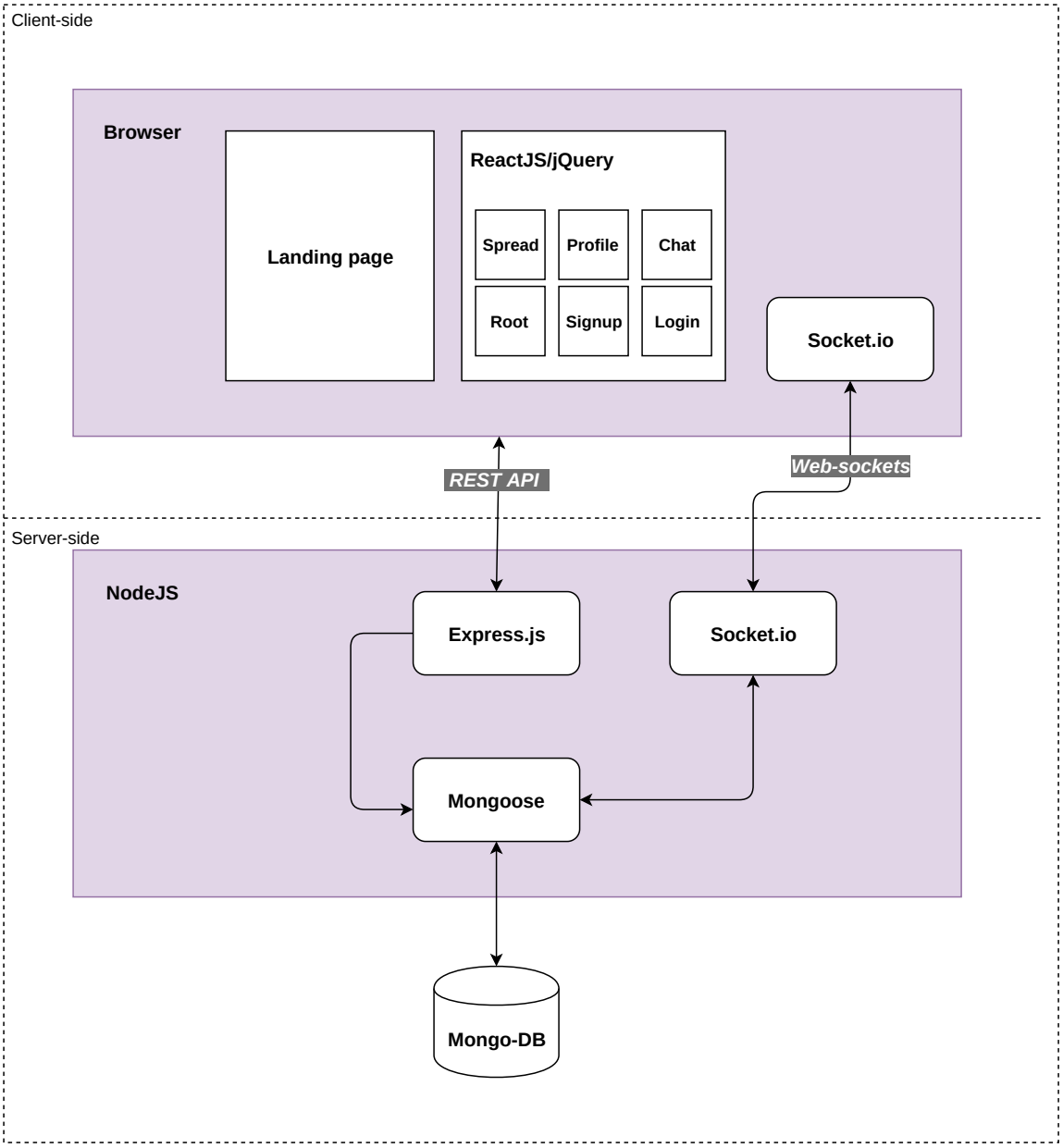
Architecture de l'Application



I) Principaux choix :

A) Server-side :

1) Mongo-DB :

C’est une base de données NoSQL et Open Source.
Justification :

- Présentation des données : les données sont modélisées sous forme de documents JSON, le système permet de faire évoluer le schéma de la base de données à la volée.
- Simplicité : le modèle de type document réduit au maximum le nombre de relation dans la base de donnée, ce qui simplifie sa structure et augmente sa lisibilité.
- Performance : le Sharding permet de répartir les données sur plusieurs serveurs soit pour simplement augmenter les performances ou pour répartir les données géographiquement.
- Fonction de recherche : MongoDB intègre un système de recherche optimisé en fonction de la langue utilisée.

