

# République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université M'hamed Bougara - Boumerdès



Faculté des Sciences  
Département d'Informatique

**Domaine** : Mathématiques Informatique

**Filière** : Informatique

**Spécialité** : Développement Web et Infographie

*N° de l'Arrêté d'habilitation de la spécialité : arrêté n°002 du 03/01/2021*

Mémoire de fin d'études en vu de l'obtention du  
Diplôme de Licence Professionnelle

## Thème

**VoiceNotion**  
**AI-Based Voice Productivity**  
**SaaS Application**

Présenté par :

BENHADJER Foudhil Abdellah  
SADAOUUI Ayoub  
HAFSAOUI Abderahim

Supervisé par :

CHAOUCHÉ Ali

Soutenu le 24/06/2024 Devant le jury composé de

SIACI Redouane : Examinateur  
HAMMICHE Mokhtar : Président

# TABLE DES MATIÈRES

.....	1
.....	2
INTRODUCTION .....	3
Objectifs du projet .....	3
Portée du document .....	4
1 ETUDE PREALABLE .....	5
1.1 Introduction .....	5
1.2 Problématiques .....	5
1.2.1 Limites de la saisie manuelle .....	5
1.2.2 Accessibilité réduite .....	6
1.2.3 Complexité de structuration .....	6
1.3 Solution et objectif .....	6
1.3.1 Concept de VoiceNotion .....	6
1.3.2 Objectifs du projet .....	7
1.4 Avantages du projet .....	7
1.4.1 Efficacité accrue .....	7
1.4.2 Accessibilité améliorée .....	7
1.4.3 Flexibilité et puissance .....	7
1.5 Conclusion .....	8
2 PLANIFICATION ET CONCEPTION UX .....	9
2.1 Introduction .....	9
2.2 Planification du projet .....	9
2.2.1 Méthodologie de planification .....	9
2.2.1.1 Approche Agile Scrum .....	9
2.2.1.2 Design Thinking pour l'UX .....	10
2.2.2 Planification .....	11
2.2.2.1 Roadmap du projet .....	11

2.2.2.2	Gestion des risques . . . . .	12
2.3	Expérience d'utilisateur (UX) . . . . .	12
2.3.1	Recherche . . . . .	12
2.3.1.1	Méthodologie de recherche . . . . .	13
2.3.1.2	Principaux insights . . . . .	13
2.3.2	Empathie . . . . .	14
2.3.2.1	Personas utilisateur . . . . .	14
2.3.2.2	Scénarios d'utilisations . . . . .	16
2.4	Conception du système d'information . . . . .	17
2.4.1	Identification des acteurs . . . . .	17
2.4.2	Diagramme de cas d'utilisation . . . . .	18
2.4.3	Diagrammes de séquence . . . . .	19
2.4.3.1	Séquence de commande vocale . . . . .	19
2.4.3.2	Séquence de création et sauvegarde de note . . . . .	20
2.4.3.3	Séquence d'authentification . . . . .	22
2.4.4	Diagrammes de base de données . . . . .	23
2.4.4.1	Diagramme entité-relation . . . . .	23
2.4.4.2	Diagramme de base de données . . . . .	23
2.5	Conclusion . . . . .	24
3	IMPLEMENTATION ET DEVELOPPEMENT . . . . .	26
3.1	Introduction . . . . .	26
3.2	Architecture globale . . . . .	26
3.2.1	Composants principaux . . . . .	27
3.2.2	Flux de données . . . . .	27
3.3	Environnement de développement . . . . .	28
3.3.1	Materiels . . . . .	28
3.3.2	Logiciels et outils de développement . . . . .	28
3.3.2.1	Visual Studio Code . . . . .	28
3.3.2.2	GitHub . . . . .	29
3.3.2.3	TypeScript . . . . .	30
3.3.2.4	Zod . . . . .	31
3.3.2.5	Tests . . . . .	32
3.4	Outils de conception et design . . . . .	32
3.4.1	Figma . . . . .	32
3.4.2	Adobe Illustrator . . . . .	33
3.4.3	Adobe Photoshop . . . . .	34
3.5	Backend et services cloud . . . . .	35
3.5.1	Supabase . . . . .	35
3.5.2	Prisma . . . . .	37
3.5.3	Hebergement et deploiement . . . . .	38
3.5.3.1	Vercel . . . . .	38
3.5.3.2	Expo EAS . . . . .	39
3.6	Site web (Front-end) . . . . .	40

3.6.1	Technologies et outils utilisés . . . . .	40
3.6.1.1	React.js . . . . .	40
3.6.1.2	Next.js . . . . .	41
3.6.1.3	TailwindCSS . . . . .	42
3.6.1.4	BlockNote . . . . .	44
3.6.2	Interfaces principales . . . . .	46
3.6.2.1	Page d'accueil . . . . .	46
3.6.2.2	Authentification . . . . .	46
3.6.2.3	Tableau de bord . . . . .	49
3.6.2.4	Éditeur de notes . . . . .	50
3.7	Application mobile . . . . .	51
3.7.1	Technologies et outils utilisés . . . . .	51
3.7.1.1	React Native . . . . .	51
3.7.1.2	Expo . . . . .	53
3.7.2	Interfaces principales . . . . .	53
3.7.2.1	Ecrans d'accueil et d'authentification . . . . .	53
3.7.2.2	Editeur et saisie vocale . . . . .	54
3.8	Intégration de la reconnaissance vocale . . . . .	55
3.8.1	Gemini API . . . . .	55
3.8.2	Flux de traitement des commandes vocales . . . . .	56
3.9	Tests et assurance qualité . . . . .	57
3.9.1	Stratégie de test . . . . .	57
3.9.2	Outils de test . . . . .	57
3.10	Déploiement et intégration continue . . . . .	58
3.10.1	Pipeline CI/CD . . . . .	58
3.10.2	Environnements de déploiement . . . . .	58
3.10.3	Plateformes de déploiement . . . . .	58
3.10.3.1	Digital Ocean . . . . .	58
3.10.3.2	Nginx . . . . .	59
3.11	Sécurité et protection des données . . . . .	59
3.11.1	Authentification et autorisation . . . . .	59
3.11.2	Protection des données . . . . .	59
3.11.3	Conformité RGPD . . . . .	60
3.12	Défis techniques et solutions . . . . .	60
3.12.1	Défis rencontrés . . . . .	60
3.12.2	Solutions implementées . . . . .	60
3.13	Conclusion . . . . .	60
INTRODUCTION	. . . . .	61
4 ETUDE PREALABLE	. . . . .	62
4.1	Introduction . . . . .	62
4.2	Problématiques . . . . .	62
4.3	Solution et objectif . . . . .	62

4.4	Avantages du projet . . . . .	62
4.5	Conclusion . . . . .	62
5	PLANIFICATION ET CONCEPTION UX . . . . .	63
5.1	Introduction . . . . .	63
5.2	Planification du projet . . . . .	63
5.2.1	Méthodologie de planification . . . . .	63
5.2.2	Planification . . . . .	63
5.3	Expérience d'utilisateur (UX) . . . . .	63
5.3.1	Recherche . . . . .	63
5.3.2	Empathie . . . . .	63
5.3.2.1	Personas utilisateur . . . . .	63
5.3.2.2	Scénarios d'utilisations . . . . .	63
5.4	Conception du système d'information . . . . .	63
5.4.1	Identification des acteurs . . . . .	63
5.4.2	Diagramme de cas d'utilisation . . . . .	63
5.4.3	Diagrammes de séquence . . . . .	63
5.4.4	Diagrammes de base de données . . . . .	63
5.4.4.1	Diagramme entité-relation . . . . .	63
5.4.4.2	Diagramme de base de données . . . . .	63
5.5	Conclusion . . . . .	63
6	IMPLEMENTATION ET DEVELOPPEMENT . . . . .	64
6.1	Introduction . . . . .	65
6.2	Architecture . . . . .	65
6.3	Environnement . . . . .	65
6.3.1	Matériel (hardware) . . . . .	65
6.3.2	Logiciel (software) . . . . .	65
6.4	API (Back-end server) . . . . .	65
6.4.1	Technologies et outils utilisés (sur Back-end) . . . . .	65
6.4.1.1	Node.js . . . . .	65
6.4.1.2	Express.js . . . . .	65
6.4.1.3	TypeScript . . . . .	65
6.4.1.4	Supabase . . . . .	65
6.4.1.5	Firebase . . . . .	65
6.4.1.6	Docker . . . . .	65
6.4.1.7	PostgreSQL . . . . .	65
6.4.2	Déploiement . . . . .	65
6.4.2.1	Digital Ocean . . . . .	65
6.4.2.2	Nginx . . . . .	65
6.4.2.3	CI/CD Pipeline . . . . .	65
6.5	Site web (Front-end) . . . . .	65
6.5.1	Technologies et outils utilisés (Front-end) . . . . .	65
6.5.1.1	React.js . . . . .	65

6.5.1.2	Next.js . . . . .	65
6.5.1.3	TailwindCSS . . . . .	65
6.5.1.4	BlockNote.js . . . . .	65
6.5.2	L'interface utilisateur du site . . . . .	65
6.5.2.1	Landing page . . . . .	65
6.5.2.2	Connexion . . . . .	65
6.5.2.3	Inscription . . . . .	65
6.5.2.4	Dashboard . . . . .	65
6.5.2.5	Note Editor . . . . .	65
6.5.2.6	Voice Command Panel . . . . .	65
6.5.2.7	Settings . . . . .	65
6.5.2.8	Search . . . . .	65
6.6	Application mobile . . . . .	65
6.6.1	Technologies et outils utilisés (app mobile) . . . . .	65
6.6.1.1	Expo . . . . .	65
6.6.1.2	React Native . . . . .	65
6.6.1.3	Expo DOM Components . . . . .	65
6.6.2	L'interface utilisateur d'application mobile . . . . .	65
6.6.2.1	Welcome page . . . . .	65
6.6.2.2	Connexion . . . . .	65
6.6.2.3	Inscription . . . . .	65
6.6.2.4	Accueil et menu . . . . .	65
6.6.2.5	Note Editor . . . . .	65
6.6.2.6	Voice Input Button . . . . .	65
6.6.2.7	Recherche . . . . .	65
6.6.2.8	Profil . . . . .	65
6.6.2.9	Rendez-vous . . . . .	65
6.7	Intégration de la reconnaissance vocale . . . . .	65
6.7.1	Gemini API . . . . .	65
6.7.2	Conversion Speech-to-Text . . . . .	65
6.7.3	Analyse d'intention (Intent Parsing) . . . . .	65
6.7.4	Actions Editor . . . . .	65
6.8	Conclusion . . . . .	65
7	VOICENOTION – IDENTITE VISUELLE . . . . .	66
7.1	Introduction . . . . .	67
7.2	Logo . . . . .	67
7.2.1	Le choix du nom . . . . .	67
7.2.2	Le choix du logo . . . . .	67
7.2.3	La méthode de conception . . . . .	67
7.3	La typographie . . . . .	67
7.3.1	Polices latines . . . . .	67
7.3.1.1	Inter font . . . . .	67
7.3.1.2	Outfit font . . . . .	67
7.3.2	Police Arabe . . . . .	67

7.3.2.1	Noto Sans Arabic . . . . .	67
7.4	Palette de couleurs . . . . .	67
7.4.1	Bleu primaire . . . . .	67
7.4.2	Gris neutre . . . . .	67
7.4.3	Accent colors . . . . .	67
7.5	Design System . . . . .	67
7.5.1	Conception . . . . .	67
7.5.2	Components . . . . .	67
7.5.2.1	Buttons . . . . .	67
7.5.2.2	Input fields . . . . .	67
7.5.2.3	Cards . . . . .	67
7.6	Mockups . . . . .	67
7.6.1	Mobile app . . . . .	67
7.6.1.1	Light theme . . . . .	67
7.6.1.2	Dark theme . . . . .	67
7.6.2	Web app . . . . .	67
7.6.2.1	Light theme . . . . .	67
7.6.2.2	Dark theme . . . . .	67
7.7	Video animation . . . . .	67
7.7.1	Scènes de vidéo . . . . .	67
7.7.1.1	Présentation du problème . . . . .	67
7.7.1.2	Présentation de la solution . . . . .	67
7.7.1.3	Description du processus . . . . .	67
7.7.1.4	Résultat . . . . .	67
7.8	Conclusion . . . . .	67
CONCLUSION	. . . . .	68
A REFERENCES	. . . . .	69
B GLOSSARY	. . . . .	70
C VOICE COMMAND REFERENCE	. . . . .	71

# TABLE DES FIGURES

2.1	Intégration de Scrum et Design Thinking dans notre méthodologie . . . . .	11
2.2	Synthèse des principaux insights de recherche utilisateur . . . . .	14
2.3	Les trois personas principaux de VoiceNotion . . . . .	16
2.4	Diagramme de cas d'utilisation pour VoiceNotion . . . . .	18
2.5	Diagramme de séquence pour le traitement d'une commande vocale . . . . .	20
2.6	Diagramme de séquence pour la création et sauvegarde d'une note . . . . .	21
2.7	Diagramme de séquence pour l'authentification . . . . .	22
2.8	Diagramme entité-relation pour VoiceNotion . . . . .	23
2.9	Schéma logique de la base de données pour VoiceNotion . . . . .	24
3.1	Architecture globale de VoiceNotion . . . . .	27
3.2	Système de design VoiceNotion dans Figma . . . . .	33
3.3	Création des éléments graphiques avec Adobe Illustrator . . . . .	34
3.4	Dashboard Supabase pour VoiceNotion . . . . .	36
3.5	Page d'accueil du site VoiceNotion . . . . .	46
3.6	Page de connexion du site VoiceNotion . . . . .	46
3.7	Page d'inscription du site VoiceNotion . . . . .	47
3.8	Tableau de bord du site VoiceNotion . . . . .	49
3.9	Éditeur de notes avec commandes vocales . . . . .	50
3.10	Ecrans d'accueil et de connexion de l'application mobile . . . . .	53
3.11	Ecrans d'inscription et d'accueil de l'application mobile . . . . .	54
3.12	Editeur de notes et interface de saisie vocale . . . . .	54
3.13	Intégration de l'API Gemini dans VoiceNotion . . . . .	56
3.14	Flux de traitement des commandes vocales . . . . .	57

# INTRODUCTION

Dans un monde en constante évolution, la technologie est devenue essentielle pour transformer divers secteurs, y compris le domaine de la prise de notes et de la création de documents. Avec l'essor des assistants vocaux et des technologies de reconnaissance vocale, le potentiel d'innovation dans la façon dont nous capturons et organisons nos pensées est immense.

VoiceNotion représente une approche innovante pour capturer, organiser et affiner les pensées principalement par commandes vocales, complétée par un éditeur intuitif basé sur des blocs. L'application vise à être la solution de référence pour les utilisateurs qui valorisent la rapidité, l'efficacité et la flexibilité de la saisie vocale, sans sacrifier les riches capacités d'édition des éditeurs de blocs modernes.

