IA Générative et Squash

Club Utilisateurs Squash

05/11/24

Auteurs : *Florian Gautier, Sophie Depierre, Laurent Mazuré*

henix

CARTOGRAPHIE DES CAS D'USAGE PERTINENTS

CHOISIR LES CAS D'USAGES OÙ L'IA GÉNÉRATIVE APPORTE DES GAINS

Planification

Objectif de l'étape:

Anticiper les futurs travaux de qualification et estimer la charge prévisionnelle associée

Principaux cas d'usage :

- Macro-planning
- Macro-chiffrage

Cadrage

Objectif de l'étape:

Définir une stratégie de recette pertinente et organiser les travaux de qualification

Principaux cas d'usage :

- Conception du plan de test

Implémentation

Objectif de l'étape:

Recenser les exigences à couvrir et concevoir les cas de test valorisés correspondants

Principaux livrables :

- Rédaction des Exigences
- Identification et rédaction de cas de tests
- Automatisation des scénarios
- Gestion et génération Jeux de données et des bouchons
- Définition du plan d'exécution

Exécution

Objectif de l'étape:

Exécuter en cycle itératif et dans un ordre pertinent les cas de test prévus dans le plan d'exécution

Principaux cas d'usage :

- Interprétation des résultats d'exécution
- Analyse de logs croisés (Anomalies et incidents)
- Déclaration d'anomalies

Finalisation

Objectif de l'étape:

Émettre le PV de recette et faire le bilan des travaux de qualification

Principaux cas d'usage :

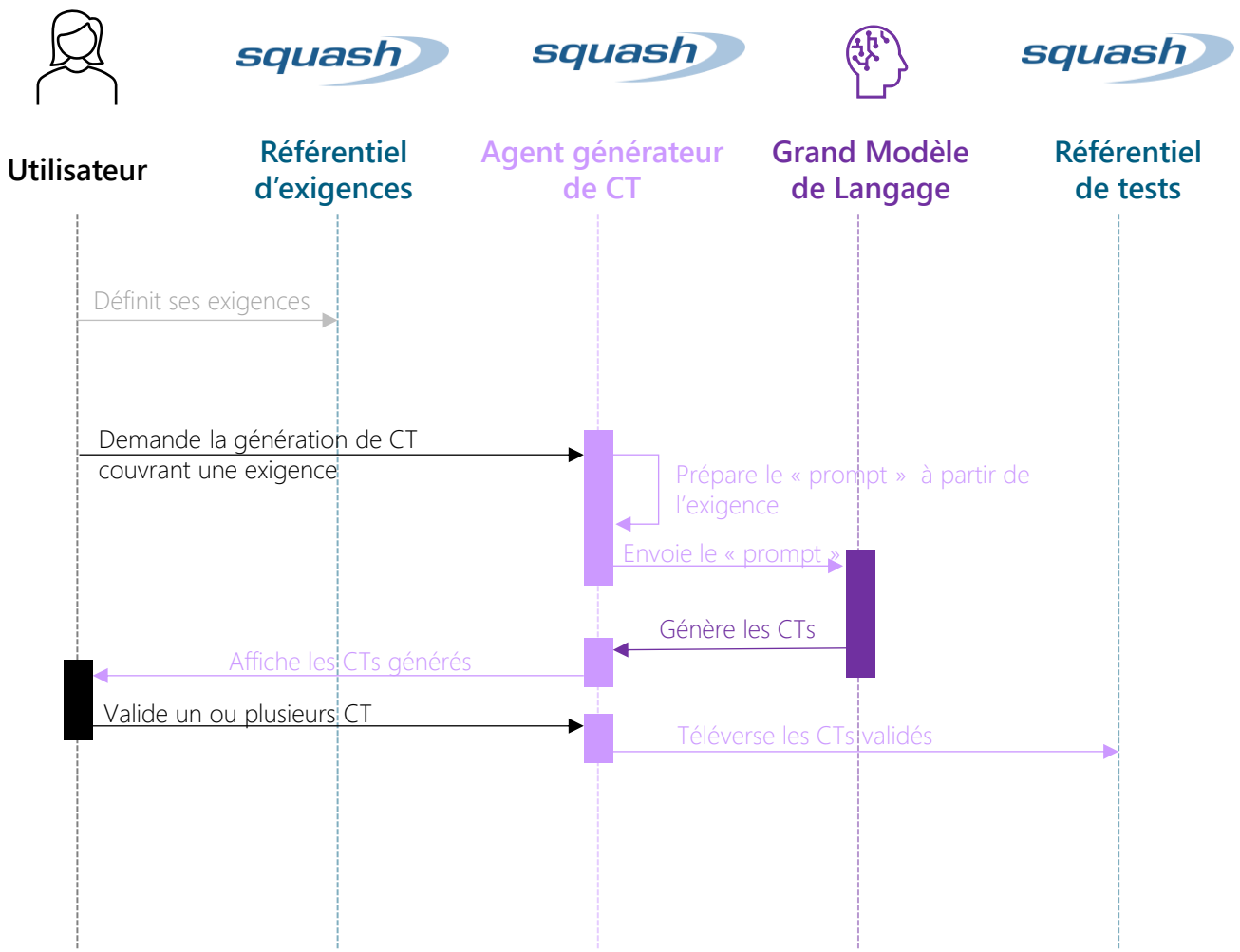
- Capitalisation et mis à jour du patrimoine de test (dont TNR)
- Analyse du patrimoine et plan d'amélioration continue