2) Mongoose :

Mongoose est une bibliothèque ODM (Object Data Modeling) pour MongoDB et Node.js.
Justification :

- Il gère les relations entre les données, fournit la validation du schéma
- Il est utilisé pour la conversion entre les objets du code et la représentation de ces objets dans MongoDB.

3) Express.js :

C’est est une librairie minimale et flexible de Node.js qui fournit un ensemble robuste de fonctionnalités pour les applications Web et mobiles.
Justification :

- Avec une myriade de méthodes utilitaires HTTP et de middleware à votre disposition, la création d'une API robuste est simple et rapide.

- Express fournit une fine couche de fonctionnalités d'applications Web fondamentales, sans masquer les fonctionnalités de Node.js que vous connaissez et aimez.

4) Socket.io :

C’ est l'une des bibliothèques les plus prisées par ceux qui développent avec Node.js.

Justification :

- Il permet une communication en temps réel, bidirectionnelle et basée sur des événements.
- Il fonctionne sur toutes les plates-formes, navigateurs et appareils, en mettant l’accent sur la fiabilité et la rapidité.

B) Client-side :

1) ReactJs :

C’est est une bibliothèque JavaScript libre.

Justification :

- Il facilite la création d'application web monopage, via la création de composants dépendant d'un état et générant une page (ou portion) HTML à chaque changement d'état.
- Il se démarque de ses concurrents par sa flexibilité et ses performances, en travaillant avec un DOM virtuel et en ne mettant à jour le rendu dans le navigateur qu'en cas de nécessité.

2) jQuery :

C’est est un framework Javascript sous licence libre qui permet de faciliter des fonctionnalités communes de Javascript.

Justification :

- Manipulation du DOM (HTML ou CSS)
- Gestion des événements (clic, survol, soumettre un formulaire ...)
- AJAX
- Effet d'animation

II) Plan de travail :

1) Configuration de l'environnement :

- Faire l'architecture de l'application.
- Configuration de l'environnement initial pour le développement de l'application, qui comprend Node.js,a MongoDB, React.js, ***Webpack 3, Babel et Yarn.***

2) Création de premières pages et des end-points de l'API de base :

- Créer une simple page d'accueil et ajouter des écrans d'inscription et de connexion.
- Développer les quelques premiers composants React.js et brancher un paquet appelé «react-router» pour permettre le routage d'URL vers notre application Web.
- Création des premiers end-points d'API pour l'enregistrement et l'authentification d'utilisateurs sans base de données.

3) Créer des pages de profil et MongoDB :

- Connexion de l'API pour utiliser MongoDB pour stocker les informations des utilisateurs.
- Création d'un schéma d'utilisateur avec Mongoose pour MongoDB et création de pages de profil d'utilisateur.

4) Implémentation de l'authentification à l'aide de Passport.js :

- Implémentation de l'authentification JWT à l'aide de Passport.js pour permettre l'identification des utilisateurs.
- Création de l'authentification
- Connecter un modèle d'utilisateur avec l'API / Register, le point de terminaison d'API Add / Register, l'écran Create UserProfile et la route Create/me
- Extraction des données utilisateur des jetons JWS

5) Créer une page de profil :

- Créer une page de profil qui affichera des informations de base sur un utilisateur particulier. Nous allons stocker les informations de l'utilisateur côté client.

6) Ajout de fonctionnalités d'enregistrement :

- Ajouter des fonctionnalités d'enregistrement à l'application et inclure la refactorisation dans le code React.js pour optimiser les processus futurs.
- Mise en œuvre de la phase d'inscription

7) Capacité des utilisateurs à travailler avec d'autres :

- Ajout de fonctionnalités répertoriant les utilisateurs et affichant leurs pages de profil les unes aux autres;
- Implémentation de la fonctionnalité «upvote / downvote» qui suit comment les utilisateurs s’apprécient, comme le font les sites de rencontres.

