DOCUMENT DE DÉFINITION D’ARCHITECTURE

Table des matières

1 Objet du document

Ce document permet de modéliser l’architecture qui sera réalisée par les équipes de développement.

Au-delà de la modélisation graphique, il s’agit également d’énoncer les principes sur lesquels l’architecture s'appuie, de justifier cette approche architecturale, mais également d’indiquer des transitions si nécessaire.

2 Objectif du projet

L’objectif de la nouvelle application est de s’adapter à l’évolution du marché qui se développe vers l’international.

Cette croissance implique l’implémentation de fonctionnalités sous forme d’actions utilisateurs.

3 Principes d’architecture

L’architecture est développée en trois composants qui constituent le cœur de l’application.

Notre premier constituant est le Front-end qui va contenir notre Interface Homme Machine (IHM) permettant à l’utilisateur d’effectuer les actions implémentés.

Ce Front-end est accompagné d’un Back-end.

C’est ce dernier qui fait l’interface entre l’IHM Front et la base de données. Il gère tout l’aspect technique des requêtes transmises par le Front-end via le protocole HTTPS.

Enfin la base de données sert à contenir les informations souhaitées ainsi que de créer un serveur autorisant l’accès aux données via des requêtes Sql envoyées par le Back-end.

4 Architecture existante

Le projet ne s'appuie pas sur un produit existant qui serait repris puis mis à jour. Il n’y a donc pas d’architecture existante à définir.

5 Architecture

Dans cette partie nous allons détailler l'architecture de notre application sous plusieurs angles à savoir un premier fonctionnel, un second autour des données et enfin technologique.

Nous allons développer une architecture orientée services qui permettra d'assurer l'utilisation de l'application à un grand nombre d'utilisateurs, point en accord avec la croissance à l'international de l'application.

5.1 Point de vue fonctionnel

Les fonctionnalités se décrivent en trois points à savoir la gestion de profil, la gestion de location ainsi que la possibilité de contacter le support. Elles concernent toutes l'utilisateur qui est l'élément central de l'application.

Notre première donne à l'utilisateur la capacité de créer, gérer et supprimer son profil. Il lui est aussi possible de rentrer en contact avec le support.

Notre seconde permet à l'utilisateur de rechercher selon un modèle spécifique ou par une région donnée. Il peut aussi consulter l'ensemble des agences de locations ainsi qu'afficher une offre ou y souscrire.

Enfin, ce même utilisateur doit pouvoir contacter le support dans le cadre de la résolution d'un problème rencontré dans le processus de réservation ou tout au long de la location.

5.2 Point de vue lié au données

L'architecture des données s'articule en plusieurs sous points qui ensemble constituent la base de données.

Tout d'abord nous avons les données relatives à **l'utilisateur** qui contienne nom, prénom, date de naissance ainsi que l'adresse.

Ensuite un **véhicule** qui lui est associé à une catégorie, un modèle et une marque particulière.

Par la suite, une **catégorie** qui elle contient le type de véhicule ainsi que son coût journalier.

Puis l'**agence** constituée d'une adresse ainsi que de la liste des véhicules disponibles lors de la consultation du client.

Pour que l'utilisateur puisse contacter le support, il est nécessaire de contenir les **messages** en base.

Ces derniers seront relatifs à un ID utilisateur, une date, un contenu ainsi qu'un titre dans le cadre d'un contact par message non instantané.

Enfin, la **location** qui contient le nom d'une ville de départ, d'une ville d'arrivée, d'une date et heure de départ, et de son équivalent pour l'arrivée, une catégorie qui lui est rattachée ainsi qu'un coût.

Toutes ces informations peuvent être résumées dans le schéma ci-dessous :

user
id
firstname
lastname
birthDate
adress

vehicule
id
category_id*
Model
brand

category
id
type
cost

agency
id
name
adress
vehicules*

message
id
user_id*
content
title
date

rental
id
DepartureCity
ArrivalCity
DepartureDate
ArrivalDate
category_id*

LÉGENDE
integer
string
Foreign key : *

5.3 Point de vue technologique

Avec le prisme technologique, notre architecture peut se schématiser comme ci-dessous.

