



Département Génie Informatique

Réf : PFA2-2024-17

Rapport de Projet de Fin d'Année

de

Deuxième année en Génie Informatique

Présenté et soutenu publiquement le 10/05/2024

Par

Cherni Rihab
Elloumi Molka
Hammemi Wiem

Application mobile E-learning accessible aux prsonnes non-voyants

Composition du jury

Madame	Salsabil GHRAIRI	Président
Monsieur	Ramzi FARHAT	Encadrant

Année universitaire : 2023-2024

Dédicaces

Nous dédions ce projet à nos chers parents, nos frères, nos sœurs, à nos meilleurs amis et à tous nos collègues et professeurs.

Nous tenons à exprimer nos sincères remerciements à nos chers parents pour le courage et la patience illimitée qu'ils nous ont accordée tout au long de nos vie surtout dans les moments difficiles et les sacrifices nous ont permis d'accomplir une éducation digne

À nos sœurs et frères pour l'encouragement continue de leur part, car ils ont su nous soutenir.

À tous nos amis pour leur disponibilité et l'immense l'aide et le soutien formidables qu'ils nous ont apportés.

Remerciements

Au terme de notre projet de fin d'année, c'est avec un grand plaisir que nous dédions ces lignes pour remercier toute personne ayant contribué, directement ou indirectement, de près ou de loin au bon déroulement et à l'aboutissement de ce modeste projet.

Nous voudrions particulièrement remercier et exprimer notre gratitude à notre encadrant, Monsieur **Ramzi FARHAT**, pour avoir toujours cru en nous. Nous vous remercions également pour votre patience, votre disponibilité, vos conseils pertinents et pour l'attention qu'il nous a accordée tout au long de notre travail. Merci de nous encourager et surtout de nous orienter dans la bonne direction avec l'aide, les informations et les conseils fournis.

Nous tenons à remercier également les membres du jury pour le grand honneur qu'ils nous ont fait en acceptant d'évaluer et juger notre travail.

Nous voudrions également remercier tous nos enseignants de l'École nationale supérieure d'ingénieurs de Tunis qui ont contribué à notre formation et nous ont bénéficié par leurs connaissances, expériences et nous ont supporté tout au long de cette année.

Nos derniers remerciements, nous tenons à les exprimer à tous nos proches qui nous ont permis de réaliser ce travail.

Table des matières

Introduction générale	1
Chapitre 1: Cadre général du projet	2
Introduction	2
1 Cadre du projet	2
2 Contexte du projet	2
3 Etude et critique de l'existant	3
3.1 Solutions existantes	3
3.2 Tableau comparatif des solutions existantes	6
4 Solution proposée	6
Conclusion	7
Chapitre 2: Etude détaillée	8
Introduction	8
1 Spécification des besoins	8
1.1 Identification des acteurs	8
1.2 Identification des besoins fonctionnels	9
1.3 Identification des besoins non fonctionnels	10
2 Analyse fonctionnelle	11
2.1 Diagramme de cas d'utilisation global	11
2.2 Raffinement des cas d'utilisation	12
Conclusion	20
Chapitre 3: Conception	21
Introduction	21

1	Modèle architectural	21
2	Diagramme de classe	22
3	Diagrammes de séquences détaillés	23
3.1	Diagramme de séquence détaillé du cas d'utilisation de «S'inscrire» . . .	24
Conclusion		25
Chapitre 4: Technologies et environnement de développement		26
Introduction		26
1	Environnement matériel	26
2	Environnement logiciel	26
Conclusion		36
Chapitre 5: Réalisation et mise en œuvre		37
Introduction		37
1	Architecture de notre système	37
2	Quelques interfaces de "Nawarny"	39
2.1	Page d'accueil	39
2.2	Page visiteur	40
2.3	Interface pour choisir le rôle dans l'application	41
2.4	Interface d'inscription	41
2.5	Interface de connexion	42
2.6	Les interfaces de l'administrateur	42
2.7	Les interfaces de l'auteur	45
2.8	Les interfaces de l'apprenant	46
2.9	Les interfaces mettant en avant l'accessibilité	49
Conclusion		50
Conclusion générale		51
Bibliographie		54

Table des figures

1.1	Logo de l'application "Canvas Student" [2]	3
1.2	Quelques interfaces de l'application "Canvas Student"[2]	4
1.3	Quelques interfaces de l'application "Blackboard Learn"[4]	4
1.4	Logo de l'application "Moodle" [6]	5
1.5	Quelques interface de l'application "Moodle"[7]	5
2.1	Diagramme de cas d'utilisation global	11
2.2	Diagramme de séquence du cas d'utilisation de «S'inscrire»	15
2.3	Diagramme raffiné de cas d'utilisation "Gérer les cours"	16
2.4	Diagramme de séquence du cas d'utilisation de «Ajouter un cours»	18
2.5	Diagramme de séquence du cas d'utilisation de «Poser une question concernant un cours»	20
3.1	Diagramme de classe	23
3.2	Diagramme de séquence détaillé du cas d'utilisation «S'inscrire»	25
4.1	Flutter [10]	27
4.2	Visual Studio Code [11]	27
4.3	Draw.io [12]	27
4.4	GitHub [13]	27
4.5	Overleaf [14]	27
4.6	StarUML [15]	27
4.7	Mongo DB [16]	28
4.8	Django [17]	28
4.9	Jira [18]	28
4.10	Comparaison entre framework flutter et d'autres framework frontend existant selon google trends[20]	31

4.11 Comparaison entre framework django et d'autres framework backend existant selon google trends [20]	33
4.12 Comparaison entre mongodb et d'autres systèmes de gestion de bases de données existant selon google trends [20]	35
5.1 Illustration des widgets d'un simple exemple d'une interface graphique.[23] . . .	38
5.2 Architecture de Django.	39
5.3 Page de Loading	40
5.4 Page d'accueil	40
5.5 Page visiteur	40
5.6 Interface de sélection des rôles	41
5.7 Page d'inscription	41
5.8 Inscription confirmée	41
5.9 Erreur d'inscription	41
5.10 Interface login auteur et admin	42
5.11 Interface erreur de connexion	42
5.12 Interfaces tableau de bord pour l'administrateur	43
5.13 Interface gestion des apprenants	43
5.14 Interface gestion des auteurs	44
5.15 Interface gestion des cours	44
5.16 Interface des cours	45
5.17 Interface de profil	46
5.18 Interface visualisant le compte de l'apprenant	47
5.19 Interface de Consultation des cours	47
5.20 Interface de consultation des détails des cours.	48
5.21 Interface de catégories de cours	48
5.22 L'interface mettant en avant le lecteur d'écran pour la page de sélection des rôles	49
5.23 L'interface mettant en avant le lecteur d'écran pour la consultation du contenu des cours	50

Liste des tableaux

1.1	Tableau comparatif des solutions existantes[9]	6
2.1	Description textuelle du cas d'utilisation : S'inscrire	14
2.2	Description textuelle du cas d'utilisation "Ajouter un cours"	17
2.3	Description textuelle du cas d'utilisation «Poser un question concernant un cours»	19
4.1	Spécifications des PC	26
4.2	Comparaison entre framework flutter et d'autres framework frontend existant [19]	30
4.3	Comparaison entre django et les autres framework backend existant [21]	32
4.4	Comparaison entre mongodb et d'autres systèmes de gestion de bases de données existants [22]	35

Introduction générale

L'accès à l'éducation est un droit fondamental pour tous, quelle que soit leur situation ou leurs capacités. Cependant, pour les personnes non voyantes et malvoyantes, ce droit peut souvent être entravé par des obstacles liés à l'accessibilité des supports éducatifs traditionnels. Dans un monde de plus en plus numérique, il est impératif de créer des solutions innovantes pour garantir que personne ne soit laissé pour compte dans sa quête de connaissances.

L'éducation est le pilier de l'autonomie et du développement personnel. Pour les personnes non voyantes et malvoyantes, avoir accès à des outils éducatifs adaptés à leurs besoins spécifiques est essentiel pour réaliser leur plein potentiel. C'est dans cette optique que notre projet d'application mobile e-learning dédiée à cette communauté prend tout son sens.

En observant les lacunes des solutions éducatives existantes pour ce public, nous avons constaté un besoin criant de combler le fossé numérique en offrant une plateforme éducative inclusive et accessible à tous. Notre application vise à lever les barrières de l'apprentissage en proposant un environnement virtuel où ce public peut accéder à du contenu pédagogique diversifié, adapté à leurs besoins spécifiques et présenté de manière accessible.

À travers ce rapport de projet, nous explorerons en détail le processus de conception, de développement et de mise en œuvre de notre application mobile e-learning. Nous mettrons en lumière les défis uniques auxquels nous avons été confrontés dans la création d'une interface utilisateur intuitive et accessible, ainsi que les solutions novatrices que nous avons développées pour garantir une expérience d'apprentissage enrichissante pour tous.

Ce rapport est structuré en cinq chapitres afin de fournir une vue d'ensemble complète de notre projet, allant de l'analyse des besoins à la réalisation concrète de l'application. Les chapitres inclus sont le cadre général du projet, l'étude détaillée, la conception, les technologies et l'environnement de développement, et enfin la réalisation et mise en œuvre. Nous examinerons également les implications futures de notre travail et les possibilités d'amélioration continue pour assurer que notre application reste à la pointe de l'accessibilité éducative.

Cadre général du projet

Introduction

Dans ce chapitre, nous aborderons le cadre général de notre projet, contextualisant son importance dans le domaine de l'éducation inclusive pour les personnes non voyantes et malvoyantes. Nous examinerons l'état actuel des solutions disponibles et proposerons une approche novatrice pour combler les lacunes identifiées.

1. Cadre du projet

Ce travail s'inscrit dans le cadre d'un projet de fin d'année mené au sein de l'école nationale supérieure d'ingénieur de Tunis (ENSIT). Le projet vise à concevoir et développer une application mobile e-learning dédiée aux personnes non voyantes et malvoyantes, afin de répondre à leurs besoins spécifiques en matière d'éducation.

2. Contexte du projet

Le sujet du projet, intitulé "**Application mobile E-learning accessible aux prsonnes non-voyants**", met l'accent sur la création d'une plateforme éducative inclusive et accessible à tous, en particulier aux personnes atteintes de déficiences visuelles. L'objectif principal est de combler le fossé numérique en offrant un environnement d'apprentissage virtuel où ces individus peuvent accéder à du contenu pédagogique diversifié, adapté à leurs besoins spécifiques et présenté de manière accessible.

Dans le cadre de ce projet, nous nous engageons à relever les défis techniques et conceptuels liés à la conception et au développement d'une application mobile e-learning accessible à un public ayant des besoins particuliers. Nous aspirons à créer une interface utilisateur intuitive

et ergonomique, qui facilite la navigation et l'interaction pour les utilisateurs non voyants et malvoyants.

3. Etude et critique de l'existant

Dans cette section, nous procéderons à une analyse approfondie des applications web et mobile dédiées à l'e-learning pour les personnes non voyantes. L'objectif est de déterminer les forces et les faiblesses de chaque solution existante, en mettant en évidence les aspects clés qui impactent l'expérience utilisateur des apprenants non voyants et malvoyants.

3.1. Solutions existantes

Cette étape implique de décrire les caractéristiques spécifiques de chaque application, en mettant l'accent sur les fonctionnalités d'accessibilité déjà mises en œuvre.

Les applications d'e-learning telles que Canvas Student, Blackboard Learn et Moodle sont des plates-formes éducatives en ligne qui offrent une variété de cours et de ressources. Ces plateformes ont mis en place des fonctionnalités d'accessibilité visant à rendre leur contenu accessible.

1- Canvas Student : est un système de gestion de l'apprentissage (LMS) utilisé par de nombreuses institutions éducatives pour créer des cours en ligne et faciliter l'interaction entre les enseignants et les étudiants.[1]



Figure 1.1 – Logo de l'application "Canvas Student" [2]

Cette application Permet de personnaliser l'apparence et les fonctionnalités de la plateforme selon les besoins et les préférences des utilisateurs, propose des outils de communication, de collaboration, de feedback, et d'évaluation pour améliorer l'expérience d'apprentissage, et offre des fonctionnalités d'accessibilité.

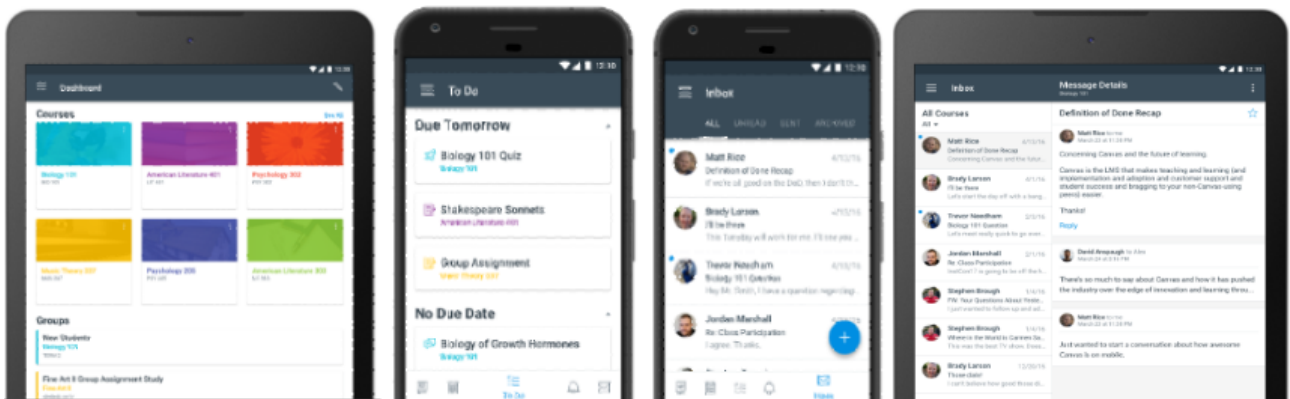


Figure 1.2 – Quelques interfaces de l'application "Canvas Student"[2]

Limites de "Canvas Student"[3] :

- Manque de compatibilité avec les lecteurs d'écran : Canvas Student ne dispose pas d'une compatibilité native avec les lecteurs d'écran, rendant difficile l'accès aux contenus visuels pour les personnes non voyantes.
- Déficit de contraste : Les couleurs et les contrastes des éléments graphiques peuvent ne pas être suffisamment distinguables pour les personnes ayant une vision limitée.

