Abdul Kévin Alexis

Analyse comparative entre les frameworks uniques Demande de page (SPA)

Belo Horizonte

2023

Abdul Kévin Alexis

Analyse comparative entre les frameworks Single Page Demande (SPA)

Travaux de fin de cours présentés au cours de génie informatique du Centre fédéral d'enseignement technologique du Minas Gerais, comme condition partielle pour l'obtention du baccalauréat en génie informatique.

Centre fédéral d'enseignement technologique du Minas Gerais – CEFET-MG

Département d'informatique

Cours de génie informatique

Conseiller : Pr. Dr Edson Marchetti da Silva

Belo Horizonte

2023

Centre fédéral d'enseignement technologique du Minas Gerais

Cours de génie informatique

Évaluation du travail de fin de cours

Étudiant : Abdul Kevin Alexis

Titre du poste : Analyse comparative entre les frameworks d'applications à page unique (SPA)

Date de soutenance : 12/06/2023

Heure : 08h00

Localisation de la défense :

https://conferenciaweb.rnp.br/conference/rooms/edson-marchetti-da-silva-2/invite?showsuggestion=false

Ce travail de fin de cours a été évalué par le jury suivant :

Professeur Edson Marchetti da Silva – Conseiller
Département d'informatique
Centre fédéral d'enseignement technologique du Minas Gerais

Professeur Kecia Aline Marques Ferreira – Membre du jury d'évaluation
Département d'informatique
Centre fédéral d'enseignement technologique du Minas Gerais

Professeur Eduardo Cunha Campos – Membre du jury d'évaluation
Département d'informatique
Centre fédéral d'enseignement technologique du Minas Gerais

Je dédie ce travail à tous ceux qui travaillent dans le domaine du génie informatique ou du logiciel.

Merci

Tout d'abord, je tiens à remercier mon encadrant et les personnes qui ont contribué à ma formation.

Ensuite, je remercie le peuple brésilien, ce pays qui m'a accueilli et soutenu tout au long de mon séjour ici. Je suis très reconnaissant pour l'opportunité et la croissance au cours mes études.

Je tiens également à remercier ceux qui m'ont permis de voyager au Brésil. Sans votre aide précieuse, je n'aurais pas pu réaliser mon rêve.

Remerciement spécial à:

- Esther Fleurijuste, ma chère, pour votre soutien inconditionnel;
- Samuel Michel, pour son aide dans l'obtention de la bourse ;
- Marie Alice Sirène, ma marraine, pour avoir toujours cru en moi ;
- Nicolas Dorvilus, qui repose en paix, pour son soutien et son soutien inconditionnels ;
- Beat Lockslina Edmond, pour sa générosité et sa solidarité ;
- Guerlovecia Pierre, pour son aide à la correction ;
- Père Benh Donais (CJMS), pour ses encouragements et ses conseils ;
- Socolavim (Fils-Aimé) et DGS Communication (Gary Dalencourt), pour un soutien financier ceiro;
- Carline Alexis, Dor Erlens, Rose Darline Camille, Maxime Termitus, @BUC (Laf, Mahatma, AMI, 2tay, Sly. . .), je vous suis très reconnaissante de faire partie de ce voyage. Votre soutien m'a été essentiel pour atteindre mes objectifs.

Merci!

« Croyance + Persévérance = Réalisation ou Succès »

(Propre paternité)

Résumé

Ce travail présente les résultats d'une étude expérimentale réalisée pour comparer trois frameworks de développement front-end : Angular, React et Vue. Les frameworks d'application à page unique (SPA) sont l'une des principales tendances du développement Web actuel . Ils permettent de créer des applications Web qui chargent une seule page, ce qui offre une expérience utilisateur plus fluide et réactive. Avec la prolifération des frameworks SPA disponibles, choisir celui idéal pour un projet donné peut être une tâche difficile. L'objectif de ce travail est de réaliser une analyse comparative entre les frameworks Angular, React et Vue , en considérant la taille du package, le temps de chargement, le temps de rendu, l'utilisation de la mémoire et le temps d'exécution. En tant que méthode, une application basée sur une architecture de microservices a été créée en utilisant chacune des alternatives, puis ces applications ont été comparées pour évaluer laquelle présente les meilleures performances. L'analyse comparative a révélé que React présente les meilleures performances globales, suivi de Vue et Angular. En termes de taille de bundle généré pour la mise en production, Vue possède le plus petit bundle, React le deuxième plus petit et Angular le plus grand des trois. En termes de temps de chargement, React et Vue sont les plus rapides, suivis d'Angular. En termes de temps de rendu, Angular et React, qui ont des performances similaires, rendent le rendu plus rapide, suivis de Vue, dont le rendu est plus lent. En termes d'utilisation de la mémoire, Vue utilise moins de mémoire, React arrive en deuxième position suivi d'Angular qui utilise le plus de mémoire. En termes de temps d'exécution, React a les meilleures performances dans toutes les opérations CRUD, suivi de Vue et Angular. Sur la base des résultats obtenus, React ou Vue ont été recommandés comme option pour les applications qui doivent se charger rapidement. Pour les applications qui nécessitent moins de m Pour les applications qui doivent bien fonctionner dans les opérations CRUD, React est la meilleure option. Les résultats de cette étude contrastent avec ceux de Kaluža, Troskot et Vukelić (2018), qui ont conclu qu'il n'existe pas de cadre JavaScript idéal pour MPA et SPA. Ziani (2021) a également réalisé une analyse comparative des frameworks JavaScript, mais son étude était qualitative, alors que la présente étude est quantitative. Par conséquent, les résultats de cette étude corroborent la déclaration de Diniz-Junior et al. (2022), qu'Angular a de meilleures performances en termes de rendu. Les résultats de cette étude sont pertinents pour les ingénieurs logiciels débutants car ils fournissent des informations précieuses pour choisir la meilleure technologie SPA pour les logiciels basés sur une arch microservice.

Mots-clés : cadres. SPA. Développement Web.Angular. Réagir. Vue

Abstrait

Ce travail présente les résultats d'une étude expérimentale réalisée pour comparer trois frameworks de développement front-end : Angular, React et Vue. Les frameworks SPA (Single Page Application) sont l'une des principales tendances du développement Web actuel. Ils permettent de créer des applications Web qui chargent une seule page, ce qui offre une expérience utilisateur plus fluide et réactive. Avec la prolifération des frameworks SPA disponibles, choisir celui idéal pour un projet donné peut être une tâche difficile. L'objectif de ce travail est de réaliser une analyse comparative entre les frameworks Angular, React et Vue, en considérant la taille du package, le temps de chargement, le temps de rendu, l'utilisation de la mémoire et le temps d'exécution. En tant que méthode, une application basée sur une architecture de microservices a été créée en utilisant chacune des alternatives, puis ces applications ont été comparées pour évaluer laquelle présente les meilleures performances. L'analyse comparative a révélé que React présente les meilleures performances globales, suivi de Vue et Angular. En termes de taille de bundle généré pour la mise en production, Vue a le plus petit tbundle, React a le deuxième plus petit et Angular a le plus grand des trois. En termes de temps de chargement, React et Vue sont les plus rapides, suivis d'Angular. En termes de temps de rendu, Angular et React, qui ont des performances similaires, rendent le rendu plus rapide, suivis de Vue, dont le rendu est plus lent. En termes d'utilisation de la mémoire, Vue utilise moins de mémoire, React arrive en deuxième position suivi d'Angular qui utilise le plus de mémoire. En termes de temps d'exécution, React a les meilleures performances dans toutes les opérations CRUD, suivi de Vue et Angular. Sur la base des résultats obtenus, React ou Vue ont été recommandés comme option pour les applications qui doivent se charger rapidement. Pour les applications qui nécessitent moins de mémoire, Vue est la meilleure option. Pour les applications qui doivent bien fonctionner dans les opérations CRUD, React est la meilleure option. Les résultats de cette étude contrastent avec ceux de Kaluža, Troskot et Vukelić (2018), qui ont conclu qu'il n'existe pas de cadre JavaScript idéal pour MPA et SPA. Ziani (2021) a également réalisé une analyse comparative des frameworks JavaScript, mais son étude était qualitative, alors que la présente étude est quantitative. Par conséquent, les résultats de cette étude corroborent la déclaration de Diniz-Junior et al. (2022), qu'Angular présente de meilleures performances en termes de rendu. Les résultats de cette étude sont pertinents pour les ingénieurs logiciels débutants, car ils fournissent des informations précieuses pour choisir la meilleure technologie SPA po
sur l'architecture des microservices.

Mots-clés : cadres. SPA. Développement Web.Angular. Réagir. Vue

Liste des illustrations

Liste des tableaux

Liste des abréviations et acronymes

AHP	Processus de hiérarchie analytique
API	Interface de programmation d'applications
CPU	Unité centrale de traitement
CRUD	Créer, lire, mettre à jour, supprimer
SOLEIL	Modèle d'objet de document
HTML	Langage Signalétique Hyper Text
EDI	Environnement de développement intégré
AMP	Demande multipage
REPOS	Transfert d'État représentatif
SI LA	optimisation du moteur de recherche
SGBD	Système de gestion de base de données
SPA	Demande d'une seule page

résumé

1. Introduction

Actuellement, le monde assiste à un processus d'évolution rapide des besoins en matière d'infrastructures d'information en termes de quantité, de qualité et d'accès. Les gens sont plus connectés partout dans le monde et peuvent travailler avec des communautés en dehors de leurs réseaux habituels (NEWMAN et al., 2016).

Compte tenu de ces faits, les développeurs doivent choisir, parmi les différents architectures et technologies existantes, qui correspondent le mieux à vos exigences dans chaque cas. Selon Kornienko, Mishina et Melnikov (2021), les solutions Web sont préférées aux plateformes mobiles ou de bureau lors de la création de nouveaux services. Ce choix est justifié par le fait qu'aucune ressource en ligne n'impose à l'utilisateur des exigences matérielles strictes (mémoire de stockage, mémoire vive, consommation électrique de l' unité centrale), ne lie pas l'interaction à un type spécifique de plateforme (personnelle ordinateur, smartphone, appareil spécialisé). En d'autres termes, les applications Web sont accessibles à l'aide d'un simple navigateur Web.

