1. Introduction :
2. Présentation de l’organisme d’accueil:

Diva Software est spécialisée dans le développement informatique de systèmes ERP, notamment pour les entreprises de confection. Elle s’appuie sur le potentiel de son équipe technique et commerciale pluridisciplinaire composée de talents issus d’écoles d’ingénierie tunisiennes. Diva Software a réussi à réaliser des produits innovants à haute valeur.

Elle s'est développée et a conquis de grands marchés internationaux, notamment en France et au Maroc

## Informations de Contact de Diva Software :

* **Adresse :**Boulevard Yasser Arafat Immeuble N° 38 - Bureau 44 Sahloul - 4054 Sousse
* **Propriétaire** : Mr Rochdi Jdidi
* **Administration :**(+216) 53 333 850  
  **Commercial :**(+216) 53 333 853
* **E-mail :**contact@diva-software.com
* **Site web :**[https://www.diva-software.com](https://www.diva-software.com/)



1. Activités :

Diva Software intervient dans la conception, le développement et le déploiement des systèmes d’information, de la fabrication à la livraison. Elle gère l’ensemble des processus d’une entreprise en intégrant toutes ses fonctions, y compris la gestion des ressources humaines, l’aide à la décision, la vente et l’approvisionnement. Grâce à une compréhension approfondie et une interaction étroite avec le client, Diva Software s'engage à long terme. En effet, cela est dû à sa réputation en tant que partenaire de choix et à son engagement absolu pour la réussite et la commercialisation de l'excellente exécution de ses projets

1. Service de Diva software :

Diva Software fournit différents services pour optimiser le cycle de production et améliorer les performances de l’entreprise, en mettant à disposition divers modules :

* 1. La gestion des stocks :

Ce module résout les problématiques liées à la gestion des stocks, des magasins et des articles. Il permet de définir l'état des stocks et de conserver un historique des mouvements de stocks dans les différents magasins et entrepôts. De plus, il autorise les entrées et sorties de stocks de manière manuelle ou automatique

* 1. La gestion de la production :

Ce module permet une gestion avancée des données techniques afin de préparer rapidement les dossiers de fabrication. Il offre une vision complète de la capacité de production (personnel, machines, matières premières, etc.). De plus, il permet de calculer les besoins nets (CBN) pour contrôler et optimiser les stocks en maintenant le lien entre les commandes clients et le lancement en fabrication associé aux commandes fournisseurs

* 1. La gestion de la qualité :

Ce module permet de collecter les informations de manière manuelle ou automatique, puis de les présenter sous forme d'indicateurs de performance de production. Cela garantit une amélioration continue du processus de fabrication.

* 1. La gestion des expéditions :

Ce module permet de gérer de manière simple les expéditions tout en générant les listes de colisage. Ces listes peuvent être automatiquement traduites en bons de livraison, qui serviront ensuite pour la facturation.

* 1. La facturation :

Ce module permet l’édition des factures.

* 1. La gestion de la maintenance :

Ce module est conçu pour gérer les différentes tâches de maintenance des équipements et assister quotidiennement les services de maintenance dans l'accomplissement de leurs missions.

1. Organigramme de l'entreprise :

L’entreprise est composée de 20 employés au total pour une gestion plus efficace et dynamique des principaux métiers de la société ; l'organigramme adopté est présenté dans la figure 2.



1. Présentation du projet :
   1. Contexte :

La maintenance industrielle joue un rôle crucial dans la durabilité et la performance des entreprises. Dans ce contexte, la Gestion de Maintenance Assistée par Ordinateur (GMAO) émerge comme un outil stratégique permettant d'optimiser la gestion des équipements, de planifier les interventions, et d'améliorer la disponibilité des ressources. Ce projet de fin d'études vise à concevoir et développer un logiciel de GMAO sur mesure pour répondre aux besoins spécifiques d'entreprises opérant dans le secteur de textile

* 1. Objectif :
* **Optimisation de la gestion des équipements :**
* **Planification des interventions :**
* **Amélioration de la disponibilité des ressources**
* **Personnalisation pour les besoins spécifiques du secteur textile :**
* **Intégration de fonctionnalités stratégiques**
* **Interface utilisateur conviviale**
* **Sécurité et fiabilité**

1. Étude et analyse de l’existant :
   1. Description de l’existant :

Dans le secteur textile, la gestion de la maintenance des équipements repose principalement sur des processus manuels et des outils traditionnels tels que les feuilles de calcul. Ces méthodes présentent toutefois des limites en termes de traçabilité des interventions, de gestion des plannings de maintenance, et de coordination des ressources.

