**Université Chouaib DOUKKALI**

**Faculté des Sciences El Jadida**

**Département d’Informatique**

Projet de fin d’etudes

Pour l’obtention du diplôme Licence en Sciences Mathématiques et Informatique (SMI)

Promotion Juin 2024

Application web d’assistance médicale d’urgences

**Réalisé Par : EL KHALFI Salma**

**Encadré par : Mr. Le professeur : DAHMOUNI Abdellatif**

**Examiné par :**

**Mr. . Le professeur : DAHMOUNI Abdellatif**

**Mr. . Le professeur : SILKAN Hassan**

**Mr. . Le professeur : EL MOHADAB Mohamed**

**Année Universitaire: 2024-2023**

Résumé

Dans le domaine de la santé publique, le service d’urgence joue un rôle critique dans la gestion des soins assurés par les autres services des hôpitaux provinciaux ou régionaux. Certes, ce service souffre de défis liés à la qualification des priorités, la limitation des outils, la surcharge des locaux et la variabilité des délais de traitement. Notre projet de fin d’étude vise à développer une application web pour l'assistance médicale d'urgence, offrant une réponse rapide et efficace aux besoins médicaux critiques. Cette application est conçue pour simplifier d’une part la coordination entre les différents intervenants médicaux ; d’autre part la communication entre le service d’urgence, les victimes et leurs accompagnants. Ceci dans le but d'optimiser les délais d'intervention et d'améliorer la qualité des soins dispensés. Les fonctionnalités clés de notre application comprennent : un système de déclaration des victimes, une catégorisation des blessures selon leur gravité, une interface utilisateur intuitive et des informations essentielles accessibles en temps réel. Les intervenants médicaux peuvent ainsi accéder rapidement aux détails des victimes et prendre les mesures nécessaires, tandis que les victimes ou leurs accompagnants peuvent fournir des informations cruciales pour une intervention rapide et appropriée.

Remerciement

Je tiens à exprimer ma profonde gratitude à toutes les personnes qui m'ont soutenu dans la réalisation de ce projet d'assistance médicale d'urgence.

Je remercie tout particulièrement mon encadrant Mr. DAHMOUNI Abdellatif pour son soutien constant, ses conseils avisés et son accompagnement tout au long de ce projet. Sa disponibilité et son expertise ont été d'une aide précieuse pour surmonter les défis et avancer dans la bonne direction.

Je souhaite également remercier le docteur Rajaa pour ses conseils précieux et ses clarifications détaillées sur le phénomène des urgences médicales. Ses explications et son expertise ont grandement enrichi ma compréhension du sujet et ont contribué de manière significative à la qualité de ce projet.

Je remercie aussi Mrs. les professeurs SILKAN Hassan, EL MOHADAB Mohamed d’avoir examiner ce travail de PFE.

Je tiens également à remercier mes proches pour leur soutien inconditionnel et leur encouragement tout au long de ce parcours. Leur présence et leur soutien moral ont été une source de motivation constante.

Enfin, je suis reconnaissante envers moi-même pour ma détermination, ma persévérance et mon engagement à réaliser ce projet. C'est grâce à ma volonté et à mon travail acharné que j'ai pu atteindre mes objectifs et concrétiser cette vision.

Merci à toutes les personnes qui ont contribué, de près ou de loin, à la réalisation de ce projet. Votre soutien et votre confiance ont été les moteurs de ma réussite.

Tables de matières

[Résumé 1](#_Toc170461961)

[Remerciement 2](#_Toc170461962)

[Tables de matières 3](#_Toc170461963)

[Table des figures 6](#_Toc170461964)

[Introduction générale 7](#_Toc170461965)

[**Chapitre 1 : Le Web au service de la médecine** 9](#_Toc170461966)

[**I.** **Introduction :** 9](#_Toc170461967)

[**II.** **Applications web** 9](#_Toc170461968)

[**2.1.** **Les types d’applications web** 10](#_Toc170461971)

[**2.1.1.** **Les applications web statiques** 10](#_Toc170461972)

[ **Avantages** 10](#_Toc170461973)

[ **Inconvénients** 10](#_Toc170461974)

[**2.1.2.** **Les applications web dynamiques** 11](#_Toc170461975)

[ **Avantages du Web Dynamique** 12](#_Toc170461976)

[ **Inconvénients du Web Dynamique** 12](#_Toc170461977)

[**2.1.3.** **Les applications e-commerce** 13](#_Toc170461978)

[**2.1.4.** **L’application de portail web** 13](#_Toc170461979)

[**2.1.5.** **Une application web animée** 13](#_Toc170461980)

[**2.1.6.** **Les systèmes de gestion de contenu** 14](#_Toc170461981)

[**2.2.** **Les architectures des applications** 14](#_Toc170461982)

[**2.2.1.** **Architecture monolithique** 14](#_Toc170461983)

[**2.2.2.** **Architecture basée sur les microservices** 15](#_Toc170461984)

[**2.2.3.** **Architecture client-serveur** 16](#_Toc170461985)

[**2.2.4.** **Architecture Serverless** 17](#_Toc170461986)

[**III.** **Médecine et télémédecine** 17](#_Toc170461987)

[**3.1.** **Introduction à la Médecine** 17](#_Toc170461989)

[**3.2.** **Télémédecine** 18](#_Toc170461990)

[**IV.** **Problématique et domaine d’étude** 19](#_Toc170461991)

[**4.1.** **Contexte et Problématique** 19](#_Toc170461993)

[**4.2.** **Objectifs du Projet** 20](#_Toc170461994)

[**4.3.** **Étude des besoins et des exigences** 20](#_Toc170461995)

[**4.4.** **Analyse des utilisateurs et des parties prenantes** 21](#_Toc170461996)

[**V.** **Positionnement du Projet** 21](#_Toc170461997)

[**5.1.** **Étude de l'État de l'Art** 21](#_Toc170461998)

[**5.1.1.** **Applications de Gestion des Urgences :** 21](#_Toc170462001)

[**5.1.2.** **Applications de Localisation des Centres Médicaux :** 22](#_Toc170462002)

[**5.1.3.** **Applications de Communication en Temps Réel** : 22](#_Toc170462003)

[**5.1.4.** **Intégration des Technologies de l'Information et de la Communication :** 22](#_Toc170462004)

[**5.2.** **Cahier de charges** 23](#_Toc170462005)

[**VI.** **Conclusion** 24](#_Toc170462006)

[**Chapitre 2 Conception UML et outils de développement** 25](#_Toc170462007)

[**I.** **Introduction** 25](#_Toc170462008)

[**II.** **Conception** 25](#_Toc170462009)

[**2.1.** **Diagramme de cas d’utilisation** 25](#_Toc170462011)

[**2.1.1.** **Les acteurs principaux** 25](#_Toc170462012)

[**2.1.2.** **Description des interactions** 26](#_Toc170462013)

[**2.2.** **Diagramme de classes** 27](#_Toc170462014)

[**2.3.** **Diagrammes des sequences** 33](#_Toc170462015)

[**2.3.1.** **Éléments Clés d'un Diagramme de Séquences** 33](#_Toc170462021)

[**2.3.2.** **Enregistrer victime** 34](#_Toc170462022)

[**2.3.3.** **Localisation des victimes** 35](#_Toc170462023)

[**2.3.4.** **Modifier les informations d’une victime** 36](#_Toc170462024)

[**2.3.5.** **Chercher fiche techniques du premier secours** 37](#_Toc170462025)

[**III.** **Outils et technologies** 38](#_Toc170462026)

[**3.1.** **Environnement de Développement** 38](#_Toc170462029)

[**3.1.1.** **Back-end (Java EE)** 38](#_Toc170462032)

[**3.1.2.** **Front-end** 40](#_Toc170462033)

[**3.1.3.** **Presentation** 42](#_Toc170462034)

[**3.2.** **Architecture du Projet (MVC)** 42](#_Toc170462035)

[**3.3.1.** **Modèle (Model)** 43](#_Toc170462038)

[**3.3.2.** **Vue (View)** 44](#_Toc170462039)

[**3.3.3.** **Contrôleur (Controller)** 44](#_Toc170462040)

[**3.3.4.** **Illustration de l'Architecture MVC** 45](#_Toc170462041)

[**IV.** **Conclusion** 45](#_Toc170462042)

[**Chapitre 3 Implémentation** 46](#_Toc170462043)

[**I.** **Introduction** 46](#_Toc170462044)

[**II.** **Présentation de l’application développée** 46](#_Toc170462045)

[**2.1.** **Interfaces Utilisateur** 46](#_Toc170462048)

[**2.1.1.** **Page d’accueil** 46](#_Toc170462049)

[**2.1.2.** **Page de connexion** 47](#_Toc170462050)

[**2.1.3.** **Page d’inscription** 48](#_Toc170462051)

[**2.1.4.** **Page saisir victime** 49](#_Toc170462052)

[**2.1.5.** **Page de témoignages** 51](#_Toc170462053)

[**2.2.** **Interfaces administrateur** 51](#_Toc170462054)

[**2.2.1.** **Page de gestion des utilisateurs** 51](#_Toc170462055)

[**2.2.2.** **Page de de gestion des victimes** 52](#_Toc170462056)

[**III.** **Test de Validation** 53](#_Toc170462057)

[**3.1.** **Types de Tests** 53](#_Toc170462059)

[**3.1.1.** **Tests Unitaires** 53](#_Toc170462060)

[**3.1.2.** **Tests d'Intégration** 54](#_Toc170462061)

[**3.1.3.** **Tests de Système** 54](#_Toc170462062)

[**IV.** **Conclusion** 55](#_Toc170462063)

[**Conclusion générale** 56](#_Toc170462064)

[**Webographie** 57](#_Toc170462065)

Table des figures

[Figure 1: web static 11](#_Toc170462066)

[Figure 2: web dynamique 13](#_Toc170462067)

[Figure 3:architecture monolithique 15](#_Toc170462068)

[Figure 4: architecture microservices 15](#_Toc170462069)

[Figure 5: différence entre monolithique et microservices 16](#_Toc170462070)

[Figure 6: Architecture client serveur 16](#_Toc170462071)

[Figure 7: architecture serverless 17](#_Toc170462072)

[Figure 8: réunion médicale à distance 18](#_Toc170462073)

[Figure 9: consultation à distance 18](#_Toc170462074)

[Figure 10: diagramme de cas d'utilisation principale 26](#_Toc170462075)

[Figure 11: diagramme de classe 32](#_Toc170462076)

[Figure 12: diagramme de séquence Authentification 34](#_Toc170462077)

[Figure 13: diagramme de séquence de enregistrer une victime 35](#_Toc170462078)

[Figure 14: Diagramme de séquence de localisation des victimes 36](#_Toc170462079)

[Figure 15: diagramme de séquence de modifier une victime 36](#_Toc170462080)

[Figure 16: Diagramme de séquence de chercher une fiche techniques 37](#_Toc170462081)

[Figure 17:Eclipse IDE 38](#_Toc170462082)

[Figure 18: Appache tomcat 39](#_Toc170462083)

[Figure 19: Hibernate 39](#_Toc170462084)

[Figure 20: XAMPP 40](#_Toc170462085)

[Figure 21: VS Code 41](#_Toc170462086)

[Figure 22: HTML 41](#_Toc170462087)

[Figure 23: CSS 42](#_Toc170462088)

[Figure 24: StarUML 42](#_Toc170462089)

[Figure 25: modèle MVC 43](#_Toc170462090)

[Figure 26: Page d'accueil utilisateur 46](#_Toc170462091)

[Figure 27: Page de connexion 48](#_Toc170462092)

[Figure 28: Page d'inscription 48](#_Toc170462093)

[Figure 29: Page de saisir victime (informations personnelles) 49](#_Toc170462094)

[Figure 30: Page de saisir victime (Etat d'urgence) 50](#_Toc170462095)

[Figure 31: Page de témoignages 51](#_Toc170462096)

[Figure 32: Page de gestion des utilisateurs 52](#_Toc170462097)

[Figure 33: Page gérer victime 53](#_Toc170462098)

Introduction générale

À l'ère numérique actuelle, la technologie occupe une place centrale dans notre quotidien, remodelant profondément nos habitudes en rendant les tâches plus accessibles et pratiques. En particulier, les applications web qui ont bouleversé notre manière d'accomplir diverses activités quotidiennes, offrant des solutions innovantes et efficaces à des besoins multiples et variés.

Dans ce contexte, notre projet se propose de répondre à une exigence croissante dans le domaine de la santé publique « Gestion des urgences médicales via une application web ». En effet, les situations d'urgence médicale exigent une intervention rapide et adéquate pour assurer la santé et la sécurité des personnes concernées. D’où, la nécessité du développement d’application web dédiée à faciliter les différentes tâches de l'assistance médicale d'urgence. Notre projet de fin d’étude s'inscrit dans ce cadre et propose une démarche de modernisation et d'optimisation des processus d'intervention médicale, en offrant une solution pratique et accessible pour la gestion des situations d'urgence à distance.