8) Implementer un algorithme de correspondance :

- Concevoir et mettre en œuvre un algorithme de correspondance simple pour permettre aux utilisateurs de se retrouver facilement - c’est la base même d’une application de datation, non?
- Création d'une interface utilisateur pour permettre aux utilisateurs d'interagir les uns avec les autres.

9) Ajout de WebSockets à l'aide de socket.io :

- Ajout du support WebSockets de base à l'application de datation à l'aide du package socket.io sur le backend. WebSockets permettra aux utilisateurs de communiquer instantanément les uns avec les autres.
- Ajout de nouveaux composants React.js pour l'interface utilisateur sur le frontend.

10) Ajout d’un template bootstrap :

- Choisir un template pour le dating website
- Introduire du jQuery pour dynamiser les pages

III) Les fonctionnalités :

NB: des fonctionnalités non pas été finis à cause de ma mal organisation et manque de temps.

1) Facilité de configuration et debuggage :

- Webpack-dev-server : re-lance automatiquement la configuration du serveur une fois un changement a été détecté et compress tous les fichiers js dans un seul fichier (bundle.js) pour que le client n’aura qu’un seul fichier à loader.
- Yarn : c'est un gestionnaire de paquets qui utilise le registre NPM comme serveur principal. Yarn crée un fichier (yarn.lock). Ce fichier stocke les versions exactes des dépendances jusqu'au dernier chiffre. Lors de l’installation, Yarn vérifie d’abord le fichier de verrouillage pour connaître les versions, puis exécute le script dans (package.json)
- Babel : c’est est un transpiler (convertisseur et compilateur) JS qui convertit le nouveau code JS en ancien. C'est un outil très flexible en termes de transpiling. Aide au débogage des code Js.

2) Formulaire d’inscription :

- Le formulaire actuelle : Nom, Mail, Mot de passe.
- Le formulaire finale : Nom, Mail, Age, genre, Interet, Photo de profile, Mot de passe.

3) Formulaire de login :

- Le formulaire contient : Mail et Mot de passe pour s’authentifier.

4) Sécurité de la base de donnée :

- Stocker les hashes des mot de passe au lieu du plaintext, en utilisant un sel aléatoire et l'algorithme SHA512.

5) Gestion de session et protection des répertoires :

- Utilisation de JWT (JSON Web Tokens) middleware pour accorder à chaque utilisateur authentifié un token valide d'une heure.
- Protection des répertoires qui demande de l'authentification avant l'accès.

6) Gestion de profile :

- Gestion actuelle :
  - possibilité de voir les informations de son profile et le profile des autres utilisateurs .
  - possibilité de liker ou disliker un autre utilisateur
- Gestion finale : possibilités de faire des changements de nom, mail, genre, interet, photo de profile.

7) Algorithme de correspondance :

- Le schéma utilisateur contient un dictionnaire de likes où la clé est l'ID d'un autre utilisateur et la valeur est le nombre de likes ou dislikes.
- Le système de point est intégré : chaque utilisateur à une note qui dépend de nombre de likes et dislikes.
- La suggestion d'utilisateurs selon un algorithme :
  Suivant la logique " Les amis de mes amis sont mes amis "

8) Messagerie :

- Messagerie actuelle : de type broadcast, tout le monde parle avec tout le monde.
- Messagerie finale : parler avec des personnes spécifiques qu'on a déjà liké et ils ont liké back

Réferences :

- Les outils : https://medium.com/front-end-weekly/what-are-npm-yarn-babel-and-webpack-and-how-to-properly-use-them-d835a758f987
- Documentation : https://docs.npmjs.com/ https://reactjs.org/tutorial/tutorial.html
- La logique de travaille est la décomposition de l'application : https://www.education-ecosystem.com/kuzzmi/lmnAV-how-to-create-a-dating-web-app-in-nodejs/Dy1LK-how-to-create-a-dating-web-app-in-nodejs-4/