Les entreprises font face à des défis tels que des temps d'arrêt non planifiés, des difficultés à suivre les historiques de maintenance, et une gestion fragmentée des données. Malgré les efforts des équipes de maintenance, il existe des opportunités d'amélioration en termes de réduction des coûts, d'optimisation des plannings, et d'augmentation de la disponibilité des équipements.

L'adoption d'une solution moderne de GMAO sur mesure s'avère ainsi essentielle pour centraliser et structurer les données de maintenance, automatiser les processus, optimiser les interventions, et améliorer la performance globale de la gestion de maintenance dans le secteur textile.

* 1. Critique de l’Existant :

Lors de notre étude précédente, nous avons identifié les problèmes suivants :

* **Fragmentation des données**
* **Manque de traçabilité**
* **Gestion des plannings des interventions**
* **Ressources sous-utilisées**
* **Manque d'analyse et de reporting**

En résumé, la gestion actuelle de la maintenance dans le secteur textile souffre de lacunes importantes en matière de données fragmentées, de suivi insuffisant, de planification peu efficace des interventions, d'utilisation suboptimale des ressources et d'analyse limitée des performances. Ces défauts ont un impact négatif sur la qualité, l'efficacité et la rentabilité des opérations de maintenance, justifiant ainsi le besoin impérieux d'une solution moderne et intégrée comme notre proposition de GMAO sur mesure.

1. Solution proposée :

De plus, en développant sa propre solution de GMAO sur mesure, Diva Software pourra personnaliser un outil de gestion de maintenance en fonction de ses besoins spécifiques. Elle pourra ajouter des fonctionnalités sur mesure pour répondre aux exigences particulières de son secteur d'activité et des processus de suivi de maintenance spécifiques à la société. Cette flexibilité accrue permettra à Diva Software d'adapter l'outil à ses spécificités de processus de gestion de maintenance et d'améliorer l'expérience utilisateur de son équipe. L’affectation des tâches aux différents acteurs, techniciens et responsables de maintenance, est assurée de manière centralisée par l’administrateur. Le suivi des états des interventions de maintenance est plus adapté aux besoins de la société. La classification des équipements est assurée pour chaque projet ainsi que leurs détails de réalisation comme la date de début, de fin, les membres de l’équipe, le budget, etc. Des recherches personnalisées sur un équipement donné peuvent fournir des informations détaillées sur ses interventions avec la société. Le suivi des tâches est assuré selon leurs états : Tâches en cours, terminées, à planifier et en retard.

1. Conclusion :

Dans ce chapitre, nous avons présenté le cadre général de projet. Nous avons fait l’étude et l’analyse de l’existant pour pouvoir proposer une solution sous forme d’une application web qui couvre les limites des projets similaires. Dans le deuxième chapitre, nous allons faire la spécification des besoins fonctionnels et non fonctionnels ainsi que le Framework agile suivi.

Chapitre2 : Spécification des besoins et pilotage du projet avec Scrum

1. Introduction :
2. Identification des acteurs :

Un acteur dans une application est une entité externe qui interagit avec le système [1].

Les trois types d’acteurs de l’Application de gestion de maintenance assisté par un ordinateur que nous proposons sont les suivants :

1. Administrateur :
2. Technicien :
3. Manager :
4. Spécification des besoins :
5. Spécification des besoins fonctionnels :

Le système doit remplir certaines actions, appelées exigences fonctionnelles, pour devenir opérationnel. Ces exigences englobent les tâches nécessaires que le système doit effectuer. Notre application a été conçue pour répondre aux besoins fonctionnels de différents types d'utilisateurs, notamment l’admin, les techniciens et les Managers.

