossier de conception. Cette étape consistera à représenter les parties du système et leurs interactions. Grace à ces diagrammes, nous serons en mesure de visualiser et de communiquer de manière claire et précise l’architecture et la structure du système que nous allons développer.

INTRODUCTION

1. OBJECTIF DE LA CONCEPTION
2. CAPTURE DES BESOINS TECHNIQUES (PRESENTATION DES DIAGRAMMES INTERVENANTS)

CONCLUSION

La phase de conception d’un projet vise à décrire les exigences du système. Son objectif est de modéliser la solution à concevoir et d’identifier les différents éléments nécessaires pour la mise en place de la base de données. Cette étape permet de définir de manière détaillée l’architecture et la structure du système, en prenant en compte les besoins fonctionnels et non fonctionnels. Elle joue un rôle essentiel dans la planification et la préparation de la mise en œuvre du système, en fournissant les bases solides pour le développement ultérieur.

**PLAN**

DOSSIER IV : DOSSIER DE CONCEPTION

**INTRODUCTION**

La phase d’analyse précédente a permis d’identifier les besoins fonctionnels du système. Dans la phase de conception qui suit, nous allons nous concentrer sur les aspects statiques du système. Nous examinerons également les différentes données à manipuler afin de déterminer la structure de la future base de données. Cette étape nous permettra de définir de manière détaillée les entités, les relations et les attributs qui seront nécessaires pour la gestion efficace des données dans le système. En analysant ces aspects, nous serons en mesure de concevoir une architecture pour la base de données, alignée sur les besoins identifiés lors de l’analyse, afin d’assurer un fonctionnement optimal du système.

1. **OBJECTIF DE LA CONCEPTION**

La conception en UML permet entre autres de représenter de façon simple tous les éléments nécessaires au bon fonctionnement de l’application et d’établir les différentes tables de données et ressortir la base de données.

1. **CAPTURE DES BESOINS TECHNIQUES**
2. **Diagramme de classe**

Un diagramme de classe permet de présenter les différents objets d’un système sous forme de classe. Il permet de mettre en exergue les différentes collaborations, et relations entre les différentes classes. C’est un diagramme statique qui fournit une vue globale du système.

1. **Formalisme du diagramme de classe**

|  |  |  |
| --- | --- | --- |
| 1. ELEMENT | DESCRIPTION | Représentation |
| La classe | Une classe est un type abstrait caractérisé par des propriétés (attributs et méthodes) communes à un ensemble d’objets et permettant de créer des objets ayant ces propriétés. |  |
| Les attributs | Un attribut représente la modélisation d’une information élémentaire représentée par son nom et son format. | **nomAttribut :type** |
| L’identifiant de la classe | L’identifiant est un attribut particulier, qui permet de repérer de façon unique la classe. | **nomAttribut :type** |
| Méthode (ou opérations) | Une opération est une fonctionnalité assurée par une classe. La description des opérations peut préciser les paramètres d’entrée et de sortie ainsi que les actions élémentaires à exécuter. | **NomMéthode() :typeRetour** |
| TYPES DE RELATION | |  |
| Association | Elle permet de représenter une relation structurelle entre deux ou plusieurs classes. |  |
| Agrégation | C’est une relation entre deux classes spécifiant que les objets d’une classe sont des composants de l’autre classe. |  |
| Composition | La composition est une agrégation forte. Ici la vie des composants est liée à celle de l’agrégat. |  |
| Généralisation / spécialisation | Relations dans laquelle des classes appelées classes filles héritent des attributs et méthodes d’une autre classe appelé classe mère. |  |

1. **Exemple de diagramme de classe**
2. **Diagramme d’état transition**

Le diagramme d’état transition permet de présenter les différents états qu’un objet peut prendre durant son cycle de vie.

1. **Formalisme du diagramme d’état transition**

|  |  |  |
| --- | --- | --- |
| Composants | Description | Représentation |
| Etat initial | Représenté par un cercle plein, il marque l’état d’un objet avant toute transition. |  |
| Etat | L’état est une situation durable dans laquelle se trouve un objet. |  |
| Transition | Il permet de modélisé le changement d’état d’un objet provoqué généralement par un évènement. Elle est symbolisée par une flèche orientée. |  |
| Eta final | Il permet de représenter la destruction d’un objet |  |

1. **Exemple de diagramme d’état transition**
2. **Diagramme de package**

Les diagrammes de package (ou diagramme de paquetages) permet de décomposer le système en catégories ou partie plus facilement observable appelés « packages ». L’utilisation la plus courante pour les diagrammes de classes. Bien que l’utilisation des Diagrammes de paquetages ne se limite pas a ces éléments UML.

