Implémentation :

* Frontend
* Adapter l’algorithme d’autocomplétion à la base de données
* Localisation
* Création de la base de données et les requêtes

Utilisation de la base de données nosql : ElasticSearch

C’est un outil puissant permettant de représenter les données en utilisant les JSON

* On installe ElasticSearch sur le site officiel, on lance le .bat et on se connecte sur le port 9200 à partir de là on peut effectuer des requêtes en modifiant juste l’url : ceci fonctionne comme une API.
* Pour faciliter les requêtes on a besoin d’installer l’utilitaire curl qui permet d’effectuer les requêtes en ligne de commande et de passer le fichier qui contient un JSON qui est enfait un query dans lequel on configure tous les critères afin d’obtenir les résultats les plus précis.
* Les attributs du query sont : « must », « must\_not », « filter », « should » ou encore « match », « match\_phrase », « range » et d’autres critères plus internes sur les attributs de index (table en nosql) tels que : fields.nomduchamp
* Utilisation des agrégats afin d’effectuer des requêtes complexes
* Distribution sur les serveurs ou machines différentes : scalabilté horizontale
* Privilégie la réactivité à la cohérence des donnéeS ²s
* La notion de nœuds et de clusters sur Elasticsarch : les nœuds sont les instances individuelles d'Elasticsearch qui forment un cluster, tandis que le cluster est l'ensemble de ces nœuds travaillant ensemble pour fournir des capacités de recherche et d'analyse de données distribuées.
* Kibana : Visualisation des données

Thème : RECHERCHE DE VOYAGE

Situation problème : Un utilisateur doit pouvoir trouve

Types de voyages : Voyage urbain, voyage inter urbain,

Moyens déplacement : Pour le voyage urbain : motos, taxis, voitures personnelles

¨Pour le voyage interurbain : agences de voyage, voitures personnelles.

Proposition de solution :

Objectifs :

* Etudier la théorie des graphes et comment ça pourra nous aider pour la résolution du problème
* Utilisation des cartes
* Recherche des moyens d’optimiser la recherche

CONNEXION A LA BASE DE DONNEES

* Installation de PostrgreSql, StackBuilder, pgAdmin4, postgis extension
* Review of sql syntaxes and how to use postgresql with the shell (psql) and pgadmin4
* Downloading from openstreetmap the binary of Cameroon roads and towns in the PBF format
* Downloading osm2pgsql and adding to systel’s environment variables with the command:

SETX PATH chemin\dacces\a\osm2pgsgl through the terminal

It is a tool that enables us to convert binaries of different formats into a database table

* Loading the data from PBF into a table in our database:

osm2pgsql -c -d postgres -U postgres -H localhost -P 5432 -W cameroon-latest.osm.pbf -S D:\Set-ups\osm2pgsql-bin\default.style

STRUCTURE DU RAPPORT DE MODELISATION

1. Analyse formelle :
2. Description du problème :
3. Proposition de solution : Recherche et tri efficace, algorithme d’autocomplétion
4. Description de l’utilisation de la solution
5. Résultats obtenus après la recherche :

Bien sûr, voici comment vous pourriez aborder chaque section avec des exemples concrets pour votre projet de recherche de voyage à partir d'une base de données :

1. \*\*Introduction\*\*--Eleonor

- Contexte : Présentez brièvement l'industrie du voyage, les défis actuels liés à la recherche de voyages et l'importance de votre projet pour améliorer cette expérience.

- Objectifs : Expliquez que le rapport vise à détailler la modélisation de la solution qui facilitera la recherche de voyages à partir d'une base de données centralisée.

- Aperçu : Décrivez brièvement les sections suivantes du rapport pour donner au lecteur une idée claire de son contenu.

2. \*\*Analyse des besoins\*\*---Eleonor

- Besoins fonctionnels : Exemple - Permettre aux utilisateurs de rechercher des vols en fonction de la destination, de la date et du prix.

- Besoins non fonctionnels : Exemple - Le système doit être convivial, offrir des temps de réponse rapides et garantir la sécurité des données utilisateur.

- Acteurs et rôles : Exemple - Utilisateurs finaux (voyageurs), administrateurs (gestion de la base de données), fournisseurs de voyages (compagnies aériennes, hôtels, etc.).

- Exigences système : Exemple - Le système doit être accessible depuis les navigateurs web courants et les appareils mobiles.

2. Principe de la solution : -- Lado

On explique en détail le principe derrière notre algorithme, notions d’indexation, modeles mathematiques utilisés

3. \*\*Modèle conceptuel\*\*--Audrey

- Description du domaine : Exemple - Le domaine du voyage inclut des entités telles que Utilisateur, Voyage, Destination, Compagnie Aérienne, Hôtel, etc.

On va citer toutes les entités et les décrire : leurs utilités sans leurs attributs

Entités : PointGeographique(id, coordonnees\_x, coordonnnees\_y, nom, type)

Agence(id, nom)

StationAgence(id, nom)

Voyage(id, HeureDepart, listePointDArrets, tarif)

Vehicule(id, marque, type, nbPlaces)

- Entités principales : Exemple - Utilisateur (ID, nom, email), Voyage (ID, destination, date de départ, prix), Destination (ID, nom, pays), Compagnie Aérienne (ID, nom, classe de vol), Hôtel (ID, nom, emplacement).

On décrit les entités avec leurs attributs.

- Relations : Exemple - Un utilisateur peut réserver plusieurs voyages, une destination peut être associée à plusieurs voyages, etc.

- Diagramme de classe conceptuel : Incluez un diagramme qui illustre les entités principales et leurs relations.

4. \*\*Modèle logique\*\*--Audrey

- Transformation : Convertissez le modèle conceptuel en modèle relationnel en créant des tables pour chaque entité et en définissant les relations à l'aide de clés primaires et étrangères.