- Attention à ne pas se limiter aux cas d'usage de l'automatisation des tests. L'IA générative est un accélérateur. C'est un effet cumulé des gains sur l'ensemble des étapes et pratiques de tests qui va in fine amener à plus d'automatisation des tests. D'autant plus que certains cas d'utilisation de l'automatisation (Génération de code) seront adressés au niveau SI pour d'autres pratiques (développement par exemple).
- Il s'agit de cartographier puis de sélectionner le premier UC en fonction
  - De la criticité du cas d'utilisation
  - Des gains potentiels
  - De la complexité de mise en œuvre (génération des jeux de données, nécessite déjà de gérer correctement la donnée)

1 Identification et génération de cas de test

GÉNÉRATION DE CAS DE TEST ASSISTÉE PAR IA GÉNÉRATRICE

UN PREMIER PAS POUR VALIDER LA PERTINENCE DU CAS D'UTILISATION



- **Objectif :**
  Générer à partir d'une exigence les cas de tests la couvrant et pour chaque cas de test (CT) le scénario correspondant.

- **Principe :**
  Intégrer un « agent » via lequel un utilisateur va demander la génération des cas de test à partir d'une exigence du référentiel d'exigences.

  L'agent, en requêtant le modèle de langage (via « prompt » pré-configuré*), fait une proposition de cas de tests à l'utilisateur.

  L'utilisateur peut alors valider tout/ou partie des test générés, qui sont alors stockés dans le référentiel de tests.

- **Avantages :**
  L'agent est une façade du modèle de langage qui masque la complexité de l'interaction avec le modèle et facilite la prise en main par les utilisateurs

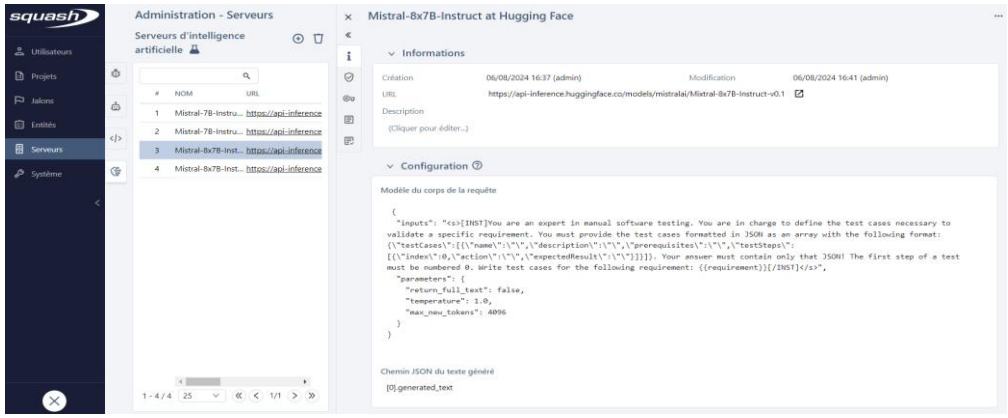
  Il est possible de tirer parti des nombreux frameworks d'intégration aux modèles de langage (Langchain) ou des Référentiels de tests proposant cette feature (ex Squash) pour :
  - Faciliter le prototypage de l'agent/la solution
  - Rester agnostique aux différents modèles de langage: Les modèles évoluent vite et le choix d'un modèle est un choix stratégique au niveau du SI. Il est alors nécessaire de créer des outils le plus agnostique possible aux différents modèles.

