Conception de gestion des tâches

- Professeurs : Mr ROUSSILLE & Mr DIENY
- Etudiantes : KOUDJA BRENDA & NGUEGANG BRINET

Table des matières

1. Introduction

Ce document décrit la conception d'un système de gestion des tâches destiné à aider les individus et les équipes à organiser et à suivre leurs tâches de manière efficace. Notre objectif est de créer une solution robuste et intuitive qui réponde aux besoins des utilisateurs en matière de gestion de leurs charges de travail.

2. Objectifs

Le principal objectif de ce projet est de créer une application de gestion des tâches qui permette aux utilisateurs de créer, de suivre et de gérer leurs tâches de manière simple et organisée. Les fonctionnalités clés incluent :

- Création des tâches
- Suppression des tâches
- Modification des tâches

Le processus de création des tâches est le suivant :

- Entrée du nom de la tâche : l'utilisateur ajoute le nom de la tâche.
- Ajout d'une description : l'utilisateur a la possibilité d'ajouter une description détaillée de la tâche, incluant toutes les informations.
- Définition du statut : ici, l'utilisateur choisit le statut initial de la tâche parmi plusieurs options telles que « À faire », « En cours », « Terminée », « Annulée »
- Ajout d'une date limite à laquelle la tâche doit être effectuée.

3. Conception du projet

Nous allons parler des différents éléments pour faire notre projet de gestion de tâches.

❖ Développement

- Une machine virtuelle : dans notre cas, nous avons utilisé la machine (172.16.20.20) suivie du numéro de notre groupe Devops1.
- L'installation des différentes dépendances dans le projet, nous pouvons citer quelques-unes : Express, bycript.
- GitLab nous a permis le développement et la gestion du projet.

- Node.js :
- Docker :
- Postman : pour pouvoir tester nos différentes routes d'api.

❖ Devops

Pour le déploiement de l'application, nous avons suivi le document fourni par le professeur et nous avons réalisé cela étapes par étapes.

Comme logiciels utilisé :

- Jenkins : Il aide à automatiser les parties du développement logiciel liées au build, aux tests et au déploiement.
- Sonarqube : est un outil d'analyse statique qui a pour but de mesurer la qualité du code d'un applicatif.
- Jmeter : permet d'effectuer des tests de performance d'applications et de serveurs Grafana.
- Les machines virtuels (devops1, haproxy, backend1, backend2)
- Gitlab : nous a permis le développement et la gestion du projet.

4. Les étapes de conception de gestion de tâches

a) besoins

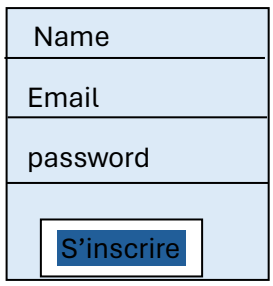
L'idée de mettre en place un projet de gestion de tâches découle d'un besoin, notamment, de simplifier notre vie quotidienne. Nous voulons créer un outil qui nous aide, en tant qu'utilisateurs, à mieux organiser nos tâches, à suivre notre progression et à rester concentrés sur ce qui compte vraiment.

b) Les différentes méthodes et actions effectuées

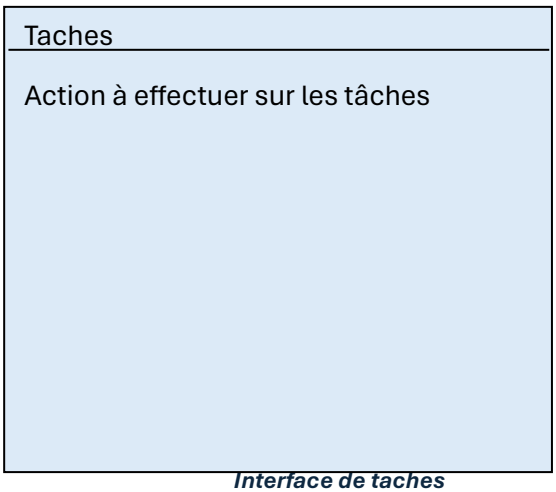
Pour commencer, nous avons tout d'abord identifié les différentes fonctionnalités de l'application, tout en identifiant les différentes classes qui devaient s'implémenter, la technologie à utiliser et bien d'autres.