* **Administrateur :**
* **Gestion des Utilisateurs :**
* **Gestion des Données** :
* **Configuration du Système** :
* **Reporting et Analyse** :
* **les techniciens :**
* **Gestion des Tâches de Maintenance** :
* **Gestion des Équipements** :
* **Gestion des Stocks** :
* **Saisie des Rapports d'Intervention** :
* **les Managers :**
* **Planification des Interventions** :
* **Suivi des Performances** :
* **Coordination des Équipes** :

1. Spécification des besoins non fonctionnelles :

Les besoins non fonctionnels sont des exigences qui définissent la qualité globale d'un système logiciel plutôt que ses fonctionnalités spécifiques. Contrairement aux besoins fonctionnels qui décrivent ce que le système doit faire, les besoins non fonctionnels se concentrent sur la manière dont le système doit fonctionner.

Les principaux besoins non fonctionnels de notre application se résument dans les points suivants :

* **Performance :** l’application doit être rapide et réactif lorsque les utilisateurs effectuent des actions ou accèdent à des informations. Cela garantit une expérience fluide et efficace pour tous les utilisateurs.
* **Fiabilité :** Minimiser les temps d'arrêt du système et garantir la disponibilité des fonctionnalités critiques en tout temps.
* **Sécurité :** L'application garantit la sécurité en attribuant des rôles à chaque utilisateur pour gérer leurs activités, et en sécurisant les comptes des utilisateurs avec des mots de passe cryptés. De plus, elle utilise un mécanisme de vérification d'authentification par Token JWT pour chaque tentative de connexion.
* **Évolutivité :**
  + Extensibilité : Concevoir le système de manière à ce qu'il puisse être étendu facilement pour intégrer de nouvelles fonctionnalités et s'adapter à l'évolution des besoins de l'entreprise.
  + Interopérabilité : Assurer que le système peut interagir de manière transparente avec d'autres systèmes et services externes, le cas échéant.
* **Convivialité :**
  + Interface utilisateur conviviale : Concevoir une interface utilisateur intuitive et conviviale pour faciliter l'adoption du système par les utilisateurs finaux et réduire le besoin de formation approfondie.
  + Accessibilité : Assurer que le système est accessible à tous les utilisateurs, y compris ceux ayant des besoins spécifiques en matière d'accessibilité.
* **Robustesse :** Le logiciel doit être solide et capable de résister à des situations difficiles sans tomber en panne.
* En intégrant ces besoins non fonctionnels dans la conception et le développement de votre logiciel de GMAO, vous pouvez garantir sa qualité globale et son succès dans l'amélioration des processus de maintenance industrielle.

1. Architecture :

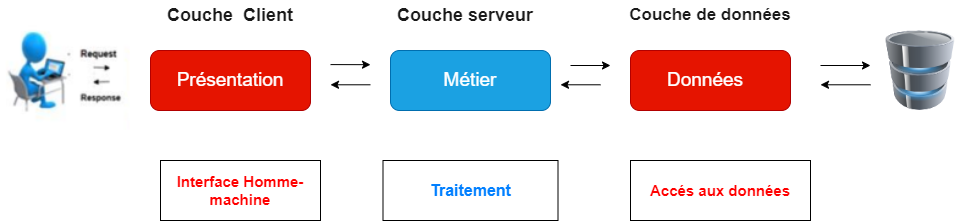
Avant de commencer à concevoir et à développer un système informatisé, il est essentiel de planifier son architecture.

* 1. Architecture fonctionnelle :



* 1. Architecture physique :

Notre application utilise une architecture en 3 tiers, une approche qui sépare clairement le système en trois couches distinctes. Ces trois couches sont représentées dans la figure 4 :



Il s’agit d’une architecture 3-tiers composé par 3 principaux couches :

**✓ Couche Client :** Elle représente l'interface utilisateur avec laquelle les utilisateurs interagissent directement.

**✓ Couche Web :** Elle fait référence à la partie de l'application qui traite les requêtes HTTP et génère des réponses pour les clients.

**✓ Couche de données :** Elle représente le serveur de base de données, fournisseur des données au serveur web.

* 1. Architecture technique :
  2. Architecture logicielle :

L’architecture d’un logiciel décrit la manière dont seront agencés les différents éléments d’une application et comment ils interagissent entre eux. Cette étape est donc l’une des premières étapes du développement logiciel et intervient lors de la phase de conception[2].

