Dossier de Conception et de Spécification : Application Help-Desk

1.0 Introduction et Vision du Produit

L'application "Help-Desk" est conçue comme un compagnon de soutien personnel, un espace numérique intime et sécurisé. Cette section inaugurale établit la mission fondamentale du projet, ses objectifs directeurs et son public cible. Elle pose ainsi les bases philosophiques et éthiques sur lesquelles reposent toutes les décisions techniques et fonctionnelles qui suivront, garantissant que la technologie reste au service de l'utilisateur, dans le respect absolu de sa vie privée.

1.1 Contexte et Objectifs Clés

La mission principale de l'application est d'offrir un espace sûr pour aider les utilisateurs à suivre leur humeur et leurs habitudes, et à dialoguer avec un assistant IA bienveillant, sans jamais compromettre la confidentialité de leurs données. Pour concrétiser cette vision, le projet est guidé par quatre objectifs fondamentaux :

- **Respect de la vie privée** : Cet objectif se traduit par une exigence non négociable : 100% des données de l'utilisateur (profil, humeurs, notes, conversations) doivent rester stockées localement sur son appareil, sans aucune transmission vers des serveurs externes.
- **Simplicité et accessibilité** : L'interface utilisateur doit être intuitive et facile à prendre en main. Cette exigence vise à minimiser la charge cognitive pour des personnes pouvant se trouver en situation de vulnérabilité.
- **Autonomie technologique** : L'intégration d'une intelligence artificielle fonctionnant localement via le moteur Ollama est une exigence clé. Cela garantit que les conversations restent entièrement privées et que l'application peut fonctionner sans connexion Internet ni dépendance à des API cloud tierces.
- **Soutien à l'introspection** : L'application doit fournir des outils de suivi et de visualisation qui aident activement les utilisateurs à mieux comprendre l'évolution de leurs émotions et de leurs habitudes quotidiennes.

1.2 Public Cible

L'application s'adresse principalement aux personnes confrontées à un Trouble du Déficit de l'Attention avec ou sans Hyperactivité (TDAH), à une dépression légère, à de l'anxiété ou à des difficultés émotionnelles du quotidien. Il est crucial de souligner que l'application est un *outil d'accompagnement et non un dispositif médical*.

1.3 Avertissement Important

Pour garantir l'intégrité éthique du projet, il est impératif de définir une limite claire à son champ d'application. L'application est un outil de soutien, et non un substitut à une expertise médicale.

Cette application est un outil de soutien personnel et ne remplace en aucun cas un suivi professionnel médical ou psychologique.

Cette vision claire et ces principes éthiques exigent une architecture technique rigoureuse et réfléchie, capable de garantir la robustesse, la sécurité et la confidentialité promises à l'utilisateur.

2.0 Architecture Générale et Socle Technologique

L'efficacité et la fiabilité de l'application Help-Desk reposent sur une architecture stratégiquement modulaire. La séparation claire des responsabilités entre l'interface utilisateur (UI), la logique métier (services) et la couche d'accès aux données garantit la maintenabilité à long terme, la robustesse face aux erreurs et l'évolutivité du projet. Ces qualités sont essentielles pour un outil destiné à être un compagnon fiable et durable pour ses utilisateurs.

2.1 Structure Modulaire du Projet

L'organisation des fichiers du projet matérialise cette approche modulaire, chaque dossier ayant un rôle précis et délimité.

```
help-desk/
├── main.py
├── requirements.txt
├── README.md
├── SECURITE.md
│
├── assets/
│   └── theme.css
│
├── data/
│   ├── journal.db
│   └── secret.key
│
├── db/
│   ├── database.py
│   └── models.py
│
├── services/
│   ├── chat_ai.py
│   ├── chat_service.py
│   ├── export_service.py
│   ├── habit_service.py
│   ├── mood_service.py
│   └── profile_service.py
│
├── ui/
│   ├── components.py
│   └── layout.py
│
└── utils/
    ├── dates.py
    ├── safety.py
    └── security.py
```