L'idée principale derrière VoiceNotion est de créer une expérience fluide de prise de notes et de création de documents, optimisée pour les appareils mobiles. L'application s'adresse aux étudiants, aux professionnels, aux écrivains et à toute personne ayant besoin de noter rapidement des idées, d'organiser des notes ou de rédiger des documents en déplacement.

La vision de VoiceNotion est de devenir l'application de choix pour ceux qui cherchent à maximiser leur productivité en transformant la parole en contenu structuré et organisé. En combinant la puissance des commandes vocales avec l'organisation intuitive des éditeurs de blocs, VoiceNotion offre une solution innovante aux défis de la prise de notes traditionnelle.

## Objectifs du projet

Les objectifs principaux du projet VoiceNotion sont :

- Développer une application mobile entièrement fonctionnelle permettant aux utilisateurs de créer et d'éditer des notes via des commandes vocales.
- Implémenter un éditeur basé sur BlockNote.js offrant une expérience d'édition par blocs similaire à Notion.
- Assurer une transcription vocale de haute fidélité et une analyse intelligente des intentions de l'utilisateur.
- Créer une interface utilisateur intuitive et conviviale optimisée pour les appareils mobiles.

- Fournir une application web complémentaire pour une expérience cross-platform complète.

## Portée du document

Ce document sert de guide complet pour l'application VoiceNotion, couvrant sa base conceptuelle, son architecture technique, les détails d'implémentation, la conception de l'expérience utilisateur et l'identité visuelle. Il est destiné aux développeurs, designers et parties prenantes impliqués dans le projet, fournissant une compréhension approfondie de la structure et de la fonctionnalité de l'application.

La documentation est organisée en plusieurs chapitres, chacun couvrant un aspect spécifique du développement et de la conception de VoiceNotion :

- **Étude préalable** : Analyse du problème, objectifs du projet et solutions proposées.
- **Planification et conception UX** : Méthodologie de développement, recherche utilisateur et conception du système.
- **Implémentation et développement** : Architecture technique, technologies utilisées et détails d'implémentation.
- **Identité visuelle** : Conception de la marque, éléments visuels et interfaces utilisateur.

Ce document servira de référence tout au long du cycle de développement du projet et pourra être utilisé comme base pour la formation, la maintenance et l'évolution future de l'application VoiceNotion.

# CHAPTER 1

## ETUDE PREALABLE

### 1.1 Introduction

Dans un monde où la productivité et l'efficacité sont devenues des priorités absolues, la prise de notes demeure un processus fondamental pour capturer et organiser l'information. Que ce soit pour les étudiants, les professionnels ou les créateurs de contenu, la capacité à saisir rapidement des idées, des concepts ou des informations est essentielle. Cependant, les méthodes traditionnelles de prise de notes présentent des limitations significatives, notamment en termes de vitesse, d'accessibilité et de flexibilité.

Cette étude préalable vise à explorer le contexte, les problématiques et les opportunités liés à la création d'une application de prise de notes pilotée par la voix, que nous avons nommée VoiceNotion. Notre analyse portera sur les défis actuels de la prise de notes, les solutions existantes sur le marché, et la manière dont notre approche innovante peut répondre aux besoins non satisfaits des utilisateurs.

### 1.2 Problématiques

La prise de notes traditionnelle, qu'elle soit manuscrite ou numérique, présente plusieurs défis majeurs que nous avons identifiés à travers notre recherche et nos observations:

#### 1.2.1 Limites de la saisie manuelle

La saisie manuelle de notes, que ce soit à l'aide d'un stylo et papier ou d'un clavier, présente plusieurs inconvénients:

- **Vitesse limitée:** La vitesse de frappe moyenne (40-60 mots par minute) ou d'écriture manuscrite (10-30 mots par minute) est souvent insuffisante pour capturer efficacement des informations lors de réunions, conférences ou sessions de brainstorming rapides.
- **Fatigue et inconfort:** La saisie prolongée peut entraîner une fatigue des mains et des poignets, particulièrement sur les appareils mobiles où les claviers virtuels offrent une expérience sous-optimale.
- **Attention divisée:** Le fait de devoir se concentrer sur la saisie divise l'attention de l'utilisateur, réduisant sa capacité à écouter activement ou à participer pleinement à une discussion.

### 1.2.2 Accessibilité réduite

Les méthodes traditionnelles de prise de notes présentent des obstacles d'accessibilité significatifs:

- **Mobilité réduite:** Les personnes souffrant de limitations motrices peuvent éprouver des difficultés considérables avec la saisie manuelle.
- **Situations multi-tâches:** De nombreux contextes (conduite, marche, exercice) rendent la saisie manuelle difficile voire impossible.
- **Barrières linguistiques:** Pour les utilisateurs non natifs d'une langue, la saisie écrite peut présenter des difficultés supplémentaires par rapport à l'expression orale.

### 1.2.3 Complexité de structuration

L'organisation et la structuration efficaces des notes représentent un défi majeur:

- **Effort post-capture:** La transformation de notes brutes en contenu structuré et organisé nécessite souvent un effort supplémentaire considérable.
- **Rigidité des formats:** De nombreuses applications de prise de notes offrent des structures rigides qui limitent la flexibilité et l'adaptabilité aux différents types de contenu.
- **Courbe d'apprentissage:** Les éditeurs avancés comme Notion requièrent un investissement initial en temps pour maîtriser leurs fonctionnalités.

## 1.3 Solution et objectif

Face à ces problématiques, VoiceNotion propose une approche novatrice qui combine la puissance de la reconnaissance vocale, de l'intelligence artificielle et d'un éditeur de notes par blocs inspiré de Notion. Notre solution vise à créer une expérience de prise de notes qui soit à la fois rapide, accessible et puissante.

### 1.3.1 Concept de VoiceNotion

VoiceNotion est une application mobile et web qui permet aux utilisateurs de créer, éditer et organiser des notes principalement à travers des commandes vocales. L'application transforme la voix en texte structuré et permet d'effectuer diverses actions d'édition sans nécessiter d'interactions manuelles complexes.

- **Saisie vocale intelligente:** Utilisation de l'API Gemini pour une transcription précise et une compréhension contextuelle des commandes.
- **Éditeur par blocs:** Interface inspirée de Notion permettant une organisation flexible du contenu en blocs (texte, listes, tableaux, images, etc.).
- **Commandes vocales complètes:** Capacité à effectuer toutes les actions d'édition courantes (formatage, création de listes, ajout de blocs, navigation) par la voix.
- **Expérience mobile-first:** Conçue d'abord pour les appareils mobiles, avec une interface adaptée aux écrans tactiles et aux contextes d'utilisation en déplacement.

### 1.3.2 Objectifs du projet

Le développement de VoiceNotion vise à atteindre plusieurs objectifs clés:

- **Accélérer la prise de notes:** Permettre une saisie jusqu'à 3 fois plus rapide qu'avec un clavier traditionnel.
- **Améliorer l'accessibilité:** Rendre la prise de notes accessible aux personnes à mobilité réduite et dans des contextes où la saisie manuelle est difficile.
- **Simplifier la structuration:** Faciliter l'organisation du contenu grâce à des commandes vocales intuitives pour la création et la manipulation de blocs.
- **Offrir une flexibilité maximale:** Supporter divers cas d'utilisation, des notes rapides aux documents structurés complexes.
- **Garantir une expérience utilisateur fluide:** Proposer une interface intuitive qui minimise la friction entre l'intention de l'utilisateur et sa réalisation.

## 1.4 Avantages du projet

VoiceNotion offre plusieurs avantages distincts par rapport aux solutions existantes sur le marché:

### 1.4.1 Efficacité accrue

- **Vitesse de saisie supérieure:** La dictée vocale permet une saisie moyenne de 120-150 mots par minute, soit 2-3 fois plus rapide que la frappe.
- **Réduction du temps de formatage:** Les commandes vocales pour le formatage et la structuration éliminent le besoin de manipulations manuelles complexes.
- **Attention préservée:** Les utilisateurs peuvent se concentrer davantage sur le contenu et moins sur le processus de saisie.

### 1.4.2 Accessibilité améliorée

- **Inclusion:** Solution accessible aux personnes à mobilité réduite ou souffrant de troubles musculosquelettiques.
- **Utilisation en contexte mobile:** Possibilité de prendre des notes en déplacement, pendant l'exercice ou dans d'autres situations où les mains sont occupées.
- **Réduction de la barrière linguistique:** Plus facile pour les utilisateurs non natifs qui s'expriment mieux oralement qu'à l'écrit.

### 1.4.3 Flexibilité et puissance

- **Structure par blocs:** Organisation flexible du contenu adaptée à différents types de notes (cours, réunions, idées, projets).
- **Multimodalité:** Possibilité de basculer facilement entre saisie vocale et manuelle selon le contexte.

- **Adaptation contextuelle:** L'intelligence artificielle permet d'adapter l'interprétation des commandes au contexte spécifique de l'utilisateur.

## 1.5 Conclusion

Cette étude préalable a permis d'identifier clairement les problématiques actuelles de la prise de notes traditionnelle et de poser les bases conceptuelles de VoiceNotion comme solution innovante. En combinant la puissance de la reconnaissance vocale, de l'intelligence artificielle et d'un éditeur par blocs flexible, VoiceNotion a le potentiel de transformer radicalement l'expérience de prise de notes pour un large éventail d'utilisateurs.

Le projet répond à des besoins réels et non satisfaits du marché, offrant une alternative plus rapide, plus accessible et plus flexible aux méthodes traditionnelles. Les chapitres suivants détailleront la planification, la conception et l'implémentation technique de cette solution, en mettant l'accent sur l'expérience utilisateur et la robustesse technique.

Dans un monde où l'information est abondante et le temps précieux, VoiceNotion vise à devenir un outil essentiel pour capturer, organiser et exploiter efficacement les idées et les connaissances.

# CHAPTER 2

## PLANIFICATION ET CONCEPTION UX

### 2.1 Introduction

Après avoir identifié les problématiques et défini le concept de VoiceNotion dans le chapitre précédent, nous nous concentrons maintenant sur la planification du projet et la conception de l'expérience utilisateur. Cette phase est cruciale pour transformer notre vision en un plan d'action concret et en une interface utilisateur intuitive qui répond aux besoins réels des utilisateurs.

La planification et la conception UX sont particulièrement importantes pour VoiceNotion en raison de son approche novatrice combinant commandes vocales et édition par blocs. L'interaction vocale, bien que naturelle pour la communication humaine, présente des défis uniques lorsqu'elle est appliquée au contrôle d'une interface numérique. Notre objectif est de créer une expérience fluide et intuitive qui tire pleinement parti des capacités vocales tout en offrant une interface visuelle cohérente et familière.

Ce chapitre détaille notre méthodologie de planification, notre approche de l'expérience utilisateur basée sur la recherche et l'empathie, ainsi que la conception technique du système d'information qui sous-tend l'application.

### 2.2 Planification du projet

#### 2.2.1 Méthodologie de planification

Pour le développement de VoiceNotion, nous avons adopté une approche agile, plus précisément la méthodologie Scrum, complétée par des éléments de Design Thinking pour la conception UX. Cette combinaison nous permet d'itérer rapidement tout en gardant l'utilisateur au centre de notre processus de conception.

##### 2.2.1.1 Approche Agile Scrum

L'approche Scrum a été choisie pour sa flexibilité et sa capacité à s'adapter aux changements inhérents au développement d'un produit innovant comme VoiceNotion. Nous avons organisé notre travail en sprints de deux semaines, chacun comportant les événements standards de Scrum:

- **Sprint Planning:** Au début de chaque sprint, l'équipe sélectionne les tâches du Product Backlog à réaliser, en se basant sur leur priorité et la capacité de l'équipe.
- **Daily Stand-up:** Réunions quotidiennes de 15 minutes pour synchroniser les activités et identifier les obstacles.

- **Sprint Review:** À la fin de chaque sprint, l'équipe présente les fonctionnalités développées aux parties prenantes pour obtenir des retours.
- **Sprint Retrospective:** L'équipe analyse ce qui a bien fonctionné et ce qui peut être amélioré pour le prochain sprint.

Cette approche nous permet de livrer régulièrement des incrémentums de produit fonctionnels et d'ajuster notre direction en fonction des retours utilisateurs et des défis techniques rencontrés.

#### 2.2.1.2 *Design Thinking pour l'UX*

En parallèle de Scrum, nous avons intégré le processus de Design Thinking pour la conception de l'expérience utilisateur, qui comprend cinq phases:

1. **Empathie:** Comprendre profondément les besoins, les frustrations et les aspirations des utilisateurs potentiels à travers des entretiens et des observations.
2. **Définition:** Synthétiser les connaissances acquises pour définir clairement les problèmes à résoudre.
3. **Idéation:** Générer un large éventail d'idées et de solutions potentielles sans contraintes initiales.
4. **Prototypage:** Créer des représentations tangibles des solutions les plus prometteuses.
5. **Test:** Recueillir les retours des utilisateurs sur les prototypes pour affiner et améliorer les solutions.

Cette approche centrée sur l'utilisateur est particulièrement pertinente pour VoiceNotion, où l'interaction vocale nécessite une compréhension fine des attentes et des comportements des utilisateurs.



Figure 2.1: Intégration de Scrum et Design Thinking dans notre méthodologie

## 2.2.2 Planification

### 2.2.2.1 *Roadmap du projet*

La roadmap de VoiceNotion a été structurée en quatre phases principales, chacune comportant plusieurs sprints:

#### 1. Phase de recherche et de conception (2 mois):

- Recherche utilisateur et analyse concurrentielle
- Définition des personas et des scénarios d'utilisation
- Conception de l'architecture du système
- Wireframing et prototypage initial

#### 2. Phase de développement MVP (4 mois):

- Mise en place de l'infrastructure technique
- Développement du backend et de l'API
- Implémentation de l'éditeur de notes par blocs (avec BlockNote.js)
- Intégration de la reconnaissance vocale de base

- Développement de l'interface utilisateur mobile (Expo/React Native)

### 3. Phase d'amélioration et d'expansion (3 mois):

- Optimisation des algorithmes de traitement du langage naturel
- Expansion des commandes vocales supportées
- Développement de l'interface web
- Implémentation des fonctionnalités de sous-pages et de hiérarchie de notes

### 4. Phase de finalisation et de lancement (1 mois):

- Tests utilisateurs approfondis
- Correction des bugs et optimisations finales
- Préparation du matériel marketing
- Lancement officiel de l'application

#### 2.2.2.2 Gestion des risques

Nous avons identifié plusieurs risques potentiels pour le projet et élaboré des stratégies d'atténuation:

Risque	Impact potentiel	Stratégie d'atténuation
Précision limitée de la reconnaissance vocale	Frustration utilisateur, abandon de l'application	Implémentation d'un mécanisme de correction, modes d'entrée alternatifs, tests extensifs avec différents accents
Complexité technique de l'intégration BlockNote	Retards de développement, problèmes de performance	Spike techniques précoces, exploration des alternatives, recrutement d'expertise spécifique
Expérience utilisateur non intuitive	Courbe d'apprentissage abrupte, faible adoption	Tests utilisateurs fréquents, approche itérative, tutoriels intégrés
Limites des API Gemini	Fonctionnalités restreintes, dépendance à un tiers	Plan de secours avec solutions alternatives, découplage de l'architecture
Problèmes de performance sur appareils mobiles	Lenteur, consommation excessive de batterie	Optimisation continue, tests sur différents appareils, métriques de performance

Table 2.1: Tableau des risques et stratégies d'atténuation

## 2.3 Expérience d'utilisateur (UX)

### 2.3.1 Recherche

La conception de VoiceNotion est fondée sur une recherche approfondie pour comprendre les besoins, les attentes et les points de friction des utilisateurs potentiels en matière de

prise de notes.