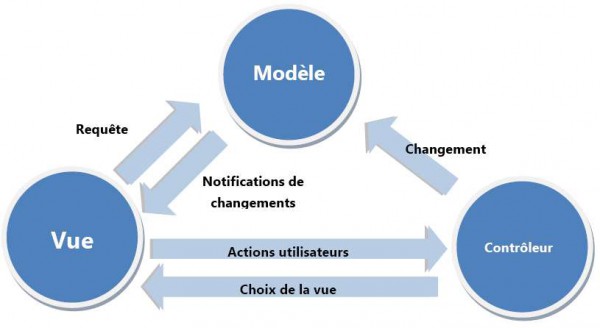
Notre système est conçu en utilisant l'architecture « MVC » pour la partie Back-end, « MVC » pour le Front-end, et « SOA » comme architecture globale de l'application.

* Choix de l’architecture côté back-end (côte serveur) :

Le choix de l'architecture côté serveur, ou back-end, dépend de plusieurs facteurs tels que les besoins fonctionnels de l'application, la scalabilité, la facilité de maintenance et la préférence des développeurs.

* **Architecture MVC (Modèle-Vue-Contrôleur)** :
* **Modèle :** cette partie gère ce qu'on appelle la **logique métier** de votre site. Elle comprend notamment la gestion des données qui sont stockées, mais aussi tout le code qui prend des décisions autour de ces données. Son objectif est de fournir une interface d'action la plus simple possible au contrôleur. On y trouve donc entre autres des algorithmes complexes et des requêtes SQL.
* **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. On y trouve essentiellement du code HTML mais aussi quelques boucles et conditions PHP très simples, pour afficher par exemple une liste de messages.
* **Contrôleur** : cette partie gère les **échanges** avec l'utilisateur. C'est en quelque sorte l'intermédiaire entre l'utilisateur, le modèle et la vue. Le contrôleur va recevoir des requêtes de l'utilisateur. Pour chacune, il va demander au modèle d'effectuer certaines actions (lire des articles de blog depuis une base de données, supprimer un commentaire) et de lui renvoyer les résultats (la liste des articles, si la suppression est réussie). Puis il va *adapter* ce résultat et le donner à la vue. Enfin, il va renvoyer la nouvelle page HTML, générée par la vue, à l’utilisateur. [3]

La figure suivante schématise le rôle de chacun de ces éléments :



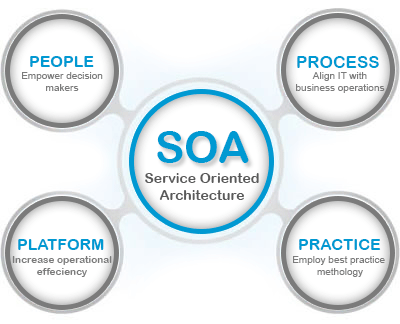
* Choix de l’architecture côté front-end (côte client) :
  1. L’architecture SOA :

Notre application est basée sur l’architecture orientée services (SOA) qui est une approche utilisée pour créer une architecture basée sur l’utilisation d’un ensemble des services.

Ces services sont conçus pour effectuer des tâches spécifiques et peuvent être appelés par d'autres composants de l'application ou par des applications externes via des protocoles de communication standardisés tels que HTTP, SOAP (Simple Object Access Protocol), ou REST (Representational State Transfer).

**SOA visant à :**

* La décomposition d'une fonctionnalité en un ensemble de services.
* L'optimisation des performances des organisations.
* La diminution des coûts.
* L'amélioration de la souplesse des processus.



* 1. Architecture de sécurité :
* **L’intérêt de développer d’une API pour un progiciel :**

L'API peut être décrite comme une solution informatique essentielle qui permet à différentes applications de communiquer entre elles de manière fluide, en échangeant des services ou des données. En réalité, elle se compose d'un ensemble de fonctions accessibles via des requêtes HTTP, facilitant ainsi l'accès aux services d'une application à travers un langage de programmation défini.

En simplifiant la communication entre deux programmes informatiques, l'API offre un moyen efficace de favoriser leur évolution tout en renforçant la distinction entre le Back-end et le Front-end. Cela simplifie également le développement d'applications web en permettant aux développeurs de créer une "surcouche" personnalisée, par exemple pour un progiciel commercial, ou en laissant aux membres la possibilité de coder des fonctionnalités supplémentaires selon leurs besoins spécifiques.