Le tableau suivant détaille la responsabilité de chaque composant principal de cette arborescence :

Dossier/Fichier	Responsabilité Principale
`main.py`	Point d'entrée de l'application. Gère la navigation principale et l'initialisation de la session utilisateur.
`db/`	Contient toute la logique relative à la base de données : connexion, création des tables et exécution des requêtes SQL.
`services/`	Renferme la logique métier de l'application. Certains fichiers, comme `habit_service.py` ou `profile_service.py`, témoignent d'un développement itératif et peuvent contenir des logiques vestigiales ou préparées pour de futures évolutions.
`ui/`	Gère la construction de l'interface utilisateur avec Streamlit, y compris la mise en page des différentes pages et les composants réutilisables.
`utils/`	Fournit des fonctions utilitaires transverses, telles que la gestion de la sécurité ou la détection de détresse émotionnelle.
`SECURITE.md`	Documente les mesures de sécurité implémentées et les bonnes pratiques recommandées pour la protection des données utilisateur.

2.2 Socle Technologique Détaillé

Le choix des technologies a été guidé par les principes de simplicité, de performance et de fonctionnement 100% local.

Technologie	Version	Rôle dans l'application	Documentation Officielle
Python	3.13	Langage de programmation principal pour toute la logique de l'application.	python.org
Streamlit	1.31.0+	Framework utilisé pour construire l'interface utilisateur web interactive et réactive.	docs.streamlit.io
SQLite	3	Système de gestion de base de données relationnelle locale, stockant les données dans un unique fichier.	sqlite.org
Ollama	-	Moteur permettant d'exécuter localement des modèles de langage (LLM) comme `llama3.1:8b`.	ollama.ai
Pandas	2.2.0+	Bibliothèque pour la manipulation et l'analyse des données, notamment pour les exports et les statistiques.	pandas.pydata.org
Matplotlib	-	Utilisé pour la génération des graphiques de visualisation de l'humeur.	matplotlib.org
FPDF2	-	Bibliothèque utilisée pour générer les rapports de suivi au format PDF.	py-pdf.github.io/fpdf2/
Cryptography	42.0.2+	Bibliothèque fournissant des outils cryptographiques, préparée pour un futur chiffrement des données.	cryptography.io

Ce socle technologique, et en particulier le choix de SQLite et d'Ollama, est la concrétisation directe des objectifs de confidentialité et d'autonomie définis dans la vision du produit. Il offre les fondations pour une gestion des données entièrement locale et sécurisée, dont la structure est détaillée ci-après.

3.0 Modèle de Données et Persistance Locale

Le choix stratégique de la base de données SQLite constitue la pierre angulaire de l'engagement de l'application en matière de confidentialité. En confinant toutes les informations dans un fichier unique (`journal.db`), cette architecture *de-risk* nativement le projet de toute vulnérabilité liée à une transmission réseau. Cette décision élimine les dépendances à une infrastructure externe et matérialise directement l'objectif d'"Autonomie technologique", garantissant à l'utilisateur un contrôle total et exclusif sur ses données.

3.1 Schéma de la Base de Données

La base de données est structurée autour de quatre tables principales, conçues pour modéliser les différentes facettes de l'expérience utilisateur.

Table `users`

- **Analyse des colonnes clés** : `prenom` permet de personnaliser l'accueil et les messages. Le champ `tdah` est de type `INTEGER` (utilisé comme un booléen, `0` pour faux et `1` pour vrai), un choix standard pour garantir la compatibilité maximale avec les bibliothèques SQLite.

```
CREATE TABLE IF NOT EXISTS users (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    prenom TEXT,
    birth_date TEXT,
    tags TEXT,
    tdah INTEGER DEFAULT 0,
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP
)
```

Table `mood`

- **Analyse des colonnes clés** : La contrainte `INTEGER NOT NULL` sur `mood_value` garantit l'intégrité de cette métrique essentielle pour la visualisation. Les colonnes `emotion` et `motivation` sont de type `TEXT` pour offrir une flexibilité maximale dans la description qualitative du ressenti.

```
CREATE TABLE IF NOT EXISTS mood (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    mood_value INTEGER NOT NULL,
    emotion TEXT,
    motivation TEXT,
    tags TEXT,
    notes TEXT,
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP
)
```