### 2.3.1.1 Méthodologie de recherche

Notre recherche a combiné plusieurs approches:

- **Entretiens qualitatifs:** Nous avons mené 15 entretiens approfondis avec des utilisateurs potentiels issus de nos groupes cibles (étudiants, professionnels, créateurs de contenu).
- **Analyse concurrentielle:** Étude détaillée des applications existantes (Notion, Evernote, OneNote, Google Keep) pour identifier les forces, les faiblesses et les opportunités d'innovation.
- **Sondage en ligne:** Un questionnaire distribué à 150 participants pour quantifier les préférences et les habitudes de prise de notes.
- **Sessions d'observation:** Observation de 8 utilisateurs dans leur environnement naturel pendant qu'ils prenaient des notes, révélant des comportements et des défis non exprimés lors des entretiens.

### 2.3.1.2 Principaux insights

Cette recherche a mis en lumière plusieurs insights clés qui ont guidé notre conception:

1. 78% des participants trouvent que la saisie manuelle limite leur capacité à capturer rapidement les informations lors de réunions ou de conférences.
2. Les utilisateurs de Notion apprécient la flexibilité de la structure par blocs, mais 65% trouvent la courbe d'apprentissage trop abrupte.
3. 92% des participants ont exprimé de l'intérêt pour les commandes vocales, mais 71% craignent qu'elles ne soient pas assez précises ou intuitives.
4. Les utilisateurs mobiles (81% de notre échantillon) prennent des notes dans des contextes variés où le clavier n'est pas toujours optimal (transports, déplacements, exercice).
5. La structuration post-capture est identifiée comme l'un des plus grands défis, avec 85% des participants qui admettent ne jamais réorganiser leurs notes brutes par manque de temps.

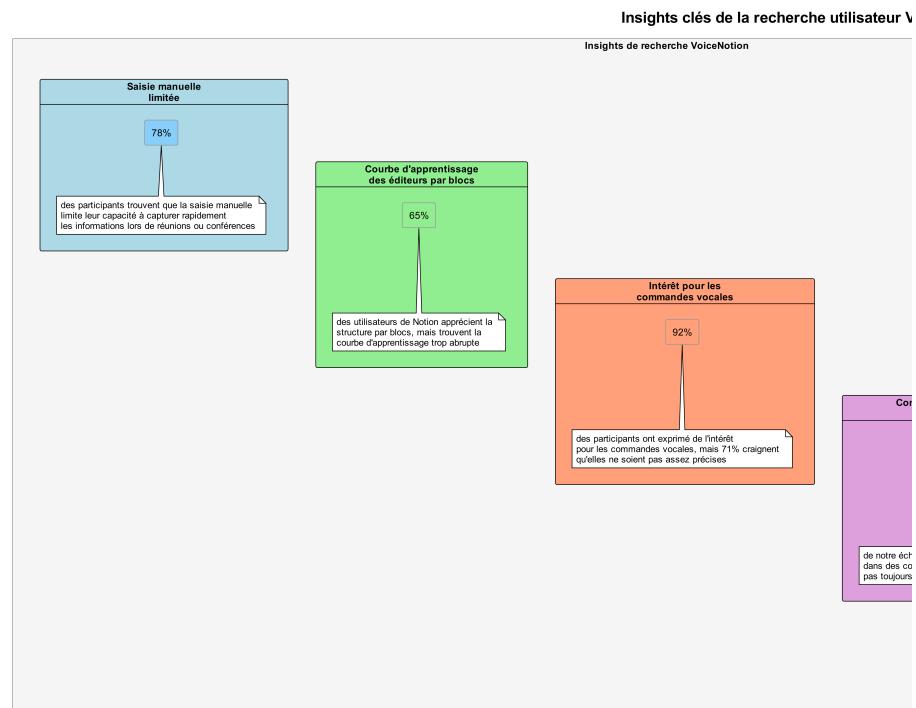


Figure 2.2: Synthèse des principaux insights de recherche utilisateur

### 2.3.2 Empathie

#### 2.3.2.1 Personas utilisateur

Basés sur notre recherche, nous avons développé trois personas principaux qui représentent nos utilisateurs cibles:

#### Persona 1: Sophie l'Étudiante

- Profil:** 22 ans, étudiante en master de droit, utilise principalement son smartphone et son ordinateur portable pour prendre des notes.
- Objectifs:** Capturer efficacement les informations en cours, organiser ses révisions, créer des liens entre différents concepts juridiques.
- Frustrations:** Difficulté à suivre le rythme des professeurs, perd du temps à reformater ses notes, trouve la navigation entre ses documents fastidieuse.
- Comportements:** Prend des notes pendant les cours, les complète en bibliothèque, révise régulièrement avec des mind maps et des fiches synthétiques.
- Citation:** "Je passe plus de temps à essayer de noter tout ce que dit le prof qu'à vraiment comprendre le cours."

#### Persona 2: Marc le Professionnel

- Profil:** 38 ans, chef de projet dans une entreprise technologique, toujours en déplacement entre réunions et sites clients.

- **Objectifs:** Capturer rapidement les décisions et actions des réunions, organiser ses projets, partager les informations avec son équipe.
- **Frustrations:** Manque de temps pour prendre des notes détaillées, difficultés à capturer des idées en déplacement, oublie des détails importants.
- **Comportements:** Utilise son téléphone pour des notes rapides, son ordinateur portable pour des documents plus structurés, souvent en multitâche.
- **Citation:** "Entre deux réunions, je n'ai parfois que 5 minutes pour noter les points clés avant d'enchaîner."

### Persona 3: Leila la Créatrice de Contenu

- **Profil:** 29 ans, auteure et créatrice de contenu indépendante, travaille depuis chez elle ou dans des cafés.
- **Objectifs:** Capturer l'inspiration quand elle survient, structurer ses idées en articles ou en scripts, organiser ses recherches.
- **Frustrations:** Perd ses meilleures idées quand elle ne peut pas les noter immédiatement, passe trop de temps à organiser son contenu, lutte avec différents formats de notes.
- **Comportements:** Alterne entre notes vocales, notes manuscrites et documents numériques, travaille de manière non linéaire avec beaucoup d'itérations.
- **Citation:** "Mes meilleures idées me viennent souvent quand je suis loin de mon ordinateur, pendant une promenade ou sous la douche."



..../assets/docs/voicenotion\_personas.png

Figure 2.3: Les trois personas principaux de VoiceNotion

#### 2.3.2.2 Scénarios d'utilisations

Pour chaque persona, nous avons développé des scénarios d'utilisation qui illustrent comment VoiceNotion répondrait à leurs besoins spécifiques:

**Scénario 1: Sophie en cours de droit constitutionnel** Sophie assiste à un cours de droit constitutionnel où le professeur parle rapidement et fait référence à de nombreux articles et précédents. Avec VoiceNotion, elle:

1. Active l'application et commence à prendre des notes textuelles de base.
2. Utilise la commande vocale "Nouveau titre: Articles constitutionnels importants" pour créer une section structurée sans interrompre sa prise de notes.
3. Dit "Ajouter liste à puces" pour commencer une liste des articles mentionnés.
4. Alterne facilement entre la saisie vocale pour les concepts généraux et la saisie manuelle pour les termes techniques précis ou les références.

5. À la fin du cours, utilise la commande "Créer une sous-page pour les cas jurisprudentiels" pour organiser les exemples mentionnés dans une structure hiérarchique.

**Scénario 2: Marc en réunion client puis en déplacement** Marc participe à une réunion importante avec un client pour discuter des spécifications d'un nouveau projet:

1. Au début de la réunion, il ouvre VoiceNotion et crée une nouvelle note avec la structure de base (objectifs, spécifications, actions).
2. Pendant la discussion, il ajoute rapidement des points à chaque section avec des commandes vocales discrètes.
3. Il utilise la commande "Ajouter liste à puces: Points à suivre" pour créer une liste d'actions à réaliser.
4. Après la réunion, dans le taxi, il révise ses notes et utilise la commande vocale "Réorganiser: déplacer la section Budget après Échéancier" pour restructurer son document.
5. Il crée une sous-page pour les détails techniques qui nécessitent une exploration plus approfondie.

**Scénario 3: Leila trouve l'inspiration pendant une promenade** Leila fait une promenade quotidienne quand elle a une idée pour un nouvel article:

1. Elle sort son téléphone et ouvre VoiceNotion, puis dit "Nouvelle note: Idée d'article sur la créativité et la routine".
2. En marchant, elle dicte ses idées principales, utilisant des commandes comme "Nouveau paragraphe" ou "Point important" pour structurer sa pensée.
3. Elle dit "Ajouter référence: livre Flow de Mihaly Csikszentmihalyi" pour ne pas oublier cette source.
4. De retour chez elle, elle reprend la note sur son ordinateur, où elle peut voir la structure déjà organisée et commencer à développer chaque section.
5. Elle utilise la fonctionnalité de bloc toggle pour cacher certaines sections et se concentrer sur l'introduction qu'elle rédige maintenant manuellement.

## 2.4 Conception du système d'information

### 2.4.1 Identification des acteurs

Le système VoiceNotion interagit avec plusieurs types d'acteurs, chacun ayant des objectifs et des interactions spécifiques:

- **Utilisateur non authentifié:** Peut explorer la landing page, créer un compte ou se connecter.
- **Utilisateur authentifié:** Le principal acteur du système, qui peut créer, modifier, organiser et exporter des notes.

- **API Gemini:** Acteur système externe qui traite les commandes vocales et retourne des instructions structurées.
- **Service de stockage:** Acteur système responsable de la persistance et de la synchronisation des données.

#### 2.4.2 Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation ci-dessous illustre les principales interactions entre les acteurs et le système VoiceNotion:



./assets/docs/voicenotion\_use\_case\_updated.png

Figure 2.4: Diagramme de cas d'utilisation pour VoiceNotion

Les principaux cas d'utilisation incluent:

- **Gestion du compte:** Incription, connexion, modification du profil.
- **Gestion des notes:** Création, édition, suppression, création de sous-pages.
- **Interaction vocale:** Dictée de contenu, commandes de formatage, navigation par la voix.

- **Édition structurée:** Manipulation des blocs, formatage du texte, insertion d'éléments riches.
- **Recherche et filtrage:** Recherche textuelle, filtrage par date.
- **Exportation:** Export des notes vers différents formats.

### 2.4.3 Diagrammes de séquence

Pour illustrer les interactions dynamiques entre l'utilisateur, l'application et les services externes, nous avons créé des diagrammes de séquence pour les processus clés.

#### 2.4.3.1 Séquence de commande vocale

Le diagramme suivant montre la séquence d'interactions lors de l'utilisation d'une commande vocale pour manipuler le contenu:

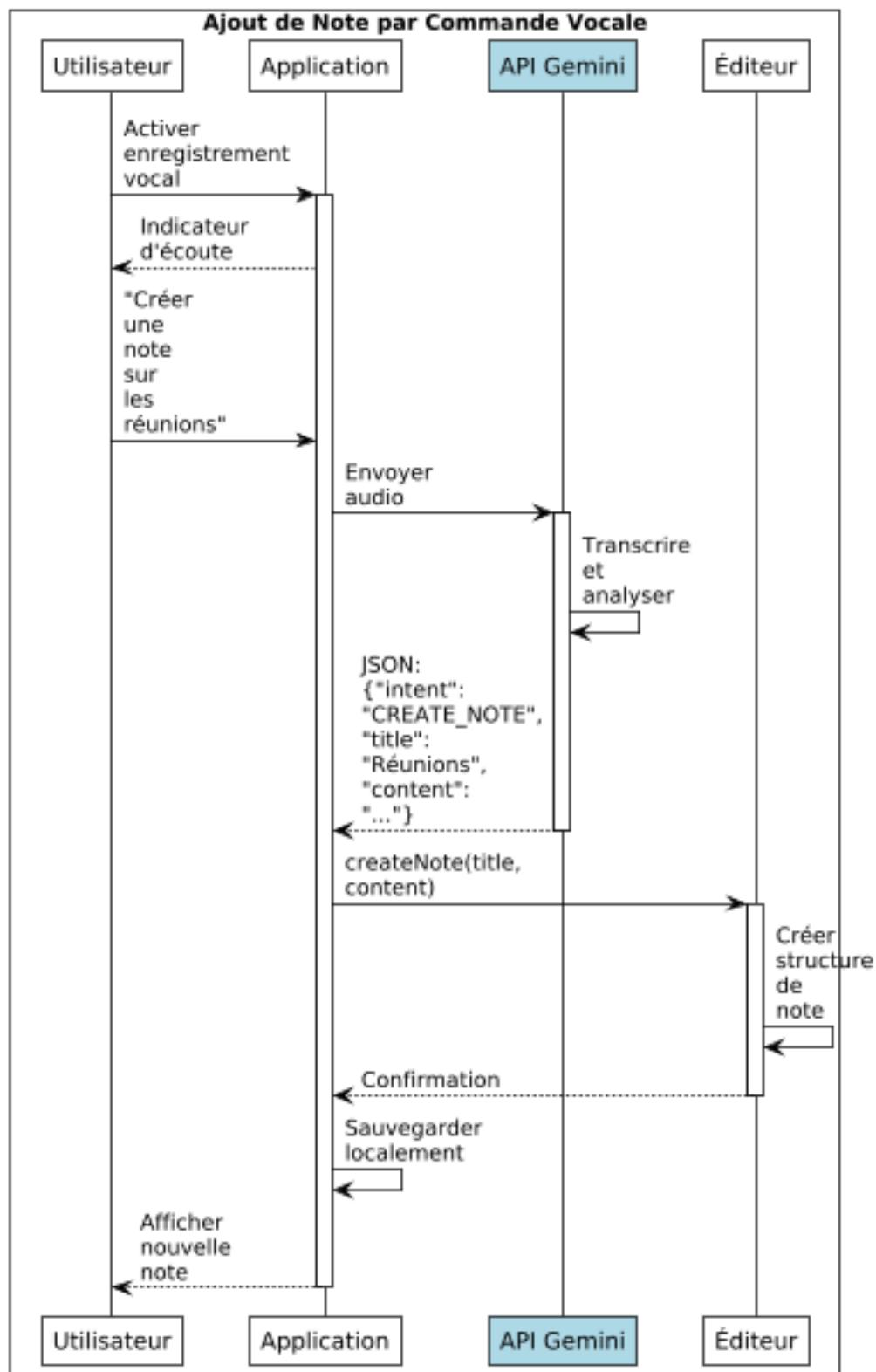


Figure 2.5: Diagramme de séquence pour le traitement d'une commande vocale

#### 2.4.3.2 Séquence de création et sauvegarde de note

Ce diagramme illustre le processus de création, d'édition et de sauvegarde d'une note:

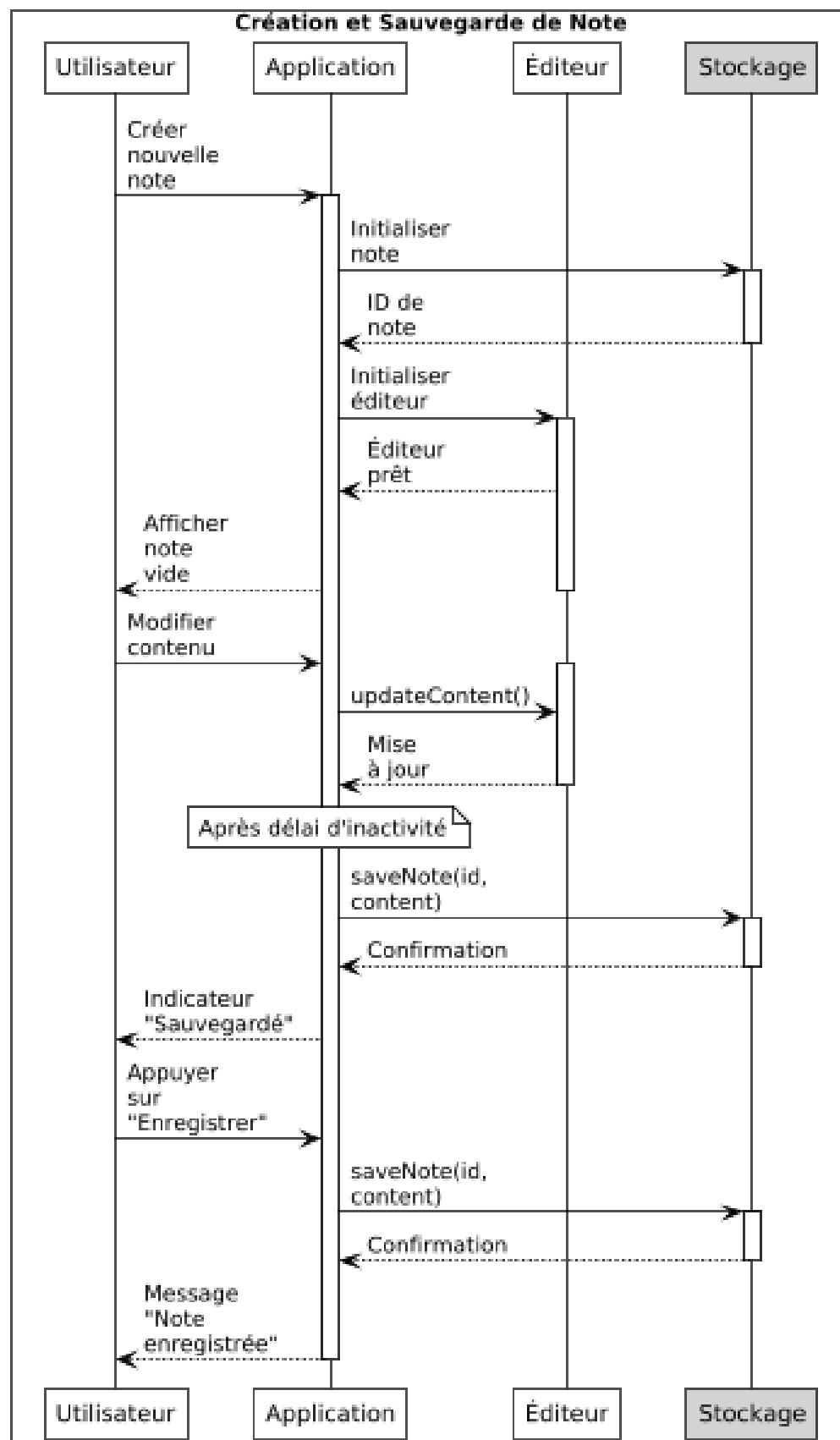


Figure 2.6: Diagramme de séquence pour la création et sauvegarde d'une note

### 2.4.3.3 Séquence d'authentification

Ce diagramme illustre le processus d'authentification des utilisateurs dans l'application:

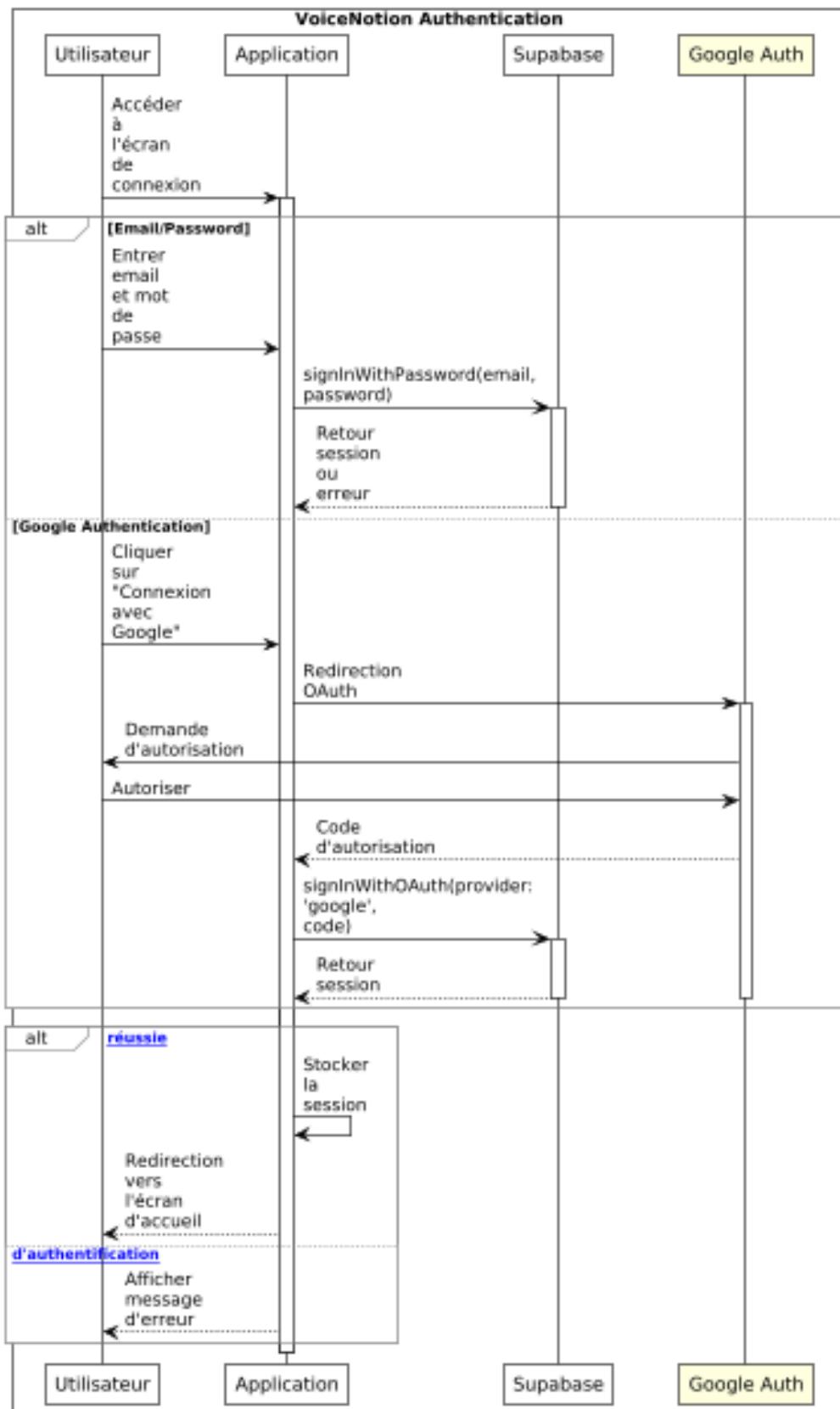


Figure 2.7: Diagramme de séquence pour l'authentification

## 2.4.4 Diagrammes de base de données

### 2.4.4.1 Diagramme entité-relation

Le modèle entité-relation ci-dessous représente la structure conceptuelle des données pour VoiceNotion:

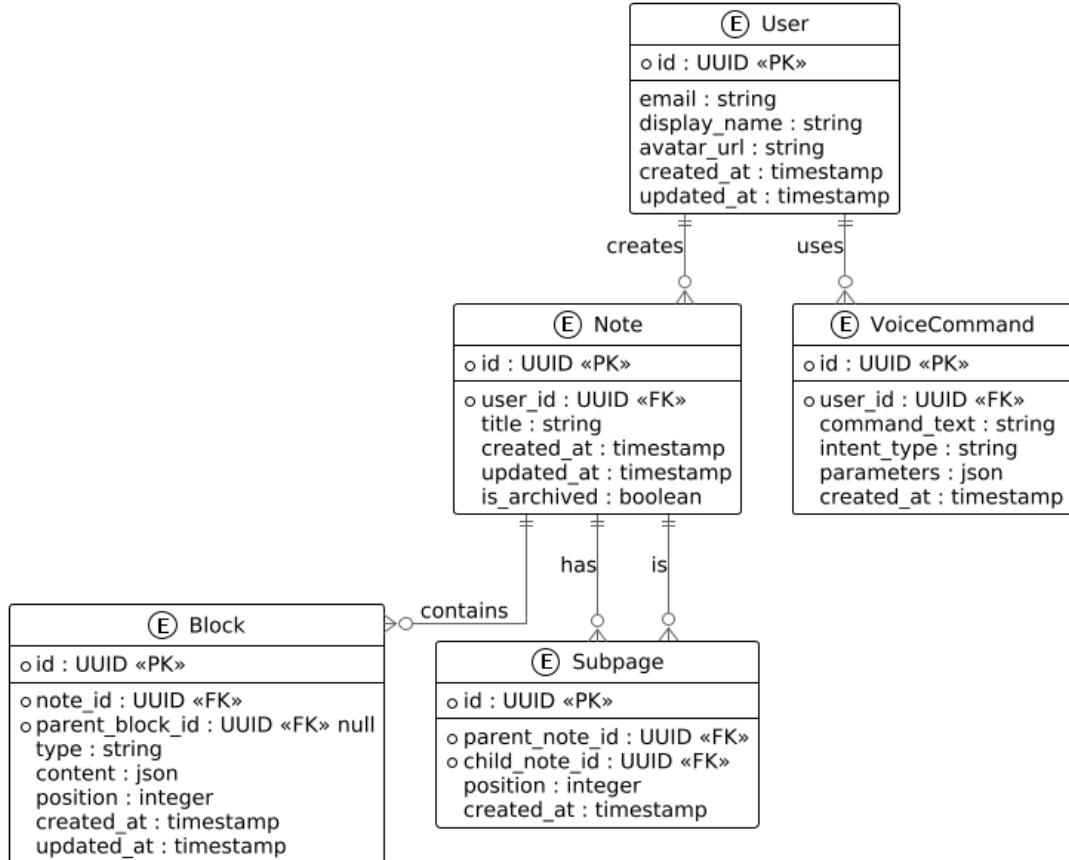


Figure 2.8: Diagramme entité-relation pour VoiceNotion

Les principales entités et leurs relations sont :

- **User**: Stocke les informations utilisateur (email, nom d'affichage, avatar).
- **Note**: L'entité centrale qui contient les métadonnées d'une note (titre, date de création/modification, propriétaire).
- **Block**: Représente un bloc individuel dans une note, avec son type, contenu et position.
- **Subpage**: Gère la relation hiérarchique entre les notes, permettant la création de sous-pages.
- **VoiceCommand**: Stocke les commandes vocales et leurs paramètres.

### 2.4.4.2 Diagramme de base de données

Le schéma logique de la base de données traduit le modèle entité-relation en une structure implantable:

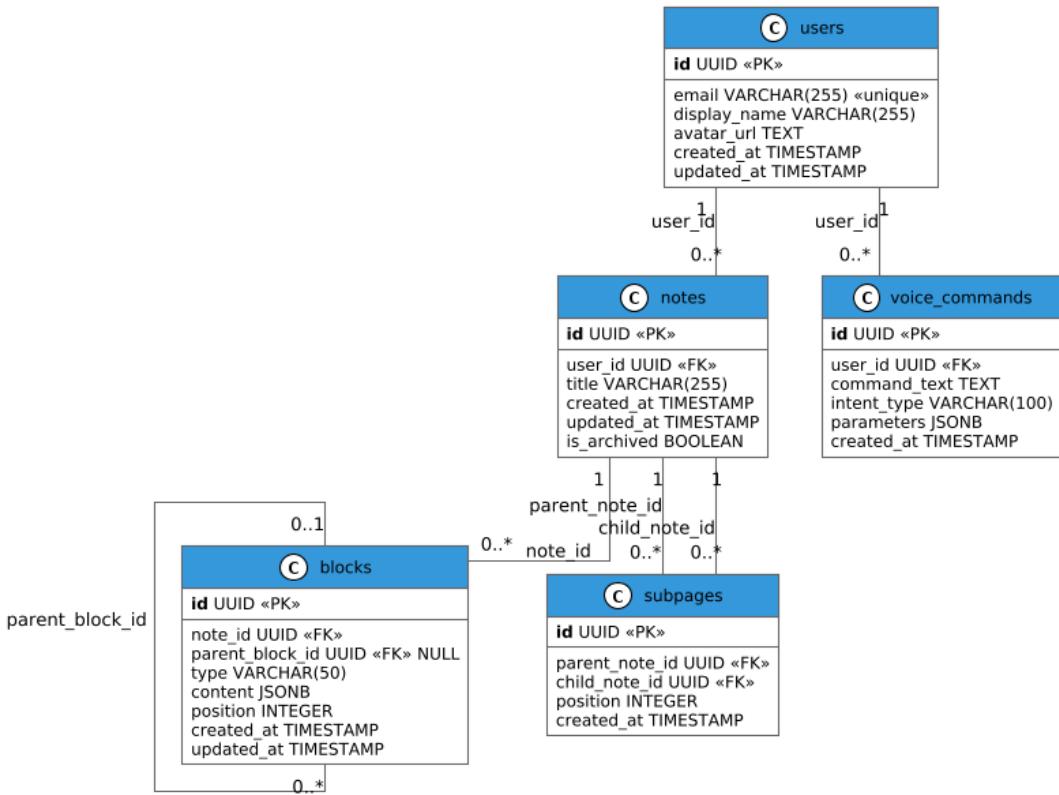


Figure 2.9: Schéma logique de la base de données pour VoiceNotion

Notre implémentation utilise une base de données PostgreSQL hébergée sur Supabase, avec les tables principales suivantes:

- **users**: id, email, display\_name, avatar\_url, created\_at, updated\_at
- **notes**: id, title, user\_id, created\_at, updated\_at, is\_archived
- **blocks**: id, note\_id, parent\_block\_id, type, content, position, created\_at, updated\_at
- **subpages**: id, parent\_note\_id, child\_note\_id, position, created\_at
- **voice\_commands**: id, user\_id, command\_text, intent\_type, parameters, created\_at

## 2.5 Conclusion

Ce chapitre a présenté notre approche méthodique de la planification du projet VoiceNotion et de la conception de son expérience utilisateur. En combinant une méthodologie agile avec une approche de Design Thinking centrée sur l'utilisateur, nous avons établi un cadre solide pour le développement d'une application qui répond véritablement aux besoins des utilisateurs.