En somme, l'API joue un rôle central dans l'architecture logicielle moderne, favorisant la flexibilité, la modularité et l'efficacité des systèmes informatiques interconnectés.

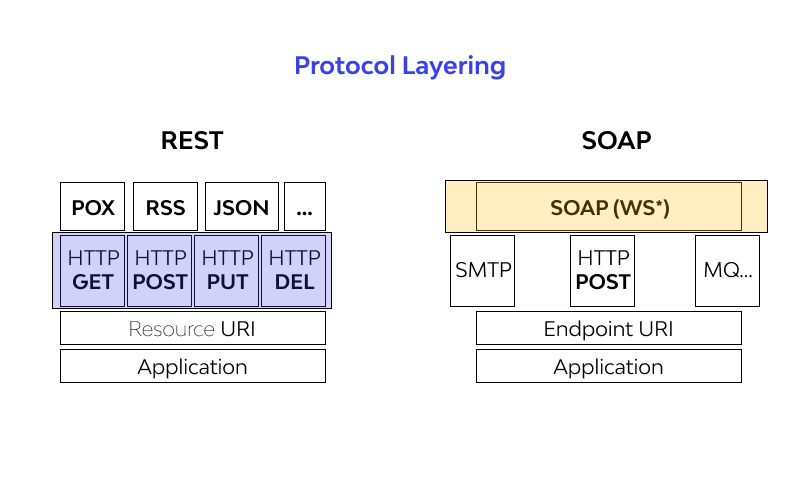
**▪ Les diverses structures d'API :**

Il existe deux principaux protocoles de communication sur lesquels reposent les API : le Simple Object Access Protocol (SOAP) et le Representational State Transfer (REST).

Le REST s'est progressivement imposé comme le choix privilégié par rapport au SOAP en raison de sa flexibilité accrue. Cette évolution a conduit à l'émergence des API REST ou RESTful.

La distinction fondamentale entre ces deux protocoles réside dans le niveau de couplage entre le client et le serveur.

La figure suivante montre la différence entre les 2 services :



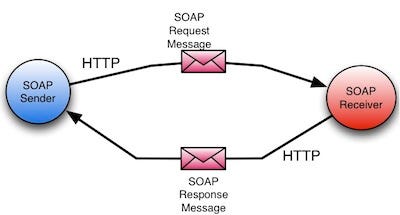
* **Modèle d'API basé sur SOAP :**

L'architecture d'API SOAP utilise le protocole SOAP (Simple Object Access Protocol) basé sur XML.

Les services sont décrits via des fichiers WSDL, assurant un contrôle élevé sur la sécurité et la fiabilité des communications, ce qui la rend adaptée aux environnements critiques.

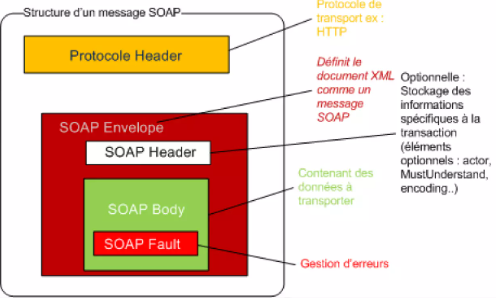
Cependant, cette architecture est souvent critiquée pour sa complexité et sa lourdeur par rapport à d'autres approches plus légères comme REST, ce qui limite son adoption dans certains cas d'utilisation.

La figure suivante présente la mode de communication :



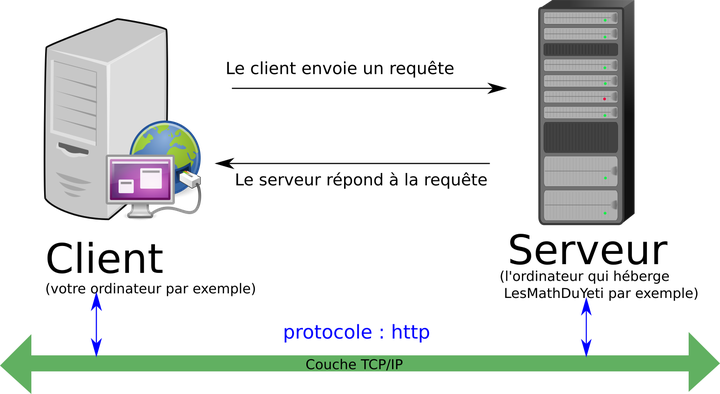
* **Structure d’un message SOAP :**

La figure suivante montre la structure d’un message SOAP :