2- Blackboard Learn :[4] est un système de gestion de l'apprentissage utilisé dans les établissements éducatifs pour créer des cours en ligne, partager des documents et faciliter la communication entre les enseignants et les étudiants.

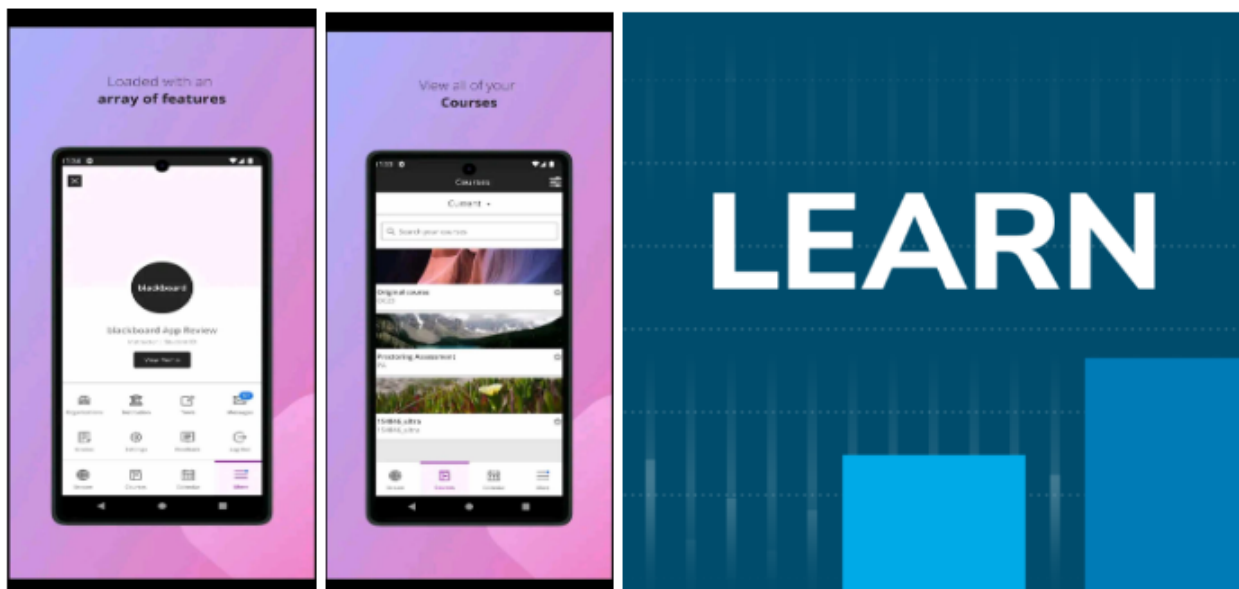


Figure 1.3 – Quelques interfaces de l'application "Blackboard Learn"[4]

Blackboard Learn offre des fonctionnalités d'accessibilité spécifiques pour répondre aux besoins des étudiants ayant des besoins spécifiques, y compris ceux qui ont des problèmes de vision.

Limites de "Blackboard Learn"[5] :

- Dépendance aux outils tiers : Blackboard Learn peut nécessiter l'utilisation d'outils tiers pour une accessibilité optimale, ce qui peut entraîner des problèmes de compatibilité et de convivialité pour les utilisateurs non voyants.
- Complexité de la navigation : L'interface de Blackboard Learn peut être complexe, avec des menus et des options multiples, ce qui peut rendre la navigation difficile pour les personnes non voyantes.

3- Moodle[6] : est une plateforme d'apprentissage en ligne open-source qui permet aux éducateurs de créer des cours en ligne et de fournir du contenu éducatif aux étudiants.



Figure 1.4 – Logo de l'application "Moodle" [6]

Moodle offre une plateforme d'apprentissage en ligne open-source qui permet aux éducateurs de créer des cours en ligne et de fournir du contenu éducatif aux étudiants, ainsi qu'elle propose des outils de communication, de collaboration, de feedback, et d'évaluation pour améliorer l'expérience d'apprentissage, et elle offre des options d'accessibilité et peut être configuré pour répondre aux besoins spécifiques des apprenants.



Figure 1.5 – Quelques interface de l'application "Moodle" [7]

Limites de "Moodle"[8] :

- Interface complexe : Bien que Moodle offre certaines fonctionnalités d'accessibilité, son interface peut être complexe et difficile à naviguer pour les utilisateurs non voyants, ce qui peut rendre l'accès aux contenus éducatifs difficile.
- Dépendance aux formats de contenu : Moodle dépend souvent du format de contenu téléchargé par l'utilisateur, ce qui peut rendre difficile la garantie d'une accessibilité optimale pour tous les types de contenu.

3.2. Tableau comparatif des solutions existantes

Distinguer les forces et les faiblesses des solutions existantes en termes d'accessibilité à travers un tableau comparatif 1.1, offrant ainsi un aperçu des différences selon la compatibilité avec les lecteurs d'écran, le contraste visuel, la convivialité et la dépendance aux formats de contenu.

Caractéristiques	Canvas Student	Blackboard Learn	Moodle
Compatibilité avec les lecteurs d'écran	Non (interfaces complexes)	Non (oui si utilisés des outils tiers)	Non
Contraste des éléments visuels	Non	Peut être configuré	Non
Navigation et convivialité	Non	Non	Non (oui si on considère l'interface conviviale, mais manque de fonctionnalités spécifiques)

Table 1.1 – Tableau comparatif des solutions existantes[9]

4. Solution proposée

Pour améliorer l'accessibilité des applications d'e-learning, des solutions spécifiques peuvent être envisagées. La solution proposée pour notre projet vise à intégrer des fonctionnalités essentielles pour faciliter leur expérience d'apprentissage.

Contrairement à des plateformes plus complexes, notre solution ne nécessitera pas de paramétrage compliqué pour activer l'accessibilité. Nous mettons l'accent sur une approche centrée sur l'accessibilité, en veillant à ce que tous les contenus et fonctionnalités soient conçus pour être utilisables par les non-voyants.

Cette approche garantira une utilisation fluide et agréable de l'application pour tous les utilisateurs, favorisant ainsi une expérience d'apprentissage équitable et inclusive.

Les outils d'accessibilité qui seront employés dans cette application sont :

- Lecteurs d'écran convertissent le texte affiché à l'écran en une forme audible, ce qui permet aux utilisateurs de naviguer et de consommer le contenu.
- Directives d'accessibilité fournissent des recommandations et des normes pour assurer que l'interface utilisateur et le contenu sont conçus de manière à être accessibles à tous.

Nous allons suivre les directives d'accessibilité telles que les WCAG (Web Content Accessibility Guidelines) pour nous assurer que notre interface utilisateur et notre contenu respectent les normes d'accessibilité établies.

Cela inclut la vérification, la fourniture de textes alternatifs pour les images, et la création de formulaires accessibles avec des étiquettes descriptives.

- Textes alternatifs pour les images : Nous devons nous assurer de fournir des descriptions textuelles alternatives pour toutes les images utilisées dans notre application.

Cela permet aux utilisateurs ayant des problèmes de vision de comprendre le contenu visuel de l'application.

- Contraste et couleur : Nous devons choisir des palettes de couleurs qui offrent un contraste suffisant entre le texte et l'arrière-plan pour améliorer la lisibilité, en particulier pour les utilisateurs malvoyants.
- Contenu audio et vidéo : Nous devons nous assurer que tout le contenu audio et vidéo de notre application est accessible aux utilisateurs malentendants en fournissant des sous-titres ou des transcriptions pour le contenu audio, et en ajoutant des descriptions alternatives pour le contenu vidéo lorsque cela est pertinent.
- Exigences sur l'auteur : L'auteur utilisant cette plateforme doit fournir le contenu du cours de manière adaptée aux besoins des apprenants handicapés. Cela signifie que le contenu doit être présenté sous forme de texte, de vidéo, d'audio ou d'images, et chaque image doit être accompagnée d'une description audio explicative ou un texte alternatif.

Conclusion

Ce chapitre a exposé les défis et les opportunités dans le domaine de l'éducation pour les personnes non voyantes et malvoyantes, en mettant en lumière les lacunes des solutions existantes. Notre approche innovante vise à combler ces lacunes en offrant une expérience d'apprentissage inclusive. Dans le prochain chapitre, nous approfondirons notre analyse en étudiant les besoins des utilisateurs et en élaborant un diagramme de cas d'utilisation pour guider le développement de notre solution.

Etude détaillée

Introduction

Dans ce chapitre, nous examinons la première phase de notre projet, qui consiste à spécifier et analyser les besoins de l'application en identifiant les acteurs ainsi que les exigences fonctionnelles et non fonctionnelles du système. Ensuite, nous progressons vers la phase de conception, où ces besoins sont mis en œuvre à travers la modélisation UML.

1. Spécification des besoins

La spécification des besoins est une étape clé dans le processus de développement d'un projet. C'est la première phase dans le cycle de vie de tout logiciel à développer. Elle consiste à formaliser les besoins identifiés lors de la phase d'analyse en spécifications techniques et fonctionnelles.

1.1. Identification des acteurs

Un acteur peut être défini comme étant une abstraction d'un rôle joué par une entité externe ou interne qui interagit directement avec le système étudié et qui est capable d'échanger des informations avec d'autres acteurs. Dans le cadre de notre projet, nous avons identifié quatre acteurs, à savoir :

- **Visiteur** : représente un internaute ordinaire désireux d'explorer et de consulter les informations disponibles, avec la possibilité de s'inscrire pour accéder à des fonctionnalités supplémentaires.
- **Apprenant** : est un utilisateur inscrit qui a accès à ses cours et peut les consulter à tout moment. Son rôle principal est de suivre les cours et d'interagir avec le contenu pédagogique proposé par l'application.
- **Auteur** : est un utilisateur inscrit qui bénéficie de privilèges supplémentaires, notamment la gestion des cours. Il peut ajouter, modifier ou supprimer des cours selon les besoins. Le

rôle de l'auteur est de contribuer au contenu de l'application en créant et en modifiant des cours.

- **Administrateur** : est un utilisateur disposant de tous les privilèges de l'administrateur, en plus de pouvoir gérer les auteurs, les apprenants et les cours. L'administrateur a un rôle de supervision et de gestion des utilisateurs et des ressources de l'application.

1.2. Identification des besoins fonctionnels

Les besoins fonctionnels décrivent les exigences de base d'un système. Cette application se divise en deux unités distinctes : le back-office pour les administrateurs et le front-office pour les auteurs, les apprenants et les visiteurs, chacune répondant à des besoins fonctionnels spécifiques.

- **La partie back-office** : elle contient les diverses fonctionnalités et opérations de gestion et d'administration internes de l'application, elle définit toutes les parties du système d'information auxquelles l'utilisateur final n'a pas accès.

Dans notre application, l'acteur qui joue le rôle d'un agent de back-office est l'administrateur.

- **Les fonctionnalités de l'administrateur** :

- Authentification : il doit s'authentifier avant d'avoir l'accès à son interface pour garantir la sécurité et la confidentialité des données, assurant ainsi un accès restreint aux utilisateurs autorisés.
- Gestion de profil : il peut consulter et modifier ses informations personnelles, offrant ainsi une personnalisation et une gestion individualisée des données.
- Consultation des statistiques : il peut accéder aux statistiques détaillées via un tableau de bord intuitif, permettant une analyse approfondie des performances et des progrès.
- Gestion des comptes utilisateurs : il peut gérer les informations relatives aux étudiants et enseignants, permettant une gestion efficace des utilisateurs inscrits.

- **La partie front-office** : elle contient les diverses fonctionnalités visibles par les utilisateurs.

Dans notre application, nous avons trois acteurs qui jouent le rôle d'un agent de front-office.

- **Les fonctionnalités du visiteur** :

- Inscription : Créer un compte sur la plateforme pour accéder aux fonctionnalités réservées aux utilisateurs enregistrés.
- Consultation des dernières actualités et la liste des cours disponibles sur l'application.

- **Les fonctionnalités de l'auteur** :

- Authentification : Avant d'accéder à son espace personnel, l'enseignant doit s'authentifier en utilisant ses identifiants.

