Cahier des Charges : Chatbot Intelligent pour l'ISET

1. Introduction et Contexte

1.1 Présentation du projet

Le projet consiste à développer un chatbot intelligent destiné au site web de l'Institut Supérieur des Études Technologiques (ISET). Ce chatbot servira d'assistant virtuel pour guider les utilisateurs (principalement des étudiants) dans leur navigation sur le site et leur fournir des réponses pertinentes à leurs questions concernant les cours, programmes universitaires, procédures administratives, et autres informations relatives à l'institut.

1.2 Objectifs principaux

- Répondre de manière pertinente et précise aux questions des utilisateurs sur des sujets liés à l'ISET
- Optimiser le temps de recherche en guidant les utilisateurs vers les pages spécifiques du site correspondant à leurs besoins
- Améliorer l'expérience utilisateur sur le site web de l'ISET
- Fournir une assistance 24/7 aux étudiants et visiteurs du site

1.3 Périmètre du projet

Le projet couvre la conception, le développement, l'intégration et l'évaluation d'un chatbot basé sur des techniques de NLP et de Machine Learning (sans deep learning), ainsi que son intégration au site web existant de l'ISET.

1.4 Sources de données

- Dataset privé de questions-réponses personnalisées relatif aux informations du site de l'ISET (données préparées dans "data_option1.csv")
- Potentiellement des datasets benchmark complémentaires pour enrichir les capacités du modèle

2. Exigences Fonctionnelles

2.1 Capacités du chatbot

- **Compréhension des requêtes textuelles** : Le chatbot doit être capable de comprendre les questions posées en langage naturel par les utilisateurs.
- **Réponse aux questions générales** : Fournir des réponses précises sur les cours, programmes, horaires, procédures administratives, événements, etc.
- **Navigation assistée** : Diriger l'utilisateur vers la page spécifique du site web qui répond à sa requête.

- **Historique de conversation** : Maintenir le contexte de la conversation pour une expérience plus cohérente.
- **Fallback approprié** : En cas d'incapacité à répondre, rediriger vers les ressources appropriées ou suggérer une reformulation.

2.2 Interface utilisateur

- **Chat intégré** : Interface de chat intuitive intégrée au site web de l'ISET.
- **Accessibilité** : Interface accessible sur tous les appareils (desktop, tablette, mobile).
- **Indicateur de statut** : Affichage visuel indiquant que le chatbot traite la requête.
- **Suggestions de questions** : Proposer des questions fréquentes pour guider l'utilisateur.
- **Possibilité de feedback** : Permettre à l'utilisateur de noter la pertinence des réponses.

2.3 Administration du chatbot

- **Dashboard administrateur** : Interface permettant aux administrateurs de visualiser les métriques d'utilisation et de performance.
- **Mise à jour des connaissances** : Possibilité d'ajouter de nouvelles paires question-réponse à la base de connaissances.
- **Analyse des conversations** : Outils pour identifier les questions fréquentes et les requêtes sans réponse satisfaisante.

3. Exigences Non Fonctionnelles

3.1 Performance

- **Temps de réponse** : Le chatbot doit répondre en moins de 2 secondes dans 95% des cas.
- **Disponibilité** : Disponibilité du service à 99,5% minimum.
- **Scalabilité** : Capacité à gérer plusieurs conversations simultanées (minimum 100).

3.2 Sécurité

- **Protection des données** : Aucune donnée personnelle sensible ne doit être stockée sans consentement explicite.
- **Authentification** : Accès sécurisé au dashboard d'administration.
- **Audit trail** : Journalisation des actions administratives pour la traçabilité.

3.3 Maintenabilité

- **Documentation** : Documentation complète du code et des processus.