La recherche utilisateur approfondie et l'élaboration de personas détaillés nous ont permis de comprendre intimement les défis et les aspirations de nos utilisateurs cibles. Les scénarios d'utilisation ont illustré comment VoiceNotion s'intégrerait naturellement

dans leurs flux de travail quotidiens, offrant une valeur réelle et des améliorations tangibles à leur expérience de prise de notes.

La conception technique du système, illustrée par les diagrammes de cas d'utilisation, de séquence et de base de données, fournit une base solide pour l'implémentation qui sera détaillée dans le chapitre suivant. Cette architecture a été conçue pour être robuste, évolutive et flexible, permettant d'accueillir les futures améliorations et extensions de l'application.

Les prochaines étapes consistent à transformer cette conception en un produit fonctionnel à travers le développement technique, en restant fidèle à notre vision d'une application de prise de notes révolutionnaire qui libère la créativité et la productivité de ses utilisateurs grâce à la puissance de la voix.

# **CHAPTER 3**

## **IMPLEMENTATION ET DEVELOPPEMENT**

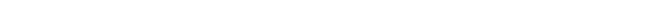
### **3.1 Introduction**

Cette partie du mémoire est consacrée à l'implémentation et au développement de l'application VoiceNotion. Nous allons détailler l'architecture technique, les technologies utilisées, et les différentes composantes de notre solution. VoiceNotion est une application de prise de notes vocale qui combine une interface web responsive et une application mobile, toutes deux partageant une base de données commune et des fonctionnalités similaires.

Notre approche de développement a été guidée par les principes de modularité, de réutilisabilité et d'expérience utilisateur fluide. Nous avons adopté des technologies modernes pour garantir la performance, la sécurité et l'évolutivité de notre solution.

### **3.2 Architecture globale**

L'architecture de VoiceNotion est construite autour d'une approche multi-plateforme avec un backend commun. Cette architecture permet de maintenir une expérience utilisateur cohérente tout en optimisant le développement pour chaque plateforme.

Figure 3.1: Architecture globale de VoiceNotion

### 3.2.1 Composants principaux

- **Application Web:** Développée avec Next.js et React, offrant une interface responsive et optimisée pour les navigateurs desktop et mobiles.
- **Application Mobile:** Construite avec React Native et Expo, permettant un déploiement natif sur iOS et Android.
- **Backend:** Utilisant Supabase comme solution Backend-as-a-Service (BaaS) pour l'authentification, la base de données, et le stockage.
- **API Gemini:** Intégration de l'API Gemini de Google pour la reconnaissance vocale et le traitement des commandes.

### 3.2.2 Flux de données

Le flux de données dans VoiceNotion suit un modèle client-serveur avec synchronisation en temps réel:

1. L'utilisateur interagit avec l'application (web ou mobile)
2. Les requêtes sont envoyées au backend Supabase via des API sécurisées
3. Les données sont stockées dans une base de données PostgreSQL
4. Les mises à jour sont synchronisées en temps réel entre les appareils grâce aux abonnements Supabase

### 3.3 Environnement de développement

#### 3.3.1 Matériels

Le développement de VoiceNotion a été réalisé sur les équipements suivants:

- MacBook Pro M1 (16GB RAM, 512GB SSD)
- iPhone 13 Pro (pour les tests iOS)
- Samsung Galaxy S21 (pour les tests Android)
- iPad Pro (pour les tests de la version tablette)

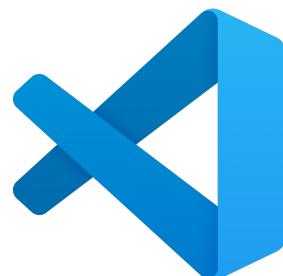
#### 3.3.2 Logiciels et outils de développement

##### 3.3.2.1 Visual Studio Code

Visual Studio Code est un éditeur de code source léger mais puissant qui s'exécute sur votre bureau et est disponible pour Windows, macOS et Linux. Il est fourni avec un support intégré pour JavaScript, TypeScript et Node.js et dispose d'un riche écosystème d'extensions pour d'autres langages et environnements de développement.

Visual Studio Code a été notre IDE principal pour le développement de VoiceNotion. Nous avons utilisé plusieurs extensions pour améliorer notre productivité:

- ESLint: Pour la vérification du code JavaScript/TypeScript
- Prettier: Pour le formatage automatique du code
- React Developer Tools: Pour le débogage des composants React
- Tailwind CSS IntelliSense: Pour l'autocomplétion des classes Tailwind
- GitLens: Pour une meilleure intégration avec Git



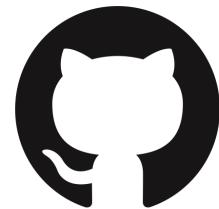
### 3.3.2.2 GitHub

GitHub est une plateforme de développement collaboratif basée sur Git, un système de contrôle de version distribué. Il est largement utilisé par les développeurs pour héberger, gérer et partager des projets de développement de logiciels.

Sur GitHub, les développeurs peuvent créer des dépôts pour stocker leur code source, collaborer avec d'autres développeurs sur des projets, suivre et gérer les problèmes, et faciliter le processus de développement via des fonctionnalités comme les pull requests, les actions, et plus encore.

Pour VoiceNotion, nous avons utilisé GitHub pour:

- Gestion du code source avec branches pour chaque fonctionnalité
- Organisation du travail d'équipe via les issues et les projets
- Intégration continue avec GitHub Actions
- Revue de code via les pull requests
- Documentation du projet dans le wiki et le README



### 3.3.2.3 TypeScript

TypeScript est un sur-ensemble de JavaScript developpe par Microsoft qui ajoute des types statiques optionnels au langage. Il est conçu pour le développement d'applications à grande échelle et se compile en JavaScript standard.

Les avantages de TypeScript que nous avons exploitez dans VoiceNotion:

- Langage type qui permet de détecter les erreurs lors de la compilation
- Compilation en différentes versions ECMAScript à partir de la version 3
- De nombreux outils disponibles et une intégration parfaite avec VS Code
- Un langage orienté objet avec l'introduction du typage, de l'héritage et des notions de public et private
- La transition du JavaScript vers TypeScript peut se faire progressivement
- La transition inverse très simple grâce à la transpilation en ECMAScript



TypeScript nous a permis de développer un code plus robuste et plus maintenable, particulièrement important pour une application comme VoiceNotion qui nécessite une gestion complexe des interactions utilisateur et des données.

### 3.3.2.4 Zod

Zod est une bibliotheque de validation de schemas axee sur TypeScript. Elle offre une API puissante et expressive pour definir et valider des schemas de donnees.

Avec Zod, nous pouvons facilement definir des regles de validation complexes pour differents types de donnees tels que les chaines de caracteres, les nombres, les tableaux, les objets, et bien d'autres. Il prend en charge des fonctionnalites avancees telles que la validation conditionnelle, les messages d'erreur personnalises et la composition de schemas.

Zod favorise le typage fort et l'inference de types, ce qui en fait un choix ideal pour les projets TypeScript. Il s'integre egale-ment parfaitement avec des frameworks et des bibliotheques populaires comme React et Express.

Dans VoiceNotion, nous utilisons Zod pour:



- Valider les donnees des formulaires utilisateur
- Verifier l'integrite des donnees provenant de l'API
- Generer des types TypeScript a partir des schemas de validation
- Assurer la coherence des donnees entre le frontend et le backend

### 3.3.2.5 Tests

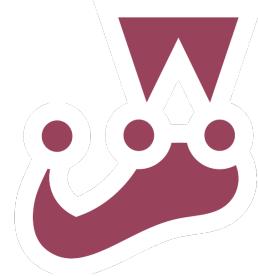
Pour assurer la qualite et la fiabilite de notre application, nous avons mis en place une strategie de tests complete avec Jest et Cypress.

**Jest** est un framework de test JavaScript concu pour assurer la correction de n'importe quel code JavaScript. Jest est complet et facile a configurer, et nous l'avons utilise pour les tests unitaires et d'integration.

**Cypress** est un framework de test end-to-end qui nous permet de tester notre application comme le ferait un utilisateur reel. Il offre une experience de test fiable, rapide et facile a comprendre.

Notre approche de test comprend:

- Tests unitaires pour les fonctions et composants individuels
- Tests d'integration pour verifier les interactions entre composants
- Tests end-to-end pour simuler les parcours utilisateur complets
- Tests d'accessibilite pour garantir que l'application est utilisable par tous

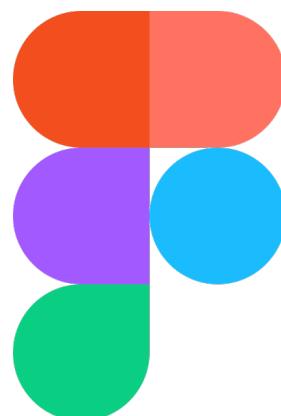


## 3.4 Outils de conception et design

### 3.4.1 Figma

Figma est un outil de conception d'interface utilisateur base sur le cloud qui permet aux equipes de collaborer en temps reel. Il est devenu notre outil principal pour la conception de l'interface utilisateur et la creation de prototypes interactifs. Il nous a permis de:

- Creer des wireframes et des maquettes haute fidelite
- Concevoir un systeme de design coherent avec des composants reutilisables
- Collaborer en temps reel sur les designs
- Tester les interactions via des prototypes cliquables
- Generer des specifications pour les developpeurs



L'interface intuitive de Figma et ses fonctionnalites avancees ont grandement facilite le processus de conception, permettant a notre equipe de travailler efficacement meme a distance.

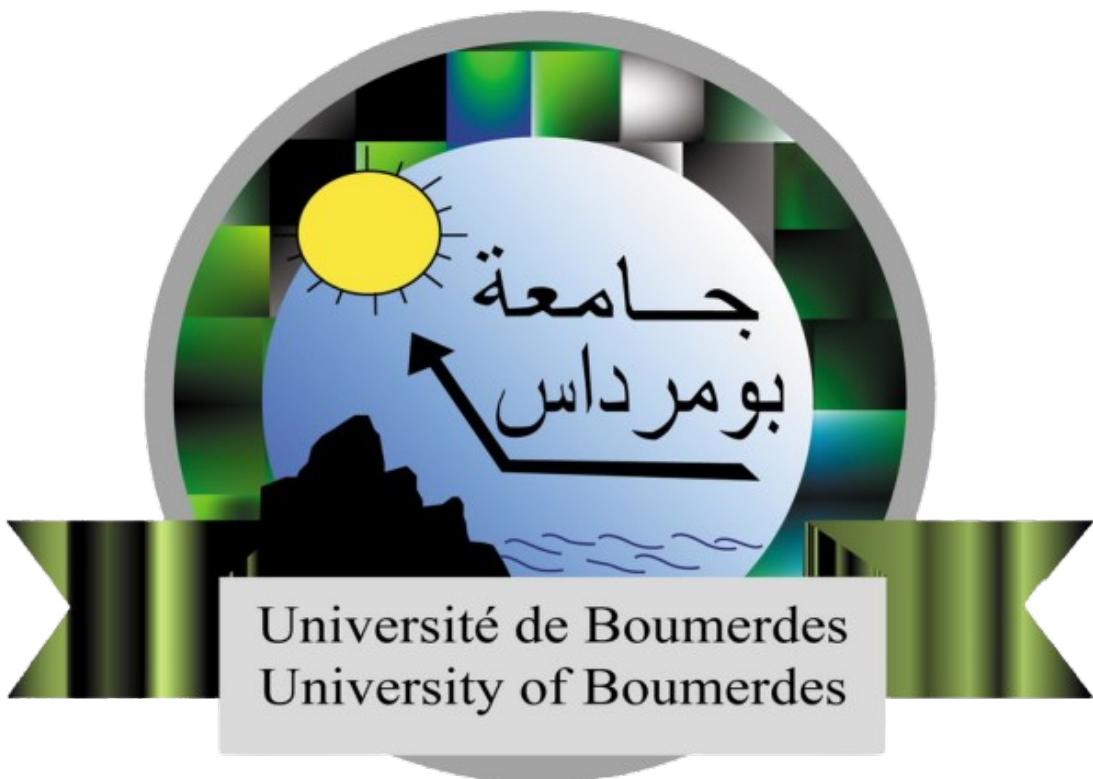


Figure 3.2: Système de design VoiceNotion dans Figma

### 3.4.2 Adobe Illustrator

Adobe Illustrator est un logiciel de création graphique et de dessin vectoriel largement utilisé dans l'industrie du design. Il offre un large éventail d'outils et de fonctionnalités qui permettent de créer des illustrations, des logos, des icônes, des graphiques et d'autres éléments visuels de haute qualité.

Illustrator utilise des vecteurs pour créer des images, ce qui signifie que les dessins peuvent être redimensionnés et modifiés sans perte de qualité. Il prend en charge la création de formes, le tracé de courbes, l'application de couleurs et de dégradés, la manipulation des calques et bien plus encore.