L'objectif principal de notre projet est de concevoir une application web intuitive et conviviale qui permettra aux intervenants médicaux, aux victimes et à leurs accompagnants de communiquer et de coordonner efficacement les actions en cas d'urgence. Cette application est conçue pour offrir un accès rapide aux informations cruciales, faciliter la prise de décision et optimiser les délais d'intervention, contribuant ainsi à l’amélioration des résultats cliniques des patients et à la minimisation des taux de mortalité des situations accidentelles.

Par ailleurs, notre application web intégrera des fonctionnalités avancées pour répondre aux besoins spécifiques des utilisateurs. Parmi ces fonctionnalités, on retrouve la déclaration des victimes, la catégorisation des blessures par gravité, la consultation en temps réel des informations médicales et l'identification des ressources médicales disponibles à proximité.

En outre, notre projet s'attachera à offrir une expérience utilisateur optimale en proposant une interface claire et facile à utiliser, ainsi qu'un processus de déclaration simplifié. Grâce à ces caractéristiques, notre application web permettra non seulement de gagner du temps, mais aussi d'offrir une solution fiable et efficace pour la gestion des urgences médicales.

En somme, notre projet s'inscrit dans une démarche d'innovation technologique et de réponse à des besoins concrets exprimés par la communauté médicale et les personnes confrontées à des situations d'urgence. En combinant les avancées technologiques avec une compréhension approfondie des besoins des utilisateurs, notre application web aspire à devenir un outil indispensable pour la gestion des urgences médicales, contribuant ainsi à sauver des vies et à améliorer la qualité des soins dispensés.

**Chapitre 1 : Le Web au service de la médecine**

1. **Introduction :**

Au cœur des enjeux de la santé publique réside la gestion efficiente des situations médicales critiques, où chaque instant compte une vie. Le Web se présente comme un outil efficace offrant des solutions réalisables et utilisables par tout publique. Dans ce chapitre, nous allons présenter les fondements principaux du projet d'assistance médicale d'urgence ; explorer l'intégration des technologies web dans les pratiques médicales contemporaines et définir les objectifs clés qui guideront notre démarche.

1. **Applications web**

Les applications web, au cœur de l'écosystème numérique moderne, offrent un moyen puissant et accessible de fournir des services et des fonctionnalités à un large éventail d'utilisateurs. Contrairement aux applications mobiles, qui nécessitent un téléchargement et une installation sur un appareil spécifique, les applications web sont accessibles via un navigateur web sur n'importe quel appareil connecté à Internet, offrant ainsi une plus grande flexibilité et une accessibilité accrue.

Grâce à leur nature basée sur le cloud, les applications web permettent également une mise à jour et une maintenance centralisées, ce qui simplifie le processus de déploiement de nouvelles fonctionnalités et garantit une expérience utilisateur cohérente sur différentes plateformes. De plus, les applications web peuvent être facilement intégrées à d'autres systèmes et services, ce qui en fait un choix idéal pour les projets nécessitant une interopérabilité avec des technologies existantes.

Dans le domaine de la santé, les applications web jouent un rôle crucial en facilitant l'accès aux services médicaux, en permettant une gestion efficace des données médicales et en favorisant la communication et la collaboration entre les professionnels de santé. Leur capacité à fournir des fonctionnalités riches et interactives, combinée à leur accessibilité universelle, en fait un outil précieux pour améliorer les soins de santé et répondre aux besoins médicaux urgents de manière efficace et efficiente.

2. 1. **Les types d’applications web**
      1. **Les applications web statiques**

Il s'agit d'une application web basique avec un contenu limité et peu d'interactivité. Ces applications sont généralement créées en utilisant du CSS et du HTML, et peuvent inclure des éléments animés tels que des GIF et des vidéos. Modifier le contenu statique sur ces applications peut être difficile, nécessitant de télécharger, modifier et uploader le code HTML sur le serveur.

Si l'application n'a pas été créée par l’entreprise lui-même, il peut être nécessaire de contacter le développeur initial pour effectuer des changements importants.

##### **Avantages**

**Simplicité et Rapidité de Développement** : Les sites statiques sont généralement plus faciles et plus rapides à développer. Ils nécessitent moins de compétences techniques et peuvent être construits avec des connaissances de base en HTML et CSS. Ils sont idéaux pour les projets simples où le contenu ne change pas fréquemment.

**Performances et Temps de Chargement** : Les sites statiques se chargent généralement plus rapidement car ils ne nécessitent pas de requêtes serveur ou de traitement côté serveur pour générer les pages. Ils peuvent être hébergés sur des serveurs plus simples et moins coûteux, améliorant ainsi l'expérience utilisateur grâce à des temps de réponse rapides.

**Sécurité** : Les sites statiques sont moins vulnérables aux attaques comme l'injection SQL ou les scripts intersites (XSS), car ils n'interagissent pas directement avec des bases de données ou des serveurs d'application. Le risque de failles de sécurité est réduit car il n'y a pas de code côté serveur qui pourrait être exploité.

**Coût d'Hébergement** : L'hébergement des sites statiques est généralement moins coûteux car ils nécessitent moins de ressources serveur. Ils peuvent être hébergés sur des services de stockage de fichiers simples, ce qui réduit les coûts opérationnels.

##### **Inconvénients**

**Manque de Dynamisme et d'Interactivité** : Les sites statiques offrent peu ou pas d'interactivité. Ils ne peuvent pas fournir de contenu personnalisé ou répondre aux actions des utilisateurs en temps réel. Les fonctionnalités avancées comme les formulaires interactifs, les moteurs de recherche internes et les systèmes de gestion de contenu ne sont pas possibles sans scripts supplémentaires.

**Difficulté de Mise à Jour** : La mise à jour du contenu sur un site statique peut être laborieuse, nécessitant des modifications manuelles du code HTML. Pour les sites avec beaucoup de contenu ou des besoins fréquents de mise à jour, cela peut devenir une tâche chronophage et sujette à des erreurs.

**Évolutivité Limitée** : Ajouter de nouvelles fonctionnalités ou étendre un site statique peut être complexe et nécessiter une refonte complète du code. Les limitations structurelles des sites statiques rendent difficile l'intégration de nouvelles technologies ou fonctionnalités sans recourir à des scripts externes.

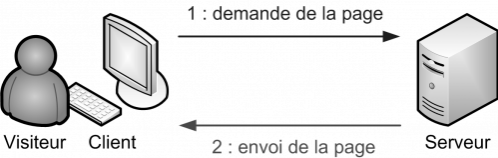


Figure 1: web static

* + 1. **Les applications web dynamiques**

Les applications web dynamiques sont plus complexes sur le plan technique que les applications statiques, car elles permettent une mise à jour constante et une modification des informations disponibles, notamment grâce à l'utilisation d'un système de gestion de contenu (CMS). Ces CMS accompagnent généralement les applications web dynamiques, facilitant ainsi la gestion et l'édition du contenu par les administrateurs ou les utilisateurs.

Différents langages web peuvent être utilisés avec des applications web dynamiques, parmi lesquels PHP et ASP sont les plus courants en raison de leur facilité de compréhension pour structurer le contenu. Bien que l'édition et la mise à jour du contenu soient relativement simples avec des applications web dynamiques, la partie backend impliquant la programmation peut être plus complexe en fonction du serveur et d'autres facteurs. Cependant, les éléments de design peuvent être facilement modifiés pour répondre aux préférences personnelles.

##### **Avantages du Web Dynamique**

**Interactivité Accrue** : Les sites dynamiques permettent des interactions en temps réel avec les utilisateurs. Les fonctionnalités comme les formulaires interactifs, les chatbots, et les notifications en temps réel rendent les sites plus engageants et améliorent l'expérience utilisateur.

**Mise à Jour Facile du Contenu** : Le contenu des sites dynamiques peut être mis à jour sans recharger l'ensemble du code HTML. Les systèmes de gestion de contenu (CMS) comme WordPress facilitent la gestion et la mise à jour du contenu par des utilisateurs non techniques.

**Personnalisation** : Les sites dynamiques peuvent offrir un contenu personnalisé basé sur les préférences et le comportement des utilisateurs. Cela permet de créer des expériences utilisateur adaptées, telles que des recommandations de produits ou des suggestions de contenu pertinentes.

**Évolutivité** : Les applications dynamiques sont plus faciles à étendre et à faire évoluer pour ajouter de nouvelles fonctionnalités. L'intégration de bases de données permet de gérer efficacement de grandes quantités de données et de répondre aux besoins croissants de l'application.

**Fonctionnalités Avancées** : Les sites dynamiques permettent d'intégrer des fonctionnalités avancées comme des moteurs de recherche internes, des systèmes de paiement en ligne, et des interfaces utilisateur riches. Les frameworks modernes comme React, Angular, et Vue.js facilitent le développement de ces fonctionnalités avancées.

##### **Inconvénients du Web Dynamique**

**Complexité de Développement** : Le développement de sites dynamiques est plus complexe et nécessite des compétences en programmation et en gestion de bases de données. Cela se traduit par des coûts et des temps de développement plus élevés par rapport aux sites statiques.

**Performances et Temps de Chargement** : Les sites dynamiques peuvent être plus lents à charger en raison de la génération du contenu côté serveur et des requêtes fréquentes à la base de données. Des optimisations telles que la mise en cache sont nécessaires pour améliorer les performances.

**Sécurité** : Les sites dynamiques sont plus vulnérables aux attaques comme l'injection SQL, le cross-site scripting (XSS), et autres failles de sécurité. Il est essentiel de mettre en place des mesures de sécurité robustes pour protéger les données et les utilisateurs.

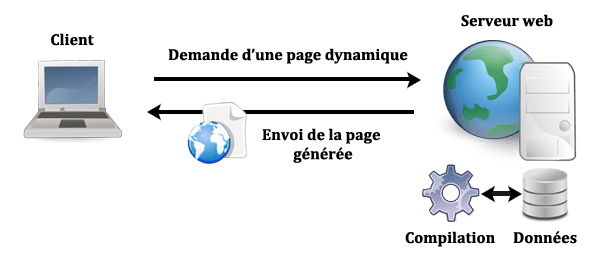


Figure 2: web dynamique

* + 1. **Les applications e-commerce**

Les applications web de commerce électronique sont des plateformes essentielles pour le shopping en ligne, permettant aux utilisateurs d'effectuer des achats, de passer des commandes et de gérer leur panier d'achats. Leur développement est plus complexe que celui des applications statiques ou dynamiques, car elles exigent une intégration de systèmes de paiement électronique et un panneau de gestion pour gérer les commandes et les articles. De plus, une version mobile de l'application web est souvent nécessaire pour assurer une expérience utilisateur cohérente sur différentes plateformes.

* + 1. **L’application de portail web**

Une application de portail web réfère simplement à une application dans laquelle différentes sections ou catégories sont accessibles par l’intermédiaire d’une page d’accueil. Ce type de page regroupe à la fois les forums, les chats, les courriels, les navigateurs et ainsi de suite.

* + 1. **Une application web animée**

Pour donner vie aux applications web, la technologie FLASH est souvent utilisée. Cependant, son utilisation peut être complexe pour les non-initiés. Malgré ses capacités à créer des animations modernes, FLASH présente des inconvénients, notamment en termes d'optimisation pour les moteurs de recherche comme Google, qui ont du mal à indexer correctement son contenu. Si votre site web est nouveau ou en phase de développement, il est conseillé de ne pas utiliser FLASH pour le moment.

* + 1. **Les systèmes de gestion de contenu**

Les systèmes de gestion de contenu (CMS) sont des outils essentiels pour maintenir et mettre à jour une application web de manière continue. Populaires pour les blogs personnels, les blogs d'entreprise et les médias, les CMS permettent d'accéder facilement à l'interface pour effectuer des modifications. Parmi les CMS les plus courants, on retrouve :

- WordPress : Célèbre pour son interface intuitive et la richesse de ses fonctionnalités, WordPress est largement utilisé aussi bien par les particuliers que par les professionnels, bénéficiant d'une vaste communauté et de nombreux tutoriels disponibles.

- Joomla : Bien qu'il soit moins populaire que WordPress, Joomla est une plateforme open source appréciée pour sa convivialité et ses fonctionnalités avancées. Bien que sa communauté soit plus petite que celle de WordPress, son interface reste intuitive et fonctionnelle.

* 1. **Les architectures des applications**

Les architectures des applications web définissent la structure et l'organisation des composants logiciels qui composent une application web. Voici quelques-unes des architectures les plus couramment utilisées :

* + 1. **Architecture monolithique**

Dans le modèle traditionnel, toute l'application est construite et déployée comme une seule unité. Toutes les fonctionnalités sont regroupées dans un seul code source et exécutées sur un seul serveur. Bien que simple à mettre en place, cette approche peut devenir difficile à maintenir et à évoluer avec le temps. En effet, la simplicité initiale provient du fait qu'il n'y a qu'une seule base de code et un seul déploiement à gérer. Cependant, à mesure que l'application grandit et que de nouvelles fonctionnalités sont ajoutées, la base de code devient de plus en plus complexe. Cette complexité rend le code difficile à comprendre, à tester et à modifier. De plus, le déploiement de nouvelles versions peut devenir risqué car une modification dans une partie de l'application peut impacter d'autres parties. En résumé, bien que le modèle d'application traditionnel soit facile à mettre en œuvre au début, il présente des défis de maintenance et de scalabilité à mesure que l'application se développe.