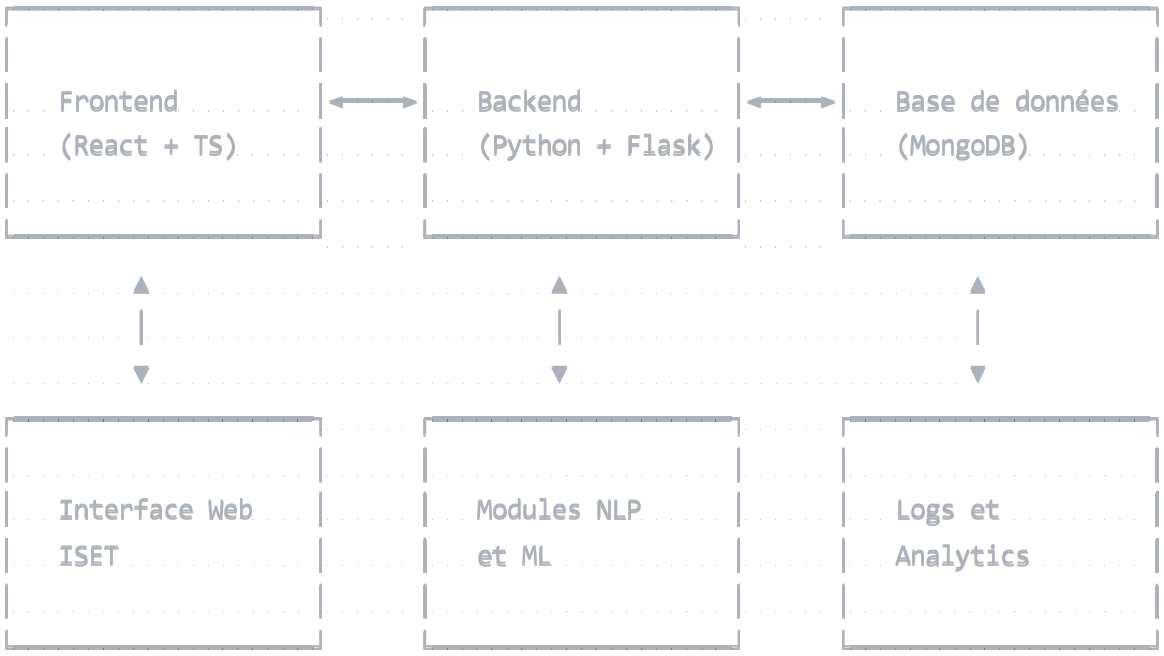
- **Tests automatisés** : Couverture de tests unitaires et d'intégration d'au moins 80%.
- **Gestion de versions** : Utilisation de Git pour le contrôle de version.

3.4 Extensibilité

- **Architecture modulaire** : Faciliter l'ajout de nouvelles fonctionnalités.
- **API extensible** : Permettre l'intégration future avec d'autres systèmes.
- **Support multilingue** : Conception permettant l'ajout futur de langues supplémentaires.

4. Architecture Technique Détaillée

4.1 Vue d'ensemble de l'architecture



4.2 Composants du Frontend (React + TypeScript)

- **Interface de chat** : Composant React permettant l'interaction avec le chatbot
- **Gestionnaire d'état** : Utilisation de Redux ou Context API pour la gestion de l'état de l'application
- **Composants UI** : Design system pour une expérience utilisateur cohérente
- **Adaptateurs API** : Services pour la communication avec le backend
- **Gestionnaire de routes** : Pour la navigation entre les différentes pages du dashboard administrateur

4.3 Composants du Backend (Python + Flask)

- **API RESTful** : Endpoints pour les interactions avec le frontend
- **Gestionnaire d'authentification** : Gestion des sessions et des droits d'accès

- **Pipeline NLP** : Traitement et analyse des requêtes utilisateurs
- **Moteur de ML** : Classification et génération de réponses
- **Connecteurs de base de données** : Interfaces pour l'accès aux données stockées
- **Système de logging** : Enregistrement des interactions et des erreurs

4.4 Base de données

- **MongoDB** : Base de données NoSQL pour le stockage flexible des paires question-réponse et des conversations
- **Collections**:
  - Questions-réponses (kb_items)
  - Conversations (conversations)
  - Feedback utilisateur (feedbacks)
  - Utilisateurs admin (admin_users)
  - Configuration du système (system_config)
  - Logs d'utilisation (usage_logs)