1. **Formalisme**

|  |  |  |
| --- | --- | --- |
| **COMPOSANT** | **DESCRIPTION** | **REPRESENTATION** |
| Package | Regroupement des cas d’utilisation catégorie |  |
| **Relation (dépendance)** |  |  |

1. **Exemple de diagramme de package**

* **Diagramme de package du système**

**CONCLUSION**

Ledossier de conception a été essentiel pour identifier les données nécessaires à la création de notre base de données et à l’implémentation de notre application. Les divers éléments modélisés dans cette phase nous ont permis d’avoir une vision globale des différents modules de notre application. A présent, l’étape suivante consistera à rédiger le dossier de réalisation en tenant compte des éléments modélisés précédemment. Ce dossier de réalisation servira de guide pour concrétiser les concepts et les spécifications définis lors de la phase de conception, en détaillant les étapes à suivre pour la mise en œuvre effective du projet.

INTRODUCTION

1. PRESENTATION DES OUTILS DE DEVELOPPEMENT
2. ARCHITECTURE DE LA PLATEFORME
3. DIAGRAMME INTERVENANTS

CONCLUSION

Cette partie du projet consiste à mettre en œuvre, dans un langage de programmation les spécifications définies lors des phases précédentes. Elle inclut également la création des diagrammes de déploiement et de composants, qui mettent en évidence l’architecture de notre application. Ces diagrammes permettent de visualiser comment les différents composants logiciels seront déployés sur l’infrastructure cibles. En utilisant le langage de programmation choisi, nous allons développer les fonctionnalités de l’application en s’appuyant sur ces diagrammes pour garantir une architecture cohérente et efficace. Cette étape est cruciale pour concrétiser le projet et créer un produit final prêt à etre déployé dans un environnement opérationnel.

**PLAN**

DOSSIER V : DOSSIER DE REALISATION

INTRODUCTION

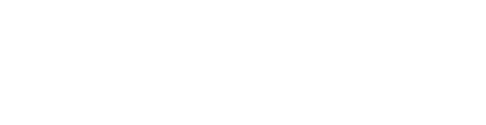
Le dossier de réalisation est celui qui contient toutes les informations relatives à l’implémentation proprement dite d’un projet. Ce dernier a pour objectif de présenter les éléments intervenants dans le développement de notre plateforme, ceci en présentant l’environnement de travail et de développement utilisé ainsi que les différents logiciels qui entreront en jeux. Dans cette partie, nous allons tour à tour parler des outils de développement que nous avons utilisé, du langage de programmation que nous avons choisi et la présentation de l’architecture physique de déploiement de la plateforme.

1. PRESENTATION DES OUTILS DE DEVELOPPEMENT

Pour la réalisation de notre application, nous avons utilisé les ressources citées dans le tableau suivant :

1. ARCHITECTURE DE LA PLATEFORME
2. Architecture physique

Le modèle d’architecture physique utilisé pour l’implémentation notre solution, est l’architecture N – tiers. Avec dans notre cas **N = 4** : **client / serveur d’application / couche métier / serveur de bases de données.**



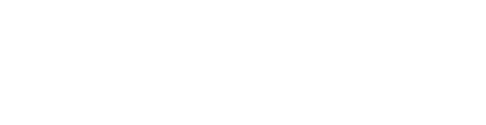
Serve

ur d

e base de

donnnn

ées



Serve

ur

d

’

application



C

lient

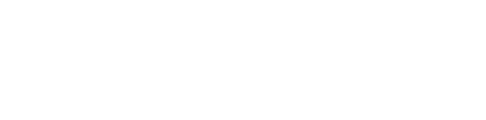
Android



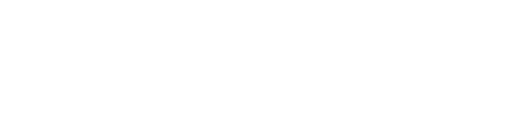
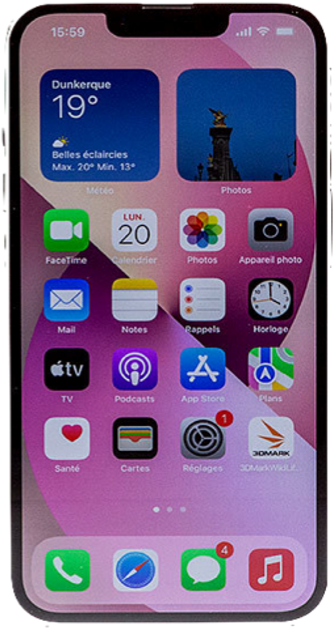
C