Figure 3:architecture monolithique

* + 1. **Architecture basée sur les microservices**

Figure 4: architecture microservices

Cette approche divise l'application en plusieurs services indépendants, chacun responsable d'une seule fonctionnalité spécifique. En d'autres termes, au lieu d'avoir un seul bloc de code contenant toutes les fonctionnalités, l'application est fractionnée en petits services autonomes. Chaque service est dédié à une tâche ou une fonctionnalité précise, telle que la gestion des utilisateurs, le traitement des paiements, ou l'envoi de notifications.

Cette architecture offre une plus grande agilité et évolutivité. Étant donné que les services sont indépendants, il est plus facile de mettre à jour ou de remplacer un service sans affecter les autres. De plus, les services peuvent être mis à l'échelle de manière individuelle. Si un service particulier reçoit beaucoup de trafic, il peut être répliqué et mis à l'échelle sans avoir à mettre à jour ou à répliquer toute l'application. Cela permet une utilisation plus efficace des ressources et une meilleure performance globale de l'application.



Figure 5: différence entre monolithique et microservices

* + 1. **Architecture client-serveur**

Cette architecture divise l'application en deux parties distinctes : le client, qui est généralement un navigateur web, et le serveur, qui fournit les données et la logique métier. Les deux parties communiquent via des requêtes HTTP.

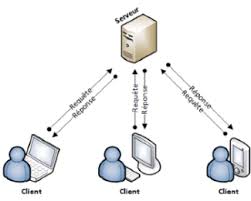


Figure 6: Architecture client serveur

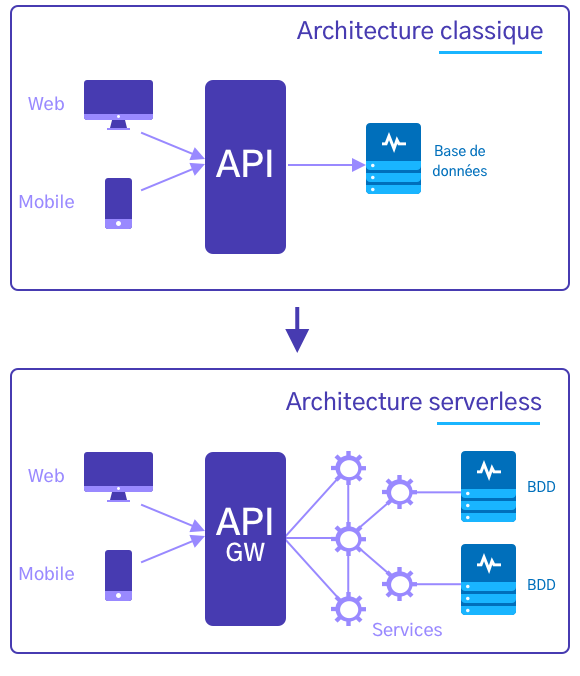
* + 1. **Architecture Serverless**

Figure 7: architecture serverless

Dans ce modèle, l'infrastructure sous-jacente est gérée par un fournisseur de services cloud, ce qui permet aux développeurs de se concentrer uniquement sur le code métier de l'application. Les applications Serverless sont souvent basées sur des fonctions cloud, qui sont déclenchées en réponse à des événements spécifiques.

1. **Médecine et télémédecine**
2. 1. **Introduction à la Médecine**

La médecine est un domaine scientifique et clinique qui se consacre à la maintenance et à la restauration de la santé par la prévention et le traitement des maladies. Elle englobe un large éventail de pratiques de soins de santé développées pour maintenir et rétablir la santé en prévention et en traitement des maladies. Les pratiques médicales traditionnelles reposent sur la médecine factuelle, où des preuves empiriques guident les décisions cliniques. Ces pratiques peuvent inclure la médecine générale, les spécialités médicales, la chirurgie, et la psychiatrie. Les avancées technologiques ont permis à la médecine d'évoluer, offrant des

méthodes de diagnostic et de traitement plus précises et efficaces.

* 1. **Télémédecine**

La télémédecine en est un parfait exemple de l’évolution constante. Cette pratique consiste à utiliser les technologies de l'information et de la communication pour permettre la prise en charge à distance des patients. Elle offre de nombreux avantages, notamment en matière d'accès aux soins et de suivi médical.

Figure 8: réunion médicale à distance

La télémédecine permet aux patients d'être pris en charge plus rapidement, notamment en cas d'urgence médicale. Grâce à des outils de communication en ligne, les médecins peuvent évaluer l'état de santé des patients à distance et leur prodiguer des conseils

****adaptés. Cette pratique est particulièrement utile dans le cadre d'une application web d'assistance médicale d'urgence, car elle permet d'optimiser la prise en charge des patients en temps réel.

Figure 9: consultation à distance

Les avantages de la télémédecine ne se limitent pas à l'urgence médicale. En effet, elle permet également de faciliter le suivi médical des patients atteints de maladies chroniques, en leur offrant un accès plus facile aux consultations médicales et en leur permettant de bénéficier de conseils personnalisés à distance. De plus, la télémédecine peut contribuer à réduire les coûts de santé en limitant les déplacements des patients et en optimisant l'organisation des consultations médicales.

En somme, la télémédecine est une pratique innovante et prometteuse qui offre de nombreuses possibilités en matière de prise en charge médicale. Dans le cadre d'une application web d'assistance médicale d'urgence, elle constitue un atout majeur pour garantir une prise en charge efficace et rapide des patients, tout en offrant un suivi médical de qualité.

1. **Problématique et domaine d’étude**
2. 1. **Contexte et Problématique**

**Présentation du Contexte :** Dans un monde où les urgences médicales se multiplient et deviennent de plus en plus complexes, la question de la prise en charge rapide et efficace des victimes demeure cruciale. La gestion des urgences médicales est un défi majeur pour les systèmes de santé partout dans le monde. Les délais dans la communication, la coordination inefficace entre les équipes médicales, et l'accès limité aux informations cruciales peuvent entraîner des retards dans les soins, compromettant ainsi la santé des patients.

**Problématique :** Face à ce défi, notre projet se positionne au croisement de la technologie et de la santé, cherchant à répondre à la problématique suivante : **Comment développer une solution numérique innovante pour optimiser la gestion des urgences médicales et garantir une intervention rapide et coordonnée des secours ?** Les principaux problèmes identifiés incluent :

* Accès limité aux soins médicaux d'urgence en temps voulu.
* Manque de coordination entre les différents acteurs du système de santé.
* Difficulté pour les patients à localiser rapidement les centres de soins d'urgence les plus proches.
* Absence de systèmes efficaces pour le suivi et la mise à jour des dossiers médicaux des patients en temps réel.
  1. **Objectifs du Projet**

**Objectifs Généraux :**

* Améliorer l'efficacité des interventions médicales d'urgence.
* Faciliter la communication et la coordination entre les patients, les équipes médicales, et les centres de soins.

**Objectifs Spécifiques :**

* Fournir une plateforme en ligne pour la localisation rapide des centres de soins d'urgence.
* Intégrer des fonctionnalités de téléconsultation pour des avis médicaux immédiats.
* Mettre en place un système de suivi en temps réel des interventions médicales.
* Assurer la mise à jour instantanée des dossiers médicaux des patients.
  1. **Étude des besoins et des exigences**

**Besoins Fonctionnels :**

* **Géolocalisation des Patients et des Centres de Soins :** Permettre aux patients de trouver rapidement les centres d'urgence les plus proches.
* **Téléconsultation :** Offrir des consultations médicales à distance via des appels vidéo.
* **Gestion des Urgences :** Enregistrer, suivre et mettre à jour les cas d'urgence en temps réel.
* **Notification et Alerte :** Envoyer des alertes instantanées aux équipes médicales et aux patients.
* **Accès aux Dossiers Médicaux :** Permettre aux médecins de consulter et de mettre à jour les dossiers médicaux des patients.

**Besoins Non Fonctionnels :**

* **Sécurité des Données :** Assurer la confidentialité et la protection des données médicales des patients.
* **Performance :** Garantir une réponse rapide du système, même en période de forte affluence.
* **Fiabilité :** Assurer un fonctionnement continu et sans interruption du service.
  1. **Analyse des utilisateurs et des parties prenantes**

**Identification des Utilisateurs :**

* **Patients :** Utilisent l'application pour localiser des centres de soins d'urgence et accéder à des réclamations urgences.
* **Médecins et Personnel Médical :** Utilisent l'application pour suivre les cas d'urgence et accéder aux dossiers médicaux des patients.
* **Administrateurs de Système :** Gèrent et maintiennent l'application.

**Analyse des Parties Prenantes :**

* **Patients et leurs Familles :** Ont besoin d'un accès rapide et fiable aux soins d'urgence.
* **Professionnels de Santé :** Nécessitent des informations précises et en temps réel pour fournir des soins adéquats.
* **Institutions de Santé :** Souhaitent améliorer la coordination et l'efficacité des soins d'urgence.

1. **Positionnement du Projet**
   1. **Étude de l'État de l'Art**

L'étude de l'état de l'art permet d'analyser les solutions existantes en matière d'assistance médicale d'urgence afin d'identifier leurs forces, leurs faiblesses et les opportunités d'amélioration. Actuellement, plusieurs applications et systèmes sont utilisés pour gérer les situations d'urgence médicale, chacun avec ses spécificités et ses limitations.

1. 1. 1. **Applications de Gestion des Urgences :**

- Système de Gestion des Informations Médicales d'Urgence (EMIS) : Utilisé principalement par les services de secours pour enregistrer et suivre les informations médicales des patients. Cependant, ce système peut manquer de flexibilité et d'interopérabilité avec d'autres systèmes de santé.

- Systèmes de Triage Médical: Ces systèmes, souvent utilisés dans les hôpitaux, permettent de classer les patients selon la gravité de leur état. Néanmoins, ils peuvent être limités par leur complexité d'utilisation et leur coût d'implémentation.

* + 1. **Applications de Localisation des Centres Médicaux :**

- Google Maps et Autres Applications GPS : Bien que très utiles pour la localisation des centres médicaux, ces outils ne fournissent pas d'informations spécifiques sur la disponibilité des services ou sur l'aptitude des centres à gérer certains types d'urgences.

- Applications Spécialisées de Localisation d'Urgence : Certaines applications offrent des fonctionnalités avancées pour trouver les centres médicaux les plus proches. Toutefois, elles peuvent ne pas intégrer des informations critiques telles que les temps d'attente ou la spécialisation des services offerts.

* + 1. **Applications de Communication en Temps Réel** :

- Systèmes de Notification d'Urgence : Utilisés pour alerter les services d'urgence, ces systèmes manquent parfois de fonctionnalités pour la communication bidirectionnelle ou la mise à jour en temps réel des informations.

- Applications de Messagerie pour les Urgences : Des applications comme WhatsApp ou Slack sont parfois utilisées pour coordonner les équipes d'urgence, mais elles ne sont pas spécifiquement conçues pour les besoins des interventions médicales et peuvent présenter des failles en termes de confidentialité des données.

* + 1. **Intégration des Technologies de l'Information et de la Communication :**

- Téléconsultation et E-Santé : L'essor des technologies de téléconsultation offre de nouvelles possibilités pour les soins à distance. Cependant, ces solutions nécessitent souvent une infrastructure technologique avancée et une formation appropriée des utilisateurs.

- Systèmes de Gestion de Contenu (CMS) : Utilisés pour la gestion des informations sur les patients et les interventions, ces systèmes peuvent être complexes à intégrer avec d'autres plateformes d'urgence.

En résumé, bien que de nombreuses solutions existent pour la gestion des urgences médicales, elles présentent souvent des limitations en termes de flexibilité, d'interopérabilité, et d'adaptation aux besoins spécifiques des utilisateurs. Notre projet vise à combler ces lacunes en développant une application web innovante et intégrée, conçue pour améliorer la coordination et l'efficacité des interventions d'urgence.

* 1. **Cahier de charges**

Notre application permettra notamment :

1. Gestion des victimes en phase Préhospitalière : Enregistrement des informations sur les victimes, y compris les données médicales pertinentes, les antécédents médicaux, et les actions entreprises par les premiers intervenants.
2. Consultation et mise à jour des informations des victimes.
3. Localisation les centres médicaux proches et adaptés (à travers la position actuelle des victimes), Orientation les victimes vers ces centres, des centres d'assurance et de police.
4. Fourniture de fiches techniques des gestes de premiers secours.
5. Transport des victimes vers les centres médicaux, en prenant en charge les zones isolées par hélicoptère.
6. Enregistrement des informations sur les professionnels de la santé intervenants dans la prise en charge des urgences.
7. Prévenir des centres Help pour aider et donner des instructions adaptées à l’état des victimes.
8. Génération de documents PDF : Capacité à générer des documents PDF contenant des informations sur les victimes, les professionnels de l'urgence, les services d'hospitalisation, les assurances, la police, ainsi que des ordonnances médicaux.
9. Identifier les centres pharmaceutiques et les pharmacies de garde et activer le service de livraison des médicaments essentiels (approbation des ordonnances).
10. Possibilité de communiquer en temps réel avec les services d'urgence et les autres professionnels de la santé (centre appelle | chat).