- Gestion de profil : Une fois connecté, l'enseignant peut gérer son compte en consultant et en mettant à jour ses informations personnelles
 - Gestion de cours : L'auteur bénéficie de la possibilité d'ajouter, de modifier et de supprimer des cours en conformité avec les exigences et les normes d'accessibilité, assurant ainsi une actualisation constante et une pertinence du contenu proposé aux utilisateurs, tout en offrant une flexibilité complète dans la gestion du contenu pédagogique.
 - Répondre aux questions des étudiants : Permet aux enseignants de répondre aux questions des étudiants de manière rapide et efficace, favorisant ainsi une communication ouverte et une assistance personnalisée.
- **Les fonctionnalités de l'apprenant** : un étudiant est un utilisateur qui a la main de :
- Authentification : Se connecter à son compte pour accéder à son espace personnel.
 - Gestion de profil : Modifier ses informations personnelles selon ses besoins.
 - Consultation des cours : Accéder à la liste des cours auxquels l'étudiant est inscrit et consulter le contenu pédagogique associé.
 - Poser des questions après chaque cours : Avoir la possibilité de poser des questions ou de demander des éclaircissements sur le contenu du cours après l'avoir suivi, favorisant ainsi l'interaction et la compréhension.

1.3. Identification des besoins non fonctionnels

Les exigences techniques et logicielles non fonctionnelles sont des besoins d'excellence du système en termes de performances et de meilleures pratiques pour rendre le travail efficace pour les utilisateurs. En plus des besoins fonctionnels, il est important que notre système réponde aux critères de qualité suivants :

- * **La sécurité** : L'application doit être sécurisée, et les informations ainsi que les ressources ne doivent être accessibles qu'aux utilisateurs authentifiés.
- * **Maintenabilité et scalabilité** : Le code de l'application doit être lisible et compréhensible afin d'assurer leur état évolutif et extensible par rapport aux besoins du marché.
- * **La performance** : Les traitements doivent être optimisés pour avoir un court temps de réponse quelque soit l'action de l'utilisateur.
- * **Adaptabilité de l'interface** : Concevoir une interface qui s'ajuste de manière optimale à différentes tailles et résolutions d'écrans.

2. Analyse fonctionnelle

L'analyse fonctionnelle commence par une vue d'ensemble des interactions entre les utilisateurs et le système à travers le diagramme de cas d'utilisation global. Ensuite, elle se concentre sur le détail et l'extension de chaque fonctionnalité identifiée, offrant ainsi un éclairage approfondi sur les exigences spécifiques à travers le processus de Raffinement des cas d'utilisation.

2.1. Diagramme de cas d'utilisation global

En utilisant le langage UML, le diagramme de cas d'utilisation global est un outil pour modéliser le comportement d'un système, définir les interactions avec les acteurs, et spécifier les exigences fonctionnelles du projet. Dans le diagramme de cas d'utilisation global représenté par la figure 2.3, nous modélisons l'ensemble des cas d'utilisations de base afin d'avoir une vue globale de fonctionnement de l'application.

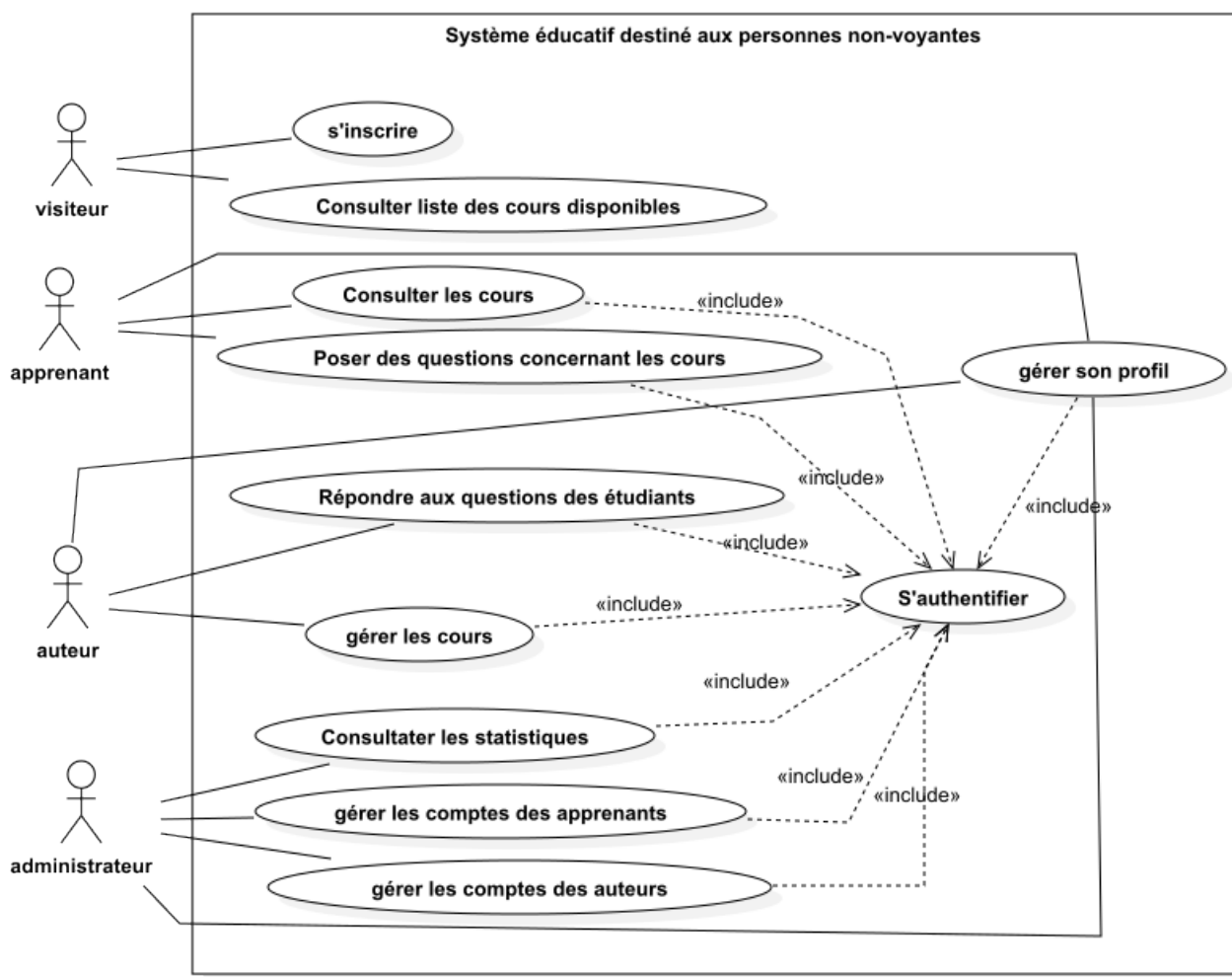


Figure 2.1 – Diagramme de cas d'utilisation global

2.2. Raffinement des cas d'utilisation

Le processus de raffinement des cas d'utilisation vise à clarifier chaque interaction entre les utilisateurs et le système, en simplifiant les descriptions tout en préservant leur exhaustivité. Cela permettra aux parties prenantes de comprendre facilement les étapes impliquées dans chaque processus et de résoudre les éventuels points de confusion dès le départ. En fin de compte, ces cas d'utilisation raffinés serviront de référence essentielle pour le développement, le test et la validation du système.

Dans cette section, nous détaillons certains cas d'utilisation en utilisant des diagrammes de cas d'utilisation raffinés, ainsi que des descriptions textuelles et des diagrammes de séquences système du scénario nominal de chaque cas d'utilisation sélectionné, lorsque cela est nécessaire.

2.2.1. Raffinement du cas d'utilisation «S'inscrire» :

Cette section illustre le flux principal du processus d'inscription, y compris les étapes telles que l'accès à la page d'inscription, la saisie des informations personnelles, la validation des informations, la génération d'identifiants de connexion, l'envoi d'un email de confirmation, et la vérification de l'adresse email via un code OTP (One-Time Password). De plus, il met également en évidence les scénarios d'erreur potentiels, tels que l'annulation de l'opération d'inscription, les informations incomplètes et les erreurs d'enregistrement. Ces scénarios d'erreur sont importants pour assurer une expérience utilisateur fluide et intuitive, même en cas de situations imprévues.

○ Description textuelle du cas d'utilisation «S'inscrire» :

Le tableau 2.1 présente la description textuelle du cas d'utilisation «S'inscrire».

Titre	S'inscrire
Acteur	Auteur
Résumé	Cas d'utilisation permettant à un auteur de s'inscrire à l'application.
Préconditions	L'auteur doit avoir accès à un dispositif connecté à Internet.

Scénario nominal	<ol style="list-style-type: none">1. L’auteur accède à la page d’inscription.2. L’auteur sélectionne l’option "S’inscrire".3. Le système affiche un formulaire de saisie des informations d’inscription.4. L’auteur remplit le formulaire avec ses informations personnelles.5. L’auteur soumet le formulaire.6. Le système valide les informations fournies.7. Le système enregistre les informations de l’auteur dans la base de données.8. Le système génère automatiquement des identifiants de connexion pour l’auteur.9. Le système envoie un email de confirmation qui contient un code OTP (One-Time Password) de 4 chiffres à l’adresse fournie par l’auteur.10. L’utilisateur doit saisir ce code dans un champ dédié sur la page d’inscription pour vérifier son adresse email et finaliser le processus d’inscription.11. Si le code est incorrect ou expiré, le système affiche un message d’erreur et invite l’utilisateur à saisir à nouveau le code.12. Une fois que l’utilisateur a saisi avec succès le code OTP et que son email est vérifié le cas d’utilisation se termine avec succès.
-------------------------	---

Scénario d'erreur	<ul style="list-style-type: none"> • Dans l'étape 4 : <ul style="list-style-type: none"> ✗ Informations Incomplètes : <p>Si des informations obligatoires ne sont pas fournies, le système affiche un message d'erreur indiquant les champs à compléter. L'utilisateur est invité à fournir les informations manquantes avant de pouvoir soumettre le formulaire.</p> ✗ Annulation : <p>À tout moment avant la soumission du formulaire, l'utilisateur peut choisir d'annuler l'opération d'inscription. Dans ce cas, le cas d'utilisation se termine et aucune modification n'est apportée à la base de données.</p> • Dans l'étapes 7 : <ul style="list-style-type: none"> ✗ Erreur d'Enregistrement : <ul style="list-style-type: none"> - Si une erreur survient lors de l'enregistrement des informations de l'utilisateur dans la base de données, le système affiche un message d'erreur et invite l'utilisateur à réessayer ultérieurement. • Dans l'étapes 9 : <ul style="list-style-type: none"> ✗ Expiration du Code OTP : <ul style="list-style-type: none"> - Le code OTP envoyé par email a une durée de validité limitée, généralement quelques minutes. - Si l'utilisateur ne saisit pas le code OTP avant son expiration, le système affiche un message indiquant que le code a expiré et invite l'utilisateur à demander un nouveau code en cliquant sur un lien fourni dans l'email de confirmation.
-------------------	--

Table 2.1 – Description textuelle du cas d'utilisation : S'inscrire

○ Diagramme de séquence du cas d'utilisation «S'inscrire» :

Les étapes de déroulement du cas d'utilisation «S'inscrire» sont décrites par le diagramme

de séquence illustré par la figure 2.5.

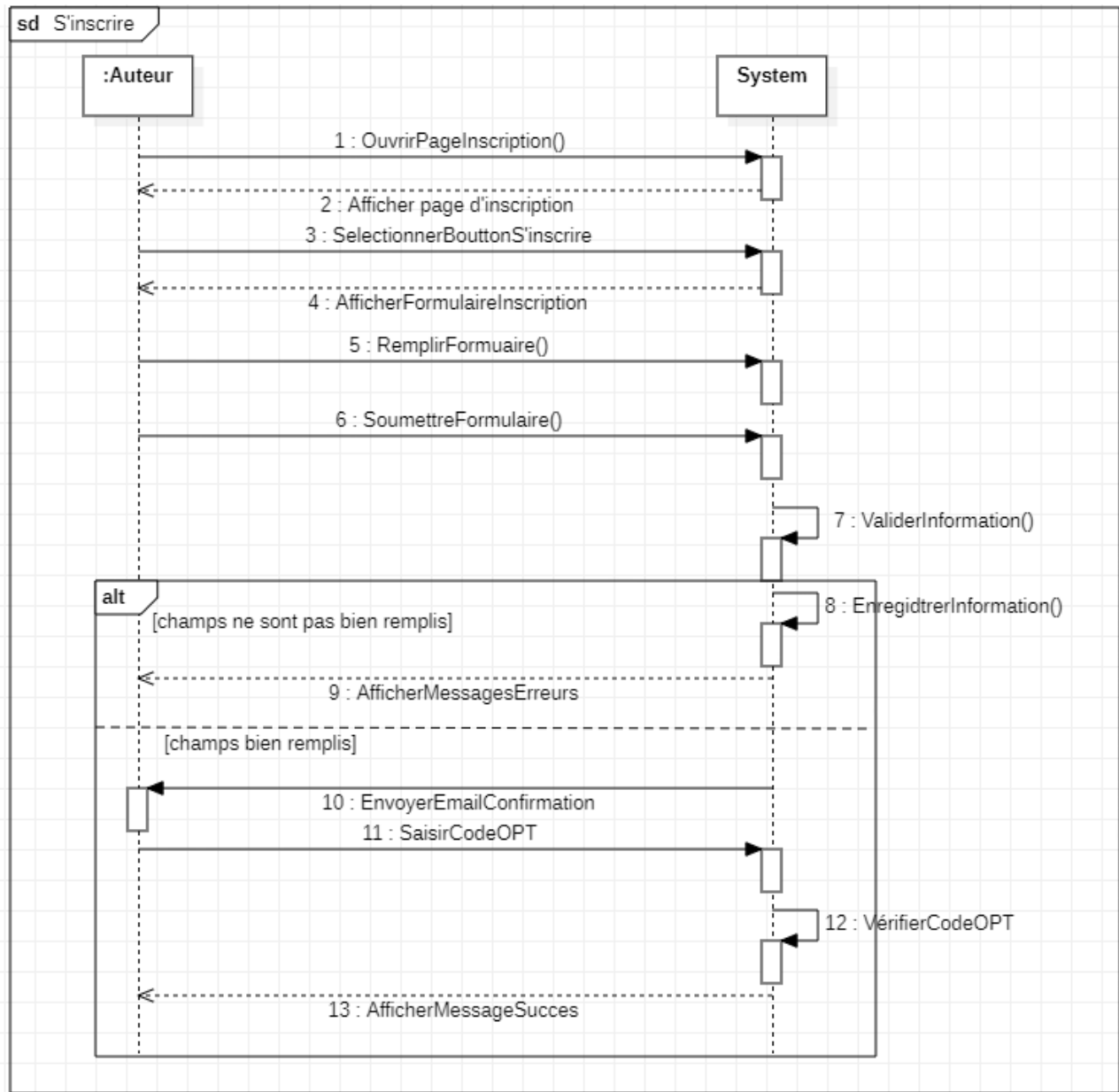


Figure 2.2 – Diagramme de séquence du cas d'utilisation de «S'inscrire»