Nous avons utilisé Adobe Illustrator pour créer les éléments graphiques vectoriels de notre identité visuelle:

- Logo VoiceNotion et ses variantes
- Icônes personnalisées
- Illustrations pour le site web et l'application
- Matériel marketing (bannières, visuels pour réseaux sociaux)



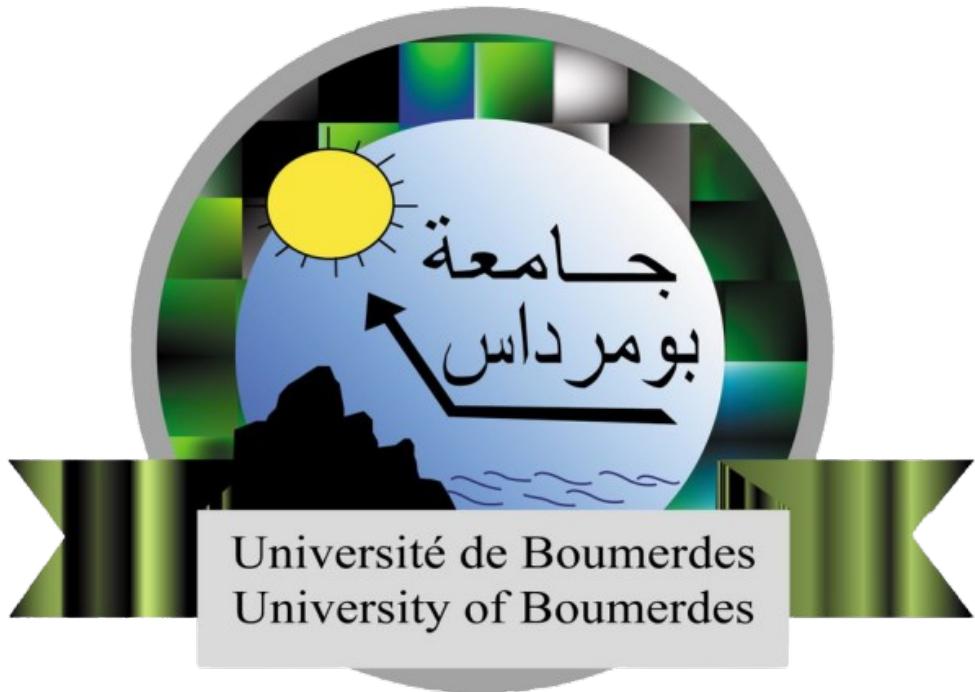


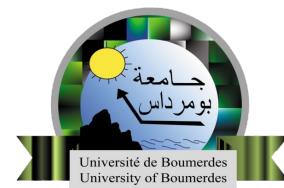
Figure 3.3: Creation des elements graphiques avec Adobe Illustrator

### 3.4.3 Adobe Photoshop

Adobe Photoshop est un logiciel de retouche d'image et de creation graphique largement utilise dans le domaine de la conception, de la photographie et du multimedia. Il offre une gamme complete d'outils et de fonctionnalites avances pour manipuler et ameliorer les images.

Avec Photoshop, nous avons effectue des retouches precises, ajuste la luminosite et le contraste, corrige les couleurs, supprime des objets indesirables, et cree des compositions complexes pour notre application. Cet outil nous a ete particulierement utile pour:

- Retoucher les captures d'écran de l'application
- Creer des mockups realistes pour presentations
- Preparer des images optimisees pour le web et les applications mobiles
- Creer des elements graphiques complexes combines avec Illustrator



### 3.5 Backend et services cloud

#### 3.5.1 Supabase

Supabase est une solution Backend-as-a-Service (BaaS) open-source qui offre une alternative à Firebase. Cette plateforme nous offre:

- Une base de données PostgreSQL performante et évolutive
- Un système d'authentification sécurisé avec plusieurs méthodes de connexion
- Des API RESTful et GraphQL générées automatiquement
- Des fonctionnalités de temps réel pour la synchronisation des données
- Un stockage de fichiers intégré

Nous avons choisi Supabase pour sa flexibilité, sa scalabilité et sa compatibilité avec PostgreSQL, ce qui nous permet de bénéficier d'une base de données relationnelle complète sans avoir à gérer l'infrastructure sous-jacente. L'authentification intégrée et les fonctionnalités en temps réel ont également considérablement accéléré notre développement.

Avec Supabase, nous avons pu implémenter rapidement des fonctionnalités essentielles comme la synchronisation des notes entre appareils, la gestion des utilisateurs et le stockage des fichiers multimédia associés aux notes.

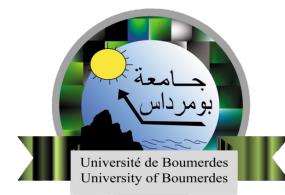




Figure 3.4: Dashboard Supabase pour VoiceNotion

#### Configuration Supabase

```
// lib/supabase.ts
import { createClient } from '@supabase/supabase-js';

// Recuperation des variables d'environnement
const supabaseUrl = process.env.NEXT_PUBLIC_SUPABASE_URL!;
const supabaseAnonKey = process.env.NEXT_PUBLIC_SUPABASE_ANON_KEY!;

// Creation et export du client Supabase
export const supabase = createClient(supabaseUrl, supabaseAnonKey);
```

### 3.5.2 Prisma

Prisma est un ORM (Object-Relational Mapping) moderne, mais concu de maniere tres differente de ce qui se fait actuellement dans l'industrie. En general, les ORM sont des bibliothèques qui font correspondre les tables de votre base de donnees aux classes du langage que vous utilisez pour ecrire votre programme.

Prisma, quant a lui, est une boite a outils de base de donnees complete. En plus, Prisma ne souffre pas des nombreux problemes qui sont communement associes aux ORM traditionnels. L'équipe estime en effet que l'approche des ORM traditionnels conduit a de nombreux problemes causes par le decalage d'impedance objet-relationnel.

C'est une situation que la conception de Prisma permet d'éviter. L'un des principaux differentiateurs entre Prisma et un ORM est son fichier de schema centralise et son langage de definition de schema. Ce schema definit a la fois les modeles de donnees de l'application et le schema de la base de donnees. Dans VoiceNotion, Prisma nous permet de:

- Definir notre schema de base de donnees de maniere declarative
- Generer des types TypeScript a partir du schema
- Effectuer des migrations de base de donnees de maniere securisee
- Interagir avec la base de donnees via une API typee



### 3.5.3 Hébergement et déploiement

#### 3.5.3.1 Vercel

Vercel est une plateforme cloud pour les sites statiques et les applications serverless. Elle permet aux développeurs de déployer des sites web et des applications web avec une configuration zéro ou minimale.

Vercel offre un flux de travail optimisé pour le développement web moderne, avec des déploiements instantanés, des domaines personnalisés, des certificats SSL automatiques, et une mise à l'échelle globale.

Pour VoiceNotion, nous avons choisi Vercel pour:

- Déployer notre application Next.js avec une configuration minimale
- Beneficier d'une intégration continue avec GitHub
- Obtenir des aperçus de déploiement pour chaque pull request
- Accéder à un réseau de diffusion de contenu (CDN) mondial
- Déployer des API serverless via les routes API de Next.js

Vercel simplifie considérablement notre workflow de déploiement et nous permet de nous concentrer sur le développement de nouvelles fonctionnalités plutôt que sur la gestion de l'infrastructure.



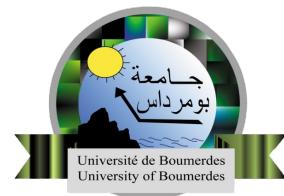
### 3.5.3.2 Expo EAS

Expo EAS (Expo Application Services) est un ensemble d'outils et de services pour le déploiement d'applications React Native. Il simplifie le processus de build, de soumission et de mise à jour des applications mobiles.

EAS offre plusieurs services clés:

- EAS Build: pour compiler les applications iOS et Android dans le cloud
- EAS Submit: pour soumettre des applications aux App Stores
- EAS Update: pour déployer des mises à jour over-the-air (OTA)

Pour VoiceNotion, EAS nous a permis de:



- Compiler notre application React Native sans avoir à configurer localement Xcode ou Android Studio
- Gérer différents environnements (développement, pré-view, production)
- Déployer rapidement des correctifs sans passer par le processus de validation des App Stores
- Automatiser le processus de soumission aux stores

## 3.6 Site web (Front-end)

### 3.6.1 Technologies et outils utilises

#### 3.6.1.1 React.js

React est une bibliothèque JavaScript open-source pour la création d'interfaces utilisateur interactives et réactives. Développée et maintenue par Facebook (Meta), React est devenue l'un des frameworks frontend les plus populaires dans l'écosystème du développement web.

Les principales caractéristiques de React qui ont guidé notre choix pour VoiceNotion sont :

- **Approche composant:** React permet de construire des interfaces utilisateur modulaires à partir de petits composants réutilisables.
- **DOM virtuel:** React utilise un DOM virtuel pour optimiser les performances en minimisant les manipulations directes du DOM.
- **Flux de données unidirectionnel:** Les données circulent du parent vers l'enfant, ce qui rend le code plus prévisible et plus facile à debugger.
- **Vaste écosystème:** React dispose d'un écosystème riche de bibliothèques et d'outils complémentaires.
- **Grande communauté:** La communauté active autour de React facilite la résolution de problèmes et l'accès aux ressources.



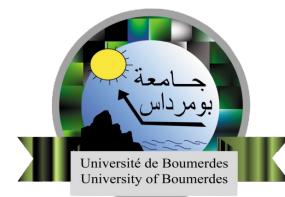
React nous a permis de créer une interface utilisateur fluide et réactive pour notre application web, avec une architecture de composants modulaire et maintenable.

### 3.6.1.2 Next.js

Next.js est un framework React qui offre des fonctionnalités avancées comme le rendu côté serveur (SSR), la génération de sites statiques (SSG), et une architecture basée sur les routes par fichiers. Développé par Vercel, Next.js simplifie considérablement la création d'applications React complexes.

Pour VoiceNotion, nous avons choisi Next.js 15 pour:

- **Performances optimisées:** Rendu hybride qui combine SSR, SSG et CSR selon les besoins
- **Routage simplifié:** Système de routage basé sur le système de fichiers
- **API Routes:** Création d'API serverless intégrées à l'application
- **Optimisation des images:** Redimensionnement, optimisation et diffusion automatiques des images
- **Server Components:** Composants exécutés uniquement sur le serveur pour réduire le JavaScript envoyé au client
- **Server Actions:** Fonctions côté serveur appelables directement depuis les composants



L'utilisation de Next.js nous a permis de construire une application web performante et optimisée pour les moteurs de recherche, tout en conservant l'expérience de développement React que nous apprécions.

#### Configuration Next.js avec TypeScript

```
// next.config.js
/** @type {import('next').NextConfig} */
const nextConfig = {
  reactStrictMode: true,
  images: {
    domains: ['localhost', 'supabase.co'],
  },
  experimental: {
    serverActions: true,
  },
}

module.exports = nextConfig
```

### 3.6.1.3 TailwindCSS

TailwindCSS est un framework CSS utilitaire qui permet de construire rapidement des interfaces personnalisées sans quitter votre HTML. Contrairement aux frameworks comme Bootstrap qui fournissent des composants prédefinis, Tailwind offre des classes utilitaires de bas niveau que vous combinez pour créer votre design unique.

Les avantages de TailwindCSS qui ont guidé notre choix sont:

- **Développement rapide:** Création d'interfaces sans écrire de CSS personnalisé
- **Personnalisation:** Facilite de personnalisation via un fichier de configuration
- **Responsivité intégrée:** Classes adaptées aux différentes tailles d'écran
- **Optimisation pour la production:** Elimination automatique des classes non utilisées
- **Conception cohérente:** Système de design intégré avec espacement, couleurs et typographie cohérents



TailwindCSS nous a permis de développer une interface utilisateur cohérente et responsive pour VoiceNotion, tout en maintenant un bundle CSS minimal pour des performances optimales.

## Exemple de composant avec TailwindCSS

```
// components/Button.tsx
interface ButtonProps {
  primary?: boolean;
  children: React.ReactNode;
  onClick?: () => void;
}

export function Button({ primary = false, children, onClick }: ButtonProps) {
  return (
    <button
      onClick={onClick}
      className={`px-4 py-2 rounded-md font-medium transition-colors
      ${primary
        ? "bg-blue-500 text-white hover:bg-blue-600"
        : "bg-gray-200 text-gray-800 hover:bg-gray-300`}
      `}
    >
      {children}
    </button>
  );
}
```

### 3.6.1.4 BlockNote

BlockNote est un éditeur de texte riche pour React, inspiré par Notion et optimisé pour la création de documents structurés. Il offre une expérience d'édition moderne avec une architecture de blocs.

Les principales caractéristiques de BlockNote qui nous ont convaincus sont :

- **Architecture de blocs:** Organisation du contenu en blocs distincts (paragraphes, listes, titres, etc.)
- **Interface intuitive:** Interface conviviale inspirée des éditeurs modernes comme Notion
- **Personnalisation complète:** Possibilité d'ajouter des types de blocs personnalisés
- **Serialisation et déserialisation:** Conversion facile entre HTML, JSON et Markdown
- **API extensible:** Possibilité d'étendre les fonctionnalités de base

Pour VoiceNotion, BlockNote était le choix idéal car il nous permettait d'intégrer des commandes vocales à un éditeur de texte riche, tout en offrant une expérience de prise de notes structurée similaire aux applications populaires comme Notion.