4.5 Communication API

- **REST API** : API RESTful pour la communication entre le frontend et le backend
- **WebSockets** : Pour les mises à jour en temps réel des conversations
- **Format des échanges** : JSON standardisé
- **Authentification** : JWT (JSON Web Tokens) pour sécuriser les échanges

4.6 Intégration au site ISET

- **Widget embarquable** : Script JavaScript pour intégrer facilement le chatbot sur n'importe quelle page
- **Style configurable** : Adaptation du style du chatbot à la charte graphique du site
- **Configuration contextuelle** : Possibilité de pré-configurer le chatbot selon la page où il est intégré

5. Choix Technologiques et Bibliothèques

5.1 Frontend

- **Framework** : React 18+ avec TypeScript
  - *Justification* : Écosystème riche, typage statique pour réduire les erreurs, performances optimales
- **État global** : Redux Toolkit

- *Justification* : Gestion prévisible de l'état, debugging facilité, middleware pour les effets de bord

- **UI/UX** :
  - Material-UI ou Chakra UI
    - *Justification* : Composants accessibles, personnalisables et réactifs
  - Styled-components
    - *Justification* : Stylisation basée sur les composants pour une meilleure isolation

- **Requêtes API** : Axios ou React Query
  - *Justification* : Gestion simplifiée des requêtes HTTP, cache intelligent, retry automatique

- **Tests** : Jest + React Testing Library
  - *Justification* : Standard de l'industrie, test orienté comportement plutôt que implémentation

5.2 Backend

- **Framework** : Flask 2.x
  - *Justification* : Légèreté, flexibilité, adapté aux API RESTful

- **Extensions Flask** :
  - Flask-RESTful pour la structuration de l'API
  - Flask-CORS pour la gestion des requêtes cross-origin
  - Flask-JWT-Extended pour l'authentification
  - Flask-SocketIO pour les WebSockets

- **ORM/ODM** : PyMongo ou MongoEngine
  - *Justification* : Interface Python adaptée à MongoDB

- **NLP** :
  - NLTK : Bibliothèque complète pour le prétraitement textuel
  - Scikit-learn : Pour les algorithmes de ML classiques
  - SpaCy : Analyse linguistique avancée
  - Gensim : Pour les modèles Word2Vec ou FastText

- **Vectorisation** :
  - TF-IDF de scikit-learn
  - Count Vectorizer pour les approches Bag-of-Words
  - FastText pour les embeddings sans deep learning

- **Classification** :
  - SVM, Random Forest, ou KNN de scikit-learn

- **Recherche sémantique** :
  - Whoosh : Moteur de recherche pure Python
  - Annoy ou FAISS : Pour la recherche rapide des vecteurs similaires

5.3 Base de données

- **SGBD** : MongoDB 5.x
  - *Justification* : Flexibilité du schéma, performances élevées, facilité d'évolution
- **Indexation** : Indexation textuelle de MongoDB pour la recherche de contenu

5.4 Déploiement et DevOps

- **Conteneurisation** : Docker + Docker Compose
  - *Justification* : Isolation des environnements, déploiement simplifié
- **CI/CD** : GitHub Actions ou GitLab CI
  - *Justification* : Automatisation des tests et du déploiement
- **Monitoring** : Prometheus + Grafana
  - *Justification* : Surveillance des performances et de la disponibilité

6. Étapes Détaillées de Développement

6.1 Phase de préparation et d'analyse

1. **Analyse du dataset**
   - Exploration des données dans data_option1.csv
   - Identification des catégories de questions
   - Analyse statistique des données (distribution des sujets, longueur des questions/réponses)
2. **Définition des intents et entités**
   - Mapping des intentions utilisateurs (demande d'information, navigation, procédure)
   - Identification des entités critiques (cours, département, document administratif, etc.)
3. **Conception UX/UI**
   - Wireframes de l'interface utilisateur
   - Flows conversationnels
   - Maquettes du dashboard d'administration