**Remarque :**

Ce cahier de charges a été techniquement validé sous l’assistance de mon encadrant avec **Madame Fatima Zahra Rifi,** infirmière spécialisée en anesthésie et réanimation, qui possède trois ans d'expérience au complexe chirurgical d'urgence de l'hôpital régional d’Assa-Zag. Que je tiens à remercier profondément.

1. **Conclusion**

En conclusion, ce premier chapitre a mis en lumière sur l'impact significatif des technologies web dans l'évolution des pratiques médicales, en mettant en évidence à la fois les opportunités et les défis associés à l'intégration des technologies numériques dans le domaine de la santé. En positionnant stratégiquement notre projet d'application web d'assistance médicale d'urgence au carrefour de ces avancées technologiques et des besoins criants en gestion efficace des urgences médicales, nous nous engageons à répondre de manière innovante aux exigences croissantes de rapidité, de sécurité et de coordination dans le traitement des situations critiques. À travers un cahier des charges rigoureux, nous avons établi les fondations nécessaires pour développer une solution qui non seulement améliore l'accessibilité et l'efficacité des soins d'urgence, mais qui également respecte les normes strictes de sécurité des données et offre une interface utilisateur intuitive pour les intervenants médicaux. Ce chapitre établit ainsi un cadre conceptuel essentiel qui guide notre approche méthodique et technologiquement avancée pour le développement de l'application, tout en posant les bases pour les chapitres à venir qui se concentreront sur la conception détaillée, les choix technologiques spécifiques et l'implémentation pratique de notre solution.

**Chapitre 2 Conception UML et outils de développement**

1. **Introduction**

Dans le domaine de la santé, la gestion des urgences médicales nécessite une approche méthodique et efficace pour assurer des soins rapides et coordonnés aux patients. Notre projet "Application Web d'Assistance Médicale d'Urgence" se concentre sur le développement d'une solution numérique innovante pour répondre à cette exigence critique. Ce chapitre combine à la fois l'analyse approfondie des besoins et des exigences du système avec la conception détaillée à l'aide de la modélisation UML (Unified Modeling Language).

1. **Conception**
2. 1. **Diagramme de cas d’utilisation**

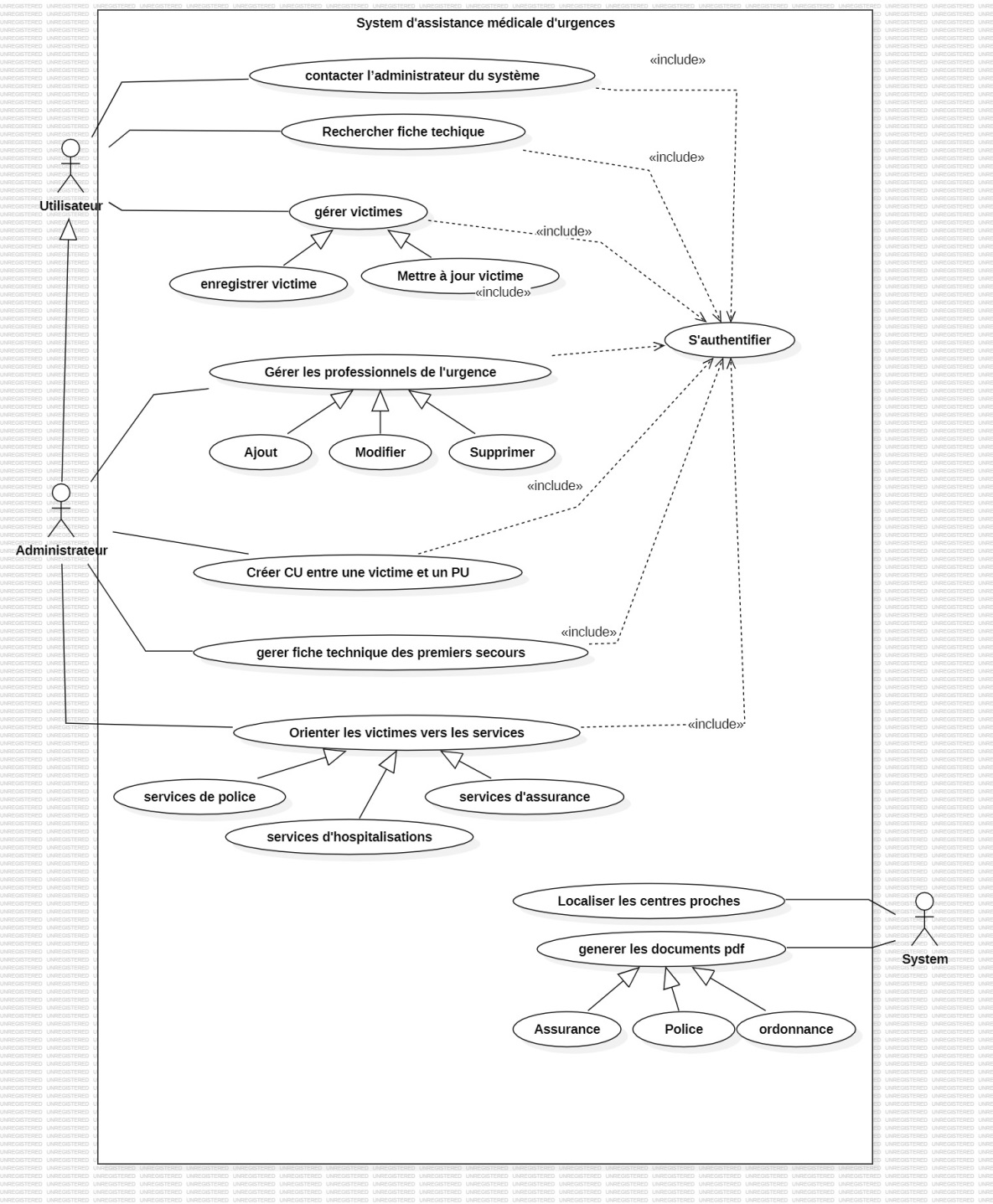
Le diagramme de cas d'utilisation est un diagramme qui représente visuellement les interactions entre les utilisateurs d'un système et ce système lui-même. Il met en lumière les différentes fonctionnalités et actions qu'un utilisateur peut effectuer avec le système, ainsi que les relations entre ces différents acteurs et scénarios. Le diagramme de cas d'utilisation est un outil essentiel dans le processus de conception et de documentation des exigences d'un système informatique.

* + 1. **Les acteurs principaux**

Comme le montre la figure-10, représentant le digramme des cas d’utilisation, comportent les éléments suivants :

1. Administrateur : C’est un utilisateur avec plus de privilèges généralement c’est un professionnel d’urgence.
2. Les cas d’utilisations :
   * 1. Enregistrer un professionnel d’urgence, modifier et supprimer.
     2. Créer consultation d’urgence.
     3. Gérer fichier technique.
     4. Orienter les victimes vers les services.
3. Utilisateurs :
   * 1. Créer compte.
     2. Enregistrer une victime, modifier .
     3. S’authentifier.
     4. Recherche fiche technique.
4. System
   * 1. Générer des documents pdf.
     2. Localiser les centres médicaux proches.

Figure 10: diagramme de cas d'utilisation principale

* + 1. **Description des interactions**

**Créer un compte :** Le but de ce cas d'utilisation est de permettre à un utilisateur de créer un compte sur la plateforme. L'utilisateur entre ses informations dans un formulaire, puis le système vérifie les informations saisies et vérifie leur exactitude, et à la fin le système affiche un message de réussite ou d'échec de création du compte.

**S’authentifier :** L’utilisateur doit s'authentifier avec un nom d'utilisateur et un mot de passe pour confirmer ses informations personnelles. Une seule authentification est requise. Pour établir son lien avec l’application. En utilisant son email comme nom d'utilisateur.

**Enregistrer victime :** Permet de santé d'enregistrer les informations d'une victime en phase préhospitalière dans le système comme le sexe, les actions entreprises par les premiers intervenants.

**Rechercher une fiche technique :** rechercher les fiches techniques de premiers secours déjà enregistré dans le system selon le cas de la victime.

**Gérer les consultations d'urgence:** un Administrateur crée une consultation d'urgence entre une victime et un professionnel de l'urgence.

**Orienter une victime vers un service :** Permet à un professionnel de santé d'orienter un blessé vers un établissement hospitalier adapté à son état et à ses besoins médicaux.

* 1. **Diagramme de classes**

Un diagramme de classes est un type de diagramme statique dans la modélisation orientée objet (UML) qui décrit la structure d'un système en représentant ses classes, attributs, méthodes et les relations entre les objets. C'est un outil essentiel pour la conception de systèmes logiciels, car il permet de visualiser les différents composants du système et leurs interactions.

1. **Classes**

**Définition :** Une classe est une représentation abstraite des objets partageant les mêmes caractéristiques (attributs) et comportements (méthodes). Elle sert de modèle pour créer des instances (objets) de cette classe.

**Éléments :**

**Nom de la Classe :** Identificateur unique de la classe.

**Attributs :** Propriétés ou caractéristiques des objets de la classe.

**Méthodes :** Comportements ou fonctions que les objets de la classe peuvent réaliser.

1. **Attributs**

**Définition :** Un attribut est une variable au sein d'une classe, représentant une propriété ou une caractéristique des objets.

**Exemple :** Dans une classe Personne, les attributs pourraient être nom, prénom, et âge.

1. **Méthodes**

**Définition :** Une méthode est une fonction définie dans une classe qui décrit le comportement des objets de cette classe.

**Exemple :** Dans une classe Personne, les méthodes pourraient inclure parler(), marcher(), et dormir().

1. **Relations**

**Associations :** Représentent les relations statiques entre les classes.

* + **Unidirectionnelles ou Bidirectionnelles :** Indiquent si la relation peut être traversée dans une ou deux directions.
  + **Multiplicité :** Indique combien d'instances d'une classe peuvent être associées à une instance d'une autre classe (par exemple, 1:1, 1, N).

**Héritage (Généralisation) :** Représente une relation hiérarchique entre une classe parent (super-classe) et une ou plusieurs classes enfants (sous-classes).

**Composition et Agrégation :** Types spéciaux d'association représentant des relations "partie-tout". La composition indique une forte dépendance entre les parties, tandis que l'agrégation indique une relation plus faible.

La figure-11 présente le diagramme des classes de notre application. A partir de cette figure, nous distinguons les classes sur lesquelles se porte notre système d'assistance médicale d'urgence suivantes :

* Classe personne

Attributs :

- CIN : Numéro d'identification de la personne.

- telephone : Numéro de téléphone.

- email : Adresse email.

- tele : Téléphone fixe.

Méthodes :

- geters() : Méthode pour obtenir les attributs.

- seters() : Méthode pour définir les attributs.

* Classe Personnel (hérite de personne)

Attributs :

- Horaires\_de\_travail : Horaires de travail du personnel.

- specialité : Spécialité du personnel médical.

- grade : Grade du personnel.

Méthodes :

- EnregistrerPersonnel() : Enregistrer un nouveau personnel.

- AffecterPersonnel() : Affecter du personnel à des tâches spécifiques.

- ConsulterDisponibilité() : Consulter la disponibilité du personnel.

* Classe P1er\_intervention (hérite de Personnel)

Méthodes :

- effectuer\_suivi() : Effectuer le suivi des patients.

- participer\_reunions() : Participer aux réunions.

* Classe Per\_medic\_Avancé (hérite de Personnel)

Méthodes :

- intervenir\_en\_urgence() : Intervenir en urgence.

- effectuerOperation() : Effectuer des opérations médicales.

* Classe Victimes

Attributs :

- age : Âge de la victime.

- sexe : Sexe de la victime.

- location\_act : Localisation actuelle de la victime.

- score : Score médical ou de gravité.

- situation\_urg : Situation d'urgence.

Méthodes :

- EnregistrerVictime() : Enregistrer une nouvelle victime.

- ConsulterDossierMedical() : Consulter le dossier médical de la victime.

- getEmplacement() : Obtenir l'emplacement de la victime.

* Classe DossierMedical

Attributs :

- idPatient : Identifiant du patient.

- hist\_trait : Historique des traitements.

- hist\_consult : Historique des consultations.

- assurance : Informations d'assurance.

Méthodes :

- rechercher() : Rechercher des dossiers médicaux.

- update() : Mettre à jour un dossier médical.

- afficher() : Afficher un dossier médical.

- enregistrer() : Enregistrer un nouveau dossier.

- imprimer() : Imprimer un dossier médical.

* Classe Véhicules

Attributs :

- Matricule : Matricule du véhicule.

- Equip\_medic\_embarqués : Équipements médicaux embarqués.

- Disponibilité : Disponibilité du véhicule.

- pos\_actuel : Position actuelle du véhicule.

Méthodes :

- isDisponibilité() : boolean : Vérifier la disponibilité.

- RéapprovisionnerFournitures() : Réapprovisionner les fournitures du véhicule.