lient

iOS



API Google Maps

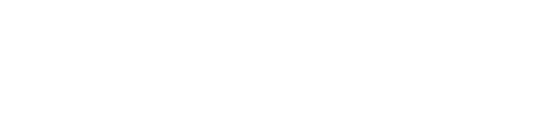


**Couche**

**d**

**’**

**application**

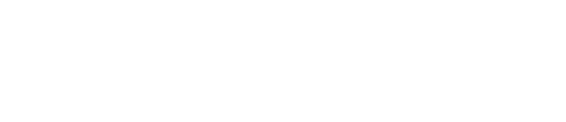


**Couche**

**m**

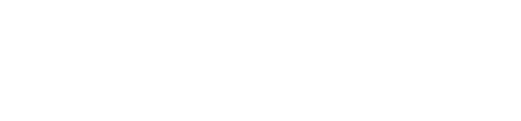
**é**

**tier**



**Couche de**

**présentation**



**Couche**

**d**

**e donn**

**ées**



**API paiement**

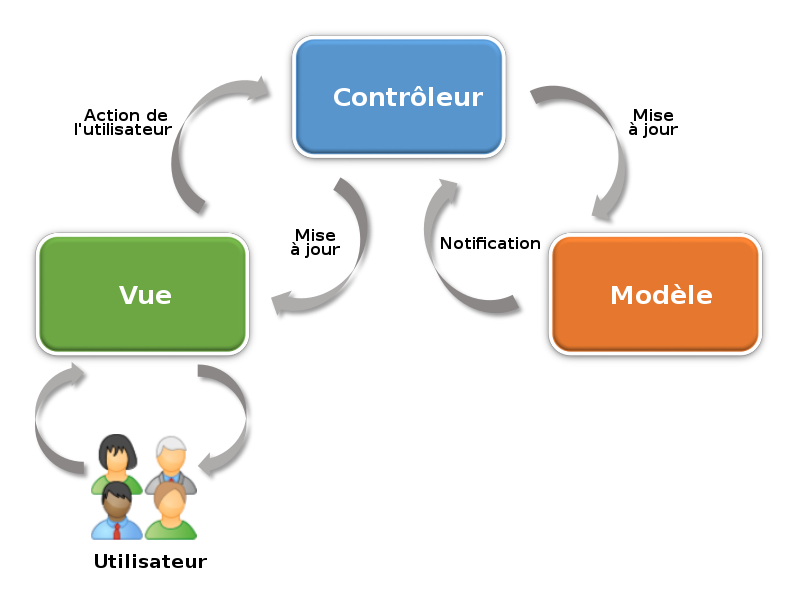
1. Architecture logique

La logique applicative de notre projet est basée sur le design pattern MVC qui est une façon très pratique d’organiser ses différentes classes dans un projet de développement d’application facilitant ainsi la détection d’erreurs, la collaboration et la réutilisabilité du code.

Le design pattern MVC (Modèle Vue Contrôleur) est une architecture et une méthode de conception qui organise l’interface homme-machine (IHM) d’une application logicielle. Ce paradigme divise l’IHM en un modèle (modèle de données), une vue (interface utilisateur) et un contrôleur (logique de contrôle, gestion des événements, synchronisation), chacun ayant un rôle précis dans l’interface.

Le design pattern MVC permet de bien organiser son code source. Il va vous aider à savoir quels fichiers créer, mais surtout à définir leurs rôles. Le but du MVC est justement de séparer la logique du code en trois parties que l’on retrouve dans des fichiers distincts.