C) Exemple de schéma pour le fonctionnement

Voici un exemplaire de marquette notre application.



Interface utilisateurs



Interface de taches

Figure 1:exemple interface

d)schéma de la base de données

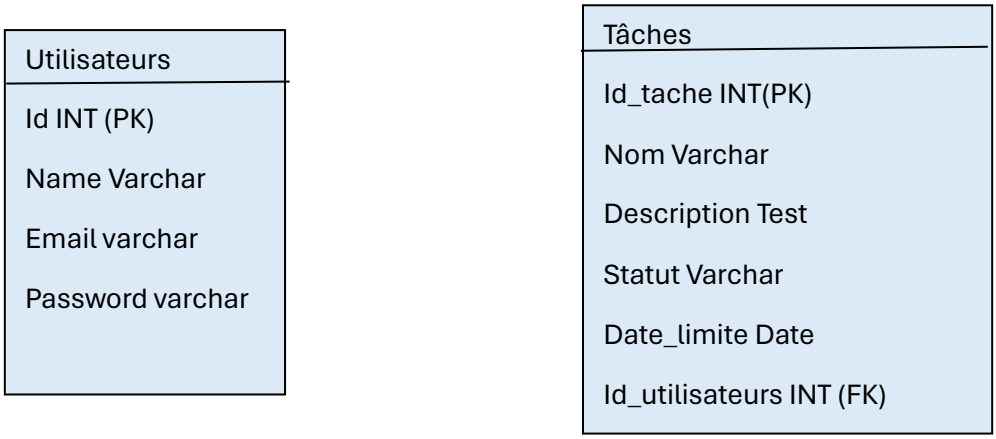


Figure 2:tables de la base de données

5.Difficultés rencontrées

Durant la conception du projet, nous avons rencontré plusieurs difficultés.

- La base de données Postgres (notamment l'ajout des différentes tables dans la base de données).
- Les différents soucis avec les ports lors de la construction des images de l'application principale, du frontend.
- Affichage de l'interface frontend sur le navigateur avec notre adresse.

- Les soucis sur notre code lors des tests.
- Interaction entre le backend et le frontend.

6.Les choix techniques de langage et d'architecture

1) langage

Pour la réalisation du projet, nous avons eu le choix de 3 langages de développement.

- RUST

Rust est un langage de programmation compilé multiparadigme qui met l'accent sur la performance, la sécurité des types et la concurrence. Il assure la sécurité mémoire, ce qui signifie que toutes les références pointent vers une mémoire valide, sans nécessiter l'utilisation de techniques de gestion de la mémoire automatisée telles que le ramasse-miettes. Afin

d'assurer la sécurité de la mémoire et d'empêcher une situation de complétion aux données, son vérificateur d'emprunts suit la durée de vie des objets de toutes les références dans un programme pendant la compilation.

- GO

Go est un langage de programmation compilé et concurrent inspiré de C et Pascal. Go veut faciliter et accélérer la programmation à grande échelle : en raison de sa simplicité, il est donc concevable de l'utiliser aussi bien pour écrire des applications, des script ou de grands systèmes. Cette simplicité est nécessaire aussi pour assurer la maintenance et l'évolution des programmes sur plusieurs générations de développeurs.

- NODEJS

Node.js est un environnement d'exécution open source et multiplateforme pour le langage JavaScript. Cela signifie que vous pouvez l'utiliser pour créer des applications serveur en JavaScript, des applications en temps réel, des outils de ligne de commande et bien plus encore.

Pour notre application de gestion de tâches, nous avons choisi comme langage Node. Pour quelques raisons :

- Les bases en Javascript : nous avons déjà quelques connaissances en Javascript.
- JS est événementiel et non bloquant : Node. JS utilise un modèle d'exécution par événements non bloquant, ce qui signifie qu'il peut traiter plusieurs requêtes en même temps sans se bloquer.
- JS est également rapide pour concevoir les applications évolutives.

2)Architectures

Nous avons utilisé 3 microservices pour le développement de l'application.

- Un microservice pour la base de données : dans notre cas, nous avons utilisé Postgres.
- Un microservice pour la backend représente notre application principale
- Un microservice pour le frontend.