- **Limitations :**
  L'**exigence** servant à la génération doit être **autoporteuse** ou le langage de modèle va être sujet aux « hallucinations ». Les réponses d'un modèle générique ne peuvent pas prendre compte le « contexte » de l'entreprise (Cas de test existent, documents et exigences liés).

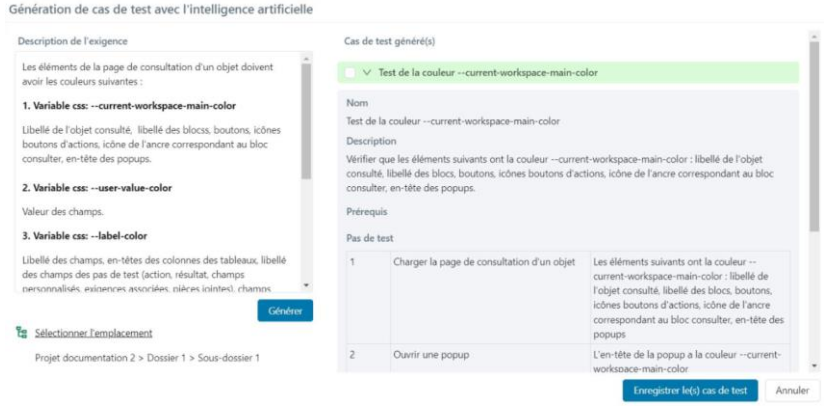
Extrait de la release note Squash 7.0

Aide à l'écriture des cas de test par l'intelligence artificielle

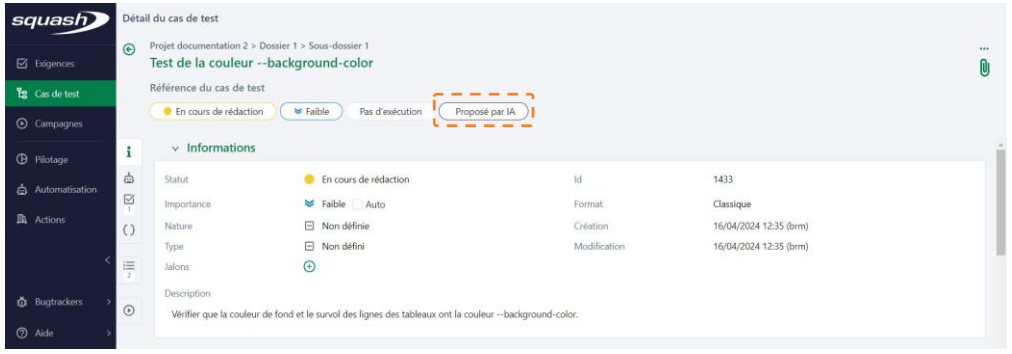
- Compatibilité avec tous les modèles (déclaration et configuration de serveur d'IA dans Squash)
- Activation par projet