- Exemple de tables : Utilisateur (ID, Nom, Email), Voyage (ID, ID\_Utilisateur, ID\_Destination, Date\_Depart, Prix), Destination (ID, Nom, Pays), etc.

- Clés et contraintes : Exemple - La clé primaire de la table Utilisateur est l'ID, et la clé étrangère dans la table Voyage est ID\_Utilisateur liée à Utilisateur(ID).

5. \*\*Modèle physique\*\*--Brayan

- Choix du SGBD : Exemple - MySQL pour la gestion de la base de données en raison de sa fiabilité et de sa compatibilité avec les langages de programmation courants.

- Schéma physique : Créez les tables dans le SGBD en utilisant les structures définies dans le modèle logique.

- Optimisations : Ajoutez des index sur les colonnes fréquemment utilisées pour améliorer les performances des requêtes de recherche.

6. \*\*Architecture système\*\*--Brayan

- Vue d'ensemble : Décrivez l'architecture à trois niveaux avec un frontend pour l'interface utilisateur, un backend pour la logique métier et la gestion de la base de données.

- Composants : Exemple - Interface utilisateur (HTML, CSS, JavaScript), Serveur backend (Node.js, Express), Base de données (MySQL).

- Communication : Utilisez des API REST pour la communication entre le frontend et le backend.

7. \*\*Sécurité et confidentialité\*\*--Brayan

- Stratégies de sécurité : Exemple - Authentification des utilisateurs avec des identifiants et mots de passe sécurisés, autorisation basée sur les rôles pour limiter l'accès aux données sensibles.

- Gestion des données : Chiffrez les informations sensibles telles que les mots de passe utilisateur dans la base de données pour garantir leur confidentialité.

- Conformité : Assurez-vous que le système respecte les réglementations telles que le RGPD en termes de protection des données personnelles.

8. \*\*Tests et validation\*\*

- Plan de test : Exemple - Testez l'inscription utilisateur, la recherche de voyages, la réservation de voyages pour garantir le bon fonctionnement du système.

- Résultats : Documentez les résultats des tests et les corrections apportées aux éventuels problèmes identifiés.

- Qualité des données : Vérifiez la qualité des données stockées dans la base de données pour éviter les erreurs et les incohérences.

9. \*\*Déploiement et maintenance\*\*

- Stratégie de déploiement : Exemple - Déployez l'application sur un serveur web en utilisant Docker pour la gestion des conteneurs.

- Procédures de maintenance : Planifiez des sauvegardes régulières de la base de données et des mises à jour logicielles pour assurer la stabilité et la sécurité du système.

- Surveillance : Utilisez des outils de surveillance pour surveiller les performances du système et détecter les problèmes potentiels.

10. \*\*Conclusion\*\*--Eléonor

- Résumé : Faites un récapitulatif des principales étapes de modélisation et de développement du système de recherche de voyage.

- Réalisations : Mettez en avant les réalisations par rapport aux objectifs initiaux et les bénéfices attendus pour les utilisateurs.

- Perspectives : Mentionnez les améliorations futures possibles et les recommandations pour optimiser le système.

11. \*\*Annexes\*\*

- Diagrammes supplémentaires : Ajoutez des diagrammes détaillés tels que les diagrammes de séquence pour illustrer le flux des opérations.

- Documentation technique : Incluez des documents techniques comme le schéma de la base de données, les API endpoints, etc.

- Références : Citez toutes les sources utilisées pour la modélisation et le développement du système.

En suivant cette structure, votre rapport de modélisation offrira une vue complète et détaillée de votre solution de recherche de voyage, de la conception à la mise en œuvre, en passant par les aspects de sécurité, de performance et de maintenance.

MODELISATION

1. Objet Voyage{

Point de depart;

Point d’arrivée;

Moyen de transport ;

Points d’arrêt ;

Tarif}

Conclusion

En conclusion, ce rapport de modélisation a été élaboré pour présenter notre solution innovante de recherche de voyages urbains et interurbains à partir d'une base de données centralisée. Notre projet s'inscrit dans le cadre d'une application de voyages complète, offrant aux utilisateurs la possibilité de trouver et de réserver facilement des déplacements en fonction de leurs besoins spécifiques.

À travers l'analyse des besoins, nous avons identifié les fonctionnalités essentielles de notre application, telles que la recherche avancée, les filtres de recherche, et la présentation détaillée des informations. Ces fonctionnalités sont conçues pour garantir une expérience utilisateur optimale, en offrant des résultats pertinents, des options de personnalisation, et une interface conviviale.

La modélisation conceptuelle, logique et physique de notre solution a permis de définir une structure robuste pour la gestion des données de voyage, en utilisant des techniques d'indexation, des algorithmes de recherche efficaces, et des filtres avancés pour optimiser les performances et la pertinence des résultats.

L'architecture système mise en place, avec ses différents composants et son approche centrée sur l'utilisateur, assure la fiabilité, la sécurité et la scalabilité de notre application. Nous avons également abordé les aspects de sécurité et de conformité pour garantir la protection des données utilisateur et le respect des normes de confidentialité.

Enfin, nos tests et validations ont permis de valider le bon fonctionnement de notre solution, en mettant en évidence sa capacité à répondre aux besoins des utilisateurs et à fournir des résultats précis et fiables.

En résumé, notre projet de recherche de voyages à partir d'une base de données représente une avancée significative dans l'industrie du voyage, en offrant une solution moderne, efficace et personnalisée pour simplifier le processus de réservation et améliorer l'expérience globale des voyageurs. Nous sommes convaincus que cette solution contribuera à répondre aux défis actuels du secteur du voyage et à satisfaire les attentes croissantes des utilisateurs en matière de déplacements.