Selon sa conception, une application Web peut être développée sous forme d' application multi- pages (MPA) ou d'application monopage (SPA) (KALUŽA ; TROSKOT ; VUKELIĆ, 2018). MPA est le modèle conventionnel de création de logiciels frontaux, dans lequel le navigateur affiche une page entière pour chaque itinéraire ou onglet de l'application afin de présenter les données. Généralement, les données sont restituées côté serveur et un seul document HTML est envoyé côté client avec chaque requête. Cependant, SPA est une combinaison de solutions, divisant une application web en deux parties (front-end et back-end) et organisant l'interaction entre elles. La partie backend gère la logique, traite les requêtes, travaille avec la base de données, tandis que la partie frontend forme la vue externe basée sur les données reçues du backend et affiche le résultat dans le L'échange d'informations entre les parties client et serveur de l'application s'effectue généralement via une interface de programmation d'application spécialement développée. Les applications Web modernes , qui ne sont pas développées de manière traditionnelle, sont basées sur le concept SPA.

Selon W3Techs (2022), JavaScript est utilisé comme langage de programmation côté client par 97,9 % de tous les sites Web. Comme il existe de nombreux frameworks JavaScript disponibles, en choisir un pour réaliser un projet peut être une tâche très difficile. Dans ce contexte, selon StackOverflow (2021), les alternatives de développement les plus utilisées sont React.js, JQuery, Express, Angular, Vue.js. Dès lors, quels critères adopter pour choisir un framework parmi les plus utilisés ?

D'après les conclusions présentées dans les travaux de Ziani (2021), il n'y a pas

réponse catégorique fiable et unanimement acceptée sur le meilleur framework JavaScript à utiliser lors du développement d'une application Web frontale. De même, Ferreira et Zuchi (2018) ont conclu que chaque framework peut apporter un certain avantage au projet, et qu'il appartient au développeur de définir quelles caractéristiques sont les plus importantes pour son application.

Il existe plusieurs alternatives SPA différentes, chacune avec ses propres forces et faiblesses, ce qui signifie que chaque framework possède son propre ensemble unique de fonctionnalités et de capacités. Angular, React et Vue sont trois des frameworks JavaScript les plus populaires pour le développement d'applications monopage. Ils sont tous basés sur des composants, ce qui facilite la modularisation et la maintenance du code. De plus, ils offrent tous une large gamme de fonctionnalités et de bibliothèques, ce qui les rend adaptés à une variété d'applications.

Le choix de ces trois cadres pour une analyse comparative se justifie par la les raisons suivantes:

- pertinence : ces frameworks sont les plus utilisés sur le marché, ce qui garantit que les résultats d'analyse sont pertinents pour un large éventail de professionnels ;
- diversité : les cadres présentent différentes approches de développement, qui ce qui permet une analyse plus complète ;
- contribution potentielle : l'analyse comparative peut fournir des informations précieuses aux ingénieurs logiciels qui évaluent le framework à utiliser pour leurs projets.

Par conséquent, faire le meilleur choix pour un projet donné dépendra des exigences spécifiques.

1.1 Objectif

Le but de ce travail est de réaliser une analyse comparative entre la plateforme SPA Angular, la bibliothèque React et le framework Vue.1 L'analyse comparative des outils de développement SPA peut être réalisée en utilisant diverses méthodes. Une méthode courante consiste à créer un tableau qui compare les différentes alternatives testées en fonction de divers critères, tels que : la facilité d'utilisation, les performances et les fonctionnalités. Une autre méthode consiste à créer une application de démonstration pour chacune des alternatives testées, puis à comparer les applications pour voir laquelle fonctionne le mieux

[1] Dans le cadre de ce travail, il convient de souligner la complexité inhérente à la recherche d'un terme consensuel qui englobe les trois technologies SPA : la plateforme Angular, la bibliothèque React et le framework Vue. Afin de simplifier la communication et d'éviter les ambiguïtés, nous adopterons le terme « framework » de manière générique pour référencer ces technologies tout au long du document.

C'est pourquoi une application a été implémentée dans une architecture de microservices en utilisant chacune de ces alternatives dans le développement du front-end. À partir de ces applications développées, des tests ont été effectués pour mesurer la taille du code généré, le temps de chargement, en plus de mesurer pour différents scénarios de chargement, le temps de rendu, l'utilisation de la mémoire et le temps d'exécution dans chaque cas.

1.2 Justification

La réalisation de ce travail est justifiée car une analyse comparative des frameworks SPA peut être un outil précieux pour les développeurs qui envisagent d'utiliser un outil SPA pour un nouveau projet. En comparant différentes alternatives, les développeurs peuvent identifier celle qui répond le mieux aux besoins de leur projet.

Vous trouverez ci-dessous quelques-uns des avantages apportés par cette analyse comparative cherché à faire :

- aider les développeurs à identifier une alternative de développement front-end SPA qui répond le mieux aux besoins de votre projet;
- aider les développeurs à économiser du temps et de l'argent, en évitant le besoin essayer différentes alternatives ;
- aider les développeurs à choisir un framework supporté par une large communauté d'utilisateurs ;
- aider les jeunes diplômés ou les débutants à choisir la meilleure option, parmi les différentes alternatives existantes, pour étudier et se développer.

Dans l'ensemble, une analyse comparative des frameworks SPA est un outil précieux pour les développeurs qui envisagent d'utiliser un framework SPA pour un nouveau projet. En effectuant cette analyse, les développeurs peuvent être sûrs qu'ils font le meilleur choix pour leurs besoins et qu'ils sont bien informés sur les différents outils et comment ils fonctionnent.

1.3 Structure de travail

Le reste de ce travail est organisé comme suit. Chapitre 2 présente le cadre théorique. Le chapitre 3 traite des travaux connexes. Le chapitre 4 discute de la méthodologie proposée. Le chapitre 5 présente les résultats, tandis que le chapitre 6 présente les conclusions.

2 Cadre théorique

Ce chapitre présente le cadre théorique qui sous-tend cette recherche, fournissant les bases conceptuelles et théoriques nécessaires à la compréhension du sujet étudié. Les principaux concepts, théories et études qui soutiennent l'analyse seront abordés.

2.1 Les débuts du Web

Il est très courant d'entendre les noms Google, Facebook, Twitter, Instagram. . . , etc. avant d'entendre l'expression « réseau social ». Après tout, ces géants de la technologie ont stimulé la recherche d'informations et les interactions entre les gens. En raison de la nécessité d' une communication continue pour rester connecté à tout ce qui se passe, il est difficile de vivre sans ces technologies dans le monde d'aujourd'hui. Ce sont des outils indispensables pour réaliser des interactions personnelles et professionnelles, faisant partie de notre vie quotidienne. Et tout a commencé avec une technologie appelée Web dans les années 1980.

Selon Berners-Lee et al. (1994), le Web représente une série de choses qui il faut distinguer. Ces éléments incluent :

– l'idée d'un monde d'informations illimitées dans lequel tous les éléments ont une référence par laquelle ils peuvent être récupérés ;

– le système d'adresses (URI) que le projet a mis en œuvre pour créer ce monde possible malgré de nombreux protocoles différents ;

– un protocole réseau (HTTP) utilisé par les serveurs web natifs proposant engagement et ressources non disponibles autrement ;

– un langage de balisage (HTML) que chaque client W3 doit comprendre et qui est utilisé pour transmettre des éléments de base tels que du texte, des menus et de simples informations d'aide en ligne sur le réseau ;

– l'ensemble de données disponible sur Internet utilisant tout ou partie des éléments répertoriés au-dessus de.

Berners-Lee est considéré comme le père du Web. Les principales fondations qu'il a inventées ont donné naissance au Web tel que nous le connaissons aujourd'hui. Cependant, ils n'ont pas suffi à rendre le Web suffisamment interactif du côté du navigateur. Selon le site Web de Mozilla (2023), chaque fois qu'une page Web présente des fonctionnalités allant au-delà de l'affichage d'informations statiques, telles que l'affichage de contenu périodiquement mis à jour, des cartes

graphiques interactifs ou animés en deux ou trois dimensions, il est fort probable que JavaScript soit impliqué dans ce processus. Par conséquent, la section 2.2 fournit une brève introduction à ce sujet.

2.2 Javascript

JavaScript, parfois abrégé en JS, est un langage léger et interprété mieux connu basé sur des objets dotés de fonctionnalités de [1] sous le nom de langage de script . premier ordre, pour les pages Web. Il est également utilisé dans plusieurs autres environnements non-navigateurs, tels que : node.js, Apache CouchDB et Adobe Acrobat. JavaScript est un langage dynamique, multi-paradigmes et basé sur des prototypes, prenant en charge les styles orientés objet, impératifs et déclaratifs comme, par exemple, la programmation fonctionnelle (JAVASCRIPT, 2021).

JavaScript est un langage de programmation interprété et dynamique de haut niveau, largement utilisé sur le Web pour créer de l'interactivité et des effets dynamiques sur les pages Web. Il a été développé par Brendan Eich en 1995 alors qu'il travaillait chez Netscape Communications Corporation. Il est basé sur C et Java et a une syntaxe similaire, mais différente des deux en termes d'objectifs et de finalité. Il est également largement pris en charge par tous les principaux navigateurs Web et peut être utilisé à la fois côté client et côté serveur. Avec la popularisation de bibliothèques et de frameworks tels que jQuery, AngularJS, React et Vue, JavaScript est devenu encore plus populaire, ce qui est une fonctionnalité appréciée par les développeurs de logiciels. L'un des principaux avantages de JavaScript est la possibilité de créer à la fois des applications multipages (MPA) et des applications monopages (SPA). Ces deux notions seront discutées ci-dessous.

Selon Kaluža, Troskot et Vukelić (2018), les SPA sont un type d' application Web qui charge tout son contenu sur une seule page HTML. Cela signifie que les utilisateurs n'ont pas besoin d'attendre que la page se recharge lorsqu'ils interagissent avec elle, ce qui peut offrir une expérience utilisateur plus transparente et plus réactive. Les SPA deviennent de plus en plus populaires car ils peuvent être utilisés pour créer des applications Web plus interactives et attrayantes que les pages Web traditionnelles. D'un autre côté, les MPA sont les pages Web traditionnelles, chaque contenu est chargé sur une page différente, qui est rechargée à chaque interaction de l'utilisateur.

Un SPA se distingue, par rapport à un MPA, par ses meilleures performances, moindres

[1] Un langage de programmation est considéré comme ayant des caractéristiques fonctionnelles de premier ordre lorsque ses fonctions sont traitées de manière équivalente aux autres variables du système. Autrement dit, dans ce type de langage, il est possible d'utiliser des fonctions comme arguments d'autres fonctions, de les renvoyer comme résultat d'une fonction et de les affecter à des variables, de la même manière que le traitement est réservé à d'autres entités de données.

consommation de ressources, une plus grande interactivité et une meilleure expérience utilisateur. Cependant, il présente une plus grande complexité de développement, des difficultés en matière d'optimisation des moteurs de recherche (SEO) et de sécurité. D'un autre côté, un MPA a la simplicité de développement, la facilité de référencement et la sécurité. Cependant, il présente de moins bonnes performances, une plus grande consommation de ressources, moins d'interactivité et une moins bonne expérience utilisateur (SHARMA et al., 2019).