Table `tasks`

- **Analyse des colonnes clés** : `done` est un `INTEGER` utilisé comme un booléen (`0` ou `1`) pour une gestion simple et performante de l'état de la tâche. `created_at` est de type `DATE` plutôt que `DATETIME` pour optimiser le stockage et l'indexation, l'heure de création n'étant pas pertinente pour cette fonctionnalité.

```
CREATE TABLE IF NOT EXISTS tasks(
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    title TEXT NOT NULL,
    done INTEGER DEFAULT 0,
    created_at DATE DEFAULT CURRENT_DATE
)
```

Table `notes`

- **Analyse des colonnes clés** : Le champ unique `content` de type `TEXT` est choisi pour sa capacité à stocker de grandes quantités de texte sans contrainte de structure, ce qui est idéal pour un bloc-notes.

```
CREATE TABLE IF NOT EXISTS notes (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    content TEXT,
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP
)
```

3.2 Sécurisation de l'Accès aux Données

Au-delà du choix d'une base de données locale, des mesures de sécurité actives sont implémentées au niveau du système de fichiers. Avant toute connexion à la base, le système s'assure que les permissions du dossier `data/` sont fixées à `0o700` et celles du fichier `journal.db` à `0o600`.

Concrètement, cela signifie :

- **`0o700` pour `data/`** : Seul le propriétaire du compte utilisateur peut lire, écrire et accéder au contenu du dossier.
- **`0o600` pour `journal.db`** : Seul le propriétaire du compte utilisateur peut lire et écrire dans le fichier de la base de données.

Cette mesure constitue une première ligne de défense essentielle, empêchant d'autres utilisateurs sur la même machine d'accéder aux données sensibles. C'est sur cette base de données sécurisée que reposent toutes les fonctionnalités interactives de l'application.

4.0 Spécifications Fonctionnelles Détaillées

Le parcours utilisateur au sein de l'application a été conçu comme un flux logique et intuitif, orchestré par le point d'entrée `main.py`. Chaque interaction est pensée pour être simple, rassurante et utile. Cette section décompose chaque fonctionnalité majeure, depuis la première configuration de l'application jusqu'à l'utilisation quotidienne des outils de suivi et de soutien.

4.1 Parcours d'Intégration (Onboarding)

Le premier contact avec l'application est crucial. Il doit être simple et poser les bases d'une relation de confiance.

1. **Création du profil** : L'utilisateur est accueilli par un formulaire (`profil_form`) qui l'invite à saisir son prénom, sa date de naissance et des informations contextuelles (tags). Ces données, enregistrées via la fonction `save_profile_to_db`, servent à personnaliser l'expérience.
2. **Écran d'introduction** : Une fois le profil créé, un écran présente la philosophie de l'application, en insistant sur l'absence de pression et d'attentes de performance. La phrase clé résume cet esprit : *"Il n'y a pas de bonne ou mauvaise journée. L'important c'est d'avancer à ton rythme"*.

4.2 Outils de Journalisation Quotidienne

Après l'intégration, l'utilisateur accède au cœur de l'application : les outils de suivi journalier.

- **Saisie de l'Humeur** : Chaque jour, l'utilisateur est invité à remplir un formulaire pour évaluer son humeur (note de 1 à 10) et décrire son ressenti émotionnel et sa motivation. C'est la porte d'entrée vers le tableau de bord principal.
- **Gestion des Tâches** : Un module de calendrier permet de planifier et suivre les objectifs quotidiens. L'utilisateur peut ajouter une tâche, la consulter par date, la marquer comme terminée ou la supprimer.
- **Prise de Notes Rapides** : Un espace simple et accessible est disponible pour la prise de notes libres, permettant de capturer une pensée ou une émotion sans friction.

4.3 Module de Chat IA : "Mathi"

"Mathi" est un assistant conversationnel conçu pour offrir un soutien bienveillant et entièrement sécurisé, fonctionnant en local.

