CHABRIER Thibaut B3C3

Plan de test TODO IT Now

L'entreprise TODO IT Now présente aujourd'hui son nouveau produit : Un gestionnaire de tache destiné à faciliter la gestion d'équipe dans des projets de différentes envergures. Ce produit pourra donc être utilisé par des managers d'entreprises ou par des personnes ayant en charges une ou plusieurs équipe(s) à leurs dispositions. Le but de ce produit est de fluidifié la gestion d'équipes et permettre de centraliser les taches et les actions de chacun afin que le chef d'équipe puisse obtenir ces informations le plus simplement possible sans avoir à passer par plusieurs applications différentes pour gérer les taches et les emplois du temps des membres de son équipe. Touchant de manières générales toutes personnes comptant gérer un projet en équipes et voulant simplifier la charge managériale quant à la répartition du travail entres les différents membres du groupe. Cette application leur permettra différentes choses comme :

-Gérer l'emploi du temps de ses collaborateurs à un niveau mensuels et hebdomadaire

- Avoir accès à la TimeSheet des membres de son ou ses équipe(s) uniquement.

- Avoir accès à son propre emploi du temps qui sera gérer par son supérieur (N+1)

Pour le développement de l'application il est possible d'utiliser de nombres technologies afin de mettre en place le produit final :

Pour la partie Frontend nous pouvons choisir un Framework JavaScript comme VueJS, Angular ou React qui sont de très bon framework permettant de gérer le frontend de l'application avec une philosophie MVVM. Pour la partie design nous pouvons utiliser des frameworks comme Quasar, Taildwindcss ou encore Vuetify et Bootstrap qui fonctionnent parfaitement avec les framework front que nous avons évoqué plutôt, ils nous permettront d'avoir accès à des components et donc ne pas forcément avoir besoin de recréer tout from scratch comme des formulaires ou des tableaux. De plus ces frameworks inclus du style qui leur est propre et nous permettront d'éviter de faire du css from scratch la aussi cependant il sera surement nécessaire de faire du css lors de design plus complexe afin de respecter la charte graphique.

Pour la partie Authentification et rôle de l'application nous pouvons utiliser des JWT (Json Web Token) qui nous permettront de gérer l'authentification en générer des tokens valides lors de la connexion d'un utilisateur. Pour la gestion de rôle et en complément avec les JWT nous pouvons stocker le JWT dans un cookie http only ou dans le cache de notre navigateur afin de pouvoir réutiliser ce token dans tout notre site lors de requetes avec l'API. Il est aussi possible de stocker cela dans le store (Vuex ou Redux). En stockant le jwt nous pouvons aussi stocker une donnée permettant de déterminer le rôle de l'utilisateur comme un simple string de son grade. Cela permettra de données plus ou moins d'accès à cette personne après s'être connecté sur l'application.

Pour la partie serveur j'ai décidé de continuer dans la philosophie JavaScript et c'est pourquoi je propose d'utiliser NodeJS et ExpressJs afin de faire fonctionnement notre serveur, ce choix n'est du qu'a ce choix de garder le même langage. A préciser qu'il est possible de faire du TypeScript avec les technologies frontend de j'ai présenté ; Le typescript permettra de garder une syntaxe est une cohérence dans le typage que le JavaScript n'a pas forcément. Cela évitera de nombreuses erreurs.

Pour la base de données nous pouvons utiliser du MongoDB afin de rester dans un format proche du JSON mais il est possible d'utiliser du SQL (MariaDB) en fonction des préférences (n'étant pas expert dans ce domaine je ne me prononcerais pas).

Maintenant pour la partie téléphone deux choix s'offre à nous : Soit nous décidons de faire une PWA (Progress Web Application) et dans ce cas la nous n'avons pas besoin de faire une application téléphone a proprement parler. Soit nous décidons de faire une vrai application téléphone et dans ce cas la je propose d'utiliser Flutter car celui nous permettra d'avoir une application fonctionnelle a la fois sur Android mais aussi sur IOS ce qui nous évitera de faire deux applications mobile.

Enfin en bonus si nous devions faire une application Desktop alors j'utiliserais Electron.

Pour les outils de testing je compte utiliser Jest pour effectuer les tests unitaires et tests d'intégration (sans interfaces graphiques). Pour le reste j'utiliserais Selenium pour les tests de compatibilité, performance et intégration (avec interfaces graphiques).

Ces technologies sont assez larges et couvres l'ensembles du projet, c'est pourquoi les connaissances seront larges et demanderont plusieurs personnes pour être utilisées correctement. Pour la partie frontend il faudra connaitre la philosophie MVVM très utilisé dans les frameworks JS mais aussi comprendre le fonctionnement des stores, cookies et JWT dans le système d'authentification qui seront les piliers de l'application pour la gestion de permissions. Il faudra aussi avoir des connaissances en sécurité afin d'éviter des injections sql ou des injection AJAX dans les formulaires notamment. La partie design quand a elle va demander des connaissances en UIX afin de faire une interface claire et lisible par tous et pour tous le but étant d'avoir une application accessible peut importe ces connaissances avec l'informatique. Pour la partie Backend de l'application il faut savoir utilisé Node JS et tous les frameworks qui lui sont propres (ExpressJS par exemple) , ainsi que comprendre et connaitre le noSQL si utilisation de MongoDB afin d'optimiser les requetes faites dans la base de données (peut import le langage SQL comme NoSQL)

Ressources (physiques) :

Pour que le produit fonctionne il faudra au minimum :

- Une base de données : afin de pouvoir y mettre des informations importantes permettant à l'application de fonctionner correctement de répondre à sa demande. Par exemple les emplois du temps des membres d'une équipe
- Un serveur : qui permettra de faire le pont entre l'application et la Base de données et qui avec un système de rôle et d'authentification pourra donner seulement les informations nécessaires à un utilisateur en fonction de son rôle
- Téléphone (Android et IOS) et ordinateur : pour la partie client/ utilisateur, nous partons du principe que l'application sera disponible sur Android et ios et pour la partie ordinateur l'application web sera disponible sur tout type d'ordinateur peu importe le système d'exploitation . Cependant si une version desktop venait à être faite alors elle verra d'abord le jour sur Windows puis après sur Linux et Mac