Le choix entre SPA et MPA dépend des exigences et des objectifs de l'application Web. Si l'application nécessite des performances, une interactivité et une convivialité élevées, un SPA peut être plus adapté. Si l'application requiert simplicité, référencement et sécurité, un MPA peut être un meilleur choix.

2.3 Angulaire

Angular est une plateforme open source pour développer des applications Web maintenues par Google. Il est basé sur TypeScript et a été publié pour la première fois en 2010. Angular est connu pour son architecture de composants, qui permet aux développeurs de créer des applications Web de manière modulaire et évolutive. Il fournit une série d'outils pour créer des applications Web, notamment la gestion de l'état, la validation des formulaires, le routage, les services, etc. Angular est utilisé par des entreprises de toutes tailles, y compris des petites startups et des grandes entreprises, pour créer des applications Web hautes performances. Il s'agit de l'un des outils JavaScript les plus populaires et les plus utilisés dans le monde (ANGULAR, 2023).

2.4 Réagir

React est une bibliothèque JavaScript open source développée par Facebook et maintenue par une communauté de développeurs. Il a été lancé en 2013 et est utilisé pour créer des interfaces utilisateur dans des applications Web et mobiles. React se concentre sur la création de composants réutilisables, permettant aux développeurs de créer des applications Web de manière efficace et évolutive. Il utilise le DOM virtuel (VDOM) qui est un concept de programmation dans lequel une représentation idéale, ou « virtuelle », de l'interface utilisateur est conservée en mémoire et synchronisée avec le « vrai » DOM par une bibliothèque telle que ReactDOM.

Ce processus est appelé réconciliation. Ce qui permet à React de restituer et de mettre à jour rapidement l'interface utilisateur sans avoir à actualiser la page entière. De plus, React fournit des fonctionnalités de gestion d'état, de routage, de validation de formulaire et d'autres fonctionnalités importantes pour la création d'applications Web. En raison de son efficacité, de son évolutivité et de sa facilité d'utilisation, React est largement utilisé par les entreprises de toutes tailles, y compris les startups et les grandes entreprises, pour développer des applications Web de haute qualité (REACT, 2023).

2.5 Vue

Vue.js est un framework JavaScript open source permettant de créer des interfaces utilisateur . Il a été créé par Evan You en 2014 et est maintenu par une communauté active de développeurs. Vue se démarque par son approche de composantisation, qui permet aux développeurs de créer des applications Web de manière modulaire et évolutive. De plus , Vue dispose d'une structure de données réactive, qui permet aux développeurs de créer des interfaces dynamiques qui se mettent automatiquement à jour à mesure que les données changent. Vue comprend également des fonctionnalités de routage, de validation de formulaires, d'animations, etc., ce qui en fait un choix populaire pour la création d'applications Web. Vue est connue pour sa courbe d'apprentissage fluide et sa documentation claire, ce qui en fait une option populaire pour les développeurs débutants et expérimentés. Il est largement utilisé par les entreprises de toutes tailles pour créer des applications Web de haute qualité (VUE, 2023).

2.6 Architecture des microservices

L'architecture des microservices est un style architectural logiciel qui se concentre sur la création d'applications sous la forme d'un ensemble de services indépendants et collaboratifs. Chaque service est construit autour d'une fonctionnalité métier unique et est déployé et géré indépendamment des autres services. Ces services communiquent généralement entre eux via une interface de programmation d'application (API) bien définie et peuvent être développés dans différents langages de programmation et en utilisant différentes technologies (MICROSOFT, 2023).

Selon Lewis et Fowler (2014), l'architecture des microservices repose sur des principes de conception tels que la séparation des responsabilités, une cohésion élevée et un couplage lâche. Ces principes garantissent que chaque service est hautement spécialisé dans une seule tâche, ce qui facilite la maintenance et l'évolution des services. L'architecture des microservices facilite également l'évolutivité horizontale des services, permettant à chaque service d'évoluer indépendamment selon les besoins.

Dans le même esprit, selon Newman (2021), l'architecture des microservices est une approche de construction de systèmes logiciels qui met l'accent sur la modularité, l'évolutivité, la maintenance et l'indépendance des services. Chaque microservice est responsable d'une seule fonction et est construit autour d'un contexte métier spécifique. Ces services sont hautement indépendants et peuvent être déployés et gérés séparément, ce qui rend le développement et la maintenance des applications plus agiles et efficaces.

En résumé, le concept d'architecture de microservices a été utilisé dans le travail pour fournir un contexte pour l'analyse comparative des frameworks Angular, React et Vue . L'architecture de microservices est une approche de conception logicielle qui divise un

système en unités plus petites, appelées microservices. Chaque microservice est responsable d'une fonction système spécifique. Cela facilite la maintenance, l'évolution et l'évolutivité des services.

2.7 Architecture REST

L'architecture REpresentational State Transfer (REST), selon Fielding (2000), est un style d'architecture logicielle basé sur les principes de communication Web et largement utilisé pour créer des API pour les applications Web et mobiles. L' architecture REST définit une série de règles et de recommandations pour structurer les API, notamment l'utilisation de ressources HTTP telles que GET, POST, PUT et DELETE pour représenter des actions sur les ressources de l'application. De plus, l'architecture REST utilise des URL pour identifier les ressources et utilise des réponses HTTP standardisées pour indiquer le résultat d'une requête. L'architecture REST est évolutive et permet la création d' applications distribuées, car différentes applications peuvent accéder et partager des ressources via une API REST. Cela fait de l'architecture REST un choix populaire pour créer des applications devant s'intégrer à d'autres systèmes ou partager des données entre différentes plates-formes. Par conséquent, l'architecture REST a été utilisée dans le travail pour définir comment les microservices communiquent entre eux. L'architecture REST est un modèle d'architecture logicielle qui définit un ensemble de contraintes pour la communication entre les systèmes. Ces contraintes garantissent que les microservices sont interopérables et faciles à intégrer.

3 ouvrages connexes

Ce chapitre présente des études liées à l'analyse comparative entre les SPA Meworks .

Kaluža, Troskot et Vukelić (2018) ont réalisé une analyse comparative entre les frameworks front-end pour le développement Web. L'objectif de l'article était de démontrer l' adéquation des frameworks disponibles pour créer des applications Web SPA vs MPA, de définir les principales fonctionnalités qui permettent un développement plus personnalisé des deux types d'applications, en plus d'identifier s'il existe un framework front-end optimisé. pour créer des applications MPA et SPA. Les auteurs ont choisi les trois frameworks front-end les plus populaires afin d'analyser : si une simple importation js est suffisante pour démarrer ; si la structure inclut beaucoup de frais généraux ; si le flux de travail est nécessaire ; s'il y a un rendu latéral ; s'il est nécessaire d'écrire et de maintenir une grande quantité de code complexe ; si le développeur doit s'appuyer sur des packages tiers ; quelle est la possibilité de compresser l'application, en minimisant le fichier bundle1 ; en plus de savoir s'il est possible de gérer l'état des données. Pour atteindre cet objectif, ils ont choisi les frameworks les plus populaires sur Github et qui étaient encore en croissance à cette époque, écartant les autres qui stagnaient. Selon ces auteurs, React comptait 86 000 étoiles et Vue.js 81 000. D'un autre côté, Angular était poussé par les géants Google et Microsoft. Huit questions ont été formulées et répondues en fonction de l'objectif. Les réponses ont été utilisées comme métriques, exprimées en texte, les valeurs possibles des réponses étaient : Non, Partiellement et Oui. De plus, chaque réponse s'est vu attribuer un poids sur une échelle de 0 à 2. Ces valeurs servent à renforcer la force de chacune des alternatives textuelles utilisées . Ils sont arrivés à la conclusion qu'Angular est le meilleur framework pour SPA, mais d'un autre côté, il s'est avéré le moins adapté au développement de MPA. Vue est le meilleur pour MPA. React était dans une situation intermédiaire dans les deux cas.

Dans les travaux de Ziani (2021), une analyse comparative et qualitative a été réalisée Cadres JavaScript . Le travail cherchait à répondre à trois questions :

– Il existe une méthode efficace pour comparer et sélectionner des frameworks JavaScript comme partie du développement d'une application Web ?

– Quels sont les critères utilisés pour comparer le framework JavaScript ?

[1] Le bundle est la version de production optimisée pour le Web créée à partir des outils de génération de chaque framework. Ses ensembles d'outils aboutissent à un fichier HTML compressé sur le Web et à un fichier JavaScript minifié, prêts pour le déploiement en production. Pour le JavaScript natif, un fichier HTML statique est utilisé qui encapsule le script équivalent , contenant les fonctions écrites dans le framework.

– Tous les frameworks JavaScript sont-ils pris en compte dans les analyses ?

Selon Ziani, les réponses aux deux premières questions ont été obtenues grâce à des recherches documentaires. Autrement dit, pour la première question, il semble qu'il n'existe pas de méthode unique et standardisée pour comparer les cadres reconnus par la communauté scientifique . Concernant la deuxième question, Ziani a identifié quatre groupes de critères

- des critères mesurables ;
- caractéristiques souhaitées/attendues ;
- le résultat de l'exécution d'un logiciel externe ;
- des critères statiques ou mesurables mais peu fiables car changeants.

Et enfin, Ziani n'a pas trouvé de solution intemporelle, car de nouveaux frameworks sont créés chaque année, les frameworks existants sont améliorés et parfois refactorisés sans offrir de rétrocompatibilité (ex. Angular V1 vs Angular V2) et enfin il peut arriver que les frameworks ne soient plus supportés. Par conséquent, Ziani a proposé d'utiliser la méthode Analytic Hierarchy Process (AHP) utilisant un arbre de décision pour comparer différents frameworks JavaScript . Cette méthode considère l'effort d'apprentissage comme un critère de sélection ; l'effort de développement; stabilité, attractivité et performance comme critères. En conclusion, Ziani a souligné Svelte comme le meilleur framework parmi ceux testés : Angular, Preact, React, Svelte et Vue.