2.2.2. Raffinement du cas d'utilisation «Gérer les cours»

Cette section détaille le processus principal de gestion des cours, depuis l'accès à la page de gestion jusqu'à l'enregistrement des modifications. Les étapes clés incluent la navigation vers la page de gestion des cours, la visualisation des cours existants, l'ajout de nouveaux cours, la modification des détails des cours et la suppression des cours obsolètes. De plus, des scénarios d'erreur sont abordés, tels que l'annulation de l'opération de gestion, les erreurs de saisie de données et les problèmes de sauvegarde des modifications. Ces scénarios d'erreur sont cruciaux pour garantir une expérience utilisateur fluide et éviter les pertes de données imprévues.

○ Diagramme raffiné du cas d'utilisation «Gérer les cours» :

Le raffinement du cas d'utilisation "Gérer les cours", qui détaille les différentes actions et interactions impliquées dans la gestion des cours, est illustré dans la figure 2.3.

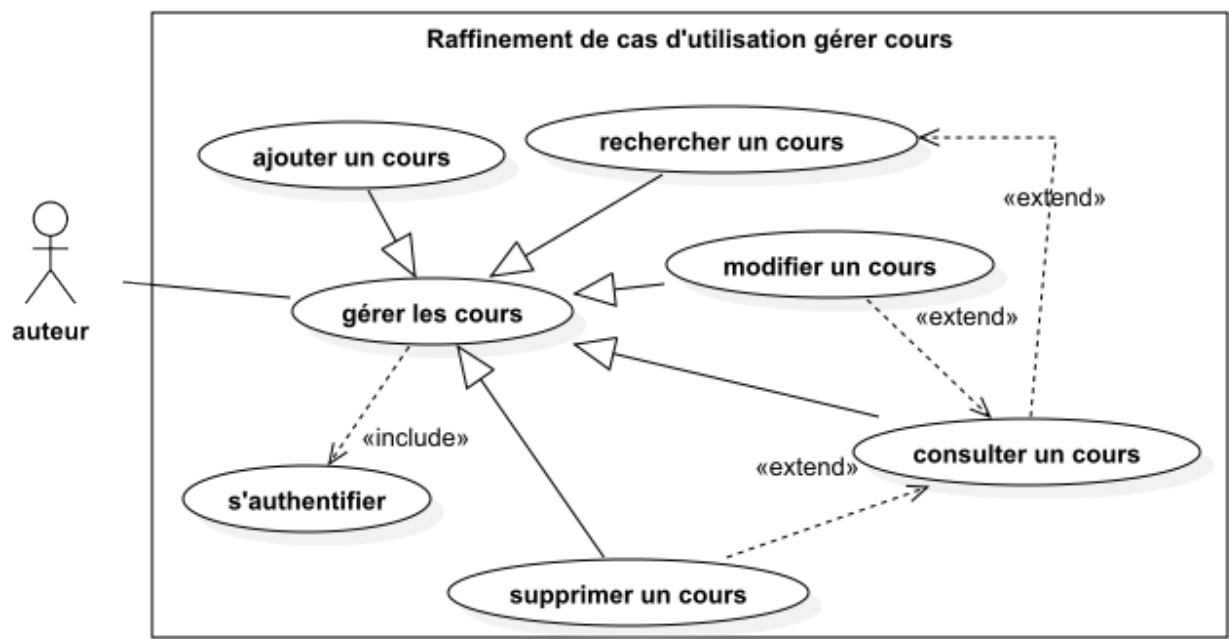


Figure 2.3 – Diagramme raffiné de cas d'utilisation "Gérer les cours"

○ Description textuelle du cas d'utilisation « Ajouter un cours » :

Le tableau 2.2 présente la description textuelle du cas d'utilisation « Ajouter un cours ».

Titre	Ajouter un cours
Acteur	Auteur
Résumé	C'est le cas d'utilisation qui permet à l'auteur de créer et ajouter un cours avec des différentes formes de contenu (image, vidéo, audio, texte) dans une catégorie spécifique pour être consulté ensuite par les apprenants.
Préconditions	- L'auteur est authentifié et a les permissions nécessaires pour ajouter un cours.
Scénario nominal	<ol style="list-style-type: none"> 1. L'auteur accède à la fonctionnalité d'ajout de cours. 2. Le système affiche un formulaire pour saisir les informations du cours telles que : (titre, description, la catégorie...) 3. L'auteur remplit les champs requis en fournissant une description détaillée pour chaque fichier téléchargé correspondant au contenu du cours (images, vidéos, audios, textes) afin de rendre le cours accessible aux personnes non voyantes. 4. L'auteur confirme l'ajout du cours. 5. Le système enregistre le cours dans la base de données. 6. Le système notifie l'auteur que le cours a été ajouté avec succès.
Scénario alternative	Dans l'étape 4 : <ul style="list-style-type: none"> - L'auteur peut annuler l'opération d'ajout de cours.
Scénario d'erreur	Dans les étapes 3 et 4 : <ul style="list-style-type: none"> - Le système affiche un message d'erreur en cas de saisie des données invalides. - Le système affiche un message d'erreur si le champ est vide.

Table 2.2 – Description textuelle du cas d'utilisation "Ajouter un cours"

○ Diagramme de séquence du cas d'utilisation «Ajouter un cours» :

Les étapes de déroulement du cas d'utilisation «Ajouter un cours» sont décrites par le diagramme de séquence illustré par la figure 2.4.

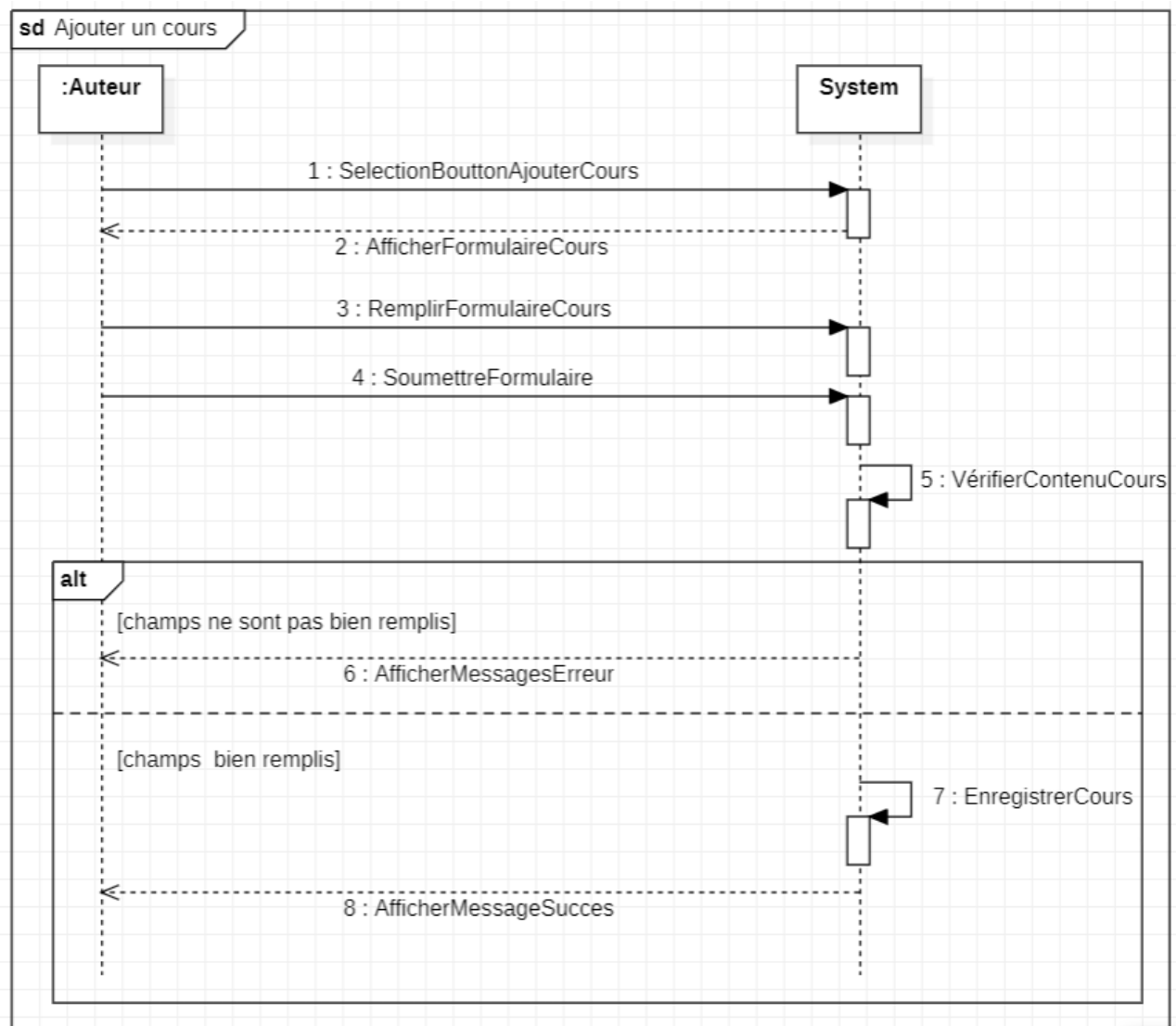


Figure 2.4 – Diagramme de séquence du cas d'utilisation de «Ajouter un cours»

2.2.3. Raffinement du cas d'utilisation «Poser un question concernant un cours»

Le cas d'utilisation "Poser une question concernant un cours" permet à un apprenant authentifié et inscrit de poser des questions spécifiques sur un cours. Après avoir saisi sa question dans un formulaire dédié, le système enregistre la question et notifie l'auteur associé au cours.

○ Description textuelle du cas d'utilisation « Poser un question concernant un cours » :

Le tableau 2.3 présente la description textuelle du cas d'utilisation «Poser un question concernant un cours ».

Titre	Poser une question concernant un cours
Acteur	Apprenant
Résumé	Cas d'utilisation permettant à un apprenant de poser une question concernant un cours.
Préconditions	1. L'apprenant est authentifié. 2. L'apprenant doit être inscrit au cours pour poser une question.
Scénario nominal	<ol style="list-style-type: none"> 1. L'apprenant se connecte au système. 2. L'apprenant navigue jusqu'à la section du cours pour lequel il souhaite poser une question. 3. L'apprenant sélectionne l'option "Poser une Question" ou un bouton similaire. 4. Le système affiche un formulaire de saisie de question. 5. L'apprenant saisit sa question dans le formulaire. 6. L'apprenant soumet la question. 7. Le système enregistre la question dans la base de données. 8. Le système notifie l'auteur associé au cours de la nouvelle question posée par l'étudiant. 9. Le cas d'utilisation se termine.
Scénario alternative	<p>Dans l'étape 6 :</p> <p>✓ Question Vide : Si l'étudiant soumet une question vide, le système affiche un message d'erreur et invite l'étudiant à saisir une question valide.</p>
Scénario d'erreur	<p>Dans l'étapes 7 :</p> <p>✗ Échec d'Enregistrement : Si une erreur survient lors de l'enregistrement de la question dans la base de données, le système affiche un message d'erreur et le cas d'utilisation se termine.</p>

Table 2.3 – Description textuelle du cas d'utilisation «Poser un question concernant un cours»

○ **Diagramme de séquence du cas d'utilisation «Poser une question concernant un cours» :**

Les étapes de déroulement du cas d'utilisation «Poser une question concernant un cours» sont décrites par le diagramme de séquence illustré par la figure 2.5.