Ce schéma fait suite aux explications données dans la partie 3.

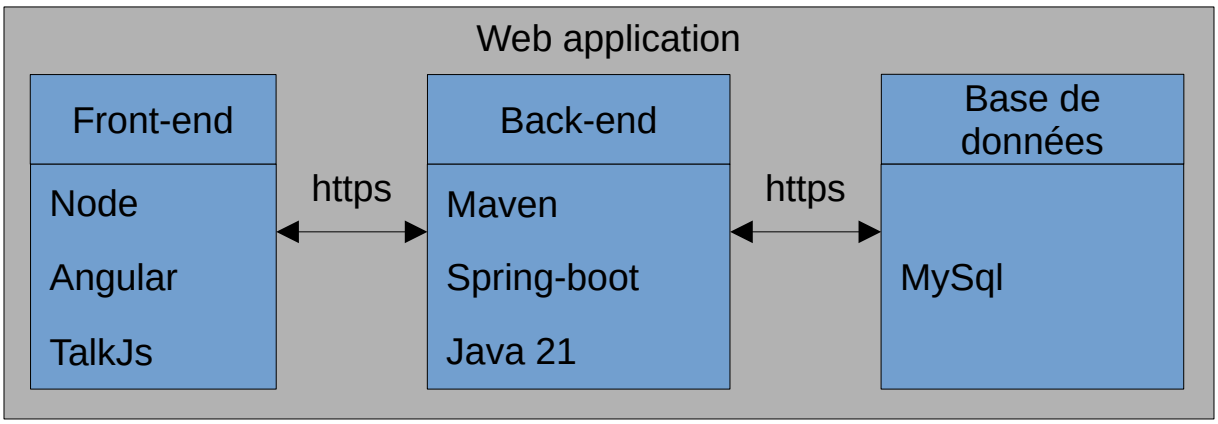

Figure 1: Schéma de l'architecture technologique

Le Front-end repose sur trois éléments clés : Node.js, utilisé pour l'installation et la compilation de l'application, Angular, un framework permettant de concevoir l'interface utilisateur (IHM) de l'application, et TalkJs, une API pré-construite implémentant une fonctionnalité de chat en ligne.

Le Front-end sera une Single Page Application (SPA) ce qui permettra d'avoir une interface légère et rapide favorisant l'utilisation de cette dernière.

Le back-end s'articule lui en trois composants : Maven, utilisé pour l'installation et la compilation de l'application, Spring-boot, un framework permettant de concevoir l'API back-end, et Java, le langage de programmation.

En troisième point, notre base de données utilise MySql pour gérer l'ensemble des données de l'application.

Ensuite, le développement des deux composantes sera régit par deux points essentiels.

Le Règlement Général de Protection des Données (RGPD) qui permettra ainsi à l'utilisateur de garder le contrôle de ses données personnelles.

Point de vue sécurité, cette réglementation impose un aspect sécurité pour lequel nous mettrons en place le protocole HTTPS pour l'ensemble des échanges de données entre le front & back ou back & base de données.

La bonne pratique Modèle Vue Contrôleur (MVC) aura pour but d'améliorer la maintenabilité de l'application.

Cette dernière sera appliquée en parallèle des principes S.O.L.I.D. qui eux rendront le code plus facilement compréhensible dans le cadre d'un maintien ou d'un changement d'équipe.

En ouverture pour la suite du projet, il sera possible de produire trois images distinctes avec docker par exemple dans le but de les héberger sur un répertoire distant tel que dockerHub.

6 Justification de l'approche architecturale

L'utilisation de l'API TalkJs va venir dé-risquer la complexité à créer un chat en ligne pour combler le besoin client sur la possibilité de contact avec le support.

Indépendamment des autres, chaque brique logicielle est reconnue et dispose d'une large communauté active qui permettrait de venir pallier à un potentiel blocage rencontré lors du développement de l'application.

Avec le respect des bonnes pratiques M.V.C. et S.O.L.I.D., l'architecture développée permettra de répondre à l'évolution croissante du marché international de l'application **Your Car Your Way** puisque cette dernière est la base architecturale de plusieurs modèles comme par exemple l'architecture orientée service ou encore client-serveur.