Diniz-Junior et coll. (2022) ont évalué les performances des technologies de rendu Web basées sur JavaScript : Angular, React et Vue. Le but du travail était d'évaluer les performances des technologies de rendu Web basées sur JavaScript, en tenant compte du temps d'interaction, du temps de manipulation du DOM et de la taille du bundle . Les auteurs ont recherché dans la littérature les principales mesures d'évaluation et analysé le DOM virtuel et incrémentiel, créant ainsi une application standardisée. Ils ont testé les manipulations DOM en utilisant trois frameworks qui représentent ces deux techniques de rendu : Angular pour l'incrémental et React et Vue pour le virtuel. En guise de résultats, les auteurs ont cherché à contribuer :

– une méthodologie pour comparer les frameworks de rendu web incrémentiels et incrémentaux Virtuel basé sur JavaScript ;

– une étude de cas comparative avec les frameworks Angular, Vue et React ;

– une application web générique capable de créer, éditer et supprimer un certain nombre d'éléments dans la fenêtre du navigateur, suivant le cycle de vie (fonction framework qui contrôle les états d'une application) proposé par chaque framework et bibliothèque.

Diniz-Junior et coll. (2022) ont conclu que React avait des difficultés à créer des éléments DOM. Angular et Vue ont pris respectivement 270 à 290 ms en moyenne, tandis que React a pris plus de 5 000 ms. Les deux premiers sont basés sur le DOM virtuel et produisent des résultats différents. Angular, à son tour, avec son implémentation incrémentielle du DOM, a produit des résultats similaires à ceux observés lors de la création d'éléments pour les structures Vue, avec une différence moyenne d'environ 20 ms en considérant 50 000 éléments et les meilleures performances lors de l'édition de scénarios DOM. De ce fait, le principal avantage par rapport aux VDOM ou aux VNodes est qu'ils peuvent être mis à jour sans avoir besoin d'un intermédiaire pour combiner les éléments en mémoire. Enfin, les auteurs rapportent que la solution proposée par le DOM virtuel implémenté dans le framework Vue présentait un petit avantage global, performant dans toutes les fonctions et maintenant des pics en dessous de la marque de 1 pour la plus grande taille d'échantillon.

Les travaux de Ziani (2021) se distinguent de ces travaux par la méthode utilisée pour attribuer du poids aux critères, qui est subjective. Autrement dit, les poids attribués à chaque variable proposée par Ziani (2021) suivent une logique définie par l'auteur qui pourrait être différente si elle était proposée par quelqu'un d'autre. Les travaux de Kaluža, Troskot et Vukelić (2018) se distinguent de ces travaux par la prise en compte du concept de MPA. Les travaux de Diniz-Junior et al. (2022) se sont beaucoup intéressés à la question du rendu. Cependant, tous ces travaux sont similaires car ils prennent en compte les trois frameworks Angular, React et Vue. Par conséquent, ce travail peut contribuer aux connaissances scientifiques sur les frameworks JavaScript pour développer des SPA basés sur des microservices. Par exemple, l'analyse peut identifier des modèles et des tendances qui pourraient être utiles pour de futures études. De plus, l'architecture des microservices est une approche de conception logicielle de plus en plus populaire, et les frameworks Angular, React et Vue sont les plus largement utilisés pour le développement d'applications monopage. Une analyse comparative de ces frameworks dans le contexte de l'architecture des microservices est pertinente pour un large éventail de professionnels. Ensuite , dans le chapitre 4, la méthodologie qui a été adoptée dans ce travail est présentée pour comparer les cadres SPA .

4 Méthodologie

Ce chapitre présente la méthodologie proposée pour atteindre les objectifs décrits dans ce travail. La méthodologie de comparaison des frameworks SPA en développant une application avec une architecture de microservices pour tester les performances comprenait une série d'étapes décrites ci-dessous :

4.1 Revue bibliographique

Une revue de la littérature a été réalisée pour obtenir des informations sur les frameworks, leurs principales caractéristiques, avantages et inconvénients, et comment les comparer par rapport à des critères tels que la taille du package, le temps de chargement, le temps de rendu, l'utilisation de la mémoire et le temps d'exécution. Trois études récentes ont analysé les frameworks SPA pour le développement Web. Kaluža, Troskot et Vukelić (2018) ont comparé les frameworks pour SPA et MPA, identifiant les principales fonctionnalités et l'adéquation à chaque type d'applica Ziani (2021) a recherché une méthode efficace pour comparer et sélectionner les frameworks JavaScript , en identifiant les critères et les frameworks les plus utilisés . Diniz-Junior et coll. (2022) ont évalué les performances des frameworks JavaScript , en tenant compte du temps d'interaction, de la manipulation du DOM et de la taille du bundle . Des études indiquent qu'il n'existe pas de framework front-end idéal pour tous les cas. Le choix du cadre doit être fait en tenant compte du type d'application, des besoins des utilisateurs et des critères d'évaluation pertinents. La revue de la littérature a également permis d'acquérir une compréhension générale du concept d' architecture de microservices et de la manière dont elle est mise en œuvre dans les applications.

4.2 Définition de l'architecture et du développement de l'application

Sur la base de l'analyse de la littérature, les critères d'évaluation, les métriques (taille du bundle , temps de chargement , temps de rendu, utilisation de la mémoire et temps d'exécution) et les frameworks à évaluer comparativement ont été définis.
De cette manière, les exigences à mettre en œuvre par l'application à construire à l'aide des trois alternatives proposées ont été définies. L'architecture des microservices a été définie pour garantir que l'application est évolutive et peut gérer le trafic excédentaire.
Ceci est nécessaire pour définir la portée des travaux. L'objectif était de développer la même application avec les trois frameworks et de les comparer à l'aide des métriques définies ;

4.2.1 Infrastructures

Pour répondre aux exigences méthodologiques proposées, il était essentiel d'employer un dispositif informatique doté d'une capacité de stockage et de traitement adéquate pour le développement et, si nécessaire, la simulation de l'application. Les technologies suivantes ont été utilisées pour réaliser ces tests :

- Ordinateur portable Vaio FE15 avec 12 Go de mémoire, processeur Intel® Core™ i5-8250U 3,40 GHz, exécutant le système d'exploitation Windows 10 ;
- Microsoft Edge 119.0.2151.58 pour exécuter le frontal ;
- Docker1 pour la conteneurisation prestations de service;
- l'IDE VS Code pour l'implémentation des codes ;
- le SGBD MYSQL 5.7 pour le stockage des données ;
- PM23 5.3.0 pour la gestion des processus ;
- Node.js 16.16.0 ;
- Angular 16.2.0, dernière version de support à long terme au moment de l'exécution de la expérience;
- React 18.2.0, dernière version de support long terme au moment de l'exécution de la expérience;
- Vue 3.3.4, dernière version de support à long terme au moment de l'exécution de la expérience.

4.2.2 Description de la demande

L'application utilisée pour les tests peut être considérée comme faisant partie d'un système de commerce électronique. L'application développée se concentre sur la gestion des produits disponibles à la vente dans la boutique en ligne, y compris leur création, lecture, mise à jour et suppression (CRUD), ainsi que sur le stockage des images associées à ces produits.

De cette manière, un e-commerce peut bénéficier du système décrit pour gérer efficacement les produits, en les rendant facilement accessibles aux clients via une interface utilisateur développée avec Vue, React et Angular. Le système peut également

[1] Docker est un ensemble de produits de plateforme en tant que service qui utilisent la virtualisation au niveau du système d'exploitation pour fournir des logiciels dans des packages appelés conteneurs. Les conteneurs sont isolés les uns des autres et regroupent leurs propres logiciels, bibliothèques et fichiers de configuration.

[2] La conteneurisation est le regroupement du code logiciel avec tous les composants nécessaires, tels que les bibliothèques, les frameworks et autres dépendances, afin qu'ils soient isolés dans leur propre conteneur.

[3] PM2 est un gestionnaire de processus pour le runtime JavaScript Node.js

contribuer à garantir l'intégrité et la cohérence des données produit, en évitant des problèmes tels que des informations incorrectes ou des images perdues.

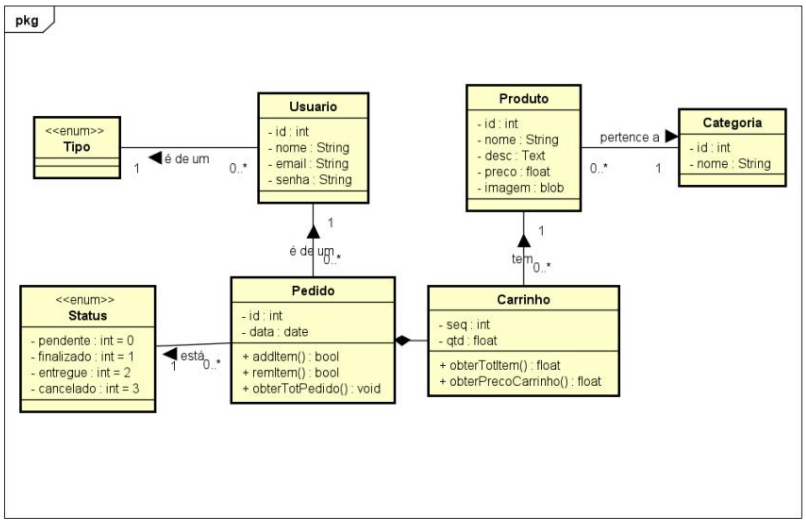
L'application est donc composée de deux couches :

- back-end : un service API qui communique avec le SGBD MySQL ;
- front-end : l'interface de l'application implémentée avec les trois frameworks (Angular, React, Vue.js).

La figure 1 montre un diagramme de classes qui représente les classes de l'application. Ce diagramme de classes correspond à la configuration système requise qui est :

- permettre aux utilisateurs autorisés de créer, lire, mettre à jour et supprimer des produits ;
- permettre d'associer des images à chaque produit ;
- permettre aux utilisateurs de rechercher des produits par nom et catégorie ;
- permettre aux utilisateurs d'ajouter des produits au panier et de finaliser l'achat.