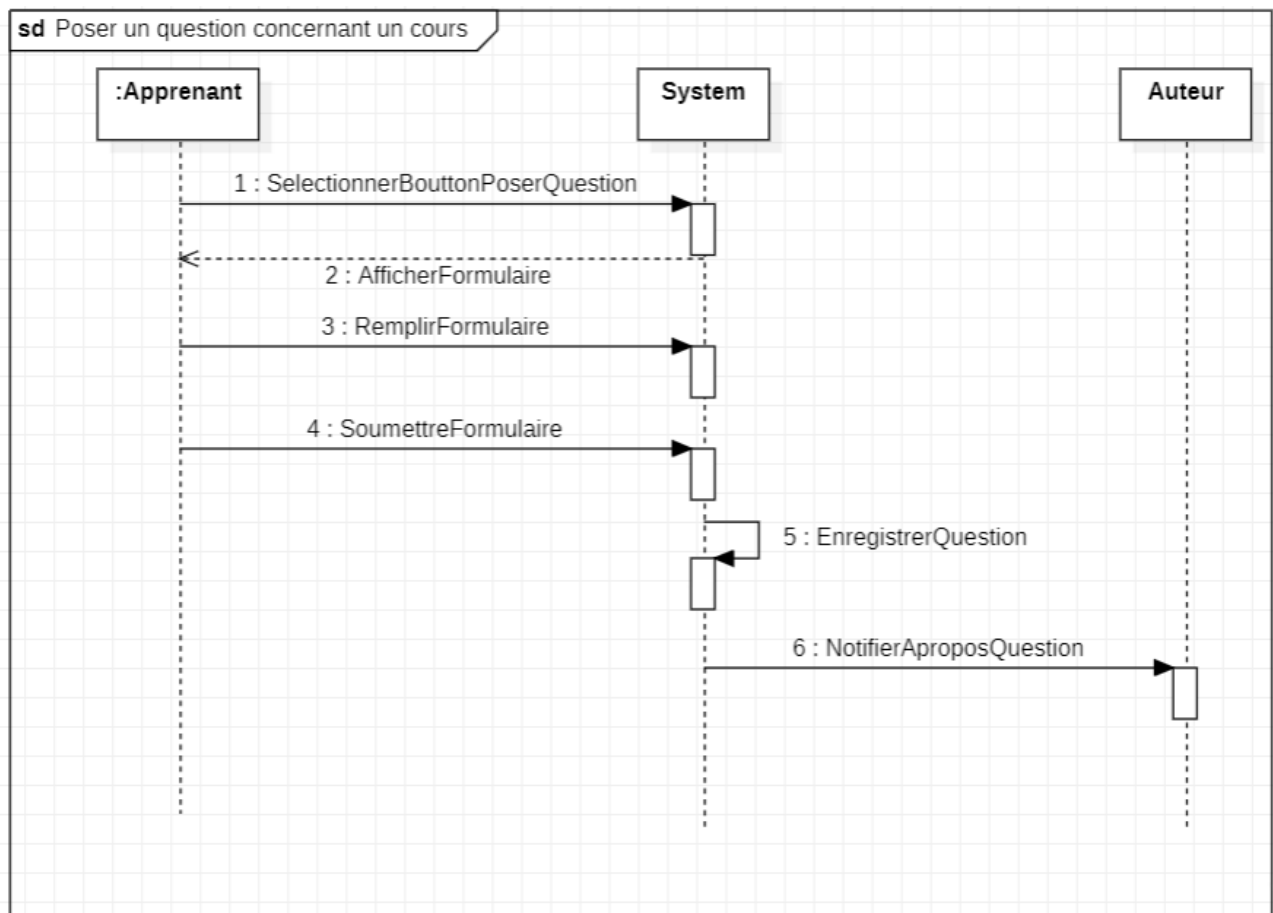


Figure 2.5 – Diagramme de séquence du cas d'utilisation de «Poser une question concernant un cours»

Conclusion

Ce chapitre focalise sur l'analyse détaillée des besoins fonctionnels et non fonctionnels de notre système. Nous avons présenté le diagramme de cas d'utilisation global et le raffinement de certains cas d'utilisation identifiés. Le prochain chapitre se concentrera sur la conception détaillée de l'application.

Conception

Introduction

Dans ce chapitre, nous explorons la phase de conception, une étape cruciale dans le développement de notre projet. Cette phase est essentielle pour définir les bases solides du système à construire. Nous commencerons par examiner le modèle architectural choisi, qui guidera la structure globale de l'application. Ensuite, nous détaillerons le diagramme de classe, qui représente les entités de données et leurs relations, ainsi que les diagrammes de séquences détaillés, qui illustrent le déroulement des interactions entre les différentes parties du système. Cette approche méthodique garantira une mise en œuvre cohérente et efficace.

1. Modèle architectural

Dans le contexte de logiciels de plus en plus complexes, il est crucial de disposer d'une architecture qui structure le code et répartit les responsabilités entre différentes couches. Les modèles architecturaux aident à isoler la logique métier de l'interface utilisateur, ce qui facilite la réutilisation du code, l'entretien et l'adaptation aux évolutions.

Dans le contexte de la conception de services web, l'architecture REST (Representational State Transfer) est un style architectural qui fournit un cadre pour organiser et structurer les systèmes en se basant sur les principes du protocole HTTP. REST vise à établir un ensemble de règles pour la communication entre un client et un serveur, et à organiser les ressources pour permettre des interactions claires et flexibles.

Les principes fondamentaux de l'architecture REST sont les suivants :

- **Ressources identifiées par des URL** : Les ressources (entités telles que les utilisa-

teurs, les articles, etc.) sont identifiées par des URLs spécifiques, ce qui permet d'accéder facilement aux données via des points d'accès bien définis.

- **Utilisation des méthodes HTTP** : Les méthodes HTTP (GET, POST, PUT, DELETE) sont utilisées pour manipuler les ressources. Par exemple, GET est utilisé pour lire les données, POST pour créer une nouvelle ressource, PUT pour mettre à jour une ressource existante, et DELETE pour supprimer une ressource.
- **Représentation des ressources** : Les ressources sont représentées dans un format standard, généralement JSON ou XML. Cela permet de garantir que les données transmises entre le client et le serveur sont facilement compréhensibles.
- **Statuts HTTP** : Les réponses des requêtes HTTP incluent des codes de statut pour indiquer le résultat de l'opération (par exemple, 200 pour succès, 404 pour non trouvé, 500 pour erreur serveur).
- **Sans état** : Chaque requête HTTP est indépendante, ce qui signifie que le serveur ne stocke pas d'état entre les requêtes successives. Cela facilite la scalabilité et la flexibilité.
- **Liens hypertextes** : Les réponses peuvent inclure des liens vers d'autres ressources, permettant de naviguer entre les ressources de manière cohérente.

REST favorise la création de services web flexibles et faciles à utiliser, permettant aux clients d'interagir avec les serveurs de manière claire et standardisée. En combinant REST avec un framework web tel que Django, les développeurs peuvent créer des API robustes et bien structurées, favorisant ainsi la maintenabilité et l'évolutivité des systèmes complexes.

2. Diagramme de classe

Le diagramme de classe est un élément central dans la modélisation orientée objet et le génie logiciel. Il offre une représentation abstraite des objets qui interagissent pour réaliser les cas d'utilisation d'un système. Le diagramme de classe est un outil essentiel pour organiser et comprendre la structure d'un logiciel, ainsi que pour faciliter la communication et la collaboration entre les membres de l'équipe de développement.

Le diagramme de classe de notre application e-learning dans la figure 3.1 présente plusieurs classes interconnectées. Nous allons décrire un segment de ce diagramme. Parmi les classes existantes, on trouve la classe "Personne", qui est une classe abstraite. Cette classe est la classe mère pour trois autres classes : "Auteur", "Administrateur" et "Apprenant". Selon l'association entre la classe "Apprenant" et la classe "Cours", nous pouvons conclure qu'un auteur peut

posséder zéro ou plusieurs cours sur la plateforme, tandis qu'un cours est détenu par un seul et unique auteur. De plus, un cours est associé à une seule catégorie, mais une catégorie peut contenir plusieurs cours. Au sein d'un cours, il peut y avoir plusieurs leçons, mais chaque leçon appartient à un seul cours.

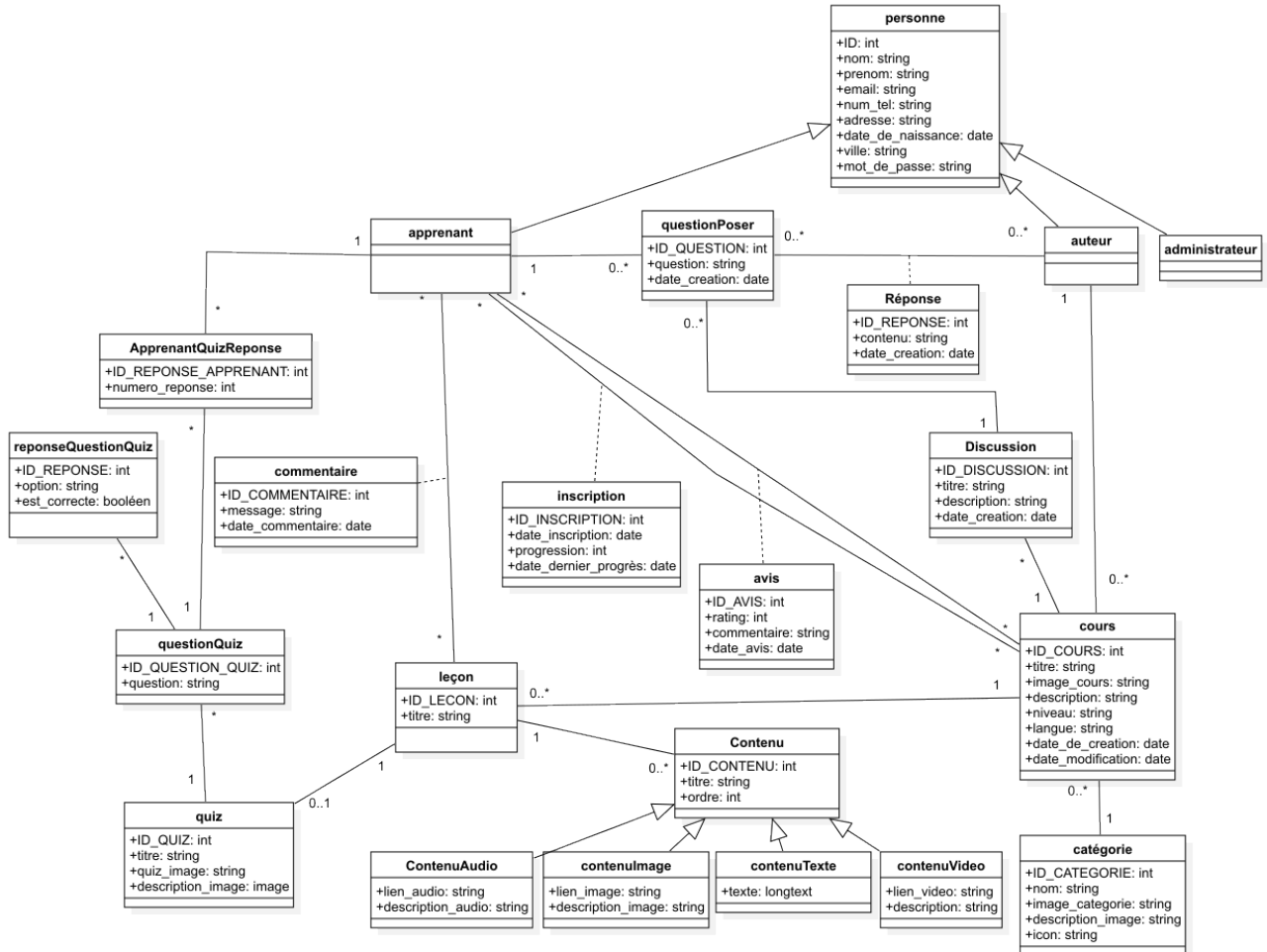


Figure 3.1 – Diagramme de classe

3. Diagrammes de séquences détaillés

Le diagramme de séquence sert à illustrer les interactions dynamiques dans un système. Il met en avant les collaborations entre les objets, en se focalisant sur l'ordre chronologique des échanges de messages. Il existe trois types de messages dans un diagramme de séquence :

- **Message asynchrone** : Représenté par une flèche continue avec une extrémité remplie, ce message indique une communication asynchrone entre les objets.
- **Message synchrone** : Représenté par une flèche continue avec une extrémité vide, ce message montre une communication synchrone entre les objets.
- **Message de réponse** : Représenté par une flèche en pointillés avec une extrémité vide, ce message montre la réponse d'un objet à un message synchrone antérieur. Cela signifie

que le destinataire a reçu le message et renvoie une réponse à l'expéditeur.

3.1. Diagramme de séquence détaillé du cas d'utilisation de «S'inscrire»

Nous présentons par la figure 3.2 un exemple de diagramme de séquences détaillé du cas d'utilisation «S'inscrire».