- **Rôle et Persona** : Le comportement de Mathi est rigoureusement défini par un `SYSTEM_PROMPT`. Elle se présente comme un "compagnon de soutien" dont le rôle est d'écouter, de valider les émotions et d'encourager. Ses interdictions sont strictes et fondamentales : ne jamais poser de diagnostic, ne jamais proposer de traitement et ne jamais se substituer à un thérapeute.
- **Intégration Technique** : L'IA est propulsée par le modèle `llama3.1:8b` exécuté via le moteur local `Ollama`. Pour garantir la fluidité de l'application, un `timeout` de 60 secondes est implémenté sur les appels au modèle.
- **Couche de Sécurité Émotionnelle** : Une protection cruciale est assurée par la fonction `detect_distress`. Celle-ci analyse les messages de l'utilisateur à la recherche de mots-clés sensibles ("suicide", "désespéré", etc.). Si une détresse est détectée, le message n'est pas envoyé à l'IA. À la place, une réponse de sécurité standard (`safety_response`) est affichée, encourageant l'utilisateur à contacter un professionnel ou un proche.

4.4 Tableau de Bord et Visualisation

Le tableau de bord (`render_dashboard`) centralise les outils et les informations dans une interface organisée par onglets :

1. **Aujourd'hui** : Affiche les tâches du jour et le module de notes rapides.
2. **Mathi** : Contient l'interface de chat avec l'IA.
3. **Historique** : Présente les visualisations et les historiques de données. Il inclut un graphique de l'humeur, des statistiques clés, l'historique des 20 dernières tâches (avec leur statut) et des 5 dernières notes prises par l'utilisateur.
4. **Export** : Permet d'extraire les données de l'application.

Dans l'onglet "Historique", la fonction `render_mood_summary` propose une visualisation claire de l'évolution de l'humeur sous forme d'un graphique en courbes, complété par des statistiques clés .

4.5 Export des Données : Reprendre le Contrôle

Cette fonctionnalité incarne le principe d'autonomie en permettant à l'utilisateur de reprendre le contrôle total de ses données pour les sauvegarder, les analyser ou les partager en toute sécurité avec un professionnel.

Format	Description
Export Excel (.xlsx)	Fournit les données brutes de l'application, organisées en feuilles (Humeurs, Tâches, Notes, Profil). Idéal pour une analyse détaillée ou une sauvegarde complète.
Export PDF (.pdf)	Génère un rapport de suivi synthétique et formaté, pensé pour être partagé. Il inclut le profil, des statistiques et les dernières entrées d'humeur et de notes.

L'ensemble de ces fonctionnalités a été conçu avec la sécurité comme fil conducteur, un principe fondateur qui mérite d'être détaillé de manière exhaustive.

5.0 Principes Fondateurs : Sécurité et Confidentialité

La sécurité et la confidentialité ne sont pas de simples fonctionnalités de l'application Help-Desk ; elles en sont les principes fondateurs. Chaque décision architecturale, chaque ligne de code a été subordonnée à l'objectif de protection absolue des données de l'utilisateur. La confiance est la base de l'aide que cet outil vise à apporter, et cette confiance se construit sur des garanties techniques tangibles.

5.1 Le Principe du Stockage 100% Local

L'architecture de l'application est fondée sur un principe de **'local-first' non négociable**. Toutes les données saisies par l'utilisateur sont stockées **uniquement et exclusivement** sur son appareil local. Aucune information personnelle, aucune note, aucune entrée d'humeur et aucune conversation n'est transmise sur Internet ou à des serveurs externes. Le modèle d'intelligence artificielle lui-même fonctionne en local, garantissant que même les discussions les plus sensibles restent entièrement privées et sous le contrôle de l'utilisateur.