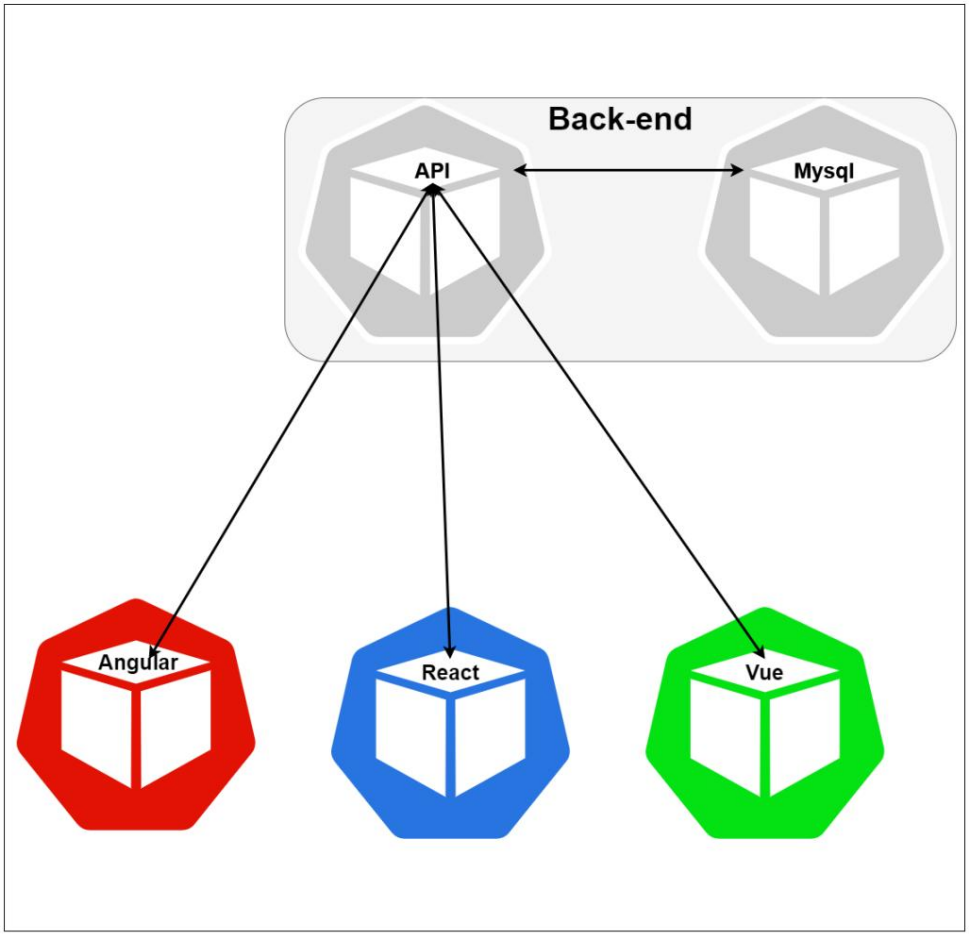
Figure 1 – Diagramme de classes



Source : de l'auteur (2023).

Pour simuler un véritable environnement d'architecture de microservices, le concept de conteneurisation a été utilisé pour implémenter chaque service. La figure 2 montre un diagramme qui représente l'architecture de l'application. Chaque cube représente un conteneur dans lequel un service qui est un composant de l'application a été implémenté. Dans ce cas, le front-end a été implémenté à l'aide des technologies SPA Angular, React et Vue pour pouvoir effectuer l'analyse comparative.

Figure 2 – Architecture des applications



Source : de l'auteur (2023).

4.3 Tests de performances

Une fois l'application développée, des tests de performances ont été effectués pour les comparer. Les tests de performances visent à mesurer la taille du package, le temps de chargement, le temps de rendu, l'utilisation de la mémoire et le temps d'exécution dans différents scénarios de chargement. Cependant, pour rendre les tests possibles, sans tenir compte de la complexité de l'interaction dans un e-commerce, les opérations dans l'interface ont été limitées à :

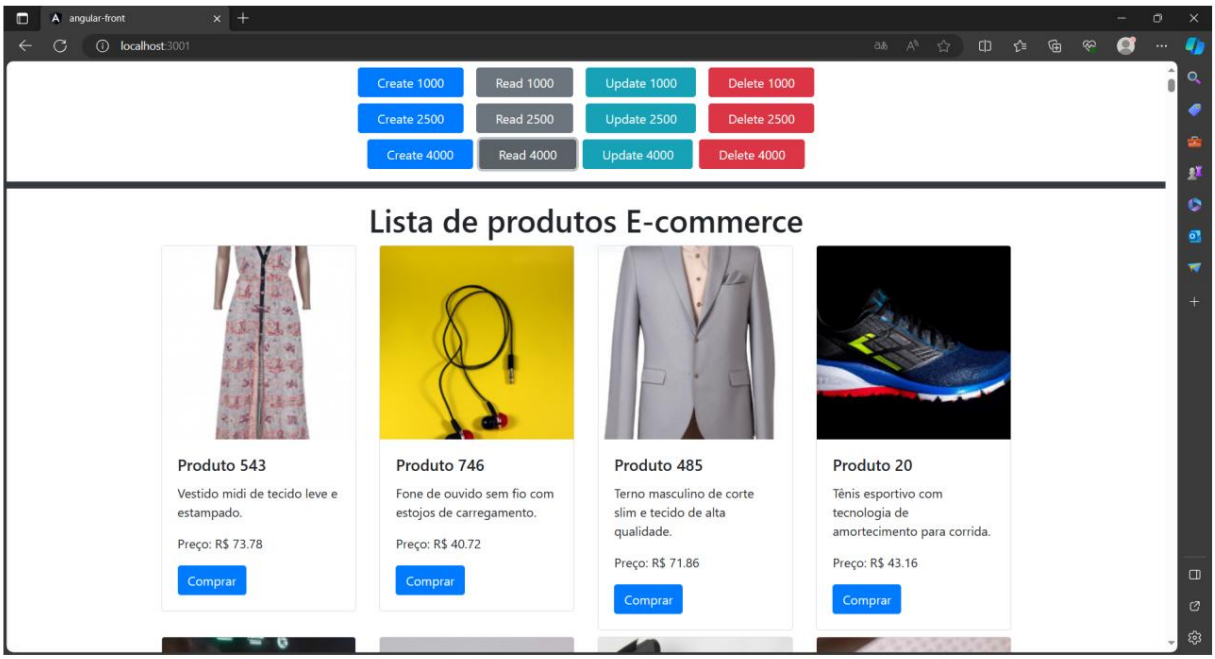
- création de 1000, 2500 et 4000 produits ;
- lecture de 1000, 2500 et 4000 produits ;
- édition de 1000, 2500 et 4000 produits ;
- retrait de 1000, 2500 et 4000 produits.

Initialement, la charge a été définie en prenant comme exemple une charge de 1 000, 5 000 et 10 000 produits. Cela représente un scénario à faible charge, un scénario à charge moyenne et

un avec une charge élevée. Cependant, la machine utilisée ne peut pas effectuer le traitement car le traitement des images nécessite de nombreuses ressources dont la machine ne disposait pas. Par conséquent, la charge a été réduite jusqu'à atteindre ces valeurs 1000, 2500 et 4000, représentant respectivement le scénario de charge petite, moyenne et élevée que la machine était capable de traiter. Le test du temps d'exécution des opérations CRUD a été effectué sur la base des interactions de l'utilisateur à l'aide d'un bouton qui exécute chacune des opérations avec la charge indiquée. Les performances de chaque framework ont été évaluées en capturant le temps d'exécution dans le code, à l'aide de l'API de performances JavaScript. La collection d'autres métriques a été obtenue à partir de l'outil de développement Microsoft Edge Devtools.

La figure 3 montre l'interface de l'application où les différents tests ont été réalisés. Cette interface est la même que celle générée à l'aide de chacun des frameworks. Afin que les produits reflètent la réalité, le produit [4] générer cent cinquante descriptions de ChatGPT a été utilisé comme le montre la figure 4 , puis l'API AI du générateur d'images DALL.E2 [5] d'OpenAIa été utilisée pour générer des images de produits basées sur les descriptions fournies par ChatGPT. Le script permettant de générer les images est présenté dans la figure 5. Au total, 1 500 images ont été générées, dix pour chaque description de produit. Ces images ont été utilisées pour réaliser le test.

Figure 3 – Interface de test

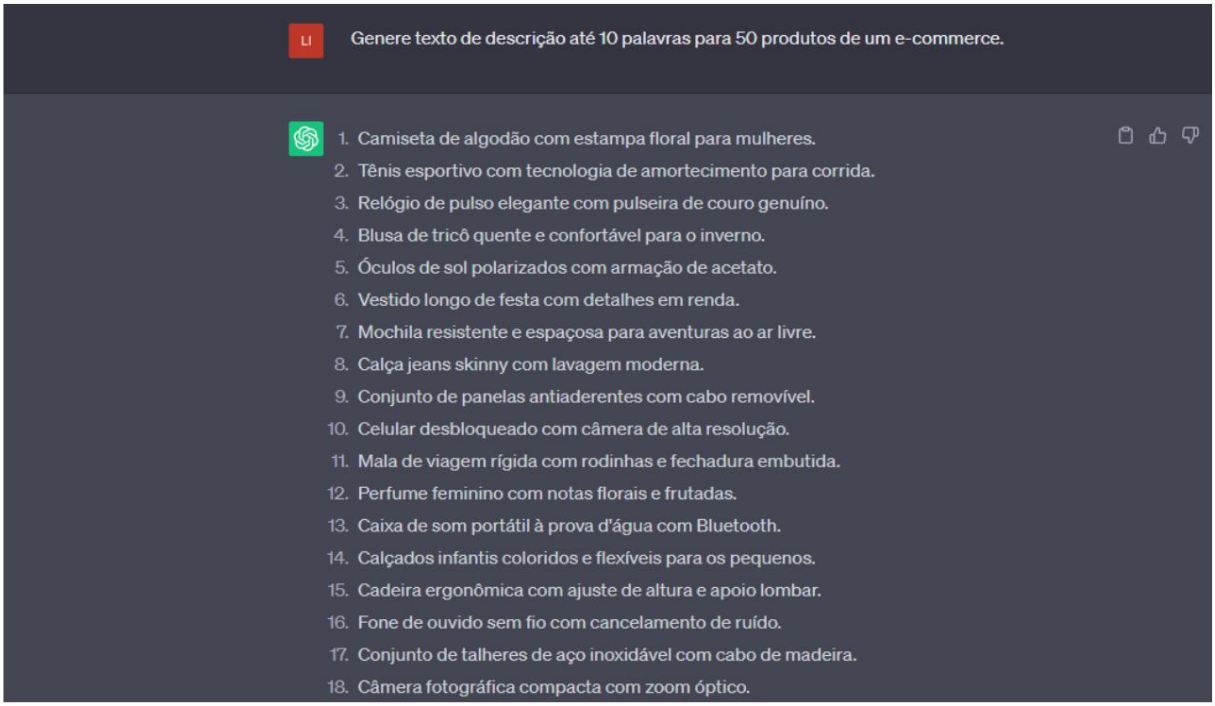


Source : de l'auteur (2023).

[4] ChatGPT est un chatbot d'intelligence artificielle en ligne développé par OpenAI, lancé en novembre 2022. Le nom « ChatGPT » combine « Chat », faisant référence à sa fonctionnalité de chatbot, et « GPT », qui signifie Generative Pre-trained Transformer, un type de grand chatbot. modèle de langage et un cadre important pour l'intelligence artificielle générative.

[5] DALL.E2 est un système d'IA capable de créer des images et des œuvres d'art réalistes à partir d'une description en langage naturel.