- Identification et Génération de cas de test par l'IA à partir d'une exigence



- Recherche des cas de test générés par l'IA



EXEMPLE DE PROMPT GÉNÉRANT UN CAS DE TEST VIA LE MODÈLE GPT-4

```
{
  "model": "gpt-4-1106-preview",
  "messages": [
    {
      "role": "system",
      "content": "Vous êtes un expert en test logiciel manuel. Vous êtes chargé de définir les cas de test nécessaires pour valider une exigence spécifique.
      Ces cas de test doivent obligatoirement être rédigés en français. Vous devez fournir les cas de test au format JSON sous forme de tableau avec le
      format suivant : \"{\"testCases\":[{\"name\":\"\",\"description\":\"\",\"prerequisites\":\"\",\"testSteps\":[{\"index\":0,\"action\":\"\",
      \"expectedResult\":\"\"}]}]}\". Votre réponse ne doit contenir rien d'autre que ce JSON ! La première étape d'un test doit être numérotée 0."
    },
    {
      "role": "user",
      "content": "Écrivez des cas de test pour l'exigence suivante : {{requirement}}"
    }
  ],
  "temperature": 0.9,
  "top_p": 0.9,
  "n": 1,
  "stream": false,
  "max_tokens": 2000,
  "presence_penalty": 0,
  "frequency_penalty": 0
}
```

Dépasser les limites des modèles génériques et les limites d'un éditeur

- La grande limitation de la génération de cas de test via une exigence, vient des hallucinations des modèles de langages génériques lorsque l'exigence n'est pas autoporteuse.
- Plusieurs techniques permettent de dépasser cela c.f. (R.A.G, Fine tunning, …)
- Nous étudions deux autres pistes car « faciles » à mettre en place.

	Context caching	« Tools »
Principe	Tirer parti du fait que la limite sur du nombre de tokens en cache tends à ne plus être un facteur limitant. L'agent met alors en cache une sélection/l'ensemble des document. https://ai.google.dev/gemini-api/docs/caching?lang=python	Proposer des « hooks » pour que le LLM "demande les infos" qui lui manquent. https://python.langchain.com/docs/how_to/#tools
Coûts	Le coût intervient sur la mise en place/le rafraichissement du cache. Mais il peut-être mutualisé.	Le coût est lié à l'implémentation du « endpoint ». Comment fournir les informations pertinentes lorsque le LLM le demande ?

- La difficulté réside dans la construction de patrimoines de tests conséquents qui puissent servir de données de tests viables pour ces fonctionnalités d'IA

Suite au prochain épisode

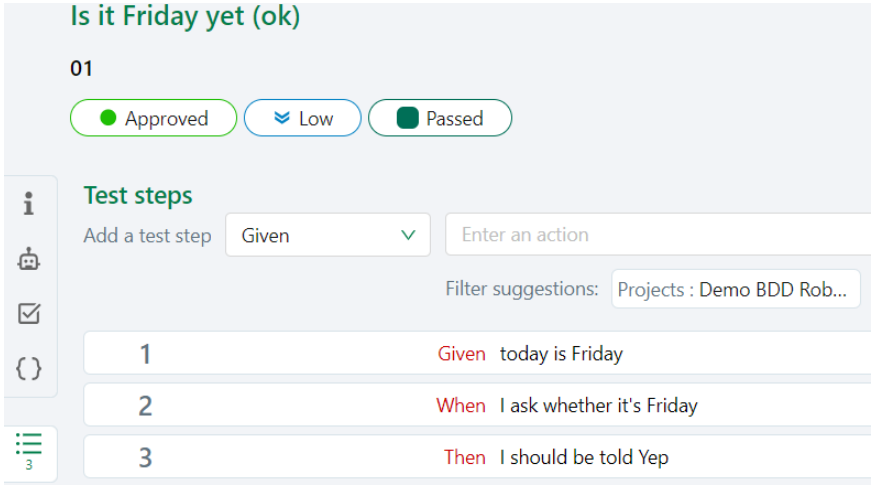
henix

2 Etude sur la facilitation de l'écriture des Cas de test Gherkin/BDD

- Auto-complétion des pas de tests (phrases d'actions) qui sont sémantiquement proches
- Analyse et suppression des doublons d'une bibliothèque de phrases d'actions

Facilitation de l'écriture des Cas de Test BDD/Gherkin

- Problématique: La mutualisation de phrase d'action (en vue ou non de l'automatisation) conduit à la mise en place de bibliothèques. Comment faciliter alors la recherche dans un « bibliothèque » d'action connues et éviter les (la création de) doublons ?



```java
@Given("^today is Sunday$")
public void today_is_Sunday()  {
    today = "Sunday";
}

@When("^I ask whether it's Friday yet$")
public void i_ask_whether_it_s_Friday_yet()  {
    actualAnswer = IsItFriday.isItFriday(today);
}

@Then("^I should be told \"([^\"]*)\"$")
public void i_should_be_told(String expectedAnswer) {
    assertEquals(expectedAnswer, actualAnswer);
   System.out.println("FRIDAYYY");
}
```