Ce diagramme illustre le processus d'inscription d'une personne, en mettant en évidence les interactions entre les différentes parties du système. Il débute par l'utilisateur sélectionnant le bouton d'inscription sur l'interface front-end, ce qui déclenche une série d'actions du back-end et de la base de données. Le processus inclut la sélection du rôle, le remplissage et la soumission du formulaire d'inscription, ainsi que la validation des données saisies. Des messages d'erreur ou de succès sont envoyés en fonction de la validité des informations fournies par l'utilisateur. Le processus peut également impliquer la vérification d'un code One-Time Password (OTP). Globalement, le diagramme offre une vue détaillée des étapes et des interactions nécessaires pour mener à bien le processus d'inscription.

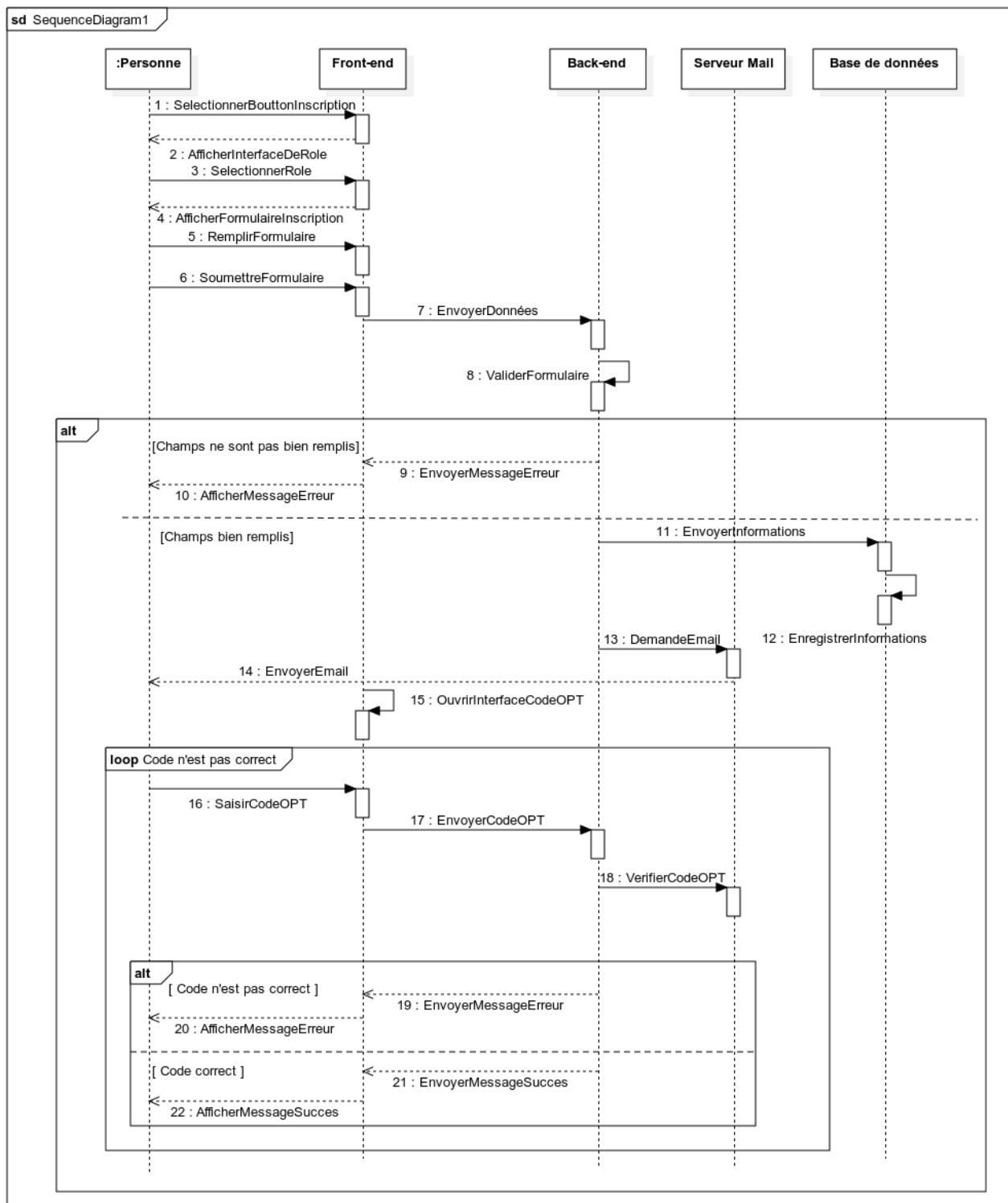


Figure 3.2 – Diagramme de séquence détaillé du cas d'utilisation «S’inscrire»

Conclusion

En conclusion, ce chapitre a établi les fondations essentielles pour la phase de réalisation et de mise en œuvre du projet. Après une étude approfondie et une documentation du modèle architectural, du diagramme de classe et des diagrammes de séquences, nous sommes désormais prêts à passer à la construction concrète du système. Le prochain chapitre abordera les détails de la réalisation, mettant en pratique les concepts abordés pour concrétiser notre projet.

Technologies et environnement de développement

Introduction

Ce chapitre est dédié à la présentation de l’environnement matériel et logiciel utilisé pour le développement de notre application.

1. Environnement matériel

L’environnement de travail informatique fait référence à l’ensemble des éléments matériels, logiciels, réseaux et paramètres qui constituent le cadre dans lequel les professionnels de l’informatique effectuent leur travail. Les PC utilisés ont les caractéristiques suivantes :

Modèle	ASUS	HP	ASUS
Type	PC portable	PC portable	PC portable
Processeur	Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz (1.80 GHz)	Intel Core i7 (11e génération)	Intel(R) Core(TM) i5-1135G7 CPU @ 2.40GHz 2.42 GHz
RAM	12 Go	8 Go	24 Go
Disque dur	SSD 256 Go	SSD 256 Go	SSD 512 Go
Système d’exploitation	Windows 10 Professionnel (64 bits)	Windows 11	Windows 11 Professionnel (64 bits)

Table 4.1 – Spécifications des PC

2. Environnement logiciel

L’environnement de développement logiciel est un terme qui désigne l’ensemble d’outils et de langage utilisé pour l’implémentation d’une solution informatique. Nous allons nous intéresser

à l'environnement logiciel. Les logiciels utilisés pour l'implémentation de notre solution sont les suivants :

Flutter : Flutter est un kit de développement logiciel (SDK) d'interface utilisateur open-source créé par Google. Il est utilisé pour développer des applications pour Android, iOS, Linux, Mac, Windows, Google Fuchsia et le web à partir d'une seule base de code.



Figure 4.1 – Flutter [10]



Figure 4.2 – Visual Studio Code [11]

Visual Studio Code [11] est un éditeur de code open-source et multi-langages léger, publié sous une licence de produit Microsoft. Il est compatible avec Windows, Linux et Mac. Il prend en charge les langages JavaScript, Java, Node.js, PHP, Python, C++ et .Net. Il fournit aux développeurs un outil de débogage, une coloration syntaxique intelligente et l'option d'auto-complétion.

Draw.io : est une application gratuite en ligne, accessible via son navigateur (protocole https) qui permet de dessiner des diagrammes ou des organigrammes. Cet outil vous propose de concevoir toutes sortes de diagrammes, de dessins vectoriels, de les enregistrer au format XML puis de les exporter.



Figure 4.3 – Draw.io [12]



Figure 4.4 – GitHub [13]

github [13] est un service web d'hébergement et de gestion de développement de logiciels open source publié en 2008, qui utilise le logiciel de gestion Git. Il présente plusieurs fonctionnalités : Il fournit un service cloud aux développeurs pour stocker, partager du contenu et gérer leur projet puisque il lui permet le suivi et le contrôle des modifications. Ainsi, il permet la récupération d'un code source sur le site web, le transférer sur une machine locale.

Overleaf est un éditeur en ligne permettant de faire l'édition du texte en LATEX de manière collaboratif en temps réel. Il permet la création des fichiers LATEX en se référant à des templates ainsi que l'importation des fichiers LATEX compressés. Latex encourage les auteurs à ne pas trop se soucier de l'apparence de leurs documents, mais à se concentrer sur l'obtention du bon contenu. Nous avons utilisé Latex pour la réalisation du présent rapport.



Figure 4.5 – Overleaf [14]



Figure 4.6 – StarUML [15]

StarUML [15] est un logiciel gratuit de modélisation Agile facile à utiliser en plus il est flexible, rapide et intégré avec de divers fonctionnalités et fonctions efficaces. Ce logiciel permet la création des diagrammes UML de manière simple.

Mongo DB[16] est un système de gestion de base de données orienté documents, répartissable sur un nombre quelconque d'ordinateurs et ne nécessitant pas de schéma prédéfini des données. Il est écrit en C++.



Figure 4.7 – MongoDB [16]



Figure 4.8 – Django [17]

Django [17] est un framework Python de haut niveau, permettant un développement rapide de sites internet, sécurisés, et maintenables. Créé par des développeurs expérimentés, Django prend en charge la plupart des tracas du développement web, vous pouvez donc vous concentrer sur l'écriture de votre application sans avoir besoin de réinventer la roue. Il est gratuit, open source, a une communauté active, une bonne documentation, et plusieurs options pour du support gratuit ou non.

Jira est un outil de développement logiciel utilisé par les ingénieurs pour suivre et gérer les tâches. Vous pouvez l'utiliser pour la gestion de projet Agile et Waterfall afin de suivre les bogues, les fonctionnalités et d'autres éléments de travail. Vous pouvez également configurer JIRA pour qu'il fonctionne avec de nombreux outils de gestion de services, ce qui en fait un outil polyvalent pour la gestion des tâches.



Figure 4.9 – Jira [18]

Étude comparative des frameworks

Cette partie vise à expliquer les choix des frameworks en se basant sur une comparaison entre nos choix et les autres frameworks existants sur le marché du travail.

1- Une comparaison entre les technologies existant dans le marché et la technologie que nous avons adaptés au niveau du frontend :

Caractéristique	Flutter	React Native	Kotlin	Ionic
Performance	Performances élevées grâce à son moteur de rendu intégré (Skia). Offre une réactivité fluide et cohérente sur une variété d'appareils.	Performances bonnes mais peuvent varier en fonction de la complexité de l'application et de la qualité des modules natifs.	Performances bonnes, dépendent de l'optimisation du code Kotlin et de l'implémentation.	Performances acceptables, mais peuvent être moins fluides que celles des autres frameworks, en particulier sur des appareils moins puissants.

Table 4.2 – Suite de la comparaison des frameworks frontend–

Caractéristique	Flutter	React Native	Kotlin	Ionic
Accessibilité pour les non-voyants	Offre un bon support pour l’accessibilité, mais nécessite une attention particulière pour garantir une expérience optimale pour les utilisateurs non voyants.	Prise en charge de l’accessibilité, mais peut nécessiter des ajustements supplémentaires pour répondre pleinement aux besoins des utilisateurs non voyants.	Kotlin en soi ne fournit pas de fonctionnalités spécifiques pour les utilisateurs non voyants, mais les applications développées avec Kotlin peuvent être rendues accessibles avec des efforts supplémentaires.	L’accessibilité peut être améliorée avec des pratiques de développement appropriées, mais cela peut nécessiter des efforts supplémentaires pour garantir une expérience optimale pour les utilisateurs non voyants.
Développement multiplateforme	Permet le développement multiplateforme natif pour iOS, Android, Web et bureau à partir d’une seule base de code, offrant ainsi une efficacité de développement et une réduction des coûts.	Basé sur JavaScript, permet de partager une grande partie du code entre les plateformes iOS et Android, mais peut nécessiter des ajustements spécifiques à la plateforme.	Principalement utilisé pour le développement Android natif, bien qu’il soit possible de partager une partie de la logique métier entre les applications Android et iOS.	Utilise Angular et TypeScript pour le développement multiplateforme d’applications mobiles et web, mais peut nécessiter des ajustements pour garantir une expérience utilisateur cohérente sur différentes plateformes.

Table 4.2 – Suite de la comparaison des frameworks frontend–

Caractéristique	Flutter	React Native	Kotlin	Ionic
Facilité de prise en main	Dart est relativement facile à apprendre pour ceux qui viennent de langages similaires comme JavaScript ou Java, avec une documentation complète et des ressources d'apprentissage disponibles.	Basé sur JavaScript, ce qui le rend accessible aux développeurs familiers avec ce langage. Bénéficie d'une grande communauté active avec de nombreuses ressources d'apprentissage.	Similaire à Java mais avec des fonctionnalités modernes, ce qui le rend relativement facile à apprendre pour les développeurs Java. Les ressources d'apprentissage peuvent être moins abondantes que pour Flutter ou React Native.	Utilise Angular et TypeScript, offrant une transition fluide pour les développeurs web. Dispose de ressources d'apprentissage abondantes grâce à la communauté active d'Angular.
Écosystème et communauté	Dispose d'une communauté active avec de nombreuses bibliothèques et plugins disponibles, ainsi qu'un soutien solide de Google.	Bénéficie d'une grande communauté avec de nombreuses ressources d'apprentissage et de développement.	En constante évolution avec une adoption croissante, mais peut avoir une communauté plus petite par rapport aux frameworks multiplateformes.	Dispose d'une communauté active, en particulier dans le domaine du développement web, mais peut être moins développé pour les applications mobiles par rapport à Flutter ou React Native.