Figure 4 – Invite de génération de description de produit à l'aide de ChatGPT



Source : ChatGPT, par l'auteur (2023).

Figure 5 – Script qui génère des images de produits à l'aide de la description fournie par ChatGPT depuis l'appel API OpenAI

```
// Função para gerar 10 produtos
async function gerarProdutos() {
  const produtos = [];
let c1=0,c2=0;
  for (let i = 0; i < descriptions.length; i++) {
    if(i!=0 && i%5==0){
      console.log('Esperando 2 minutos');
      await new Promise((resolve) => {
        setTimeout(() => {
          resolve();
        }, 2*60000); // 60000 milissegundos = 1 minuto
      });
    }
    // console.log(`'description ${i+1}:>> '`, descriptions[i]);
    const response = await axios.post('https://api.openai.com/v1/images/generations',
    {
      'prompt':  descriptions[i],
      'n': 10,
      'size': '256x256'
    },
    {
      headers: {
        'Content-Type': 'application/json',
        'Authorization': 'Bearer ' + process.env.OPENAI_API_KEY
      }
    }
  );

```

Source : de l'auteur (2023).

4.4 Évaluation comparative des cadres

Sur la base des résultats des tests de performance, une évaluation comparative des applications mises en œuvre a été réalisée. L'évaluation a montré laquelle des alternatives est la meilleure option en termes de taille du package, de temps de chargement, de temps de rendu, d'utilisation de la mémoire et de temps d'exécution.

Ensuite, au chapitre 5, les résultats attendus de ces travaux sont présentés, exprimés à travers des tableaux et des graphiques. Ensuite, une analyse des résultats obtenus est effectuée.

5 résultats

Ce chapitre présente les résultats de l'étude par métriques évaluées.

5.1 Taille du paquet

Le tableau 1 montre l' ensemble des applications des trois frameworks, mesuré en Ko. Vue a généré le plus petit package, React a généré le deuxième plus petit et Angular a généré le plus grand. En d'autres termes, le bundle Vue fait 404 Ko, le bundle React fait 488 Ko et le bundle Angular fait 500 Ko. Ko. Par conséquent, avoir le plus petit paquet généré est un avantage, car tous les appareils n'ont pas capacité mémoire pour stocker une application.

Tableau 1 – Données obtenues dans les expériences

	Vue de réaction angulaire
Offre groupée (Ko) 500 488 404	
Charge (ms) 7 6 6	
Rendu (ms) 4 4 6	
Tas JS (Mo) 2.4-6.2	2,8-3,8 2,5-3,0

Source : de l'auteur (2023).

Le tableau 1 montre que Vue est le cadre avec le plus petit bundle , React with le deuxième plus petit et angulaire avec le plus grand paquet. La taille du paquet généré par Angular est 2,46 % plus grand que React et 23,76 % plus grand que Vue. Cela signifie que l'application implémentée par Vue nécessite moins de bande passante, celle de React entre en jeu deuxième place et Angular est celui qui a le plus besoin de bande passante.

Ces différences peuvent s'expliquer par un certain nombre de facteurs, notamment caractéristiques des frameworks, des bibliothèques qu'ils incluent et des options de configuration qu'ils proposent. Angular est un framework complet , qui comprend un large éventail de des fonctionnalités telles que l'injection de dépendances, le routage et la création de modèles. Cela peut contribuer à sa plus grande taille. React est un framework plus axé sur le rendu des composants, tandis que Vue est un framework plus modulaire , qui permet aux développeurs de choisir quelles bibliothèques et ressources inclure. Cela peut expliquer la plus petite taille de ces deux cadres.

Il est important de souligner que le forfait n'est pas le seul facteur à prendre en compte lors du choix. d'un cadre. D'autres facteurs importants incluent la facilité d'utilisation, la flexibilité et la communauté de soutien. Toutefois, le forfait peut être un facteur important dans

les applications qui doivent se charger rapidement ou qui ont des contraintes de bande passante .

5.2 Temps de charge

Le tableau 1 montre le temps de chargement des applications pour les trois frameworks, mesuré en millisecondes. Les applications React et Vue ont le temps de chargement le plus court, suivis par Angular. En d'autres termes, React et Vue ont un temps de chargement de 6 millisecondes, tandis qu'Angular a un temps de chargement de 7 millisecondes. Cette différence de 1 milliseconde représente une augmentation de 16,67 % par rapport au temps de chargement de React et Vue.

Cette différence est faible, mais peut être significative dans les applications où le temps de charge est critique. Par exemple, dans les applications mobiles, où la bande passante est limitée, une augmentation d'une milliseconde peut faire une différence notable dans l' expérience utilisateur.

Il existe plusieurs raisons pour lesquelles Angular a un temps de chargement légèrement plus long que React et Vue. L'une des raisons est qu'Angular est un framework plus complexe , avec plus de code et de bibliothèques. Une autre raison est qu'Angular utilise un compilateur pour convertir le code TypeScript en JavaScript, ce qui ajoute une étape supplémentaire au processus de chargement.

Dans l'ensemble, la différence de temps de chargement entre React, Vue et Angular est faible. Toutefois, cette différence peut être significative dans les applications où le temps de charge est critique.

5.3 Temps de rendu

Le tableau 1 montre le temps de rendu des applications des trois frameworks, mesuré en millisecondes. Angular et React ont le temps de rendu le plus court, suivis de Vue avec le plus long. Les différences de temps de rendu entre les trois frameworks sont faibles mais significatives. Angular et React ont le même temps de rendu, 4 millisecondes. Vue, quant à elle, a un temps de rendu 50 % plus long de 6 millisecondes.

Cette différence de 2 millisecondes peut sembler minime, mais elle peut être significative dans les applications nécessitant des performances de rendu élevées. Par exemple, dans les applications qui nécessitent une mise à jour rapide du contenu de l'écran, comme les jeux ou les applications de streaming vidéo , une différence de 2 millisecondes peut être perceptible pour l'utilisateur.

De plus, la différence de 50 % entre Vue et les deux autres frameworks peut être significative dans les applications nécessitant un rendu fréquent. Par exemple, dans les applications qui nécessitent que le contenu de l'écran soit mis à jour en fonction de données externes, telles que la météo ou la météo, une différence de 50 % peut entraîner une plus grande consommation de ressources.

En résumé, les différences de temps de rendu entre les trois frameworks sont faibles mais significatives. Le temps de rendu de Vue est 50 % plus long que celui d'Angular et de React. Cette différence peut être significative dans les applications nécessitant des performances de rendu élevées ou des rendus fréquents.

Voici quelques exemples de la façon dont les différences de temps de rendu peuvent affecter l'expérience utilisateur :

Dans un jeu, par exemple, une différence de 2 millisecondes peut entraîner un retard notable dans la réponse du jeu aux commandes de l'utilisateur. Dans une application de streaming vidéo , une différence de 2 millisecondes peut entraîner un retard notable dans la mise à jour de l'image à l'écran. Dans une application qui affiche l'heure actuelle, une différence de 50 % pourrait entraîner une mise à jour de l'heure 25 % plus lent
Par conséquent, lors du choix d'un framework pour votre application, il est important de prendre en compte le temps de rendu , surtout si votre application nécessite des performances élevées ou un rendu fréquent.

5.4 Utilisation de la mémoire

Le tableau 1 montre l'utilisation de la mémoire des trois applications implémentées à l'aide des frameworks, mesurée en mégaoctets. La figure 6 montre que l'application implémentée à l'aide de Vue est celle qui utilise le moins de mémoire, avec une consommation allant de 2,5 à 3,0 mégaoctets. Cette différence peut être attribuée à un modèle de composant similaire à React, mais avec quelques optimisations qui réduisent la consommation de mémoire.

L'application React a la deuxième utilisation de mémoire la plus élevée, avec une consommation allant de 2,8 à 3,8 mégaoctets. Cette différence peut être attribuée à un modèle de composant plus simple, qui nécessite moins de mémoire pour stocker les informations sur les composants et leurs états.

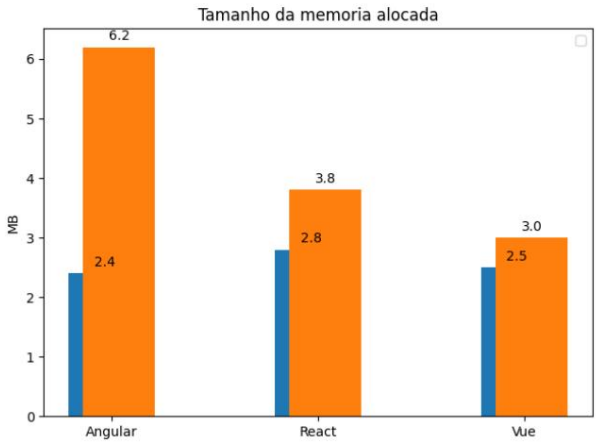
L'application Angular est celle qui utilise le plus de mémoire, avec une consommation allant de 2,4 à 6,2 mégaoctets. Cette différence peut être attribuée à un certain nombre de facteurs, notamment :

- l'utilisation d'un modèle de composant plus complexe, qui nécessite plus de mémoire pour stocker des informations sur les composants et leurs états ;
- l'utilisation d'un système d'injection de dépendances plus complet, qui nécessite également

plus de mémoire ;

- l'utilisation d'une bibliothèque plus large de composants et de services, qui peut également augmenter la consommation de mémoire.

Figure 6 – Figure de l'espace (Mo) alloué en mémoire initialement par chaque framework , où le bleu représente le plus petit espace alloué et l'orange, le plus grand espace alloué.



Source : de l'auteur (2023).

En termes de pourcentage, Angular utilise environ 63,15 % de mémoire en plus que que React et 106,67 % de mémoire en plus que Vue.

Cette différence d'utilisation de la mémoire doit être prise en compte lors du choix d'un framework pour développer une application Web. Le choix du framework idéal dépendra d'un certain nombre de facteurs, notamment de la taille et de la complexité de l'application, des ressources requises et des performances attendues.

Dans les petites applications simples, la différence d'utilisation de la mémoire peut ne pas être significative. Cependant, dans les applications vastes et complexes, la différence peut devenir importante, surtout si l'application s'exécute sur des appareils capables de limité.

5.5 Temps d'exécution

Le tableau 2 montre le temps d'exécution moyen de chaque tâche pour chaque technologie. On peut voir que, en général, Angular a obtenu les pires résultats, suivi de React et Vue.