- 1er axe: Proposer une auto-complétion « intelligente » à partir d'une recherche dans une bibliothèque d'actions connues par termes similaires et pas seulement par caractères identiques
  - Soit en demandant, via prompt et/ou agent, au LLM de fournir les résultats les plus pertinents
  - Soit en utilisant des techniques de NLP (embedding)
- 2e axe: Identification des doublons dans la bibliothèque d'actions connues

Recherche de phrases d'actions similaires
Première méthode: Via prompt « direct » du LLM

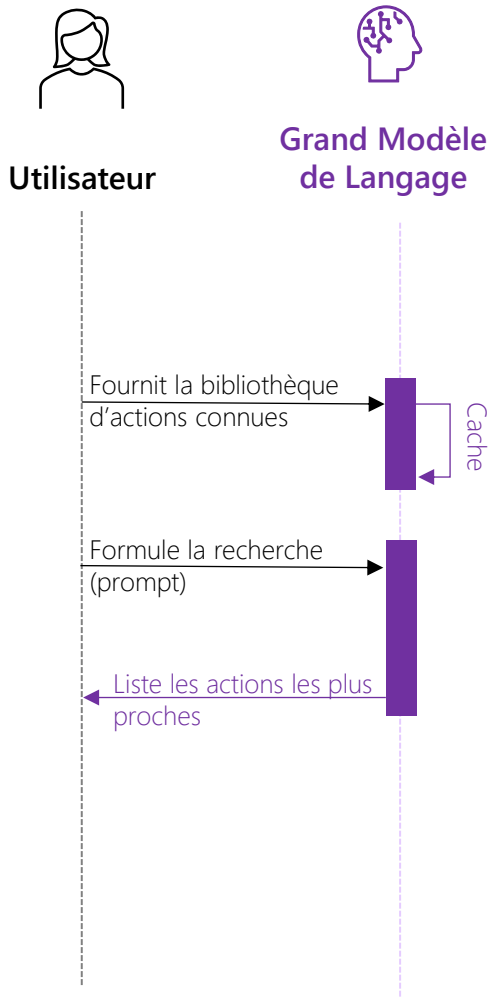
- Principe: Fournir au LLM un prompt avec la liste des phrases d'actions connues (directement dans le prompt ou en contexte) en demandant quelle phrase d'action se trouve sémantiquement le plus proche de celle que je recherche. On compare alors avec ce que « nous humains » nous aurions proposé.
- Jeux de données

Bibliothèque d'action

```
"action_words": [
  {"id": 1, "word": "Je me connecte"},
  {"id": 2, "word": "Je suis en ligne"},
  {"id": 3, "word": "Je suis hors ligne"},
  {"id": 4, "word": "Je crée un dossier"},
  {"id": 5, "word": "J'ouvre un répertoire"},
  {"id": 6, "word": "Je change d'onglet"},
  {"id": 7, "word": "Je saisis mes informations"},
  {"id": 8, "word": "J'ai le rôle observateur"},
  {"id": 9, "word": "La page est affichée"},
  {"id": 10, "word": "Le bouton est bleu"},
  {"id": 11, "word": "La page est en français"},
  {"id": 12, "word": "Je clique sur le bouton"},
  {"id": 13, "word": "J'ajoute un fichier"},
  {"id": 14, "word": "Je cherche un produit"},
  {"id": 15, "word": "J'ajoute un produit au panier"},
  {"id": 16, "word": "La page de paiement s'affiche"},
  {"id": 17, "word": "le paiement est validé"},
  {"id": 18, "word": "je crée un compte"},
  {"id": 19, "word": "j'obtiens un token"},
  {"id": 20, "word": "je modifie la couleur"}
]
```