* Classe Ambulance (hérite de Véhicules)

Attributs :

- capacite\_transport : Capacité de transport.

Méthodes :

- modifierCapacite() : Modifier la capacité de transport.

* Classe Hélicoptère (hérite de Véhicules)

Attributs :

- Numéro\_série : Numéro de série de l'hélicoptère.

- Capacité\_charge : Capacité de charge de l'hélicoptère.

Méthodes :

- voler() : Faire voler l'hélicoptère.

* Classe Service

Attributs :

- Description : Description du service.

- Nom : Nom du service.

- CapacitéAccueil : Capacité d'accueil du service.

- derecteur : Directeur du service (de type personnel).

Méthodes :

- ajouter\_service() : Ajouter un nouveau service.

- ModifierDerecteur() : Modifier le directeur du service.

* Classe CentreMédical

Attributs :

- Capacite : Capacité du centre médical.

- Dispo\_de\_lits : Disponibilité des lits.

- Emplacement : Emplacement du centre médical.

Méthodes :

- assignerLit(patient) : Assigner un lit à un patient.

- gererVictime() : Gérer les victimes.

- gererPersonnel() : Gérer le personnel.

* Classe ÉquipementMédical

Attributs :

- id : Identifiant de l'équipement.

- spécialité : Spécialité de l'équipement.

- etat : État de l'équipement.

Méthodes :

- isDisponible() : Vérifier si l'équipement est disponible.

* Classe CentreDAppelUrgence

Attributs :

- numFixe : Numéro fixe du centre d'appel.

Méthodes :

- recevoirAppel() : Recevoir un appel d'urgence.

- transférerAppel() : Transférer un appel d'urgence.

- enregistrerRapport() : Enregistrer un rapport d'appel.

**Relations entre les classes :**

La classe « Victimes » présente des interactions avec d’autres classes.

1. Victimes ↔ Véhicules

- Les Victimes sont transportées par des Véhicules.

- La relation est 1 à 0..\* (une victime peut être transportée par plusieurs véhicules, et un véhicule peut transporter plusieurs victimes).

2. Victimes↔Personnel

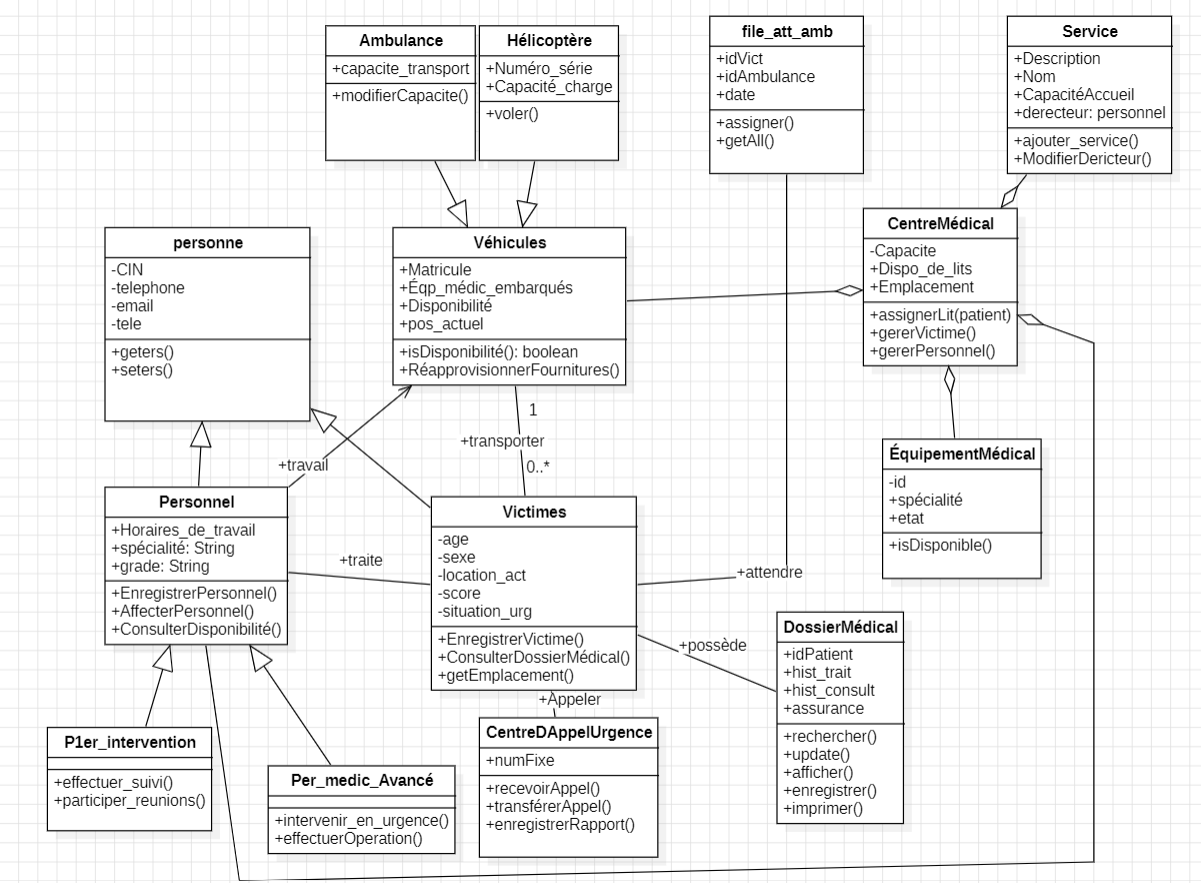
- Les Victimes sont traitées par du Personnel.

- La relation est 0..\* à 1 (une victime peut être traitée par plusieurs membres du personnel, et un membre du personnel peut traiter plusieurs victimes).

3.Victimes↔DossierMédical

- Chaque Victime possède un DossierMédical.

- La relation est 1 à 1 (une victime a un dossier médical unique, et un dossier médical appartient à une seule victime).

Figure 11: diagramme de classe

4. Victimes ↔ CentreDAppelUrgence

- Les Victimes peuvent appeler un CentreDAppelUrgence.

- La relation est 0..\* à 1 (une victime peut appeler plusieurs fois le centre d'appel d'urgence, et le centre d'appel d'urgence peut recevoir des appels de plusieurs victimes).

5. Victimes ↔ CentreMédical

- Les Victimes sont prises en charge par un CentreMédical.

- La relation est 0..\* à 1 (une victime peut être traitée par plusieurs centres médicaux au cours du temps, et un centre médical peut traiter plusieurs victimes).

**Relations entre les autres classes :**

- Ambulance et Hélicoptère héritent de Véhicules.

- Véhicules est une classe générale pour les moyens de transport médical.

- Personnel a des horaires de travail, spécialités, et grades. Ils peuvent être affectés à des véhicules ou des victimes.

- CentreDAppelUrgence reçoit des appels et transfère des informations aux Personnel et Véhicules.

- CentreMédical a une capacité de lits, disponibilités, et gère des Personnel et Victimes.

- EquipementMédical est utilisé dans des CentreMédical.

- Service fait partie de CentreMédical et a des descriptions et capacités d'accueil.

En résumé, les Victimes sont au centre des relations entre les différentes entités du système, telles que les Véhicules, Personnel, DossierMédical, CentreDAppelUrgence, et CentreMédical.

* 1. **Diagrammes des sequences**

Un diagramme de séquences est un type de diagramme UML (Unified Modeling Language) qui montre comment les objets interagissent entre eux en fonction d'une séquence de messages dans le temps. Ce type de diagramme est principalement utilisé pour représenter la dynamique d'un système, en mettant en évidence l'ordre chronologique des interactions entre les différents éléments du système.

2. 3. 1. **Éléments Clés d'un Diagramme de Séquences**

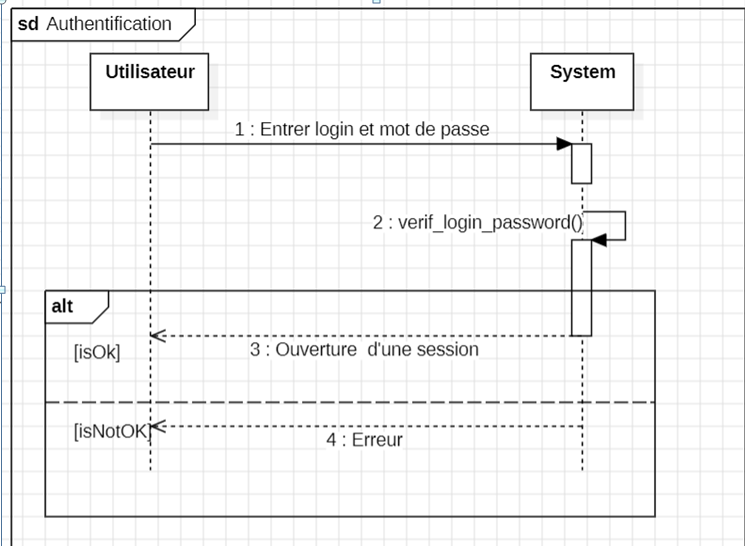
* **Acteurs** : Les acteurs représentent les utilisateurs ou les systèmes externes qui interagissent avec le système en question.
* **Objets** : Les objets représentent les instances des classes impliquées dans l'interaction.
* **Lignes de Vie (Lifelines)** : Les lignes de vie sont des lignes verticales qui descendent à partir des objets et des acteurs, représentant leur existence à travers le temps.
* **Messages** : Les messages sont des flèches horizontales ou obliques qui montrent les interactions entre les objets/acteurs. Ils peuvent représenter des appels de méthode, des envois de signaux, des réponses, etc.
* **Barres d'Activation** : Les barres d'activation sont des rectangles fins sur les lignes de vie qui montrent la période pendant laquelle un objet est actif pour traiter un message.
  + 1. **Authentification**
* L’utilisateur accède à l'interface d'authentification de l'application.
* Le système affiche les champs d'identification (nom d'utilisateur, mot de passe).
* L’utilisateur saisit son nom d'utilisateur et son mot de passe.
* L’utilisateur appuie sur le bouton "Se connecter".
* Le système vérifie si les informations d'identification sont correctes et correspondent à un compte utilisateur existant.
* Si les informations sont correctes, le système authentifie le user et lui permet d'accéder à son compte.
* Si les informations sont incorrectes, le système affiche un message d'erreur et invite l’utilisateur à saisir à nouveau ses informations d'identification.
* L’utilisateur peut réessayer de se connecter en saisissant à nouveau ses informations d'identification.

Figure 12: diagramme de séquence Authentification

* + 1. **Enregistrer victime**
* L’utilisateur a la possibilité de saisir des informations sur les victimes en remplissant un formulaire dédié.

*Le formulaire contient des cases à cocher pour indiquer l'état de la victime :*

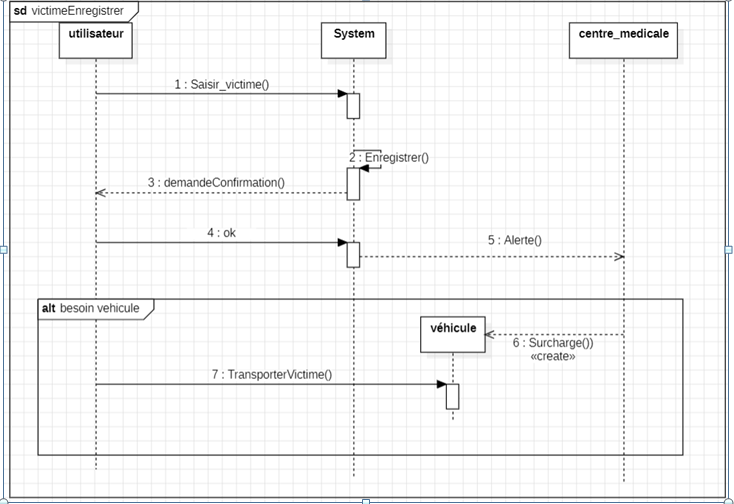
* Conscience (cochée si la victime est consciente, décochée si elle ne l'est pas)
* Respiration (cochée si la respiration est normale, décochée si il y a des difficultés respiratoires)
* Pouls (cochée si le pouls est présent, décochée si il est absent)
* Une fois le formulaire rempli, l'utilisateur peut le soumettre en appuyant sur un bouton dédié.
  + Les informations saisies par l'utilisateur sont enregistrées dans la base de données de l'application. L'utilisateur est invité à confirmer les informations avant l'envoi final.
  + Le système envoie automatiquement une alerte au centre médical dès que le formulaire est validé et soumis.
  + Si la victime a besoin d'un véhicule d'urgence, le centre médical s'occupera de dépêcher un véhicule équipé pour transporter la victime.

Figure 13: diagramme de séquence de enregistrer une victime

* + 1. **Localisation des victimes**
  + L'opérateur reçoit un appel d'urgence.
  + L'opérateur demande et obtient la permission d'accéder à la localisation du téléphone de l'appelant.
  + Le système GPS localise le téléphone et envoie les coordonnées au système.
  + L'opérateur visualise la localisation sur une carte et initie les secours.