Tableau 2 – Temps d'exécution moyen x Processus

Exécution		Temps moyen (ms) Vue de réaction angulaire		
Temps de création	1000	17836	17603	16705
	2500	46553	44097	43906
	4000	77313	73034	73317
Temps de lecture	1000	43	41	44
	2500	67	57	68
	4000	102	94	97
Temps de mise à jour	1000	38234	36347	34932
	2500	101166	90668	89543
	4000	169862	153900	169808
Supprimer l'heure	1000	9283	15682	8298
	2500	86612	23845	67686
	4000	178000	38799	151553

Source : de l'auteur (2023).

Le tableau 3 montre l'écart type d'exécution de chaque tâche pour chaque technologie. L'écart type est une mesure de la variabilité des résultats. Valeurs d'écart type Les valeurs faibles indiquent que les résultats sont plus cohérents, tandis que les valeurs d'écart type des valeurs plus élevées indiquent que les résultats sont plus dispersés.

Tableau 3 – Écart type d'exécution x Processus

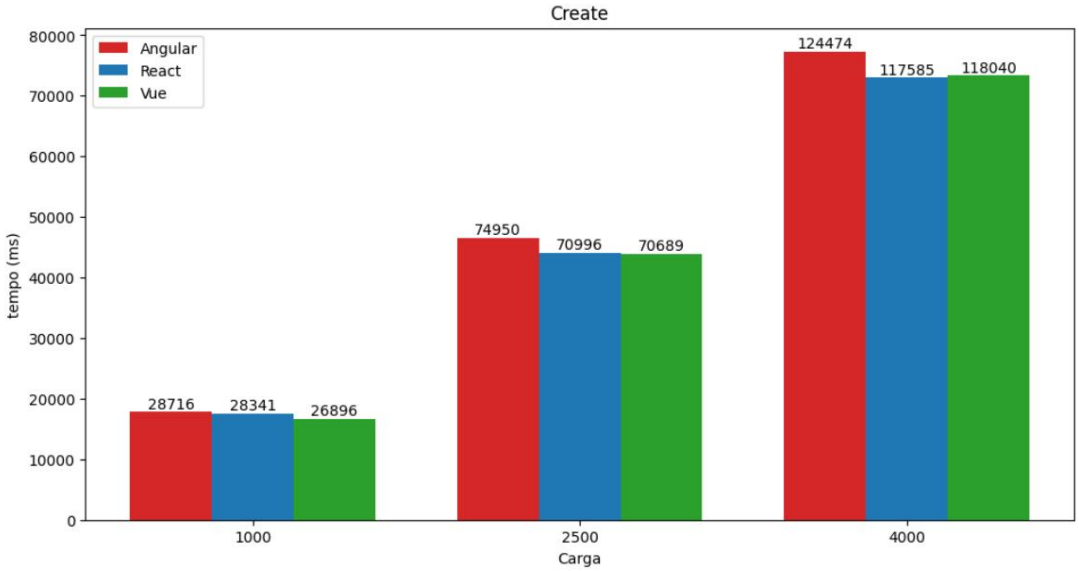
Exécution		Écart type (ms) Vue de réaction angulaire		
Temps de création	1000	43	1504	21
	2500	474	383	1098
	4000	1596		601
Temps de lecture	1000		4	456
	2500	4	14	1
	4000	7	9	12
Temps de mise à jour	1000	661	1734	179
	2500	1711	898	1149
	4000	18031	5760	19270
Supprimer l'heure	1000	242	131	265
	2500	702	615	2048
	4000	2141	710	14769

Source : de l'auteur (2023).

La figure 7 montre que le temps moyen consacré à l'opération de création de produit pour différentes charges. Les applications React et Vue fonctionnent mieux, avec la tendance qui montre que React peut être meilleur que Vue si la charge continue

en augmentant. Il est possible de vérifier que l'application Angular a les pires performances à cet égard.

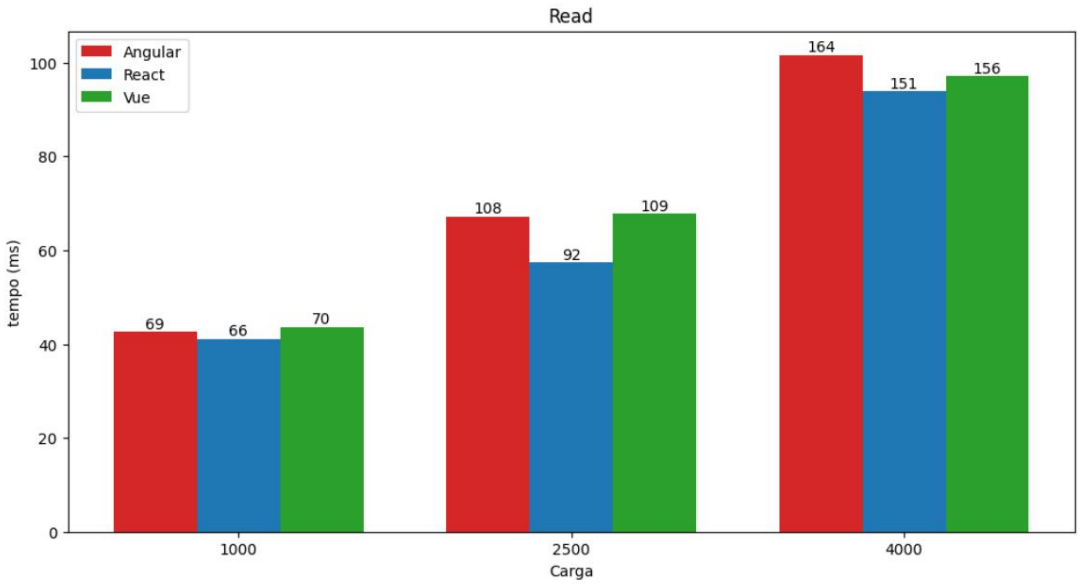
Figure 7 – Temps d'exécution (ms) de l'opération de création de produit



Source : de l'auteur (2023).

La figure 8 montre que le temps moyen consacré à l'opération de lecture du produit performances sous pour Il est possible de vérifier que l'application React a les meilleures différentes charges. à cet égard. On peut dire que Vue a la deuxième meilleure performance, bien qu'Angular ait de meilleures performances compte tenu de la charge de 2500 produits, mais la différence de 1 ms peut être insignifiante si l'on considère une marge d'erreur dans la mesure. Par conséquent, Angular a eu les pires performances à cet égard.

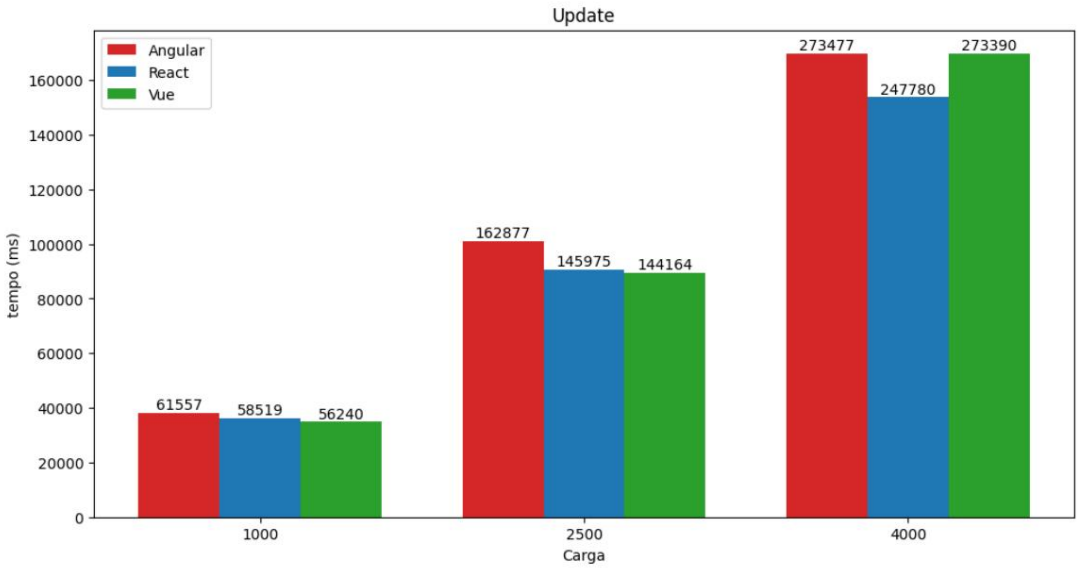
Figure 8 – Temps d'exécution (ms) de l'opération de lecture du produit



Source : de l'auteur (2023).

La figure 9 montre que le temps moyen passé sur l'opération de mise à jour du pour les différentes produit permet de vérifier que l'application React est la meilleure charges. performances pour le chargement de 4000 produits. Cependant, Vue a les meilleures performances respectivement dans des charges de 1 000 et 2 500 produits. Si vous considérez la petite différence entre Vue et React, on peut dire que React a les meilleures performances et Vue a la deuxième meilleure performance. Par conséquent, l'application Angular a les pires performances.

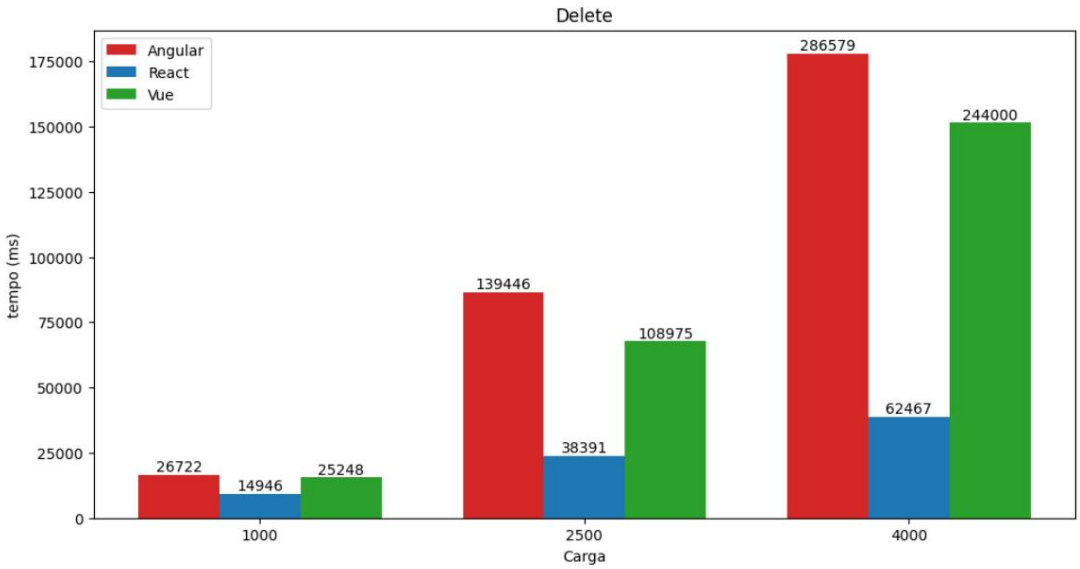
Figure 9 – Temps d'exécution (ms) de l'opération de mise à jour du produit



Source : de l'auteur (2023).