6.2 Phase de développement NLP/ML

1. **Prétraitement des textes**

- Tokenization des questions et réponses
- Suppression des stopwords
- Lemmatisation ou stemming
- Normalisation des textes (casse, accents, caractères spéciaux)

2. **Vectorisation et représentation**
   - Implémentation de TF-IDF pour la vectorisation des questions
   - Création des matrices de caractéristiques
   - Alternative avec Word Embeddings (FastText pré-entraîné pour le français)
3. **Développement des modèles ML**
   - Implémentation d'un classifieur d'intention (SVM ou Random Forest)
   - Développement d'un extracteur d'entités (règles + ML)
   - Création d'un système de ranking de réponses basé sur la similarité cosinus
   - Système de fallback intelligent
4. **Moteur de recherche**
   - Indexation des pages du site ISET
   - Développement d'un moteur de recherche avec Whoosh
   - Mécanisme de mapping entre questions et pages du site
5. **Évaluation et optimisation des modèles**
   - Split train/test du dataset
   - Validation croisée
   - Optimisation des hyperparamètres
   - Tests de performance et pertinence

6.3 Phase de développement Backend

1. **Mise en place de l'architecture Flask**
   - Structure du projet
   - Configuration des environnements (dev, test, prod)
   - Mise en place des routes API
2. **Développement des endpoints API**
   - Endpoints pour le chat (soumission de question, historique)
   - Endpoints pour le dashboard administrateur

- Authentification et autorisations

3. **Intégration du pipeline NLP/ML**
   - Encapsulation des modèles ML dans des services
   - Développement du processeur de messages
   - Gestion du contexte conversationnel
4. **Système de logging et analytics**
   - Mise en place du système de logs
   - Développement des métriques d'utilisation
   - Création des alertes de performances

6.4 Phase de développement Frontend

1. **Structure de l'application React**
   - Configuration du projet (CRA ou Vite)
   - Mise en place de TypeScript
   - Structure des dossiers et composants
2. **Développement de l'interface de chat**
   - Composant de chat réutilisable
   - Gestion des états de conversation
   - Animations et transitions pour une UX fluide
3. **Dashboard administrateur**
   - Tableaux de bord et statistiques
   - Interface de gestion des connaissances
   - Visualisation des métriques
4. **Tests et optimisation**
   - Tests unitaires des composants
   - Tests d'intégration
   - Optimisation des performances (lazy loading, memoization)

6.5 Phase d'intégration et déploiement

1. **Intégration frontend-backend**
   - Configuration CORS
   - Tests d'intégration end-to-end

- Gestion des erreurs

2. **Développement du script d'intégration**
   - Widget JavaScript pour l'intégration au site ISET
   - Système de configuration du widget
3. **Conteneurisation**
   - Création des Dockerfiles
   - Configuration Docker Compose
   - Scripts de build et déploiement
4. **Mise en place du CI/CD**
   - Configuration des pipelines
   - Tests automatisés
   - Déploiement automatique

7. Mécanismes d'Intégration au Site Web de l'ISET

7.1 Widget JavaScript

```javascript
// Exemple simplifié du widget d'intégration
(function(w, d, s, o, f, js, fjs) {
    w['ISETChatWidget'] = o;
    w[o] = w[o] || function() { (w[o].q = w[o].q || []).push(arguments) };
    js = d.createElement(s), fjs = d.getElementsByTagName(s)[0];
    js.id = o; js.src = f; js.async = 1;
    fjs.parentNode.insertBefore(js, fjs);
}(window, document, 'script', 'isetChat', 'https://chat.iset.tn/widget.js'));

isetChat('init', { theme: 'light', position: 'bottom-right' });
```

7.2 Integration contextuelle

- **Data attributes** : Utilisation d'attributs data-* pour configurer le chatbot selon la page
- **Contexte automatique** : Détection automatique de la page courante pour contextualiser les réponses
- **Paramètres d'URL** : Possibilité d'initialiser le chatbot avec des paramètres spécifiques via l'URL