* **Modèle :** cette partie gère les données de votre application. Son rôle est d’aller récupérer les informations « brutes » dans une base de données locale ou distante, de les organiser et de les assembler pour qu’elles puissent ensuite être traitées par le contrôleur. Ces données peuvent être stockées dans des bases de données, dans le Cloud ou dans des fichiers binaires.
* **Vue :** cette partie se concentre sur l'affichage. Elle ne fait presque aucun calcul et se contente de récupérer des variables pour savoir ce qu'elle doit afficher. Dans notre application On y trouve essentiellement du code React Native mais aussi quelques boucles et conditions JavaScript très simples.
* **Contrôleur :** cette partie gère la logique du code qui prend des décisions. C'est en quelque sorte l'intermédiaire entre le modèle et la vue : le contrôleur va demander au modèle les données, les analyser, prendre des décisions et renvoyer le texte à afficher à la vue. Le contrôleur contient exclusivement du JavaScript. C'est notamment lui qui détermine si l’utilisateur a le droit de voir la page ou non (gestion des droits d'accès).



1. DIAGRAMME INTERVENANTS
2. Diagramme de déploiement

Ce diagramme modélise les composants matériels utilisés pour implémenter un système et l’association entre ces composants.

1. Formalisme du diagramme de déploiement

|  |  |  |
| --- | --- | --- |
| 1. **Eléments** | **Description** | **Formalisme** |
| Composant | Il modélise une entité logicielle du système. |  |
| Nœud | Le nœud représente un ensemble d’éléments matériels du système. |  |
| Association | Permet de montrer le lien de communication entre les nœuds |  |
| Dépendance | Il permet de traduire un lien de dépendance entre des nœuds |  |

1. Diagramme de Composant

Le diagramme de composants permet de représenter les composants logiciels d’un système ainsi que les liens existants entre ces composants. Un composant est un élément logiciel autonome, qui fournit ou reçoit un service bien précis : fichier sources (.java, .cpp, .h…), librairies, exécutables… Les composants fournissent des services via des interfaces.

* 1. Composants du diagramme de composants

Les différents éléments constitutifs de ce diagramme sont les suivants :

|  |  |  |
| --- | --- | --- |
| **Composants** | **Description** | **Représentation** |
| Composant | Le composant représente une entité logicielle d’un système. Il est représenté par une boîte rectangulaire, avec deux rectangles dépassant du côté gauche. |  |
| Dépendance | Elle modélise la relation entre deux composants. Elle est représentée par une flèche pointillée, se dirigeant d’un composant donné au composant dont il dépend |  |

CONCLUSION

DOSSIER VI : TESTS DES FONCTIONNALITES

INTRODUCTION

1. TEST FONCTIONNELS
2. TEST NON FONCTIONNELS
3. TEST DE PERFORMANCE

CONCLUSION

Cette partie consiste à effectuer des tests partiels du système, en mettant sur les différentes approches et méthodologies utilisée pour tester les fonctionnalités de l’application. L’Objectif est de garantir que ces fonctionnalités fonctionnent correctement et sont conformes aux spécifications définies. Pendant cette phase de test, diverses techniques telles que les tests unitaires, les tests d’intégration et les tests de validation peuvent etre utilisées pour évaluer le comportement et la performance de l’application. L’objectif ultime est de détecter et de corriger les éventuels problèmes ou erreurs avant le déploiement final de l’application, afin d’assurer sa qualité et sa fiabilité.

**PLAN**

INTRODUCTION

1. TEST FONCTIONNELS
2. Tests d’interface utilisateur
3. Tests de saisie et de validation
4. Tests d’intégration
5. TESTS NON FONCTIONNELS
6. TESTS DE PERFORMANCE

CONCLUSION

DOSSIER VI I: GUIDE D’UTILISATION

Ce document est un guide d’utilisation clair et concis pour l’application. Il fournit des instructions détaillées sur les précautions à prendre avant de lancer l’application et les ressources logicielles requises pour assurer son bon fonctionnement. Ce guide vise à faciliter l’utilisation de l’application en fournissant des informations essentielles et des conseils pratiques pour les utilisateurs. Il garantit également que les utilisateurs sont bien informés des exigences techniques et des mesures à prendre pour une expérience utilisateur optimal.

**PLAN**

INTRODUCTION

1. GUIDE D’INSTALATION
2. GUIDE D’UTILISATION

CONCLUSION

INTRODUCTION

1. GUIDE D’INSTALATION
2. La langue
3. Ressources minimal requises
4. Matériel requis
5. Débit internet
6. GUIDES D’INSTALATION ET DES FONCTIONNALITES DEJA DEVELOPEES
7. Page d’accueil principal
8. Inscription du contribuable

CONCLUSION

CONCLUSION GENERAL

BIBLIOGRAPHIE

WEBBOGRAPHIE

ANNEXES

TABLES DE MATIERE