* **Architecture d’API REST :**

L'architecture REST est un style d'architecture qui offre une approche plus légère que SOAP. Contrairement à SOAP qui utilise XML pour les requêtes, REST se base souvent sur des URLs simples. Bien qu'il puisse nécessiter des informations supplémentaires dans certains cas, la plupart des services web REST se concentrent sur la récupération des données nécessaires via des URLs spécifiques. Cette simplicité et cette approche centrée sur les ressources en font un choix attrayant pour de nombreux cas d'utilisation, offrant une communication efficace entre les clients et les serveurs avec une architecture flexible et évolutive.



* **L'architecture d'API utilisée :**

Nous avons choisi d'utiliser une architecture d'API REST avec Laravel pour notre backend pour plusieurs raisons clés. REST offre une approche légère et flexible, en accord avec la philosophie de développement rapide de Laravel. Elle est polyvalente pour de nombreux types d'échanges d'informations entre logiciels, utilisant uniquement le protocole HTTP. Cette API REST spéciale a été développée avec Laravel, permettant aux utilisateurs représentés par d'autres applications d'appeler notre API RESTful et d'obtenir des résultats. Cependant, étant donné la sensibilité des données échangées, cette API ne peut pas être publique sans sécurisation. Pour garantir la sécurité de notre API, nous avons mis en place une technique utilisant Laravel Passeport pour l'authentification et la gestion des jetons JWT (JSON Web Tokens). Cette approche assure un accès sécurisé à notre API REST de manière stateless, grâce à la gestion des autorisations par Laravel Passeport et l'utilisation des JWT pour l'authentification et l'autorisation des requêtes.

L'environnement de développement fait référence à l'ensemble des outils, des ressources et des configurations utilisés pour concevoir, coder, tester et déployer des logiciels. Cela inclut l'environnement logiciel ainsi que les technologies et langages utilisés .

* 1. Outils de conception :

|  |  |
| --- | --- |
| Outils | Description |
| **Visual paradigm**  visual | Est un logiciel de modélisation et de conception visuelle pour le développement logiciel et la gestion de projets. |
| **Microsoft Word** | Est un logiciel de traitement de texte par Microsoft, idéal pour créer et éditer des documents. |

* 1. Outils de développement :

|  |  |
| --- | --- |
| Outils | Description |
| **Visual Studio Code** | Est un éditeur de code source léger, extensible et multiplateforme |
| **HTML 5** | C’est un langage qui permet de composer des pages web. |
| **CSS3** | Est un langage de style qui contrôle la manière dont les documents HTML sont présentés, incluant des aspects comme les polices, les couleurs, et les marges. |
| **Bootstrap** | Est un framework front-end pour le développement web, offrant des outils et des modèles prêts à l'emploi pour créer des sites réactifs et mobiles. |
| **Postman** | Les applications modernes reposent largement sur des APIs. Celle-ci est considérée comme l'API la plus complète, bénéficiant de l'adhésion de plus de 100 000 entreprises à travers le monde. |
| **PhpMyAdmin** | C'est une application Web de gestion pour les bases de données MySQL, développée en PHP et distribuée sous licence GPL. Elle est réputée comme l'une des interfaces les plus populaires pour gérer efficacement une base de données MySQL sur un serveur PHP. |
| **Xampp** | Est un logiciel libre et gratuit qui facilite la création et la gestion de serveurs web locaux. |
| **PowerBI** | Power BI est une suite d'outils d'analyse de données et de visualisation développée par Microsoft. |

Dans la suite, nous donnerons plus de détails sur les principales technologies utilisées, à savoir

Angular 17, Laravel.

* 1. Environnements de développement utilisés :

[Grab your reader’s attention with a great quote from the document or use this space to emphasize a key point. To place this text box anywhere on the page, just drag it.]