Figure 14: Diagramme de séquence de localisation des victimes

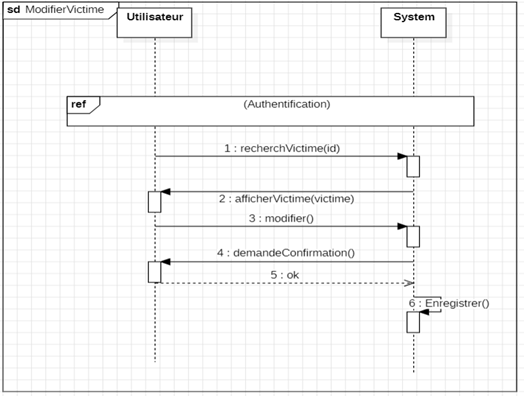
* + 1. **Modifier les informations d’une victime**
* L’utilisateur se connecte à la page qui gère les informations des victimes. Il s'identifie

Figure 15: diagramme de séquence de modifier une victime

avec les informations d'identification appropriées pour accéder au dossier de la victime concernée.

* L’utilisateur localise le dossier de la victime et identifie les informations à modifier. Il met à jour les détails pertinents, tels que l'état de la victime ou tout autre élément important.
* Avant de finaliser les modifications, l’utilisateur est invité à confirmer les changements apportés. Cela peut impliquer une étape de validation ou de confirmation pour s'assurer que les modifications sont correctes et justifiées.
  + 1. **Chercher fiche techniques du premier secours**

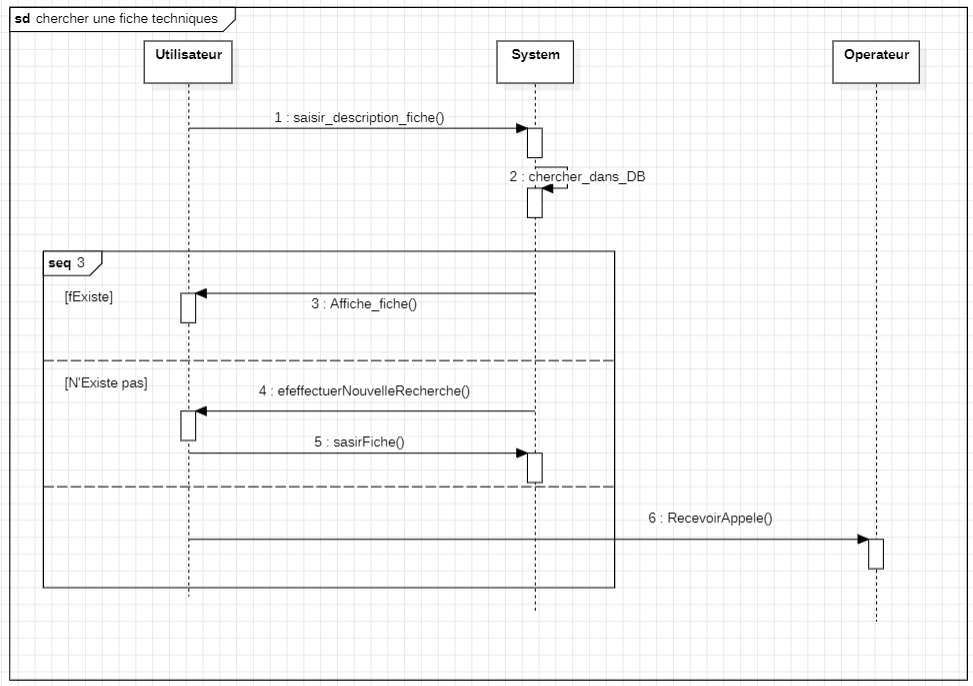


Figure 16: Diagramme de séquence de chercher une fiche techniques

* + L’utilisateur utilise la fonction de recherche pour trouver la fiche technique appropriée en fonction de la situation d'urgence spécifique. Il peut entrer des mots-clés, des symptômes ou des types d'incident pour affiner les résultats.
  + Si la fiche technique existe, le système affiche la fiche à l'utilisateur.
  + L’utilisateur accède à la fiche technique de premiers secours correspondante et consulte les instructions détaillées sur la manière de traiter la victime. Il prend connaissance des étapes à suivre, des gestes à effectuer et des précautions à prendre.
  + Si la fiche technique n'existe pas, l'utilisateur est invité à effectuer une nouvelle recherche ou à consulter un responsable pour obtenir des directives supplémentaires

1. **Outils et technologies**

Le développement d'une application web d'assistance médicale d'urgence nécessite l'utilisation d'outils adaptés pour assurer la robustesse, la sécurité et la performance du système. Ce chapitre explore les différents outils technologiques et logiciels utilisés dans le cadre de notre projet, visant à optimiser la gestion des urgences médicales et à fournir une plateforme efficace pour les intervenants médicaux et les patients.

2. 1. **Environnement de Développement**

Pour mener à bien ce projet en tant que développeur unique, j'ai sélectionné une combinaison d'outils adaptés aux exigences de développement back-end et front-end.

1. 1. 1. **Back-end (Java EE)**

* **Eclipse IDE :**



Figure 17:Eclipse IDE

Environnement de développement intégré pour Java EE, offrant des fonctionnalités avancées telles que la compilation, le débogage et la gestion de projet.

**Avantages :**

**Environnement Intégré :** Facilite la gestion des projets complexes grâce à une interface utilisateur conviviale et des fonctionnalités étendues.

**Compatibilité :** Intégration avec divers serveurs d'application Java EE comme Apache Tomcat, simplifiant le déploiement.

**Extensibilité :** Supporte une large gamme de plugins pour répondre aux besoins spécifiques de développement.

* **Apache Tomcat (ou autre serveur Java EE) :**

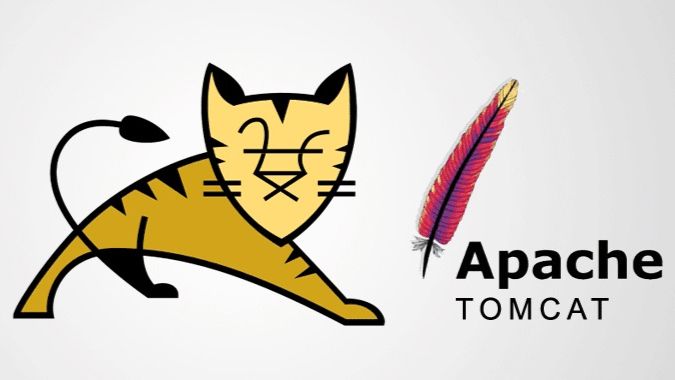


Figure 18: Appache tomcat

Serveur d'application Java EE utilisé pour déployer et exécuter l'application web.

**Avantages :**

**Léger et Rapide :** Optimisé pour une exécution rapide et une faible consommation de ressources.

**Facilité de Configuration :** Configuration simple et administration aisée, réduisant le temps de déploiement et de maintenance.

**Compatibilité :** Conforme aux spécifications Java EE, assurant une bonne interopérabilité avec d'autres composants Java.

* **Hibernate :**



Figure 19: Hibernate

Framework ORM facilitant la gestion de la persistance des données en général gere la cohérence entre les deux mondes objets et relationnels.

**Avantages :**

**Simplicité :** Simplifie le mapping objet-relationnel, réduisant ainsi la quantité de code nécessaire pour interagir avec la base de données.

**Productivité :** Automatise les tâches de gestion des données, améliorant la productivité du développeur.

**Performance :** Optimise les requêtes SQL et propose des mécanismes de mise en cache pour améliorer les performances globales de l'application.

* **Xampp (pour le mapping objet-relationnel) :**



Figure 20: XAMPP

Outil de mapping objet-relationnel pour faciliter la manipulation des données.

**Avantages :**

**Facilité d'Utilisation :** Simplifie le mapping des objets Java aux tables de la base de données via des annotations ou des fichiers de configuration XML.

**Flexibilité :** Offre une grande flexibilité dans la définition des relations et des requêtes complexes entre les objets Java et la base de données.

**Performance :** Optimisé pour des opérations rapides de lecture et d'écriture, améliorant ainsi les performances de l'application.

* + 1. **Front-end**

**Visual Studio Code (VS Code) :** Éditeur de code principal pour le développement front-end.

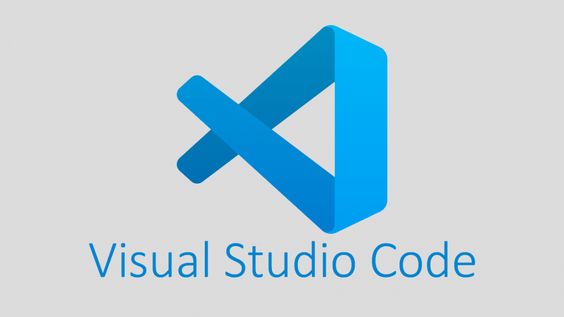


Figure 21: VS Code

**Avantages :**

**Léger et Rapide :** Interface légère et réactive, idéale pour le développement JavaScript et l'édition de fichiers HTML/CSS.

**Extensibilité :** Large écosystème d'extensions pour ajouter des fonctionnalités supplémentaires selon les besoins du projet.

**Intégration avec Git :** Intégration native avec Git pour le suivi des modifications du code et la gestion des versions directement depuis l'éditeur.

* **HTML/CSS :**

Langages de base pour la création de l'interface utilisateur (UI) et la présentation des données.

**Avantages :**

**HTML5 :** Structure de base pour organiser le contenu de la page web de manière sémantique et accessible.



Figure 22: HTML

**CSS3 :** Permet de styliser et de mettre en forme le contenu HTML, offrant un contrôle précis sur l'apparence et le layout de l'interface utilisateur.



Figure 23: CSS

**Compatibilité :** Prise en charge par tous les navigateurs modernes, assurant une expérience utilisateur cohérente.

* + 1. **Presentation**
* **StarUML :**



Figure 24: StarUML

StarUML est un logiciel de modélisation UML, commercialisé en open source par son éditeur, à la fin de son exploitation commerciale, sous un licence modifiée de GNU GPL. L'objectif de la reprise de ce projet était de remplacer des solutions commerciales comme IBMRationalRose ou BorlandTogether. StarUML prend en charge la plupart des diagrammes spécifiés dans la norme UML 2.0. (7)

* 1. **Architecture du Projet (MVC)**

Pour assurer une architecture solide et modulaire j’ai utilisé dans ce projet le modèle MVC (Modèle-Vue-Contrôleur), c’est une approche qui sépare une application en trois composants

principaux : la logique de présentation des données (le Modèle), l'interface

utilisateur (la Vue) et la gestion des actions utilisateur (le Contrôleur). Cette séparation permet de gérer la complexité de l'application en divisant les responsabilités, facilitant ainsi la maintenance, l'évolutivité et la réutilisation du code.

* 2. 1. **Modèle (Model)**

Le Modèle représente la logique métier et les données de l'application. Il est responsable de la gestion des données, des règles métier et de la logique de l'application. Le Modèle ne sait rien de la Vue ni du Contrôleur, ce qui permet une séparation nette des préoccupations.

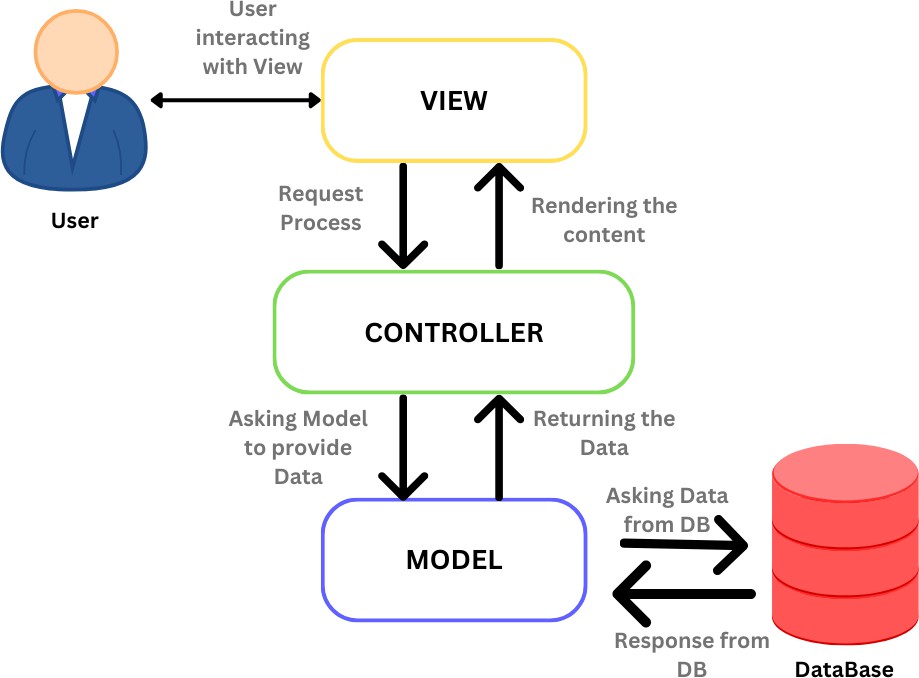


Figure 25: modèle MVC

* **Responsabilités :**
  + **Gestion des données** : Accès et manipulation des données, généralement à travers une base de données.
  + **Règles métier** : Logique spécifique à l'application qui doit être appliquée aux données.
  + **État de l'application** : Maintien de l'état courant de l'application, indépendamment de l'interface utilisateur.
* **Exemples dans le contexte du projet :**
  + **Classes d'entité Java** : Représentent les tables de la base de données, définies à l'aide de Java Persistence API (JPA) et gérées par Hibernate.
  + **DAO (Data Access Objects)** : Classes qui encapsulent les interactions avec la base de données, offrant des méthodes pour créer, lire, mettre à jour et supprimer des enregistrements.
    1. **Vue (View)**

La Vue représente l'interface utilisateur de l'application. Elle est responsable de l'affichage des données du Modèle et de la réception des interactions utilisateur. La Vue interagit uniquement avec le Contrôleur et ne contient aucune logique métier.