7.3 Adaptabilité visuelle

- **Thèmes configurables** : Light/dark mode et personnalisation des couleurs
- **Responsive design** : Adaptation à tous les formats d'écran
- **Modes d'affichage** : Mode plein écran, mode réduit, mode flottant

7.4 Hooks d'intégration

- **Événements JavaScript** : Publication d'événements pour permettre l'interaction avec le site hôte
- **Callbacks** : Fonctions de callback pour des actions spécifiques (ouverture, fermeture, nouvelle réponse)
- **API côté client** : Méthodes JavaScript pour contrôler le chatbot depuis le site

8. Plan d'Évaluation des Performances

8.1 Métriques d'évaluation technique

- **Précision du modèle** : Taux de bonnes réponses (accuracy, precision, recall, F1-score)
- **Temps de réponse** : Latence moyenne et percentiles (p50, p90, p99)
- **Utilisation des ressources** : CPU, mémoire, réseau
- **Taux d'erreur** : Pourcentage de requêtes en erreur

8.2 Métriques d'expérience utilisateur

- **Taux de satisfaction** : Pourcentage de réponses notées positivement
- **Taux de complétion** : Pourcentage de conversations aboutissant à une résolution
- **Engagement** : Nombre moyen de messages par conversation
- **Taux de rebond** : Pourcentage d'utilisateurs quittant après la première réponse

8.3 Méthodes d'évaluation

- **Tests A/B** : Comparaison de différentes versions du modèle
- **Évaluation humaine** : Panel d'utilisateurs testant le système
- **Tests de non-régression** : Vérification que les nouvelles versions n'altèrent pas les performances
- **Benchmark** : Comparaison avec des solutions existantes

8.4 Outils d'évaluation

- **Dashboard de métriques** : Tableau de bord pour visualiser les performances en temps réel
- **Rapports automatisés** : Génération hebdomadaire de rapports de performance
- **Alertes de dégradation** : Système d'alerte en cas de baisse significative des performances

9. Optimisations et Améliorations Possibles

9.1 Support multilingue

- **Pipeline NLP multilingue** : Extension du pipeline pour supporter le français et l'anglais
- **Détection automatique de la langue** : Identification de la langue de la requête
- **Ressources linguistiques** : Dictionnaires et corpus spécifiques à chaque langue

9.2 Système de feedback avancé

- **Feedback contextualisé** : Questions spécifiques selon le type de réponse
- **Apprentissage continu** : Utilisation du feedback pour améliorer automatiquement les réponses
- **Modération du feedback** : Interface de validation des suggestions d'amélioration

9.3 Graphe de connaissances

- **Représentation structurée** : Organisation des connaissances en graphe
- **Relations sémantiques** : Modélisation des relations entre entités
- **Navigation conceptuelle** : Exploration des sujets connexes

9.4 Mode d'auto-apprentissage

- **Apprentissage supervisé actif** : Identification des questions mal répondues pour amélioration
- **Extraction automatique** : Génération de nouvelles paires question-réponse à partir du contenu du site
- **Suggestions de connaissances** : Proposition de nouvelles entrées de connaissances aux administrateurs

9.5 Personnalisation utilisateur

- **Profils utilisateurs** : Adaptation des réponses selon l'historique de l'utilisateur
- **Préférences mémorisées** : Mémorisation des sujets d'intérêt
- **Recommandations personnalisées** : Suggestions proactives basées sur le profil

9.6 Intégration avancée

- **Intégration avec le système d'authentification** : Réponses personnalisées selon le statut étudiant
- **API pour applications mobiles** : Extension du service aux applications mobiles de l'ISET
- **Intégration avec les systèmes de notification** : Alertes et rappels via le chatbot