La figure 10 montre le temps moyen nécessaire à l'opération de suppression. du produit pour les différentes Vous pouvez vérifier que l'application React dispose charges utiles. de meilleures performances à cet égard. Vue a des performances intermédiaires et Angular a les pires performances. Par conséquent, pour la suppression de produits, React est plus performant, avec une marge considérable.

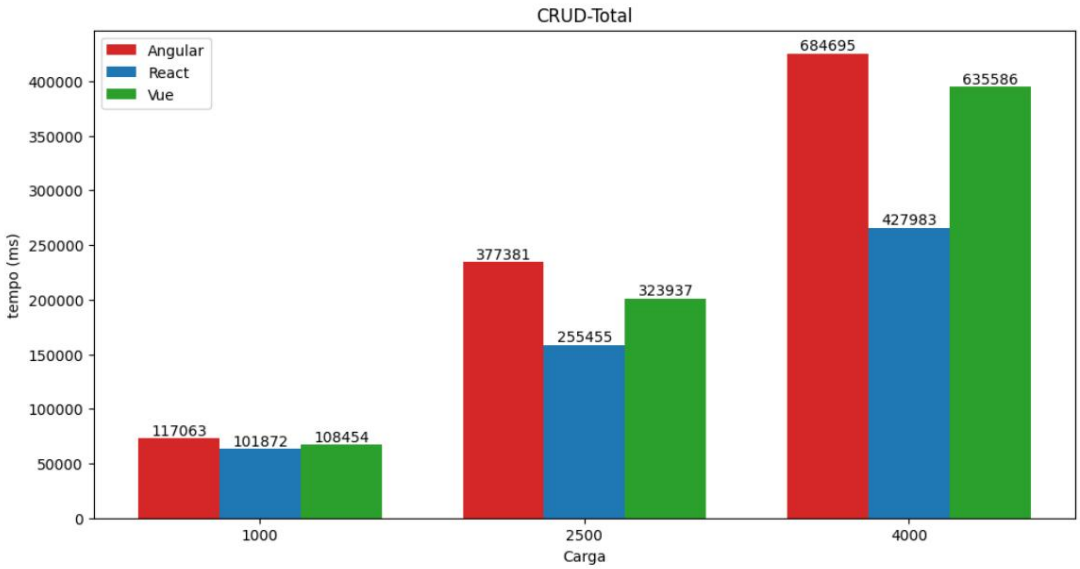
Figure 10 – Temps d'exécution (ms) de l'opération de retrait du produit



Source : de l'auteur (2023).

La figure 11 montre que le temps total moyen passé pour l'opération CRUD Il est produit pour les différentes charges. possible de vérifier clairement que l'application du React a les meilleures performances. Vue a des performances intermédiaires et Angular a les pires performances. Si vous considérez l'opération CRUD comme unique, React se démarque considérablement, surtout lorsque la charge est importante.

Figure 11 – Temps d'exécution total (ms) de l'opération CRUD du produit



Source : de l'auteur (2023).

Les résultats de cette recherche diffèrent de ceux obtenus par Kaluža, Troskot et Vukelić (2018), qui ont conclu qu'il n'existe pas de cadre JavaScript idéal pour SPA et MPA. Bien que Ziani (2021) ait mené une analyse comparative des frameworks JavaScript, il convient de noter que son étude était qualitative, tandis que le présent travail adopte une approche quantitative. Ainsi, les résultats présentés ici corroborent la déclaration de Diniz-Junior et al. (2022), indiquant qu'Angular démontre des performances supérieures en termes de rendu.

Il convient de noter que les résultats obtenus dans cette analyse sont basés sur un ensemble spécifique de données et de conditions. Les résultats peuvent être différents pour d'autres ensembles de données ou conditions. Par conséquent, cette étude présente certaines limites dont il faut tenir compte lors de l'interprétation des résultats.

Tout d'abord, l'étude a été réalisée dans un environnement contrôlé, avec un ensemble de données spécifique. Étant donné que ce travail a adopté l' architecture des microservices comme approche, il est possible que les résultats soient différents avec une architecture différente, ou dans différents environnements tels que Linux ou MacOS, ou avec des ensembles de données.

beaucoup de différents.

Deuxièmement, l'étude s'est concentrée sur un ensemble limité de mesures de performance. Il est possible que d'autres facteurs qualitatifs, tels que la facilité d'utilisation ou la maintenabilité, soient plus importants pour certaines applications que les facteurs pris en compte dans cette étude.

Troisièmement, l'étude n'a pas pris en compte les installations et les ressources disponibles pour les cadres qui peuvent aider au développement.

6. Conclusion

Des résultats obtenus dans cette analyse comparative entre les frameworks Angular, React et Vue , on peut conclure que l'application implémentée à l'aide de React présente les meilleures performances globales, suivie de Vue et Angular en dernière position.

En termes de taille du bundle généré pour la mise en production, Vue a généré le plus petit bundle, React a généré le deuxième plus petit et Angular a généré le plus grand des trois. Cela signifie que l'application Angular nécessite plus de bande passante pour se charger, ce qui peut affecter les performances de l'application sur les appareils ayant moins de puissance de traitement et un accès Internet plus lent.

En termes de temps de chargement, les applications React et Vue sont les plus rapides, suivies par Angular. Cela signifie que les applications développées avec ces frameworks se chargeront plus rapidement, ce qui peut améliorer l'expérience.
de l'utilisateur.

En termes de temps de rendu, l'application Angular et l'application React effectuent la même chose. Ils s'affichent plus rapidement que l'application Vue. Cela signifie que les applications développées avec ces frameworks auront des performances similaires en termes de rendu de contenu.

En termes d'utilisation de la mémoire, l'application Vue utilise le moins de mémoire, suivie de React et Angular en dernière position. Cela signifie que les applications développées avec Angular peuvent consommer plus de mémoire dans d'autres technologies testées, ce qui peut affecter les performances des applications sur les appareils dotés d' une capacité de traitement inférieure.

En termes de temps d'exécution, l'application React présente les meilleures performances dans toutes les opérations CRUD, suivie par Vue et Angular en dernière place.
Cela signifie que les applications développées avec React fonctionneront mieux dans les opérations CRUD.

Sur la base des résultats obtenus, les recommandations suivantes peuvent être faites :

- pour les applications qui doivent se charger rapidement, React ou Vue sont la solution
  de meilleures options ;
- pour les applications qui nécessitent moins de mémoire, Vue est la meilleure option ;
- pour les applications qui doivent fonctionner correctement dans les opérations CRUD, le
  React est la meilleure option.

Quelques travaux futurs qui pourraient être menés pour élargir les résultats de cette étude comprennent :

- réaliser l'étude dans un environnement cloud, avec différents types d'applications et données;
- Tenir compte d'autres facteurs lors de la prise de décision, comme la facilité d'utilisation ou maintenabilité;
- comparer les frameworks SPA sur du matériel mobile tel qu'Android ou IOS.

Les références

ANGULAIRE. Google, 2023. Disponible sur : <https://angular.io/guide/what-is-angular>.

BERNERS-LEE, T. et al. Le World Wide Web. Communications de l'ACM, ACM New York, NY, USA, v. 37, non. 8, p. 76-82, 1994.

DINIZ-JUNIOR, RN et al. Évaluation des performances des technologies de rendu Web basées sur javascript : Angular, React et Vue. P. 1-9, 2022.

FERREIRA, Hong Kong ; ZUCHI, JD Analyse comparative entre les frameworks frontend basés sur javascript pour les applications Web. Magazine d'interface technologique, v. 15, non. 2, p. 111-123, déc. 2018. Disponible sur : <https://revista.fatectq.edu.br/interfacetecnologica/article/view/502 >.

FIELDING, transfert d'état représentatif RT (REST). Université de Californie, 2000. Disponible sur : <https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm>.

JAVASCRIPT. Mozilla Corporation's, 2021. Disponible sur : <https://developer.mozilla.org/pt-BR/docs/Web/JavaScript>.

KALUŽA, M. ; TROSKOT, K. ; VUKELIĆ, B. Comparaison des frameworks front-end pour le développement d'applications Web. Zbornik Veleučilišta u Rijeci, Veleučilište u Rijeci, vol. 6, non. 1, p. 261-282, 2018.

KORNIENKO, D. ; MISHINA, S. ; MELNIKOV, M. L'architecture d'application monopage lors du développement de services Web sécurisés. Dans : ÉDITION IOP. Journal of Physics : Série de conférences. [SI], 2021. v. 2091, non. 1, p. 012065.

LEWIS, J. ; FOWLER, M. Microservices : une définition de ce nouveau terme architectural. Martin Fowler, 2014. Disponible sur : <https://martinfowler.com/articles/microservices.html>.

MICROSOFT. Microsoft, 2023. Disponible sur : <https://learn.microsoft.com/pt-br/azure/architecture/guide/architecture-styles/microservices >.

MOZILLA. Mozilla Corporation's, 2023. Disponible sur : <https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/First_steps/What_is_JavaScript >.

NEWMAN, R. et coll. Web 2.0 : le passé et le futur. Journal international de gestion de l'information, Elsevier, v. 36, non. 4, p. 591-598, 2016.

NEWMAN, S. Création de microservices. 2e éd. Sébastopol : O'Reilly Media, Inc., 2021.

RÉAGIR. Meta Platforms, Inc., 2023. Disponible sur : <https://pt-br.reactjs.org/>.

SHARMA, D. et al. Un bref aperçu de l'optimisation des moteurs de recherche. Dans : 2019 9e Conférence internationale sur le Cloud Computing, Data Science Engineering (Confluence). [Sl : sn], 2019. p. 687-692.

DÉBORDEMENT DE PILE. Enquête auprès des développeurs de débordement de pile 2021. 2021.
Disponible sur : <https://insights.stackoverflow.com/survey/2021#section-most-popular-technologies-web-frameworks >.

VUE. MEvan You, 2023. Disponible sur : <https://vuejs.org/>.

W3TECHS. Statistiques d'utilisation de JavaScript comme langage de programmation côté client sur les sites Web. 2022. Disponible sur : <https://w3techs.com/technologies/details/cp-javascript>.

ZIANI, A. Comparaison quantitative et qualitative des frameworks de développement Web. Mémoire (Master) — Université de Namur, 2021.

Annexes

ANNEXE A – Code

<https://github.com/lildiop2/tcc>

Source : de l'auteur (2023).