Projet 4 – projet final. CHOIX A. Moteur de recherche d'une bibliothèque.

BM Bui-Xuan

Application web/mobile : Une application web/mobile se distingue d'une page web par ses fonctionnalités avec interaction utilisateur alors qu'une page web est par définition statique. Cette interaction peut être explicite dans le sens où le résultat retourné par l'application est en réponse immédiate à une action que l'utilisateur vient d'effectuer. Dans le cas d'une fonctionnalité dérivée d'une action, ou des actions plus anciennes, on parlera de fonctionnalité implicite.

Bibliothèque et moteur de recherche : Dans ce projet, on appelle bibliothèque toute base de données de taille assez conséquente de documents textuels. Un tel exemple est la base de Gutenberg disponible à l'adresse suivante : `http://www.gutenberg.org/` où les documents sont présentés sous différentes formes, dont le format textuel. A l'instar de la base Gutenberg, une bibliothèque peut contenir des dizaines de milliers de documents, rendant une recherche manuelle impossible. Un moteur de recherche dans une base de données est une application web/mobile permettant à l'utilisateur d'accéder plus rapidement à un document textuel par recherche de mot-clef, ou par historique de recherche en lien avec ce document (système de *best-seller*, de suggestion de publicité ciblée, de favoris utilisateur, et cetera).

Analyse de pertinence et analyse de performance : La pertinence d'un moteur de recherche est un critère subjectif déterminé à la suite de test utilisateur. En revanche, la performance d'un moteur de recherche est un critère objectif mesurable par des tests de charge sur différentes volumes de données.

1 L'énoncé du projet final – CHOIX A

ATTENTION : Il est obligatoire d'indiquer à la première page du rapport le "CHOIX A" ou le "CHOIX B" dans le rendu de projet.

Il s'agit de proposer une application web/mobile de moteur de recherche de document dans une bibliothèque de livres sous format textuel. Pour cela, la première étape est de s'occuper de la couche *data* du projet, i.e. la construction d'une bibliothèque personnelle de livres, stockés sous forme de documents textuels. La taille minimum de la bibliothèque est 1664 livres. La taille minimum de chaque livre est 10000 mots. Dans un second temps, il s'agit de construire une application web/mobile (couches *serveur* et *client*) afin de munir sa bibliothèque d'un moteur de recherche. Chaque groupe est libre de déterminer la présentation *frontend* et les *userstory* de son application, cependant, les fonctionnalités principales de l'application doivent obligatoirement comprendre :

— **Une fonctionnalité explicite de "Recherche" :** Recherche de livre par mot-clef. A la suite d'une entrée texte S de l'utilisateur, l'application retourne la liste de tous les documents textuels dont la table de l'indexage contient la chaîne de caractères S.

— **Une fonctionnalité explicite de "Recherche avancée" :** Recherche dite "avancée" de livre par RegEx. A la suite d'une entrée texte $RegEx$ de l'utilisateur, l'application retourne : soit la liste de tous les documents textuels dont la table de l'indexage contient une chaîne de caractères S qui vérifie l'expression régulière $RegEx$; soit la liste de tous les documents textuels dont le contenu textuel contient une chaîne de caractères S qui vérifie l'expression régulière $RegEx$ (attention à la dégradation en performance).

— **Une fonctionnalité implicite de classement :** Classement des réponses des deux fonctionnalités de recherche précédentes. A la suite d'une réponse à la fonctionnalité recherche, l'application retourne la liste des documents triée par un certain critère de pertinence : nombre d'occurrences du mot-clef dans le document ; par indice de centralité décroissant dans le graphe de Jaccard, et cetera. Le critère de centralité est laissé libre à chaque groupe d'interpréter. Cependant, il est obligatoire d'utiliser au moins un critère parmi les trois indices vu en cours, à savoir *closeness*, *betweenness*, ou *pagerank*. Concernant l'indice de centralité utilisé, il également est important de bien le décrire dans le rapport. On veillera en particulier à expliciter la définition, le calcul, ainsi que le résultat de cet indice sur des extraits pertinents des > 1664 livres présents dans la base de données.

— **Une fonctionnalité implicite de suggestion :** Suggestion de réponses similaires à la dernière recherche. A la suite d'une réponse à la fonctionnalité recherche, l'application retourne également : soit la liste des sommets dans le graphe de Jaccard qui sont voisins avec les deux ou trois documents textuels les plus pertinents qui contiennent le mot-clef ; soit la liste des documents textuels les plus choisis (les plus *cliqués* par d'autres utilisateurs) parmi les sommets dans le graphe de Jaccard qui sont voisins avec les deux ou trois documents textuels les plus pertinents qui contiennent le mot-clef.