10. Calendrier Prévisionnel et Répartition des Tâches

10.1 Planning global

- **Phase 1 (2 semaines)** : Analyse et préparation
- **Phase 2 (4 semaines)** : Développement des composants NLP/ML
- **Phase 3 (3 semaines)** : Développement du backend
- **Phase 4 (3 semaines)** : Développement du frontend
- **Phase 5 (2 semaines)** : Intégration et tests
- **Phase 6 (1 semaine)** : Déploiement et documentation
- **Phase 7 (1 semaine)** : Formation et transfert de compétences

10.2 Répartition détaillée des tâches

Semaines 1-2 : Analyse et préparation

- **Tâche 1.1** : Analyse du dataset et des besoins (2j)
- **Tâche 1.2** : Définition des intents et entités (3j)
- **Tâche 1.3** : Conception UX/UI (5j)
- **Tâche 1.4** : Architecture technique détaillée (3j)
- **Livrable** : Document de spécifications techniques et maquettes UX/UI

Semaines 3-6 : Développement NLP/ML

- **Tâche 2.1** : Prétraitement des textes (5j)
- **Tâche 2.2** : Vectorisation et représentation (5j)
- **Tâche 2.3** : Développement des modèles ML (8j)
- **Tâche 2.4** : Moteur de recherche (5j)
- **Tâche 2.5** : Évaluation et optimisation (5j)
- **Livrable** : Pipeline NLP/ML fonctionnel avec modèles entraînés

Semaines 7-9 : Développement Backend

- **Tâche 3.1** : Architecture Flask (3j)
- **Tâche 3.2** : Développement API (7j)
- **Tâche 3.3** : Intégration pipeline NLP/ML (5j)
- **Tâche 3.4** : Système de logging et analytics (5j)
- **Livrable** : API backend complète et documentée

Semaines 10-12 : Développement Frontend

- **Tâche 4.1** : Structure React/TypeScript (3j)
- **Tâche 4.2** : Interface de chat (7j)
- **Tâche 4.3** : Dashboard administrateur (7j)
- **Tâche 4.4** : Tests et optimisation (3j)
- **Livrable** : Application frontend complète

Semaines 13-14 : Intégration et tests

- **Tâche 5.1** : Intégration frontend-backend (3j)
- **Tâche 5.2** : Script d'intégration au site ISET (3j)
- **Tâche 5.3** : Tests d'intégration end-to-end (4j)
- **Livrable** : Solution complète intégrée et testée

Semaine 15 : Déploiement et documentation

- **Tâche 6.1** : Déploiement en production (2j)
- **Tâche 6.2** : Documentation utilisateur et administrateur (3j)
- **Livrable** : Solution déployée et documentée

Semaine 16 : Formation et transfert

- **Tâche 7.1** : Formation des administrateurs (2j)
- **Tâche 7.2** : Support post-déploiement (3j)
- **Livrable** : Rapport de clôture et transfert de compétences

10.3 Jalons critiques

- **J1 (fin semaine 2)** : Validation des spécifications et de l'architecture
- **J2 (fin semaine 6)** : Validation des modèles ML
- **J3 (fin semaine 9)** : Validation du backend API
- **J4 (fin semaine 12)** : Validation de l'interface utilisateur
- **J5 (fin semaine 14)** : Recette fonctionnelle complète
- **J6 (fin semaine 16)** : Mise en production et clôture du projet

11. Conclusion

Ce cahier des charges définit le cadre complet pour le développement d'un chatbot intelligent pour le site web de l'ISET, basé sur des techniques de NLP et de ML sans deep learning. L'architecture technique proposée, combinant un frontend React TypeScript et un backend Python Flask, permettra de créer une solution performante, évolutive et facilement maintenable.

Le projet respecte les contraintes initiales tout en proposant des axes d'amélioration qui pourront être implémentés progressivement pour enrichir les fonctionnalités du chatbot. Le planning proposé permet une réalisation progressive et maîtrisée, avec des jalons clairs pour suivre l'avancement du projet.

La solution finale offrira aux utilisateurs du site de l'ISET un assistant virtuel capable de répondre à leurs questions et de les guider efficacement, améliorant ainsi significativement leur expérience utilisateur.