## Integration de BlockNote

```
// components/Editor.tsx
import { BlockNoteEditor, PartialBlock } from '@blocknote/core';
import { BlockNoteView, useBlockNote } from '@blocknote/react';
import '@blocknote/react/style.css';
import { useEffect } from 'react';

export default function Editor({ initialContent, onChange }) {
    // Initialisation de l'éditeur BlockNote
    const editor = useBlockNote({
        initialContent,
        onEditorContentChange: onChange ? (editor) => onChange(editor)
            : undefined,
        // Configuration personnalisée pour VoiceNotion
        editorOptions: {
            enableSpeech: true, // Activation de la fonctionnalité
            // vocale
            placeholderText: "Commencez à taper ou utilisez le bouton
                microphone...",
        }
    });

    // Hooks pour la sauvegarde automatique
    useEffect(() => {
        const interval = setInterval(() => {
            if (editor && onChange) {
                onChange(editor);
            }
        }, 5000); // Sauvegarde toutes les 5 secondes

        return () => clearInterval(interval);
    }, [editor, onChange]);
}

return (
    <BlockNoteView
        editor={editor}
        theme="light"
        className="min-h-[300px] border rounded-md shadow-sm"
    />
);
}
```

### 3.6.2 Interfaces principales

#### 3.6.2.1 Page d'accueil

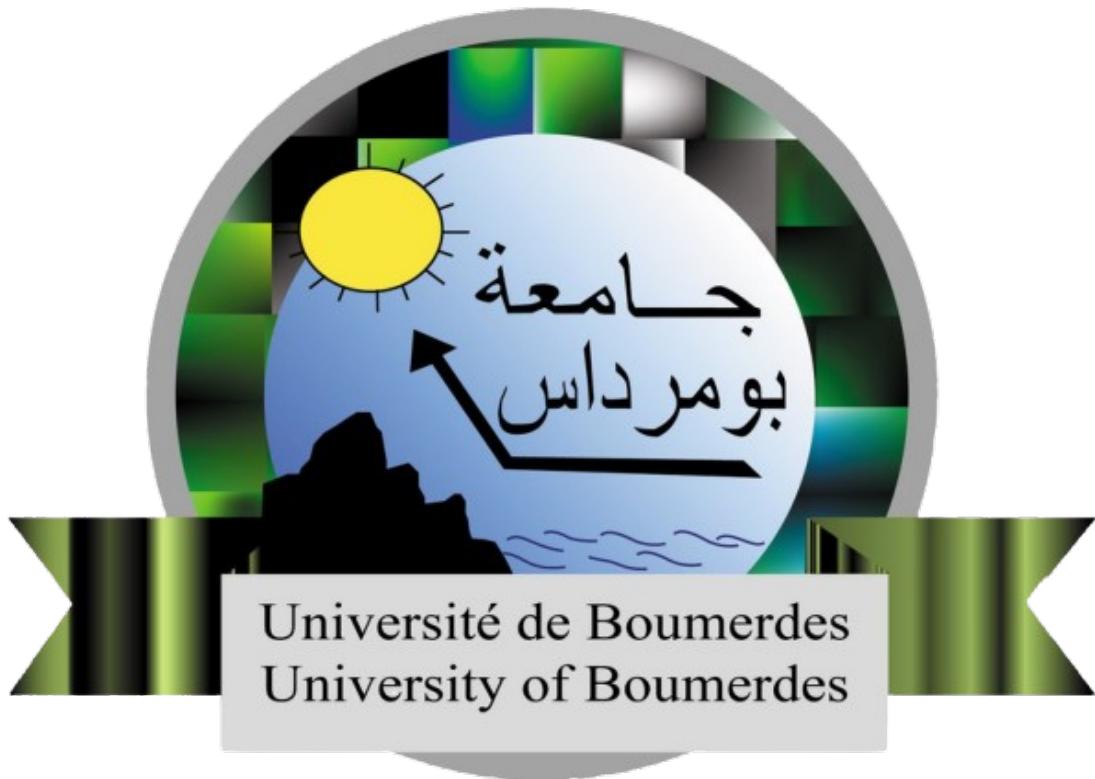


Figure 3.5: Page d'accueil du site VoiceNotion

#### 3.6.2.2 Authentification

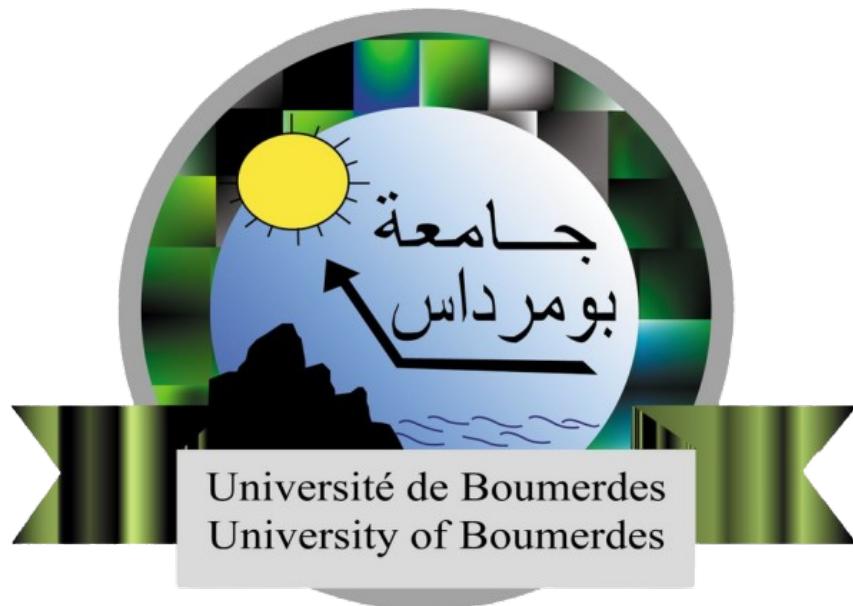


Figure 3.6: Page de connexion du site VoiceNotion



Figure 3.7: Page d'inscription du site VoiceNotion

## Composant d'authentification

```
// components/AuthForm.tsx
"use client";

import { useState } from "react";
import { supabase } from "@lib/supabase";
import { Button } from "@components/ui/Button";
import { Input } from "@components/ui/Input";

export default function AuthForm({ mode = "login" }) {
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
  const [loading, setLoading] = useState(false);
  const [error, setError] = useState("");

  const handleSubmit = async (e) => {
    e.preventDefault();
    setLoading(true);
    setError("");

    try {
      if (mode === "login") {
        const { error } = await supabase.auth.signInWithEmailAndPassword(
          email,
          password,
        );
        if (error) throw error;
      } else {
        const { error } = await supabase.auth.signUp({
          email,
          password,
        });
        if (error) throw error;
      }
    } catch (error) {
      setError(error.message);
    } finally {
      setLoading(false);
    }
  };
}

return (
  <form onSubmit={handleSubmit} className="space-y-4">
    {error && <div className="p-3 bg-red-100 text-red-700 rounded">{error}</div>}

    <div>
      <label htmlFor="email" className="block text-sm font-medium">
        Email
      </label>
      <Input
        id="email"
        type="email"
        value={email}
        onChange={(e) => setEmail(e.target.value)}
        required
      />
    </div>
    <div>
      <label htmlFor="password" className="block text-sm font-
medium">
```

3.6.2.3 Tableau de bord

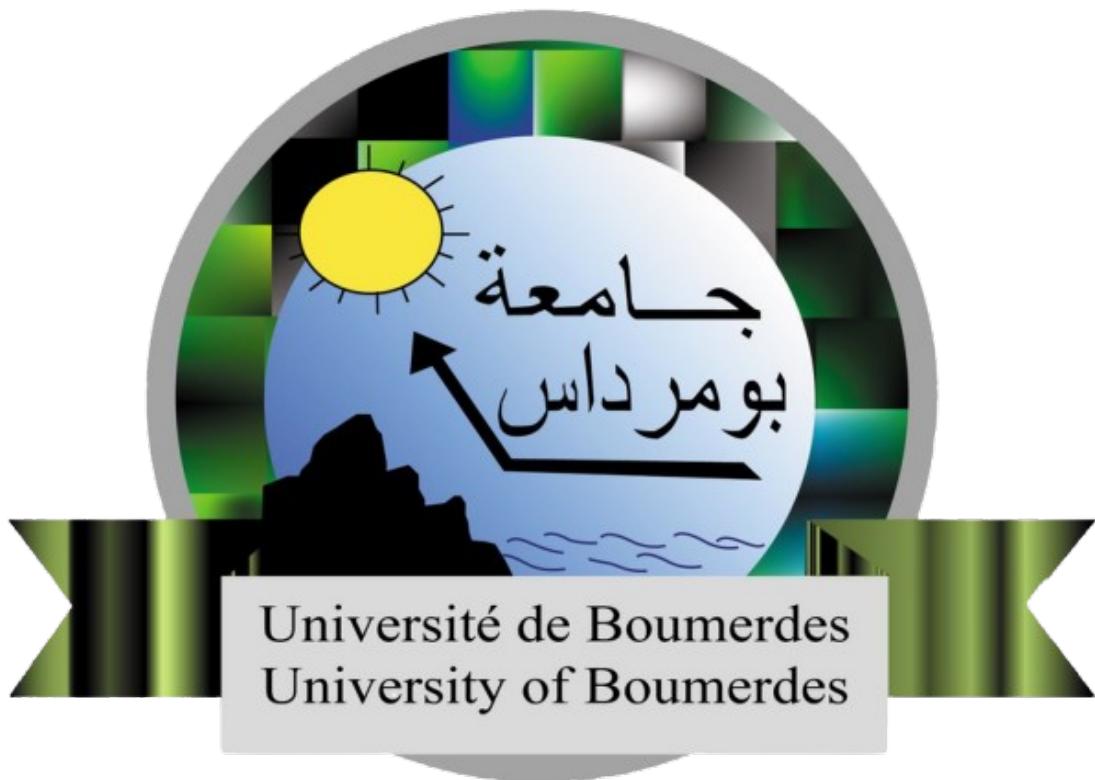


Figure 3.8: Tableau de bord du site VoiceNotion

3.6.2.4 Éditeur de notes



Figure 3.9: Éditeur de notes avec commandes vocales

### 3.7 Application mobile

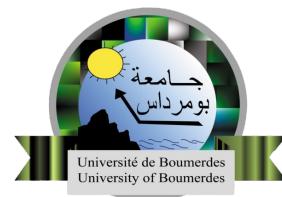
#### 3.7.1 Technologies et outils utilises

##### 3.7.1.1 React Native

React Native est un framework de développement d'applications mobiles créé par Facebook (Meta) qui permet de construire des applications natives pour Android et iOS à partir d'une base de code JavaScript/React commune. Il offre une approche "apprendre une fois, écrire partout" pour le développement mobile.

Les principales caractéristiques de React Native qui ont guidé notre choix pour VoiceNotion sont:

- **Composants natifs:** React Native compile le code JavaScript en composants natifs, offrant des performances proches des applications natives
- **Partage de code:** Possibilité de partager une grande partie du code entre les plateformes iOS et Android
- **Hot Reloading:** Visualisation instantanée des modifications du code pendant le développement
- **Communauté active:** Large écosystème de bibliothèques et support communautaire
- **Approche déclarative:** Interface utilisateur construite de manière déclarative, similaire à React



React Native nous a permis de développer l'application mobile VoiceNotion pour iOS et Android à partir d'une seule base de code, tout en offrant une expérience utilisateur native et performante sur chaque plateforme.

## Configuration du projet React Native

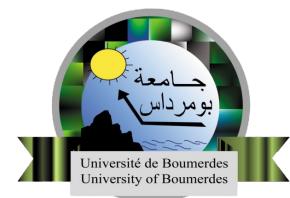
```
// app.json
{
  "expo": {
    "name": "VoiceNotion",
    "slug": "voicenotion",
    "version": "1.0.0",
    "orientation": "portrait",
    "icon": "./assets/icon.png",
    "userInterfaceStyle": "light",
    "splash": {
      "image": "./assets/splash.png",
      "resizeMode": "contain",
      "backgroundColor": "#ffffff"
    },
    "assetBundlePatterns": ["**/*"],
    "ios": {
      "supportsTablet": true,
      "bundleIdentifier": "com.voicenotion.app"
    },
    "android": {
      "adaptiveIcon": {
        "foregroundImage": "./assets/adaptive-icon.png",
        "backgroundColor": "#ffffff"
      },
      "package": "com.voicenotion.app"
    },
    "plugins": [
      [
        "expo-av",
        {
          "microphonePermission": "Autoriser VoiceNotion à accéder à votre microphone."
        }
      ]
    ]
  }
}
```

### 3.7.1.2 Expo

Expo est une plateforme et un ensemble d'outils construits autour de React Native qui simplifient considérablement le développement, le test et le déploiement d'applications mobiles. Il offre une approche "batteries included" pour React Native.

Les avantages d'Expo qui ont guidé notre choix pour VoiceNotion sont:

- **Configuration simplifiée:** Pas besoin de configurer manuellement le SDK natif d'Android ou iOS
- **Modules préintégrés:** Accès à des APIs natives courantes (caméra, géolocalisation, etc.) via des modules préconfigurés
- **Expo Go:** Application de test qui permet de visualiser les changements en temps réel sur des appareils physiques
- **EAS (Expo Application Services):** Services de build, déploiement et mise à jour des applications
- **Over-the-air updates:** Possibilité de déployer des mises à jour sans passer par les App Stores



Expo nous a permis d'accélérer considérablement le développement de l'application mobile VoiceNotion, en simplifiant l'accès aux fonctionnalités natives et en offrant un flux de travail optimisé du développement au déploiement.

### 3.7.2 Interfaces principales

#### 3.7.2.1 Ecrans d'accueil et d'authentification

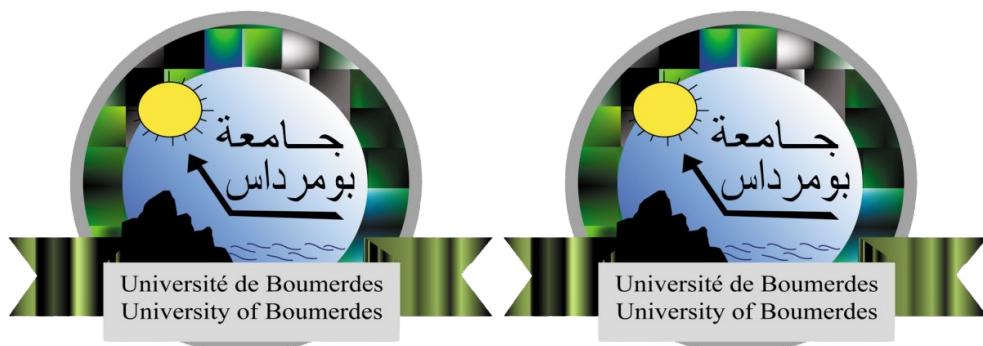


Figure 3.10: Ecrans d'accueil et de connexion de l'application mobile

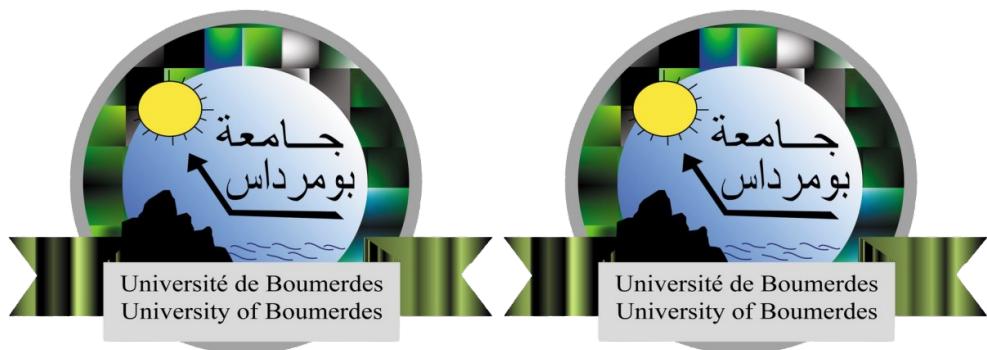


Figure 3.11: Ecrans d'inscription et d'accueil de l'application mobile

### 3.7.2.2 Editeur et saisie vocale

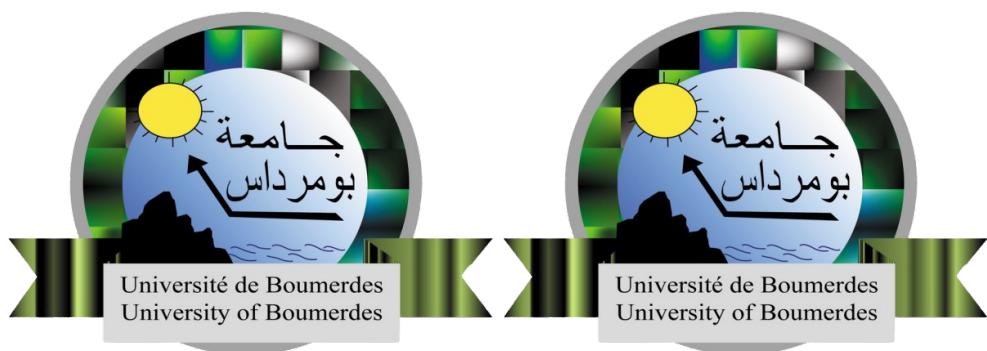


Figure 3.12: Editeur de notes et interface de saisie vocale

### 3.8 Integration de la reconnaissance vocale

#### 3.8.1 Gemini API

L'API Gemini de Google est un modèle d'IA multimodal avancé qui nous permet d'intégrer des capacités de traitement du langage naturel et de compréhension contextuelle dans VoiceNotion.