* **Responsabilités :**
  + **Affichage des données** : Présentation des données sous une forme compréhensible pour l'utilisateur.
  + **Interaction utilisateur** : Captation des entrées utilisateur, telles que les clics, les saisies de texte, etc.
* **Exemples dans le contexte du projet :**
  + **Pages JSP (JavaServer Pages)** : Génèrent du HTML dynamique en intégrant du code Java, utilisé pour afficher les données et interagir avec l'utilisateur.
  + **HTML/CSS/JavaScript** : Technologies utilisées pour structurer, styliser et rendre interactives les pages web.
    1. **Contrôleur (Controller)**

Le Contrôleur agit comme un intermédiaire entre le Modèle et la Vue. Il traite les requêtes utilisateur, met à jour le Modèle en conséquence et sélectionne la Vue appropriée pour la réponse. Le Contrôleur contient la logique de l'application qui répond aux interactions utilisateur.

* **Responsabilités** :
  + **Gestion des requêtes** : Réception et traitement des requêtes de l'utilisateur.
  + **Mise à jour du Modèle** : Modification des données du Modèle en réponse aux actions de l'utilisateur.
  + **Sélection de la Vue** : Choix de la Vue appropriée pour présenter les résultats à l'utilisateur.
* **Exemples dans le contexte du projet :**
  + **Servlets Java** : Gèrent les requêtes HTTP, appellent les méthodes du Modèle pour manipuler les données et dirigent vers les pages JSP pour la réponse.
    1. **Illustration de l'Architecture MVC**
* **L'utilisateur interagit avec la Vue** (par exemple, en remplissant un formulaire de réservation).
* **La Vue envoie une requête au Contrôleur** (par exemple, en soumettant le formulaire).
* **Le Contrôleur traite la requête** (par exemple, en validant les données du formulaire).
* **Le Contrôleur met à jour le Modèle** (par exemple, en enregistrant la réservation dans la base de données).
* **Le Modèle notifie la Vue** (via le Contrôleur) des changements de données.
* **La Vue affiche les données mises à jour** à l'utilisateur (par exemple, confirmation de la réservation).

1. **Conclusion**

En conclusion, la combinaison réfléchie de conception orientée utilisateur, de technologies robustes et d'outils de développement avancés forme un cadre solide pour l'application web d'assistance médicale d'urgence. Ce chapitre a posé les fondements nécessaires pour la réussite du projet, en assurant une approche méthodique et technologiquement avancée pour répondre aux défis complexes du domaine médical moderne.

**Chapitre 3 Implémentation**

* 1. **Introduction**

Le chapitre d'implémentation marque une étape cruciale dans la concrétisation de notre projet d'application web d'assistance médicale d'urgence. Après avoir défini les fondements théoriques et conceptuels dans les chapitres précédents, ce chapitre se concentre sur la mise en œuvre pratique de notre solution. Nous explorerons en détail les choix technologiques spécifiques, les décisions d'architecture système et les défis rencontrés lors du développement de l'application. À travers ce processus, nous visons à transformer notre vision en réalité fonctionnelle, tout en assurant la cohérence avec les objectifs établis et en répondant aux exigences rigoureuses du cahier des charges. Ce chapitre est essentiel pour comprendre comment les concepts abstraits se matérialisent dans un système interactif et opérationnel, prêt à répondre aux besoins urgents de la communauté médicale et des patients.

* 1. **Présentation de l’application développée**

2. 1. **Interfaces Utilisateur**
      1. **Page d’accueil**

Barre de menu (navigation)

Logo de notre application crée par ps Photoshop

Figure 26: Page d'accueil utilisateur

En cliquant sur ce bouton, l'utilisateur sera redirigé vers la page de connexion pour enregistrer des victimes.

La page d'accueil de l'application "Assistance Médicale d'Urgence" est conçue pour faciliter la gestion des urgences médicales. En haut, le menu de navigation propose des sections telles que Accueil, Gestion des Blessés, Professionnels, Rapports, Connexion, et Aide. La bannière principale accueille les utilisateurs avec le message "Bienvenue sur Assistance Médicale d'Urgence" et le slogan "Votre Solution pour la Gestion des Urgences Médicales".

En dessous de la bannière principale, quatre sections clés décrivent les fonctionnalités de l'application : Gestion des Blessés, permettant d'enregistrer et de suivre en temps réel les informations des blessés, Coordination des Intervenants, facilitant la coordination efficace des équipes médicales sur le terrain, Génération de Rapports, où les utilisateurs peuvent générer et télécharger des rapports PDF tels que des rapports d'assurance ou des ordonnances, représentée par une icône de feuille de papier ; et enfin Accès Mobile, mettant en avant l'application web mobile pour rester connecté et gérer les urgences où que l'on soit, symbolisée par une icône de téléphone portable. Chacune de ces sections est illustrée par une icône pertinente et une courte description, offrant une vue d'ensemble claire et concise des capacités de l'application.

* + 1. **Page de connexion**

La page de connexion de l'application "Assistance Médicale d'Urgence" offre plusieurs fonctions essentielles pour assurer une connexion sécurisée et facile des utilisateurs. Les utilisateurs doivent entrer leur numéro d'AMO dans le champ de saisie "Username" et leur CIN dans le champ de saisie "Password". Une fois ces informations saisies, en cliquant sur le bouton "Connecter", l'utilisateur envoie ses identifiants pour être authentifié et accéder à son compte.

Si un utilisateur a oublié son mot de passe, en cliquant sur le lien "J'ai oublié le mot de passe, cliquez ici", une boîte d'alerte s'affiche, indiquant que le mot de passe est en réalité leur CIN. Cette fonctionnalité simplifie la récupération du mot de passe et assure que les utilisateurs puissent rapidement retrouver l'accès à leur compte. Pour les nouveaux utilisateurs, le lien "Créer un nouveau compte" permet de s'inscrire sur la plateforme en créant un nouveau compte.

Figure 27: Page de connexion



Ce champ représente à la fois le nom d'utilisateur (username) et le numéro amo dans la base de données

Lorsqu'un utilisateur clique sur ce bouton, une servlet de vérification est appelée. Cette servlet récupère les données des deux champs, crée un objet de l'entité User, puis vérifie si cet utilisateur existe dans la base de données. Si les informations sont correctes et l'utilisateur est authentifié, il est redirigé vers la page de saisie des victimes.

* + 1. **Page d’inscription**

Le lien "J'ai déjà un compte" permet aux utilisateurs existants de revenir à la page de connexion s'ils se rendent compte qu'ils ont déjà un compte.



Si l'utilisateur fait une erreur en remplissant le formulaire, il peut cliquer sur le bouton "Réinitialiser" pour vider tous les champs et recommencer la saisie.

.

Lorsque l'utilisateur clique sur le bouton "Envoyer", les informations saisies sont envoyées au serveur pour créer un nouveau compte. Si les données sont valides et que l'utilisateur n'existe pas déjà, un nouveau compte est créé dans la base de données.

Figure 28: Page d'inscription

La page d'inscription de l'application "Assistance Médicale d'Urgence" permet aux nouveaux utilisateurs de créer un compte en saisissant leurs informations personnelles. Le formulaire comprend plusieurs champs : le champ "Nom" pour entrer le nom de l'utilisateur, le champ "Prénom" pour entrer le prénom, le champ "AMO" pour entrer le numéro d'Assurance Maladie Obligatoire, le champ "Password (CIN)" pour entrer le mot de passe correspondant au numéro de Carte d'Identité Nationale, et le champ "Téléphone" pour entrer le numéro de téléphone.

* + 1. **Page saisir victime**

La première partie de la page de déclaration d'une victime permet de saisir les informations générales sur la victime. L'utilisateur doit entrer le nom, l'âge, la date et l'heure de l'incident, ainsi que le lieu de l'incident dans les champs respectifs.



Si utilisateur a selectioné oui le system va le inserer une nouvelle ligne dans la table file\_attente\_amb

Cette icône permet de sélectionner la date et l'heure facilement au lieu de les saisir manuellement.

Figure 29: Page de saisir victime (informations personnelles)

La seconde partie de la page concerne l'évaluation de l'état de la victime. Elle est subdivisée en différentes sections pour une évaluation détaillée : le niveau de conscience (conscient, semi-conscient, inconscient), la présence et le type de saignement (contrôlable ou non), la présence de fractures et de brûlures, et l'évaluation cardiologique indiquant des symptômes comme des douleurs thoraciques ou des difficultés respiratoires. Une zone de texte permet d'ajouter d'autres observations pertinentes concernant les blessures et symptômes de la victime.

Pour chaque critère, nous attribuons des points afin de calculer un score permettant de déterminer la gravité de l'urgence. Ce score doit être persisté avec le degré de blessure dans des champs spécifiques de la table des victimes.

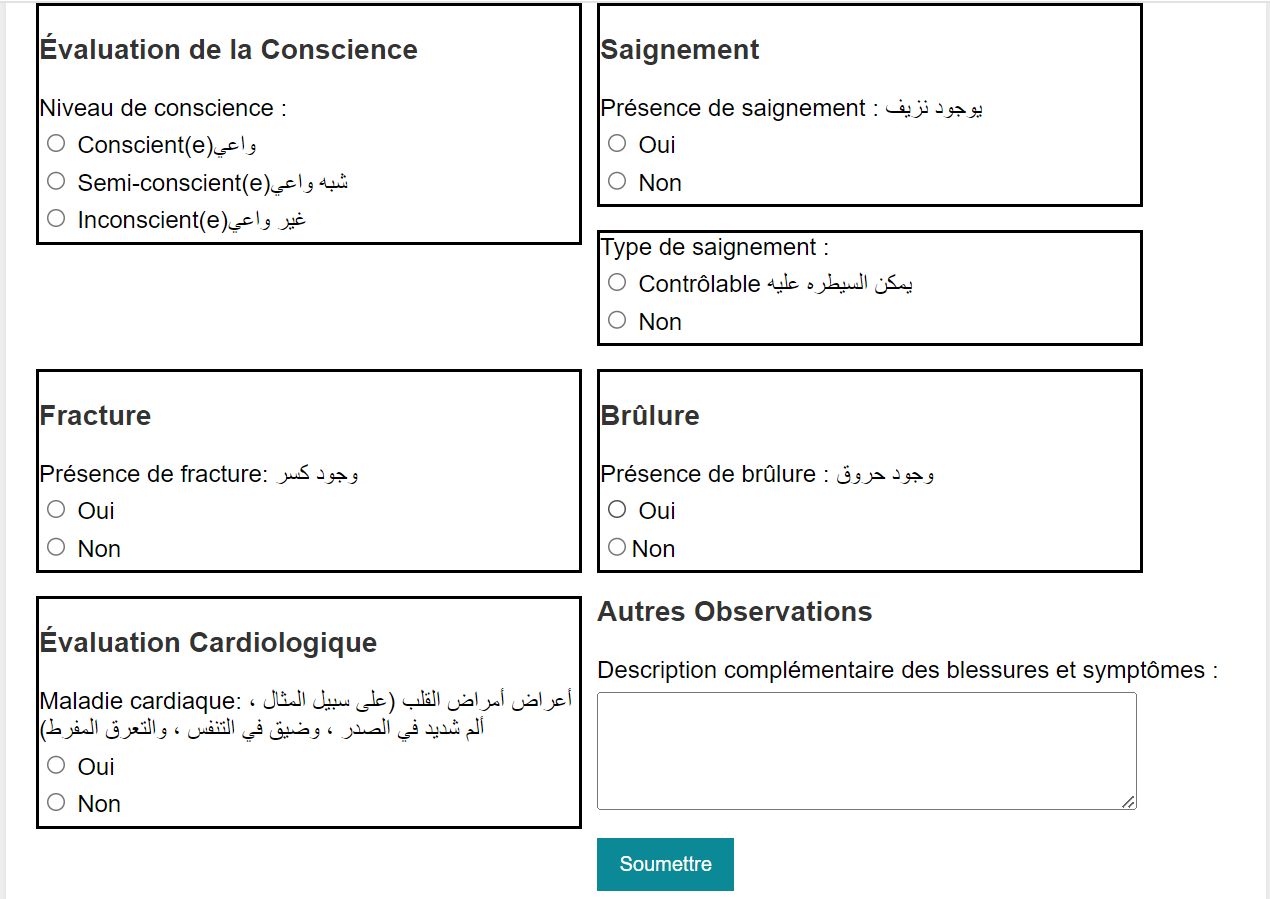


Figure 30: Page de saisir victime (Etat d'urgence)

Après avoir rempli ces informations, l'utilisateur peut soumettre le formulaire en cliquant sur "Soumettre". Les informations seront ensuite envoyées vers une servlet qui se chargera de traiter les données en profondeur. Cette servlet utilisera les fonctions calculateScore() et getState() de la couche DAO\_Victime pour calculer le score de gravité et déterminer le degré de blessure de la victime. Ces calculs permettront d'évaluer rapidement l'état de la victime et d'orienter les actions nécessaires. Ensuite, les données traitées seront enregistrées dans la base de données, garantissant ainsi une prise en charge adéquate et facilitant le suivi médical de la victime. Ce processus automatisé assure une réponse rapide et efficace aux urgences, en fournissant aux intervenants des informations cruciales et en réduisant le temps de réaction.