Table 4.2 – Comparaison entre framework flutter et d'autres framework frontend existant [19]

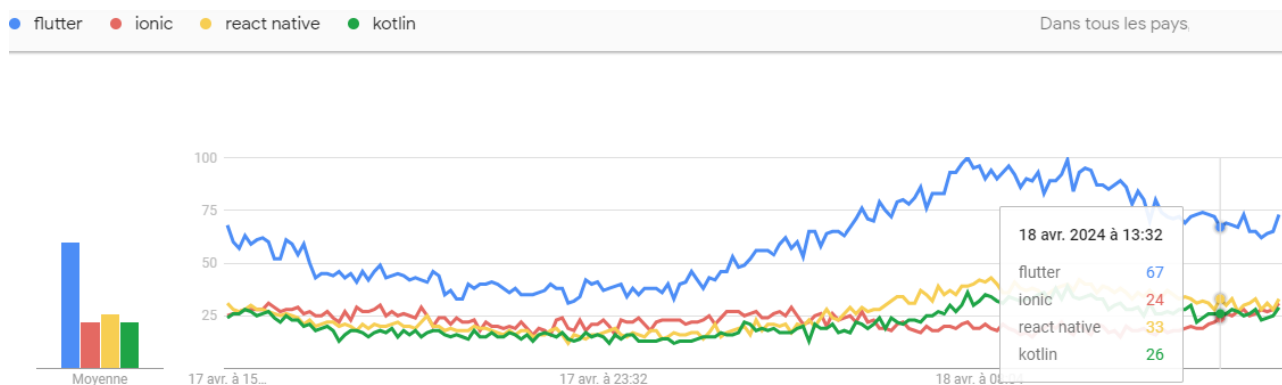


Figure 4.10 – Comparaison entre framework flutter et d'autres framework frontend existant selon google trends[20]

Après cette analyse comparative entre Flutter et les autres frameworks, et en gardant à l'esprit notre objectif de développer une application mobile pour les personnes non voyantes et aveugles, nous avons choisi Flutter pour sa performance supérieure par rapport aux autres, ainsi que sa capacité à offrir un bon support pour garantir l'accessibilité. De plus, la vaste communauté existante nous aidera à trouver différentes ressources nécessaires.

2- Une comparaison entre les technologies existant dans le marché et la technologie que nous avons adaptés au niveau du backend :

Critères	Django	Spring Boot	Node.js
Langage	Django utilise Python, un langage clair et expressif, favorisant la lisibilité du code et la productivité du développeur.	Spring Boot utilise Java, un langage bien établi avec une forte typage statique et un écosystème mature.	Node.js utilise JavaScript, un langage dynamique et flexible, avec une grande communauté et de nombreuses bibliothèques disponibles.
Écosystème	Django dispose d'un écosystème robuste avec de nombreuses bibliothèques et modules pour faciliter le développement, ainsi qu'une documentation complète et une communauté active.	Spring Boot bénéficie d'un écosystème solide avec une large gamme de bibliothèques et de modules, ainsi qu'un support étendu de la part de la communauté Java.	Node.js offre un écosystème dynamique et en croissance constante, avec une vaste gamme de modules disponibles via npm et une communauté active de développeurs.

Table 4.3 – Suite de la comparaison des frameworks backend–

Critères	Django	Spring Boot	Node.js
Sécurité	Django intègre des fonctionnalités de sécurité avancées telles que l'authentification, l'autorisation et la protection contre les attaques courantes, ce qui en fait un choix sûr pour le développement d'applications sensibles.	Spring Boot offre des fonctionnalités de sécurité avancées et une intégration transparente avec les outils de sécurité Java, garantissant la sécurité des applications.	Node.js nécessite des modules supplémentaires pour renforcer la sécurité, mais offre une grande flexibilité dans le choix des solutions de sécurité.
Performances	Django offre de bonnes performances grâce à son architecture basée sur le pattern MVC, sa gestion efficace des requêtes HTTP et ses capacités d'optimisation.	Spring Boot offre des performances élevées grâce à son architecture légère et modulaire, ainsi qu'une intégration transparente avec les technologies Java.	Node.js offre des performances élevées grâce à sa nature asynchrone et non bloquante, idéale pour les applications web en temps réel.
Accessibilité	Django permet une conception et un développement accessibles, avec des outils et des pratiques pour garantir une expérience utilisateur optimale pour les personnes non voyantes ou aveugles.	Spring Boot offre des possibilités d'accessibilité avec des ajustements supplémentaires pour répondre pleinement aux besoins des utilisateurs non voyants.	Node.js peut être utilisé pour créer des applications accessibles, mais nécessite des efforts supplémentaires pour garantir une expérience optimale pour les personnes non voyantes ou aveugles.

Table 4.3 – Comparaison entre django et les autres framework backend existant [21]

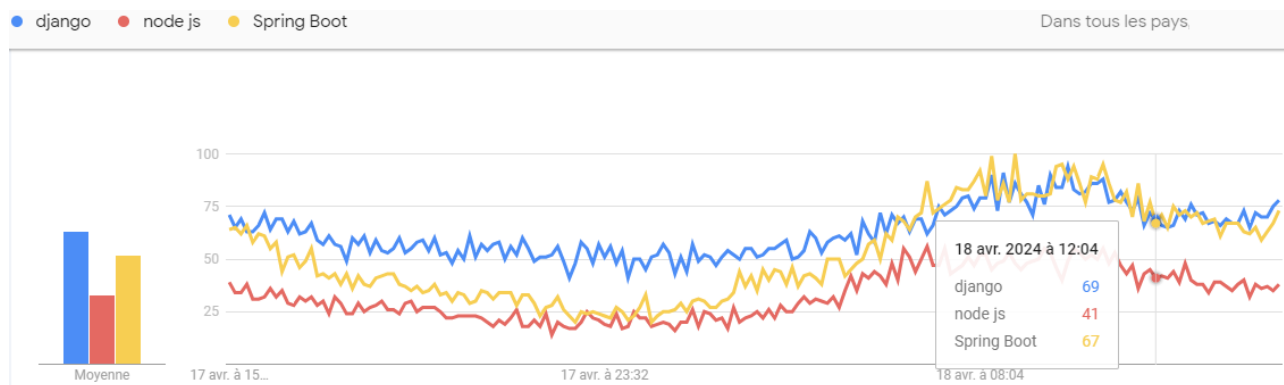


Figure 4.11 – Comparaison entre framework django et d'autres framework backend existant selon google trends [20]

Après avoir examiné plusieurs frameworks pour le développement de notre application mobile, nous avons choisi Django comme backend pour plusieurs raisons essentielles. Django offre des performances solides grâce à sa conception basée sur Python, garantissant une expérience utilisateur fluide et réactive. De plus, il intègre des fonctionnalités de sécurité avancées, assurant la confidentialité des données sensibles de nos utilisateurs. En conclusion, en choisissant Django, nous nous assurons de bénéficier de performances fiables, d'une sécurité avancée, d'opportunités d'accessibilité et d'un support communautaire robuste, garantissant ainsi une expérience utilisateur inclusive et sécurisée pour notre public cible.

3- Une comparaison entre les technologies existant dans le marché et la technologie que nous avons adaptés au niveau du base de données :

Critères	MongoDB	SQLite	PostgreSQL
Modèle de données	MongoDB utilise un modèle de données NoSQL basé sur des documents JSON flexibles, ce qui permet une modélisation des données plus dynamique et évolutive.	SQLite utilise un modèle de données SQL relationnel stocké dans un fichier local, adapté aux applications simples nécessitant une base de données légère et autonome.	PostgreSQL utilise un modèle de données SQL relationnel robuste avec prise en charge de nombreuses fonctionnalités avancées telles que les contraintes de clé étrangère et les transactions ACID.

Table 4.4 – Suite de la comparaison des frameworks base de données–

Critères	MongoDB	SQLite	PostgreSQL
Évolutivité	MongoDB offre une évolutivité horizontale facile à mettre en œuvre grâce à la réplication et au partitionnement automatique, ce qui permet de gérer facilement de gros volumes de données.	SQLite est conçu pour les petites applications nécessitant une base de données simple et autonome, mais peut manquer d'évolutivité pour les applications à grande échelle.	PostgreSQL offre une évolutivité verticale et horizontale grâce à la réplication, aux clusters et à la répartition des données, adaptée aux applications de taille moyenne à grande.
Performance	MongoDB offre des performances élevées pour les opérations de lecture et d'écriture grâce à son architecture distribuée et à son stockage des données sous forme de documents JSON.	SQLite offre des performances élevées pour les opérations locales sur des fichiers de base de données, mais peut être moins adapté pour les applications nécessitant une base de données serveur et des opérations réseau.	PostgreSQL offre des performances élevées pour les charges de travail complexes grâce à son optimiseur de requêtes avancé et à ses fonctionnalités d'indexation efficaces.

Table 4.4 – Suite de la comparaison des frameworks base de données–

Critères	MongoDB	SQLite	PostgreSQL
Flexibilité	MongoDB offre une flexibilité élevée avec la possibilité de stocker des données semi-structurées et non structurées dans des documents JSON, adaptée aux besoins changeants des applications modernes.	SQLite offre une flexibilité limitée avec un support pour les types de données SQL standard et une structure de base de données statique, adaptée aux applications simples avec des besoins de stockage de données basiques.	PostgreSQL offre une flexibilité moyenne avec une prise en charge de nombreuses fonctionnalités SQL avancées, mais peut nécessiter une planification minutieuse de la conception de la base de données pour répondre aux besoins évolutifs.
Communauté	MongoDB bénéficie d’une communauté active de développeurs, de contributeurs et de supporters, avec une documentation complète et de nombreuses ressources disponibles.	SQLite a une communauté active mais plus restreinte, avec une documentation complète et un support disponible.	PostgreSQL bénéficie d’une communauté large et active, avec une documentation complète, des forums de discussion et un support professionnel disponible.

Table 4.4 – Comparaison entre mongodb et d’autres systèmes de gestion de bases de données existants [22]

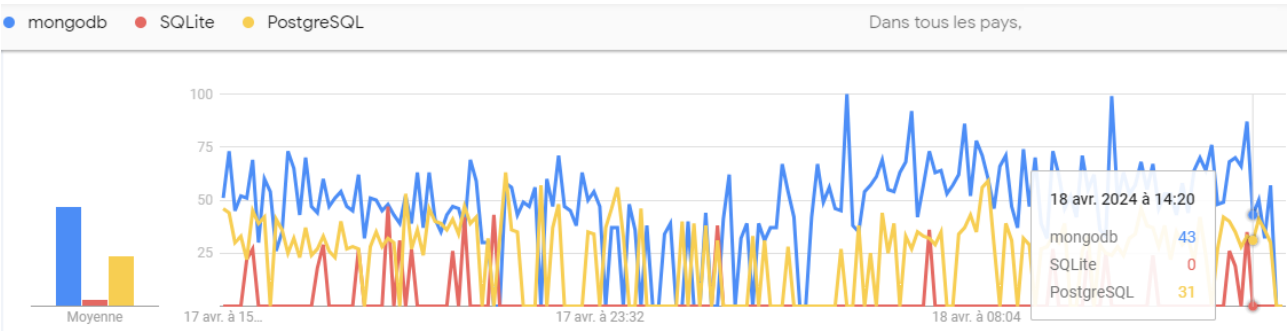


Figure 4.12 – Comparaison entre mongodb et d’autres systèmes de gestion de bases de données existant selon google trends [20]

En comparant MongoDB aux autres solutions de base de données telles que SQLite et PostgreSQL, nous constatons que MongoDB offre plusieurs avantages significatifs qui répondent aux besoins spécifiques de notre application. Tout d'abord, MongoDB se distingue par son modèle de données NoSQL basé sur des documents JSON, offrant une flexibilité et une évolutivité qui correspondent parfaitement à la nature dynamique du contenu éducatif. Cette flexibilité permet de stocker des données structurées et non structurées de manière cohérente, ce qui est essentiel pour fournir un contenu riche et varié aux utilisateurs non voyants et aveugles. De plus, MongoDB offre une évolutivité horizontale facile à mettre en œuvre, ce qui permet de gérer efficacement les vastes quantités de données générées par une plateforme d'e-learning. En termes de performances, MongoDB offre des performances élevées pour les opérations de lecture et d'écriture, garantissant ainsi une expérience utilisateur fluide et réactive. En outre, MongoDB bénéficie d'une communauté active de développeurs et de supporteurs, offrant une documentation complète et de nombreuses ressources disponibles pour soutenir le développement et la maintenance de notre application.

Conclusion

Pour conclure dans ce chapitre, nous avons examiné l'environnement matériel et logiciel nécessaire, ainsi que les langages de programmation et les technologies utilisées pour développer notre application web. Le prochain chapitre portera sur l'examen de l'architecture globale de l'application et sa mise en œuvre.

Réalisation et mise en œuvre

Introduction

Ce chapitre est consacré à la mise en place de notre application, durant lequel nous examinerons son architecture. Nous présenterons et expliquerons également quelques interfaces clés de l'application.

1. Architecture de notre système

Dans cette section, nous aborderons l'architecture que nous avons adoptée pour notre application, en détaillant à la fois l'architecture frontend et backend. L'adoption d'une architecture spécifique devient de plus en plus courante dans le domaine du développement en raison de la complexité croissante des logiciels. Cette architecture vise à organiser le code et à répartir les responsabilités entre plusieurs couches afin d'augmenter la cohésion et de réduire le couplage.