Cette API est au cœur de notre fonctionnalité de reconnaissance vocale et nous l'utilisons pour deux fonctions principales:

- **Transcription de la parole en texte (Speech-to-Text):** Conversion précise de l'audio vocal en texte écrit
- **Analyse d'intention:** Compréhension et interprétation des commandes vocales pour exécuter les actions appropriées dans l'application

Gemini nous offre plusieurs avantages clés:

- **Compréhension contextuelle:** Capacité à comprendre le contexte des commandes vocales
- **Support multilingue:** Prise en charge de multiples langues pour une utilisation internationale
- **Adaptabilité:** Possibilité d'affiner le modèle pour notre cas d'utilisation spécifique
- **Intégration simple:** API REST facile à intégrer dans nos applications web et mobile

Grâce à Gemini, VoiceNotion peut offrir une expérience de dictée vocale naturelle et intuitive, permettant aux utilisateurs de créer et de modifier du contenu à l'aide de commandes vocales complexes.





Figure 3.13: Integration de l'API Gemini dans VoiceNotion

### 3.8.2 Flux de traitement des commandes vocales

Le traitement des commandes vocales dans VoiceNotion suit un flux bien défini:

1. Capture de l'audio via le microphone de l'appareil
2. Conversion de l'audio en texte via l'API Gemini
3. Analyse de l'intention de la commande (également via Gemini)
4. Execution de l'action correspondante dans l'éditeur
5. Retour visuel et auditif à l'utilisateur

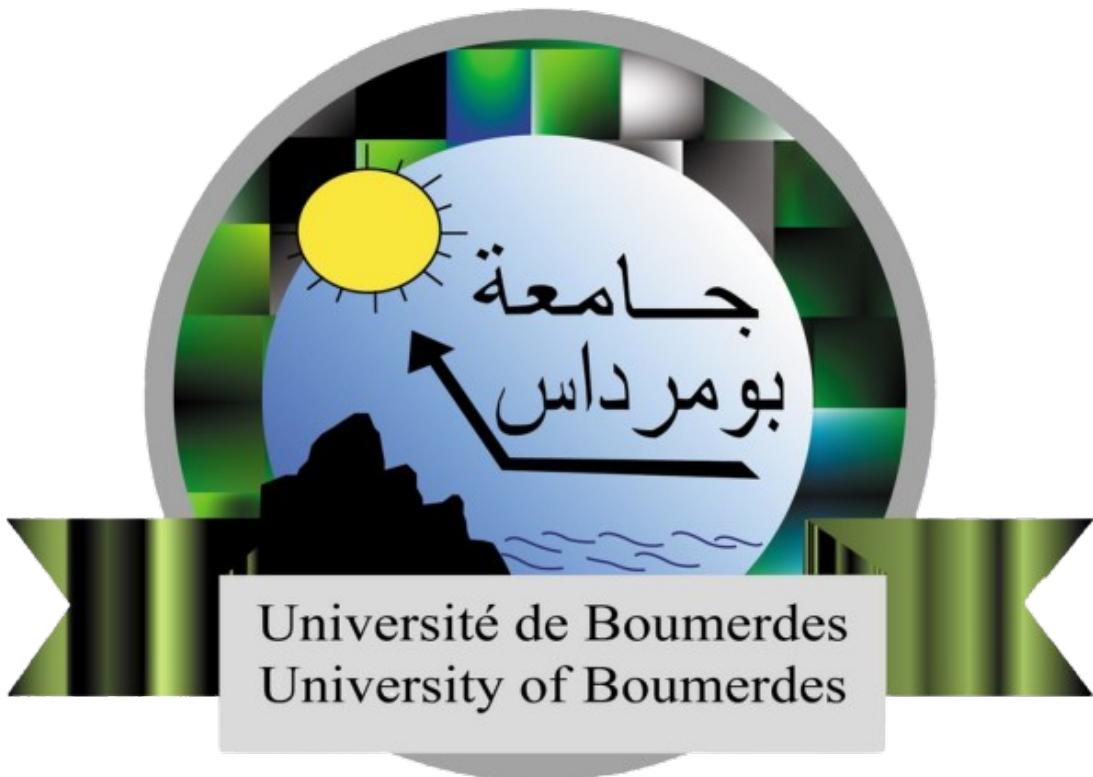


Figure 3.14: Flux de traitement des commandes vocales

### 3.9 Tests et assurance qualité

#### 3.9.1 Stratégie de test

Notre stratégie de test pour VoiceNotion comprend plusieurs niveaux:

- **Tests unitaires:** Vérification du comportement des composants individuels
- **Tests d'intégration:** Validation des interactions entre les différentes parties du système
- **Tests end-to-end:** Simulation des parcours utilisateur complets
- **Tests de performance:** Évaluation des temps de réponse et de la consommation de ressources
- **Tests d'accessibilité:** Vérification de la conformité aux standards WCAG

#### 3.9.2 Outils de test

- **Jest:** Framework de test pour les tests unitaires et d'intégration
- **React Testing Library:** Bibliothèque pour tester les composants React
- **Cypress:** Outil pour les tests end-to-end de l'application web
- **Detox:** Framework pour les tests end-to-end de l'application mobile
- **Lighthouse:** Outil pour évaluer les performances et l'accessibilité

## 3.10 Deploiement et integration continue

### 3.10.1 Pipeline CI/CD

Nous avons mis en place un pipeline d'integration continue et de déploiement continu (CI/CD) pour automatiser le processus de build, test et déploiement:

- **GitHub Actions:** Automatisation des workflows de CI/CD
- **ESLint et Prettier:** Vérification de la qualité du code
- **Tests automatisés:** Exécution des tests à chaque pull request
- **Preview Deployments:** Déploiement de versions de prévisualisation pour chaque branche

### 3.10.2 Environnements de déploiement

- **Développement:** Environnement local pour les développeurs
- **Staging:** Environnement de préproduction pour les tests
- **Production:** Environnement final accessible aux utilisateurs

### 3.10.3 Plateformes de déploiement

#### 3.10.3.1 Digital Ocean

Digital Ocean est une solution de service en cloud populaire, dotée d'une infrastructure robuste et offrant de multiples services.

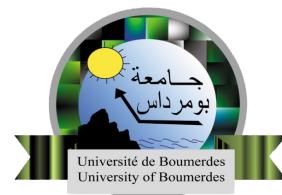
Il est principalement utilisé pour l'hébergement d'applications et de sites web et est préféré par les utilisateurs en raison de sa facilité d'utilisation. Les centres de données Digital Ocean offrent un haut niveau de sécurité pour les applications.

Les serveurs privés virtuels ou VPS offerts par Digital Ocean aux utilisateurs sont connus sous le nom de Droplets. Les utilisateurs de la plateforme peuvent gérer leurs applications par le biais d'une interface utilisateur Web ou d'une interface en ligne de commande (CLI).

La plateforme IaaS de Digital Ocean est un choix populaire pour de nombreuses grandes entreprises clientes dans le monde entier en raison de sa fiabilité. Elle permet aux utilisateurs de choisir des paramètres tels que les centres de données pour les applications, la taille des droplets et la région géographique.

Pour VoiceNotion, nous avons utilisé Digital Ocean pour:

- Héberger notre base de données PostgreSQL de sauvegarde
- Gérer des services auxiliaires comme Redis pour la mise en cache
- Exécuter des tâches de traitement en arrière-plan



### 3.10.3.2 Nginx

Nginx, prononce comme "engine-ex", est un serveur web open-source qui, depuis son succès initial en tant que serveur web, est maintenant aussi utilisé comme reverse proxy, cache HTTP, et load balancer.

Nginx a été créé à l'origine par Igor Sysoev, avec sa première sortie publique en octobre 2004. Igor a d'abord conçu le logiciel comme une réponse au problème du C10k, qui est un problème de performance lié à la gestion de 10.000 connexions simultanées.

Pour VoiceNotion, nous utilisons Nginx pour:

- Servir de reverse proxy pour notre application
- Gérer le load balancing entre les instances de l'application
- Configurer le SSL/TLS pour la sécurité des communications
- Optimiser la diffusion des assets statiques
- Mettre en place des règles de cache pour améliorer les performances



Nginx nous a permis d'améliorer considérablement les performances et la sécurité de notre application, tout en offrant une flexibilité importante pour la configuration de notre infrastructure.

## 3.11 Sécurité et protection des données

### 3.11.1 Authentification et autorisation

La sécurité de VoiceNotion repose sur plusieurs mécanismes:

- **Authentification multi-facteurs:** Pour une sécurité renforcée des comptes
- **JWT (JSON Web Tokens):** Pour la gestion des sessions
- **Contrôle d'accès basé sur les rôles:** Pour limiter les actions selon le profil utilisateur

### 3.11.2 Protection des données

- **Chiffrement des données:** En transit (HTTPS) et au repos
- **Anonymisation:** Des données utilisées pour l'analyse et l'amélioration du service
- **Sauvegarde régulière:** Pour prévenir la perte de données

### 3.11.3 Conformite RGPD

VoiceNotion est conçu dans le respect du Règlement Général sur la Protection des Données:

- **Consentement explicite:** Pour la collecte et l'utilisation des données
- **Droit à l'oubli:** Possibilité de supprimer toutes les données utilisateur
- **Portabilité des données:** Export des notes dans des formats standards

## 3.12 Défis techniques et solutions

### 3.12.1 Défis rencontres

Durant le développement de VoiceNotion, nous avons rencontré plusieurs défis techniques:

- **Precision de la reconnaissance vocale:** Particulièrement dans des environnements bruyants
- **Synchronisation en temps réel:** Entre les différents appareils d'un utilisateur
- **Performance de l'éditeur:** Avec des documents volumineux
- **Fonctionnement hors ligne:** Gestion des conflits lors de la reconnexion

### 3.12.2 Solutions implementees

- **Filtrage audio:** Algorithmes de réduction du bruit pour améliorer la reconnaissance vocale
- **Système de replication:** Basé sur CRDT (Conflict-free Replicated Data Types) pour la synchronisation
- **Virtualisation:** Rendu optimisé des longs documents pour maintenir la performance
- **Queue de synchronisation:** Pour gérer les modifications hors ligne et les synchroniser lors de la reconnexion

## 3.13 Conclusion

L'implémentation de VoiceNotion a nécessité l'intégration de nombreuses technologies modernes et la résolution de défis techniques complexes. L'architecture choisie nous a permis de créer une application performante, sécurisée et évolutive, disponible sur le web et sur mobile.

La combinaison de React, Next.js, React Native et Supabase nous a fourni une base solide pour construire notre solution, tandis que l'intégration de l'API Gemini a rendu possible la fonctionnalité centrale de reconnaissance vocale.

Les prochaines étapes de développement incluront l'amélioration continue de la précision de la reconnaissance vocale, l'ajout de nouvelles fonctionnalités collaboratives, et l'optimisation des performances sur tous les appareils.

## **INTRODUCTION**

# **CHAPTER 4**

## **ETUDE PREALABLE**

- 4.1 Introduction**
- 4.2 Problématiques**
- 4.3 Solution et objectif**
- 4.4 Avantages du projet**
- 4.5 Conclusion**

# **CHAPTER 5**

## **PLANIFICATION ET CONCEPTION UX**

### **5.1 Introduction**

### **5.2 Planification du projet**

#### 5.2.1 Méthodologie de planification

#### 5.2.2 Planification

### **5.3 Expérience d'utilisateur (UX)**

#### 5.3.1 Recherche

#### 5.3.2 Empathie

##### *5.3.2.1 Personas utilisateur*

##### *5.3.2.2 Scénarios d'utilisations*

### **5.4 Conception du système d'information**

#### 5.4.1 Identification des acteurs

#### 5.4.2 Diagramme de cas d'utilisation

#### 5.4.3 Diagrammes de séquence

#### 5.4.4 Diagrammes de base de données

##### *5.4.4.1 Diagramme entité-relation*

##### *5.4.4.2 Diagramme de base de données*

### **5.5 Conclusion**



# CHAPTER 6

## IMPLEMENTATION ET DEVELOPPEMENT

### 6.1 Introduction

### 6.2 Architecture

### 6.3 Environnement

6.3.1 Matériel (hardware)

6.3.2 Logiciel (software)

### 6.4 API (Back-end server)

6.4.1 Technologies et outils utilisés (sur Back-end)

6.4.1.1 *Node.js*

6.4.1.2 *Express.js*

6.4.1.3 *TypeScript*

6.4.1.4 *Supabase*

6.4.1.5 *Firebase*

6.4.1.6 *Docker*

6.4.1.7 *PostgreSQL*

6.4.2 Déploiement

6.4.2.1 *Digital Ocean*

6.4.2.2 *Nginx*

6.4.2.3 *CI/CD Pipeline*

### 6.5 Site web (Front-end)

6.5.1 Technologies et outils utilisés (Front-end)

6.5.1.1 *React.js*

6.5.1.2 *Next.js*

6.5.1.3 *TailwindCSS*

6.5.1.4 *BlockNote.js*

---

6.5.2 L'interface utilisateur du site 70

6.5.2.1 *Landing page*

6.5.2.2 *Connexion*



# CHAPTER 7

## VOICENOTION – IDENTITE VISUELLE

### 7.1 Introduction

### 7.2 Logo

7.2.1 Le choix du nom

7.2.2 Le choix du logo

7.2.3 La méthode de conception

### 7.3 La typographie

7.3.1 Polices latines

7.3.1.1 *Inter font*

7.3.1.2 *Outfit font*

7.3.2 Police Arabe

7.3.2.1 *Noto Sans Arabic*

### 7.4 Palette de couleurs

7.4.1 Bleu primaire

7.4.2 Gris neutre

7.4.3 Accent colors

### 7.5 Design System

7.5.1 Conception

7.5.2 Components

7.5.2.1 *Buttons*

7.5.2.2 *Input fields*

7.5.2.3 *Cards*

### 7.6 Mockups

7.6.1 Mobile app

7.6.1.1 *Light theme*

7.6.1.2 *Dark theme*

7.6.2 Web app

7.6.2.1 *Light theme*

## **CONCLUSION**

## **APPENDIX A**

## **REFERENCES**

## **APPENDIX B**

## **GLOSSARY**

## **APPENDIX C**

### **VOICE COMMAND REFERENCE**