7. Améliorations potentielles

Comme améliorations potentielles de l'application gestion de tâches, nous avons :

- **Priorisation des tâches** : mettre en place un système de priorisation des tâches avec des niveaux (par exemple, haut, moyen, bas) pour aider les utilisateurs à se concentrer sur les tâches les plus importantes.

- **Système de notification :** Mettre en place un système de notification pour informer les utilisateurs des tâches assignées, des échéances imminentes et des commentaires.
- **Recherche et filtrage avancés :** offrir des options de recherche et de filtrage avancées pour permettre aux utilisateurs de trouver rapidement des tâches spécifiques parmi de nombreux éléments.

- **Partage de listes de tâches :** permettre aux utilisateurs de partager des listes de tâches avec d'autres personnes.

8.Possibilité d'intégration dans un environnement professionnel

Pour une utilisation efficace dans un environnement professionnel, l'application de gestion de tâches doit offrir des fonctionnalités et des options qui répondent aux besoins spécifiques des entreprises et des équipes. Voici quelques points clés à prendre en compte pour l'intégration professionnelle :

- **Collaboration en équipe :** faciliter la collaboration entre les membres de l'équipe en permettant l'assignation de tâches, le partage de listes de tâches, les commentaires et les notifications.
- **Suivi de la progression et reporting :** Fournir des outils de suivi de la progression et de reporting pour mesurer l'avancement des tâches, la productivité individuelle et de l'équipe, et identifier les goulots d'étranglement potentiels.
- **Gestion des utilisateurs et des rôles :** offrir des fonctionnalités de gestion des utilisateurs et des rôles pour contrôler l'accès aux données et aux fonctionnalités en fonction des niveaux d'autorisation définis.

9. Analyse d'architecturale

Pour l'architecture à adapter pour le projet, nous avons utilisé un MVC. Pour analyser l'architecture MVC d'un projet de gestion de tâches, nous avons dû examiner les éléments suivants :

- Le modèle : dans cette partie, nous avons défini les différents champs de nos tables, notamment utilisateurs et taches. Nous avons utilisé un ORM Sequelize pour une rapidité dans notre travail.
- Routes : c'est ici que nous avons défini toutes les routes de notre API avec les différentes méthodes (DELETE, POST, PUT, GET).

- Controller : C'est dans cette partie que nous avons implémenté les différentes logiques de l'application.

❖ Avantages

- **Flexibilité et réutilisabilité : l**'architecture MVC est flexible et permet de réutiliser facilement des composants dans d'autres parties de l'application.
- **Maintenance simplifiée :** La séparation des couches rend la maintenance du code plus simple, car les modifications dans une couche n'affectent généralement pas les autres couches.
- **Facilité de développement :** de nombreux frameworks MVC populaires existent, fournissant des structures et des outils pour simplifier le développement d'applications Web.

❖ Inconvénients

- **Couplage entre les couches :** Bien que MVC encourage la séparation des préoccupations, il existe un certain couplage entre les couches, car le contrôleur doit interagir avec le modèle et la vue.
- **Difficulté de débogage :** les problèmes liés à l'interaction entre les couches peuvent être plus difficiles à déboguer, car les erreurs peuvent se manifester dans différentes parties du code.

❖ Extension de notre projet

Pour l'extension de gestion de projet nous pouvons avoir :

- **Ajout de nouvelles fonctionnalités :** De nouvelles fonctionnalités peuvent être implémentées en créant de nouveaux modules dans les couches modèle, contrôleur.

- **Intégration avec d'autres systèmes :** L'application peut être intégrée avec d'autres systèmes et d'autres fonctions en utilisant des API.

10.Conclusion

Le projet de gestion de tâches a pour objectif de fournir aux utilisateurs une plateforme conviviale pour organiser et suivre leurs tâches efficacement. Nous avons répondu aux besoins

des utilisateurs en intégrant des fonctionnalités telles que la création, la modification et la suppression de tâches.

Malgré les défis techniques rencontrés, nous avons surmonté ces obstacles grâce à une approche collaborative. En fin de compte, notre application représente une solution efficace pour améliorer la productivité des utilisateurs dans la gestion de leurs tâches quotidiennes.