5.2 Mesures de Sécurité Actives

Plusieurs couches de protection sont actives pour sécuriser les données locales :

1. **Permissions de Fichiers Restrictives** : Le dossier `data/` et le fichier de base de données `journal.db` sont créés avec des permissions qui en restreignent l'accès au seul compte utilisateur qui exécute l'application. Cette mesure protège les données contre les accès indésirables par d'autres utilisateurs sur la même machine.
2. **Absence de Transmission Externe** : L'application n'initie aucune connexion réseau pour transmettre des données utilisateur. Toutes les opérations, y compris l'interaction avec l'IA "Mathi", sont autonomes et locales.
3. **Protection et Responsabilisation des Exports** : L'utilisateur est informé que les fichiers PDF et Excel exportés contiennent ses données personnelles. Il est de sa responsabilité de conserver ces fichiers en lieu sûr et de ne les partager qu'avec des personnes de confiance, comme des professionnels de santé.

5.3 Chiffrement : Une Fonctionnalité d'Avenir

Bien que non encore active par défaut, une fonctionnalité de chiffrement de la base de données est préparée pour une future version afin d'ajouter une couche de sécurité supplémentaire. Le code nécessaire est déjà présent dans `utils/security.py`. Les fonctions `get_cipher`, `encrypt_data` et `decrypt_data`, basées sur la bibliothèque `Cryptography`, sont prêtes à être intégrées. Ce mécanisme s'appuiera sur une clé de chiffrement stockée localement dans le fichier `data/secret.key`.

5.4 Recommandations Essentielles pour l'Utilisateur

Pour atteindre un niveau de sécurité maximal, il est recommandé à l'utilisateur de suivre les bonnes pratiques documentées dans le fichier `SECURITE.md` :

- Activer le chiffrement complet du disque de son système d'exploitation (ex: BitLocker sur Windows, FileVault sur macOS) pour protéger l'intégralité de ses données en cas de vol de l'appareil.
- Utiliser un mot de passe de session fort et verrouiller son ordinateur lorsqu'il n'est pas utilisé.
- Effectuer des sauvegardes régulières et **chiffrées** du dossier `data/`. Les fichiers `data/journal.db` et `data/secret.key` sont particulièrement sensibles. Ne partagez jamais ces fichiers.
- Partager les exports de données (PDF, Excel) uniquement via des canaux de communication sécurisés.
- Supprimer les fichiers exportés après utilisation si leur conservation n'est plus nécessaire.

L'engagement du projet Help-Desk est de fournir un outil qui non seulement aide, mais protège activement la vie privée de ceux qui lui font confiance. Cet engagement est rendu possible par un écosystème de développement solide et des méthodologies éprouvées.

6.0 Écosystème de Développement et Méthodologie

La qualité d'une application ne dépend pas seulement de son code source, mais aussi de la rigueur des outils, des méthodologies et des principes qui guident sa création. Cette section offre un aperçu des coulisses du développement du projet Help-Desk, mettant en lumière l'environnement structuré qui a permis de transformer une vision en un outil fonctionnel et fiable.

6.1 Outils de Développement

Un ensemble d'outils standards et éprouvés a été utilisé pour garantir un développement efficace et reproductible :

- **Visual Studio Code** : Éditeur de code principal, utilisé pour le développement, le débogage et la gestion du projet, notamment via sa fonctionnalité de connexion à distance (SSH).
- **`pip`** : Gestionnaire de paquets standard de Python, utilisé pour installer et gérer toutes les dépendances du projet listées dans `requirements.txt`.
- **`venv`** : Outil pour la création d'environnements virtuels isolés, garantissant que les dépendances du projet n'entrent pas en conflit avec d'autres applications sur le système.

6.2 Concepts et Méthodologies Appliqués

Le développement a suivi des principes architecturaux et méthodologiques clairs :

- **Architecture inspirée de MVC** : Le projet adopte une structure inspirée du modèle MVC (Model-View-Controller) pour séparer les préoccupations : la base de données (`db/`) agit comme le Modèle, l'interface utilisateur (`ui/`) comme la Vue, et la logique métier (`services/`) comme le Contrôleur.
- **Gestion de l'état** : L'état de la session utilisateur est géré via `st.session_state` de Streamlit, un mécanisme essentiel pour maintenir la cohérence des données et de l'interface entre les interactions de l'utilisateur.
- **Bonnes pratiques de sécurité** : Les principes de l'OWASP (Open Web Application Security Project) ont servi de guide pour la conception des fonctionnalités, même dans un contexte d'application locale.