Le rendu de ce projet est avant tout une application web/mobile. Il est donc important lors de la démonstration (cf. présentation orale) que l'application web/mobile en question fonctionnent sur plusieurs machines clientes en même temps (laptops, smartphones, etc).

Un effort particulier doit être mis sur la phase de test, la rédaction du rapport, et la présentation orale du projet. Il y aura ainsi trois notations distinctes pour le rendu de ce projet : une note pour la réalisation ($\approx 20\%$ de la note totale de l'UE) ; une note pour le rapport ($\approx 10\%$ de la note totale de l'UE) ; et une note pour la présentation orale du projet ($\approx 10\%$ de la note totale de l'UE).

Rapport : On veillera à expliciter les points suivants, pour chaque algorithme implémenté (y compris les algorithmes décrits pendant les séances de cours tels de ceux dans le livre Aho-Ullman, algorithme KMP, ceux de *radix tree*, et ceux de Jaccard, *closeness*, *betweenness* ou *pagerank*) :

— définition du problème et la structure de données utilisée.

— analyse et présentation théorique des algorithmes connus dans la littérature.

— argumentation concise appuyant toute appréciation, amélioration, ou critique à propos de ces algorithmes existants dans la littératures.

— partie test : méthode d'obtention des *testbeds*. En particulier, il est important de citer la provenance des fichiers sources, e.g. pour les documents textuels.

— test de performance : mieux vaut privilégier les courbes, diagrammes bâton (moyenne + écart type) et diagrammes de fréquence, plutôt qu'exhiber les colonnes de chiffre sans fins...

— une discussion sur les résultats de test de performance est toujours la très bienvenue.

— présenter également les tests d'utilisateur et test de pertinence, s'il y a eu lieu

— conclusion et perspectives sur ce problème de moteur de recherche de bibliothèque.

Il est prudent d'avoir entre 10 et 15 pages pour un rapport avec un contenu moyen. Le nombre de pages recommandé pour ce rapport est 12 pages. Il convient de bien respecter cette limitation : les pages 15+ ne seront pas lues !

Présentation : La présentation doit comprendre exactement 20 minutes. On veillera à expliciter les points suivants :

— Partie introductive ($\approx$ 7 minutes) : membres de l'équipe ; objectif du projet ; analyse fonctionnelle (usecase et/ou userstory) ; illustration avec un *wireframe*, on peut penser à utiliser les outils tels que balsamiq, framerX, sketch, etc ; technologies utilisées (+justification pourquoi, ici, un tableau comparatif entre les technologies les plus fréquentes sur le marché est recommandé) ; éventuellement un bilan de ressource déployée pour le projet (Gantt chart ou Scrum backlog ou simplement un tableau résumant le nombre de jour-hommes par tâche principale du projet).

— Partie technique ($\approx$ 10 minutes) : architecture générale de l'application ; présentation de la couche *client* (nombre de vues, rendu graphique vs. maquette de départ, etc) ; présentation de la couche *data* (technologies de stockage utilisée pour le document textuel, les index, le graphe de Jaccard) ; présentation de la couche *serveur* (algorithmes) ;

— Partie démonstration et conclusion du projet ($\approx$ 3 minutes) : une démonstration est obligatoire pour ce projet où l'application web/mobile doit fonctionner sur au minimum 2 machines différentes connectées entre elles. Le mieux serait d'utiliser 3 machines pour la démonstration : une machine servant de *serveur* de l'application ; deux machines *client* connectant à l'application via un navigateur et/ou une App Android et/ou une App iOS/iPadOS.

Il est prudent d'avoir entre 18 et 22 pages de support de présentation pour 20 minutes de présentation orale.

Contraintes :

— A réaliser en trinôme ou en binôme.

— Archiver la totalité du rendu en un seul fichier compressé contenant de la documentation (rapport $\approx$ 10-15 pages), la codebase contenant du code source commenté, et tout ce dont on juge utile à la lecture du projet sans toute fois dépasser la dizaine de Méga-octet.

— Envoyer ce fichier à `buixuan@lip6.fr`, 3 emails maximum par groupe. L'utilisation des hébergeurs en ligne (drive et compagnies) est proscrite. La nomination de préférence est daar-CHOIX-A-ou-CHOIX-B-projet-final-NOM1-NOM2-NOM3.piki, où piki peut être un élément de $\{tgz, zip, rar, 7z, etc\}$.

— Présentation orale : 01 Février 2021, le même jour que le dernier cours de l'UE. L'ordre de passage sera déterminé pendant les séances TME10-13.

— Deadline rapport : 07 Février 2021, 23h59, cachet de serveur de messagerie faisant foi. Pénalité de retard : malus de $2^{h/24}$ points pour h heures de retard.