Recherches

Contenu de la recherche	Résultat attendu
je crée un répertoire	je crée un dossier
je suis connecté	je suis en ligne
je passe à la page suivante	je change d'onglet
j'obtiens du bleu	le bouton est bleu
j'obtiens un bouton bleu	le bouton est bleu
je suis observateur	j'ai le rôle obesrvateur
la commande est terminée	le paiement est validé
je fais une recherche	je cherche un produit
authentification	je me connecte / je suis en ligne / je saisis mes informations / je crée un compte / j'obtiens un token
je change la langue	la page est en français

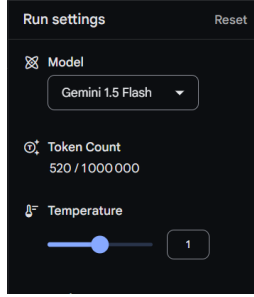


- Configuration:
  - Grand Modèle de Language: Gemini flash 1.5 https://aistudio.google.com/app/prompts
  - prompt:

```
donne en JSON les 3 actions les plus proches de '{{recherche}}' dans cette liste : {
  "action_words": [
    {"id": 1, "word": "Je me connecte"},
    {"id": 2, "word": "Je suis en ligne"},
     ...
    {"id": 20, "word": "je modifie la couleur"}
  ]
}
```

  - System instructions: « Réponds uniquement avec le JSON, sans explications »
- Remarque: Ici comme dans le cas de génération de cas de test, le prompt peut-être « caché » aux utilisateurs via un « agent »

RÉSULTATS GEMINI 1.5 FLASH CHATBOT -TEMP 1.0



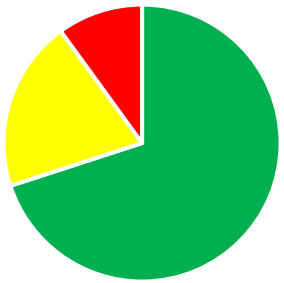
Moyenne tokens par réponse : 519

Contenu de la recherche	Résultat attendu	Résultat obtenu	Position bon résultat	Pertinence premier résultat si pas attendu
je crée un répertoire	je crée un dossier	je crée un dossier	1	-
je suis connecté	je suis en ligne	Je me connecte	2	acceptable
je passe à la page suivante	je change d'onglet	je change d'onglet	1	-
j'obtiens du bleu	le bouton est bleu	je modifie la couleur	2	acceptable
j'obtiens un bouton bleu	le bouton est bleu	Je clique sur le bouton	2	peu pertinent
je suis observateur	j'ai le rôle obesrvateur	j'ai le rôle observateur	1	-
la commande est terminée	le paiement est validé	le paiement est validé	1	-
je fais une recherche	je cherche un produit	je cherche un produit	1	-
authentification	je me connecte / je suis en ligne /je saisis mes informations / je crée un compte / j'obtiens un token	je me connecte / je suis en ligne /je saisis mes informations / je crée un compte / j'obtiens un token	1	-
je change la langue	la page est en français	la page est en français	1	-

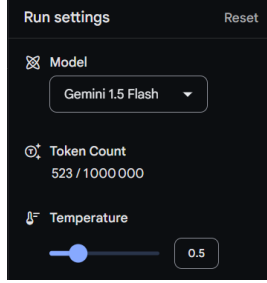
70% de résultats OK

20% de résultats où la bonne proposition est en 2e résultats et la première proposition reste pertinente

10% de résultats où la bonne proposition est en 2e résultats et la première proposition est peu pertinente



RÉSULTATS GEMINI 1.5 FLASH CHATBOT -TEMP 0.5



Moyenne tokens par réponse : 519

Contenu de la recherche	Résultat attendu	Résultat obtenu	Position bon résultat	Pertinence premier résultat si pas attendu
je crée un répertoire	je crée un dossier	je crée un dossier	1	-
je suis connecté	je suis en ligne	Je me connecte	2	acceptable
je passe à la page suivante	je change d'onglet	je change d'onglet	1	-
j'obtiens du bleu	le bouton est bleu	je modifie la couleur	2	acceptable
j'obtiens un bouton bleu	le bouton est bleu	Je clique sur le bouton	2	peu pertinent
je suis observateur	j'ai le rôle obesrvateur	j'ai le rôle observateur	1	-
la commande est terminée	le paiement est validé	la page de paiement s'affiche	2	acceptable
je fais une recherche	je cherche un produit	je cherche un produit	1	-
authentification	je me connecte / je suis en ligne /je saisis mes informations / je crée un compte / j'obtiens un token	je me connecte / je suis en ligne /je saisis mes informations / je crée un compte / j'obtiens un token	1	-
je change la langue	la page est en français	je change d'onglet	2	peu pertinent