6.3 Principes de Design et d'Expérience Utilisateur (UX)

L'interface a été conçue en gardant à l'esprit la simplicité et l'accessibilité :

- **Stylisation CSS3** : Une feuille de style personnalisée (`assets/theme.css`) est utilisée pour affiner l'apparence visuelle de l'application et garantir une identité cohérente.
- **Principes d'accessibilité** : Une attention particulière a été portée à la lisibilité et au contraste des couleurs pour rendre l'application utilisable par le plus grand nombre.
- **Interface adaptative (Responsive)** : L'utilisation des fonctionnalités de mise en page de Streamlit, telles que `st.columns` et `st.tabs`, permet à l'interface de s'adapter de manière fluide à différentes tailles d'écran.

Cet écosystème structuré a permis un développement itératif et organisé, dont l'histoire et les choix fondateurs sont retracés dans la section finale de ce document.

7.0 Historique, Contexte et Avenir du Projet

Cette dernière section offre une rétrospective du projet, détaillant son parcours depuis la conception jusqu'à sa forme actuelle, ainsi que les choix techniques qui l'ont façonné. Il est important de noter que ce projet a été réalisé dans un cadre d'apprentissage, ce qui souligne la valeur de la documentation, du partage des connaissances et du processus itératif qui a mené au résultat final.

7.1 Historique de Développement

Le projet a évolué à travers quatre phases distinctes :

1. **Phase 1 : Conception** : Analyse des besoins des utilisateurs cibles (TDAH, anxiété), choix du socle technologique (Streamlit, SQLite, Ollama) et définition de l'architecture modulaire.
2. **Phase 2 : Développement** : Mise en place de la structure de la base de données, création de l'interface utilisateur avec Streamlit et implémentation des fonctionnalités de base (saisie d'humeur, gestion des tâches).
3. **Phase 3 : Améliorations** : Intégration du module de chat IA "Mathi" avec Ollama, ajout des fonctionnalités d'export en Excel et PDF, développement des graphiques de suivi d'humeur et application du style CSS personnalisé.
4. **Phase 4 : Déploiement** : Création des scripts de lancement, rédaction de la documentation complète (`README.md`, `SECURITE.md`) et du guide d'installation.

7.2 Ce qui n'a PAS été utilisé, et pourquoi

Les choix technologiques ne se définissent pas seulement par ce qui est utilisé, mais aussi par ce qui est délibérément écarté.

- **Django/Flask écartés** pour Streamlit, privilégiant la rapidité de développement d'interfaces de données interactives.
- **PostgreSQL/MySQL écartés** pour SQLite, qui est parfaitement adapté à un usage local, sans nécessiter de serveur de base de données.
- **Docker écarté** car non nécessaire pour une application conçue pour un usage local et non pour un déploiement complexe.

7.3 Remerciements et Communauté

Ce projet n'aurait pas été possible sans l'écosystème open source et les outils modernes de développement. Des remerciements particuliers sont adressés à :

- La **communauté open source** pour les innombrables bibliothèques qui forment le socle de ce projet.
- L'équipe de **Streamlit** pour avoir créé un framework puissant et accessible.
- La **Python Software Foundation** pour le maintien et l'évolution du langage Python.

7.4 Auteur et Licence du Projet

Ce projet a été réalisé par **IJustStartPython** dans le cadre d’un parcours d’apprentissage en programmation.

Il est recommandé de distribuer ce projet sous la **MIT License**, qui est permissive et bien adaptée à ce type d'initiative.

- Licence recommandée : MIT License
- Alternative : GPL-3.0

Ce document constitue la synthèse complète du projet Help-Desk, un outil conçu pour être non seulement fonctionnel et utile, mais aussi respectueux et protecteur, reflétant une philosophie où la technologie est au service de l'humain.