Pour établir un indice de priorité basé sur les critères fournis, nous pouvons attribuer des points à chaque contrainte. Voici un modèle de points pour chaque critère :

Évaluation de la Conscience {Conscient(e)): 1point, Semi-conscient(e) :2 points, Inconscient(e) :3 points}

Présence de saignement { Oui : 3 points, Non : 0 point, non Contrôlable :ajouter une point}

Présence de fracture { Oui : 2 points, Non : 0 point }

Présence de brûlure { Oui : 2 points, Non : 0 point}

Évaluation Cardiologique { Maladie cardiaque: Oui : 3 points, Non : 0 point}

* + 1. **Page de témoignages**

La page de témoignages de l'Assistance Médicale d'Urgence présente un espace dédié aux retours des utilisateurs, illustré par une citation d'un secouriste qui loue l'accessibilité et l'utilité de l'outil. Cette citation vise à renforcer la crédibilité et la valeur ajoutée du service en mettant en avant une expérience utilisateur positive



Figure 31: Page de témoignages

* 1. **Interfaces administrateur**
     1. **Page de gestion des utilisateurs**

La page de gestion des utilisateurs permet à l'administrateur de visualiser, modifier et supprimer des comptes utilisateurs enregistrés dans le système. La section principale de la page affiche un tableau récapitulatif des utilisateurs, qui inclut les colonnes pour le Numéro AMO, le Nom, le Prénom, le CIN, et le Téléphone. Ce tableau est dynamique et se remplit automatiquement à partir d'une liste d'utilisateurs récupérée de la base de données, grâce à la fonction getAll() de la couche dao\_Users. Cette automatisation assure que les informations affichées sont toujours à jour, reflétant les modifications les plus récentes effectuées dans le système.



Tableau de bord

"Modifier" permet de mettre à jour les informations de l'utilisateur en appelant la méthode update() du DAO des utilisateurs.

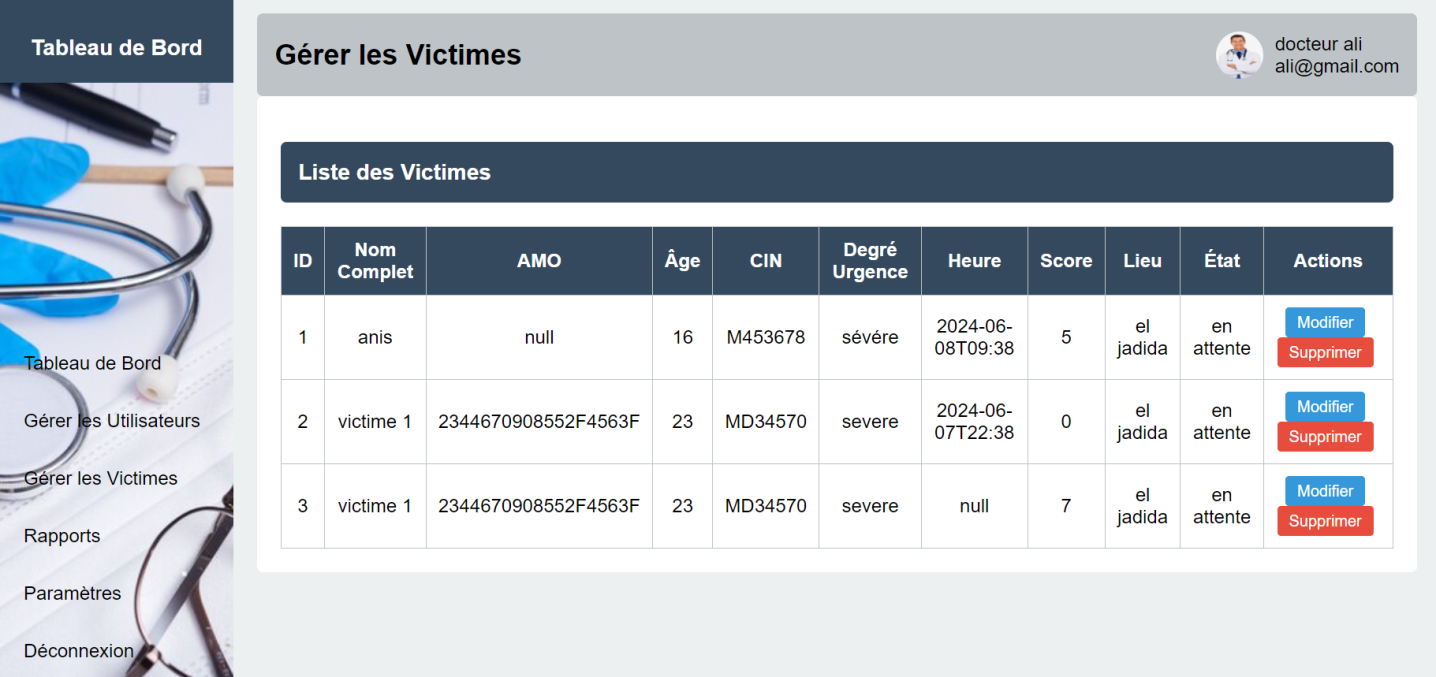
Ce bouton "Supprimer" permet de retirer un utilisateur du système. Cette action est généralement accompagnée d'une confirmation pour éviter les suppressions accidentelles

Figure 32: Page de gestion des utilisateurs

* + 1. **Page de de gestion des victimes**

La page de gestion des victimes permet aux administrateurs et aux professionnels de santé de suivre et de gérer efficacement les informations des victimes. Elle présente un tableau détaillé affichant des informations clés pour chaque victime, telles que l'ID, le Nom Complet, le Numéro AMO, l'Âge, le CIN, le Degré d'Urgence, l'Heure de l'incident, le Score, le Lieu de l'incident et l'État de la victime. Le tableau est automatiquement alimenté par une liste de victimes récupérée de la base de données via la fonction getAll() de la couche dao\_victimes. Cette fonctionnalité garantit que les informations affichées sont toujours à jour, offrant une vue précise et actuelle de la situation des victimes. Les colonnes du tableau permettent une évaluation rapide et efficace de chaque victime, facilitant ainsi la gestion et le suivi des urgences

Les données de l'administrateur connecté telles que l'email, le nom et le profil sont récupérées à partir de la session créée au moment du login.

Figure 33: Page gérer victime

* 1. **Test de Validation**

Le processus de test et de validation est crucial pour s'assurer que le système fonctionne correctement et répond aux exigences définies. Ce chapitre décrit les méthodes utilisées pour tester les différentes fonctionnalités de l'application, ainsi que les résultats obtenus. Les tests couvrent les aspects fonctionnels et non fonctionnels du système, y compris la performance, la sécurité et l'ergonomie.

1. 1. **Types de Tests**
      1. **Tests Unitaires**

Les tests unitaires sont réalisés pour vérifier le bon fonctionnement des composants individuels de l'application. Chaque fonction ou méthode des différentes classes a été testée de manière isolée pour s'assurer qu'elle produit les résultats attendus.

**Exemples de tests unitaires :**

* Vérification de la création d'un utilisateur (fonction createUser()).
* Vérification de la récupération d'une victime (fonction getVictimById()).
* Test de la fonction de calcul du score (fonction calculateScore()).
  + 1. **Tests d'Intégration**

Les tests d'intégration ont été effectués pour s'assurer que les différents modules de l'application fonctionnent bien ensemble. Par exemple, la communication entre les servlets et la couche DAO a été testée pour vérifier que les données sont correctement transmises et enregistrées dans la base de données.

**Exemples de tests d'intégration :**

* Soumission du formulaire de saisie de victime et vérification de l'enregistrement dans la base de données.
* Connexion de l'utilisateur et redirection vers la page appropriée.
* Modification des informations d'un utilisateur et mise à jour dans la base de données.
  + 1. **Tests de Système**

Les tests de système impliquent la vérification du système complet pour s'assurer qu'il répond aux exigences fonctionnelles et non fonctionnelles. Ces tests incluent des scénarios réalistes où plusieurs fonctionnalités sont utilisées de manière séquentielle.

**Exemples de tests de système :**

* Création d'un nouvel utilisateur, connexion avec cet utilisateur, saisie d'une victime, et vérification de l'affichage correct des informations dans la gestion des victimes.
* Test de la navigation à travers les différentes pages du tableau de bord pour vérifier l'ergonomie et la fluidité.
  1. **Conclusion**

La mise en œuvre des interfaces utilisateur et administrateur est une étape critique dans le développement de notre application. En concevant des interfaces conviviales et intuitives pour les utilisateurs finaux, nous visons à améliorer leur expérience d'utilisation tout en facilitant l'accès aux fonctionnalités essentielles. De plus, en développant des interfaces administratives robustes, nous nous assurons que les gestionnaires et administrateurs peuvent efficacement superviser et gérer les aspects critiques du système.

Les tests de validation jouent un rôle crucial dans ce processus. Ils nous permettent de vérifier que toutes les fonctionnalités implémentées répondent aux spécifications définies et fonctionnent de manière fiable dans divers scénarios d'utilisation. Cela garantit non seulement la qualité du produit final mais aussi sa performance optimale.

En combinant une conception réfléchie avec des tests exhaustifs, nous nous efforçons de fournir une solution logicielle qui répond pleinement aux besoins et aux attentes de nos utilisateurs. Cette approche nous permet de construire non seulement des interfaces utilisateur attrayantes mais aussi des interfaces administratives puissantes, contribuant ainsi au succès global de notre application.

**Conclusion générale**

La conception et l'implémentation d'une application web dynamique pour gérer le service d'urgence au sein des hôpitaux provinciaux ou régionaux représentent un effort significatif visant à répondre efficacement aux besoins critiques de différents services médicaux. En intégrant des fonctionnalités avancées telles que la géolocalisation, la communication en temps réel avec les services d'urgence, et la gestion sécurisée des données personnelles et médicales des patients, notre projet s'efforce de fournir une solution complète et fiable pour les utilisateurs finaux, notamment les patients, les premiers intervenants et les professionnels de la santé.

L'application ainsi implémentée vise à améliorer l'accessibilité au service d'urgence et la qualité des soins médicaux assurés par ce service. Elle permet de générer des réponses rapides et efficaces pour toutes les situations d'urgence affrontées. En outre, la conception d'interfaces utilisateur conviviales et d'interfaces administratives robustes garantit une expérience utilisateur optimale et une gestion efficace du système pour les administrateurs.

Les tests de validation rigoureux ont été essentiels pour assurer la fonctionnalité, la sécurité et la performance de l'application, en conformité avec les normes et réglementations médicales en vigueur. Ces tests ont permis de vérifier que toutes les fonctionnalités répondaient aux exigences spécifiques et fonctionnaient de manière fiable dans des scénarios réels.

En conclusion, notre application « Gestion du service médicale d'urgence » peut représenter une avancée significative dans le domaine des technologies de la santé publique. Elle vise améliorer la gestion des urgences médicales et d’offrir une meilleure qualité de d’hospitalisation aux patients. Outre, elle illustre notre engagement entant que programmeurs à utiliser les technologies du web pour répondre aux besoins critiques de notre société et à promouvoir des soins de santé accessibles et efficaces pour tous les malades.

**Webographie**

<https://www.w3schools.com/java/>

<https://learn.microsoft.com/fr-fr/azure/architecture/web-apps/serverless/architectures/web-app>

<https://kinsta.com/fr/blog/architecture-application-web/>

<https://www.medaviz.com/la-coordination-des-soins-par-la-telemedecine/>

<https://www.vie-publique.fr/eclairage/18473-la-telemedecine-une-pratique-en-voie-de-generalisation>

<https://docs.staruml.io/>

<https://fr.wikipedia.org/wiki/Diagramme_de_cas_d'utilisation>

<https://www.lucidchart.com/pages/fr/diagramme-de-classes-uml>

<https://aws.amazon.com/fr/what-is/web-application/>

<https://hibernate.org/>

<https://developer.mozilla.org/fr/docs/Web/CSS>

<https://www.mysql.com/>

<https://www.pole-medical-moreac.org/wp-content/uploads/2015/10/Gestes-1ers-secours.pdf>