50% de résultats OK
30% de résultats où la bonne proposition est en 2e résultats et la première proposition reste pertinente
20% de résultats où la bonne proposition est en 2e résultats et la première proposition est peu pertinente

Recherche de phrases d'actions similaires

Recherche de documents avec représentations vectorielles continue « Embedding Google »

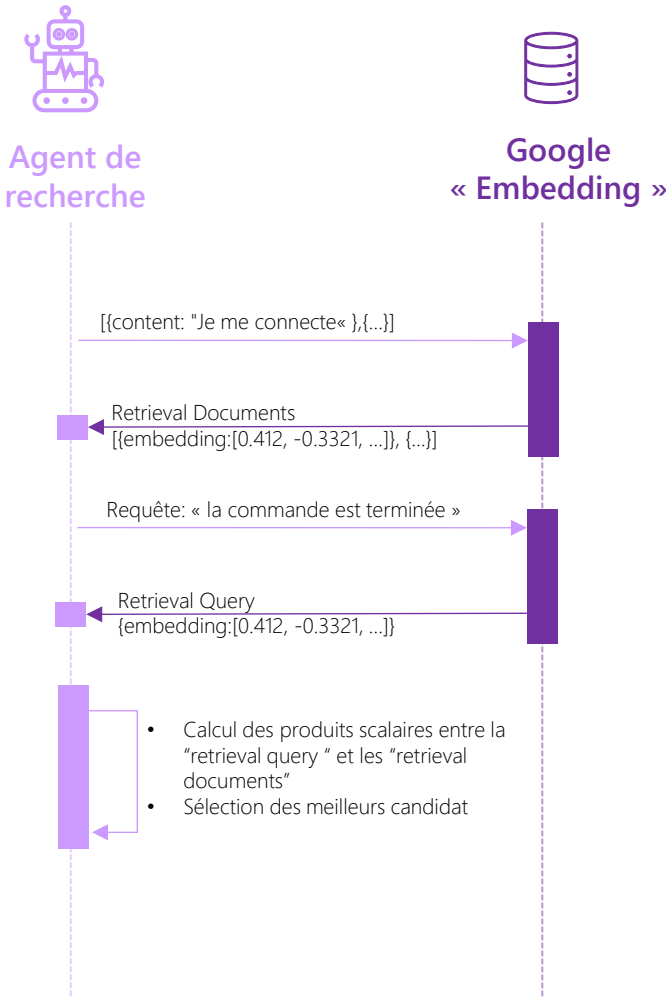
- Principe: Tirer parti des méthodes de « Natural Langage Processing ». Vectoriser les phrases d'actions ainsi que la recherche. Via « scoring » et produit scalaire retrouver les « documents » les plus proches de la recherche.
- Prototype: Pour une mise en place rapide, utilisation des bibliothèques « Embedding » de Google (**google.generativeai**) le modèle **text-embedding-004** mpour générer des représentations vectorielles continues des documents (bibilothèque d'actions) et des query (recherche) https://ai.google.dev/gemini-api/tutorials/document_search?hl=fr
- Détails
  - Embedding de la bibliothèque en tant que "retrieval_documents«
  - Embedding de la recherche en tant que "retrieval_query«
  - Produit scalaire « simple » pour trouver les "retrieval documents" les plus proche
  - Les trois premiers résultats sont retenus.
  - Jeux de données:

Bibliothèque d'action

```
"action_words": [
  {"id": 1, "word": "Je me connecte"},
  {"id": 2, "word": "Je suis en ligne"},
  {"id": 3, "word": "Je suis hors ligne"},
  {"id": 4, "word": "Je crée un dossier"},
  {"id": 5, "word": "J'ouvre un répertoire"},
  {"id": 6, "word": "Je change d'onglet"},
  {"id": 7, "word": "Je saisis mes informations"},
  {"id": 8, "word": "J'ai le rôle observateur"},
  {"id": 9, "word": "La page est affichée"},
  {"id": 10, "word": "Le bouton est bleu"},
  {"id": 11, "word": "La page est en français"},
  {"id": 12, "word": "Je clique sur le bouton"},
  {"id": 13, "word": "J'ajoute un fichier"},
  {"id": 14, "word": "Je cherche un produit"},
  {"id": 15, "word": "J'ajoute un produit au panier"},
  {"id": 16, "word": "La page de paiement s'affiche"},
  {"id": 17, "word": "le paiement est validé"},
  {"id": 18, "word": "je crée un compte"},
  {"id": 19, "word": "j'obtiens un token"},
  {"id": 20, "word": "je modifie la couleur"}
]
```

Recherches

Contenu de la recherche	Résultat attendu
je crée un répertoire	je crée un dossier
je suis connecté	je suis en ligne
je passe à la page suivante	je change d'onglet
j'obtiens du bleu	le bouton est bleu
j'obtiens un bouton bleu	le bouton est bleu
je suis observateur	j'ai le rôle obesrvateur
la commande est terminée	le paiement est validé
je fais une recherche	je cherche un produit
authentification	je me connecte / je suis en ligne / je saisis mes informations / je crée un compte / j'obtiens un token
je change la langue	la page est en français



RÉSULTATS NLP EMBEDDING GOOGLE TEXT

Contenu de la recherche	Résultat attendu	Résultat obtenu	Position bon résultat	Pertinence premier résultat si pas attendu	score résultat attendu
je crée un répertoire	je crée un dossier	j'ouvre un répertoire	2	acceptable	0.62
je suis connecté	je suis en ligne	Je me connecte	2	acceptable	0.6
je passe à la page suivante	je change d'onglet	la page de paiement s'affiche	10	acceptable	0.56
j'obtiens du bleu	le bouton est bleu	le bouton est bleu	1	-	0.61
j'obtiens un bouton bleu	le bouton est bleu	le bouton est bleu	1	-	0.72
je suis observateur	j'ai le rôle obesrvateur	j'ai le rôle obesrvateur	1	-	0.71
la commande est terminée	le paiement est validé	j'ajoute un produit au panier	2	acceptable	0.61
je fais une recherche	je cherche un produit	je crée un dossier	5	sans rapport	0.55
authentification	je me connecte / je suis en ligne /je saisis mes informations / je crée un compte / j'obtiens un token	je me connecte / je suis en ligne /je saisis mes informations / je crée un compte / j'obtiens un token	1	-	entre 0.5 et 0.4
je change la langue	la page est en français	je change d'onglet	15	sans rapport	0.51

40% de résultats OK (dont un dont le score est inférieur à 0.5)

40% de résultats où la bonne proposition n'est pas en première proposition et la première proposition reste pertinente

20% de résultats où la bonne proposition n'est pas en première proposition et la première proposition est peu pertinente

RÉSULTATS PRÉLIMINAIRES

- Chatbot IA : Fonctionne sur le principe mais est un poil « overkill », car l'on utilise tout un LLM pour faire de la recherche alors que la partie « générative » ne vous interesse pas.

- Embedings: Utiliser les techniques de « NLP » (qui sont la première couche des LLM)
  - Fonctionnement validé sur le principe.
  - Pour la recherche, les « stop words » (je, le, la, les, un, …) ne sont pas pertinents
  - De façon préliminaire, il semblerait que l'on se raccroche aux branches avec les fautes d'orthographe et les « début » de mots =>On peut s'en servir comme auto-complétion.

- Industrialisation de l'étude (en cours) – Effet de différent « Embedings »
  - Génération d'un jeu de données plus conséquents (via IA générative) (Bibliothèques, recherches et attendus associés)
  - Utilisation de Chroma DB (base de données vectorielle)
  - Choix et tests de plusieurs « embedding » (Word2Vec, Glove, BERT, CamemBERT …)
  - Besoin d'aller plus loin que le « produit scalaire » pour récupérer les termes sémantiquement proches