Les patrons d'architecture jouent un rôle crucial dans ce processus. Un patron est utilisé dans le but de séparer la logique métier de la présentation d'un logiciel, assurant ainsi une indépendance entre les différents composants du système. Cette approche garantit une meilleure réutilisation du code, une plus grande maintenabilité et une résistance accrue aux changements.

Architecture globale de l'application

Dans notre application, nous avons utilisé deux frameworks : Flutter pour le frontend et Django pour le backend.

Frontend (Flutter) :

- Cette partie de l'application gère l'interface utilisateur et les interactions avec l'utilisateur.
- Les vues, les widgets et la logique de présentation sont implémentés dans Flutter.

- Le frontend communique avec le backend via des requêtes HTTP (API RESTful).

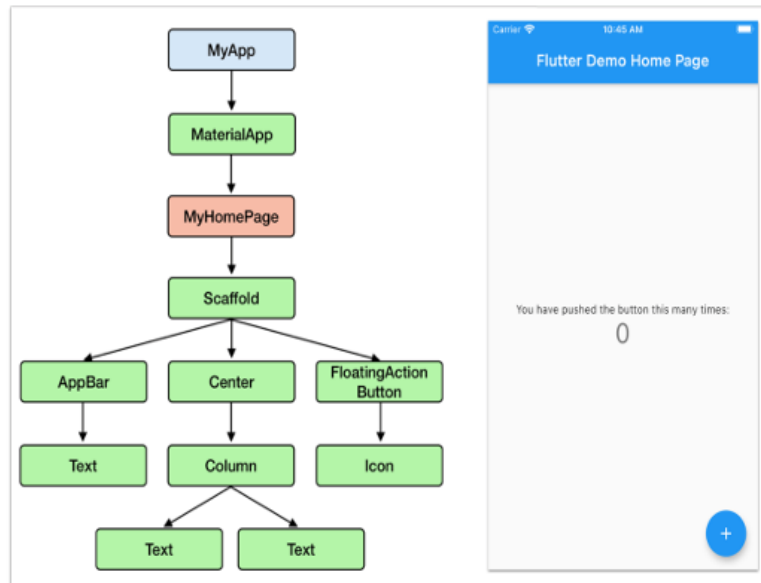


Figure 5.1 – Illustration des widgets d’un simple exemple d’une interface graphique.[23]

Backend (Django) :

- Le backend est responsable du traitement des requêtes provenant du frontend et de la gestion des données.
- Le code Django est organisé en modèles, vues (views), serializers et URLs.
- Les serializers permettent la sérialisation et la désérialisation des données pour les échanges avec le frontend.
- Nous utilisons des endpoints RESTful dans Django pour établir une communication avec l’application Flutter. Ces endpoints implémentent des méthodes HTTP standard telles que GET, POST, PUT, DELETE pour effectuer des opérations CRUD (Create, Read, Update, Delete) sur les données.
- Nous avons également adopté le modèle Repository dans Django pour encapsuler la logique d’accès aux données et interagir avec la base de données. Cela sépare la logique métier de l’accès aux données, rendant ainsi le code plus modulaire et testable.

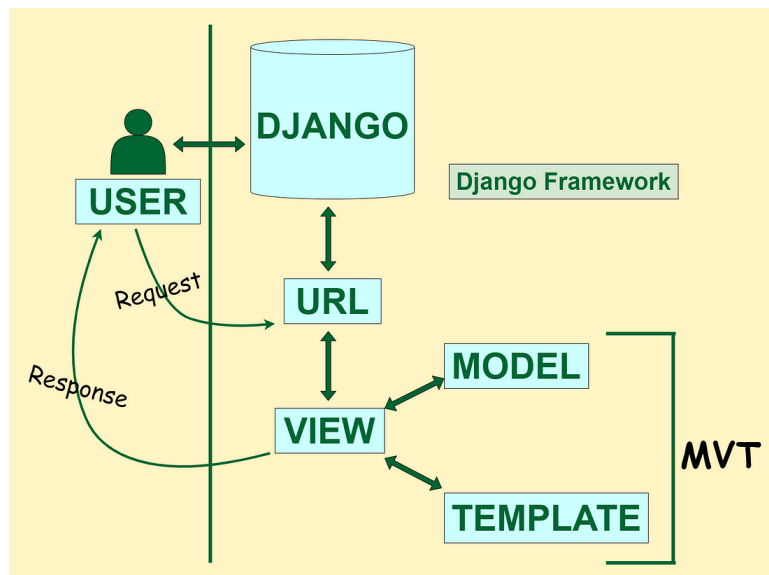


Figure 5.2 – Architecture de Django.

Base de données (MongoDB) :

Pour la base de données, nous avons opté pour MongoDB, qui stocke les données de manière non structurée sous forme de documents JSON. Les données sont organisées en collections, et nous utilisons le pilote MongoDB pour Python (pymongo) dans Django pour interagir avec la base de données.

En ce qui concerne la communication entre le frontend et le backend, nous utilisons une API RESTful, qui permet aux deux parties de communiquer de manière efficace en utilisant des requêtes HTTP standard.

2. Quelques interfaces de "Nawarny"

Dans cette partie de notre rapport, nous allons consacrer une attention particulière à l'explication et à la présentation des interfaces de notre application.

2.1. Page d'accueil

Les deux interfaces ci dessous représentent les deux premières interfaces de notre application : l'une affiche un loader pendant le chargement de l'application, tandis que l'autre présente un bouton permettant de commencer.



Figure 5.3 – Page de Loading

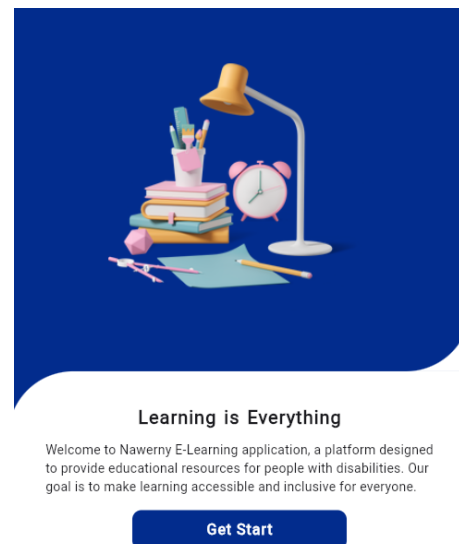


Figure 5.4 – Page d'accueil

2.2. Page visiteur

Cette interface présente la page visiteur où un étudiant peut découvrir les différents cours que la plateforme propose, classés par catégories.

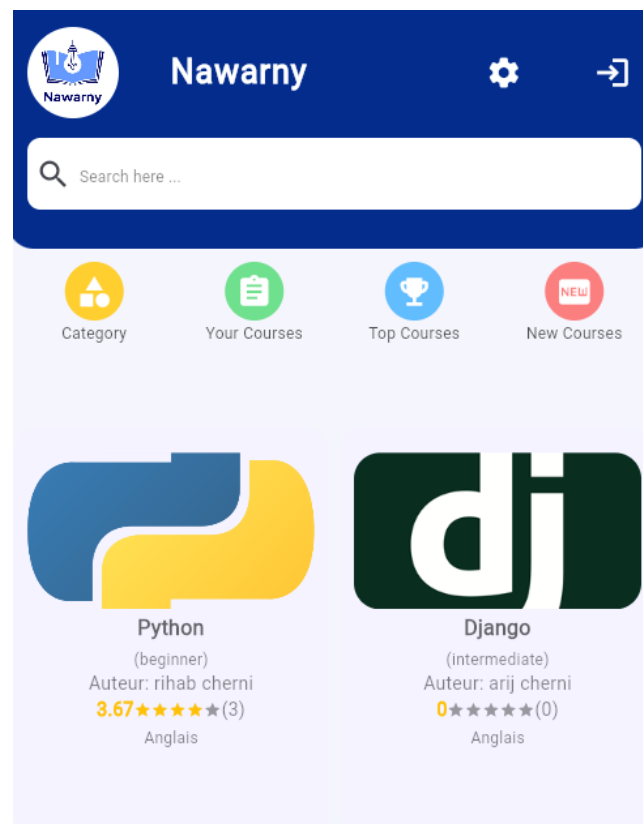


Figure 5.5 – Page visiteur

2.3. Interface pour choisir le rôle dans l'application

Cette interface expose les différents rôles accessibles à un utilisateur externe au sein de l'application, lui permettant d'endosser le statut d'apprenant ou d'auteur.

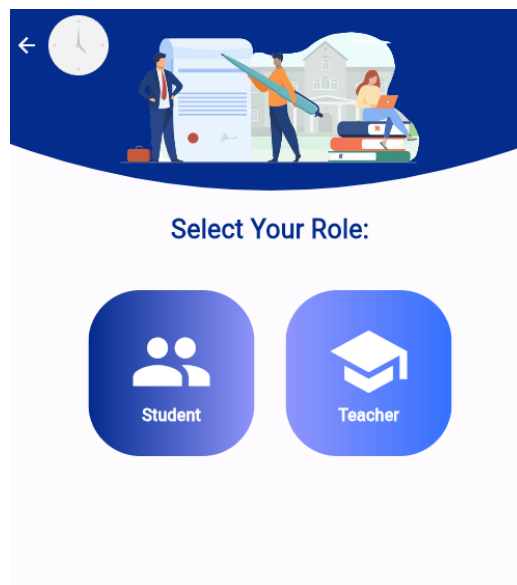


Figure 5.6 – Interface de sélection des rôles

2.4. Interface d'inscription

L'interface d'inscription est celle par laquelle l'utilisateur enregistre ses données personnelles dans le but de créer un compte sur la plateforme.

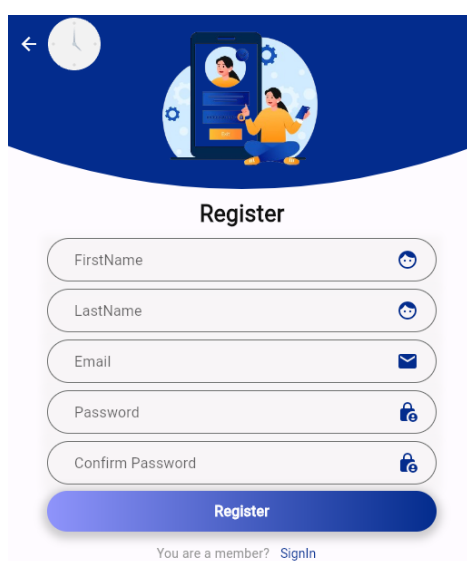


Figure 5.7 – Page d'inscription

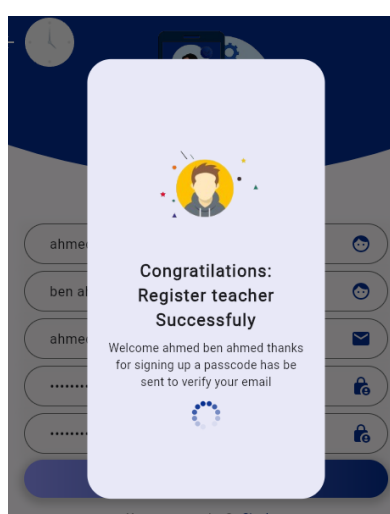


Figure 5.8 – Inscription confirmée

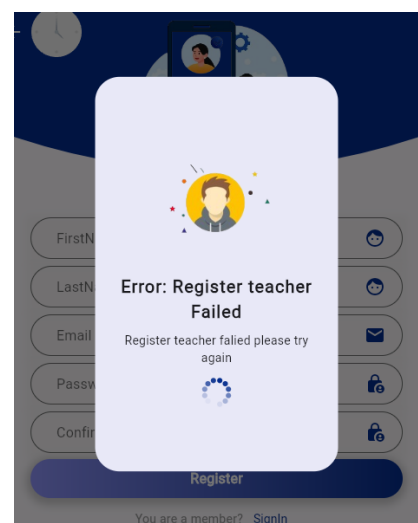


Figure 5.9 – Erreur d'inscription

2.5. Interface de connexion

Cette interface de connexion permet à l'administrateur (de manière directe) et aux autres acteurs de se connecter après s'être identifiés en tant qu'auteur ou apprenant.

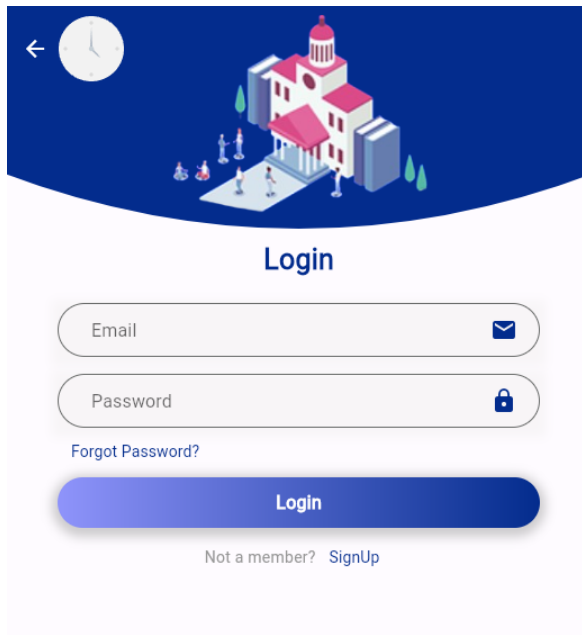


Figure 5.10 – Interface login auteur et admin

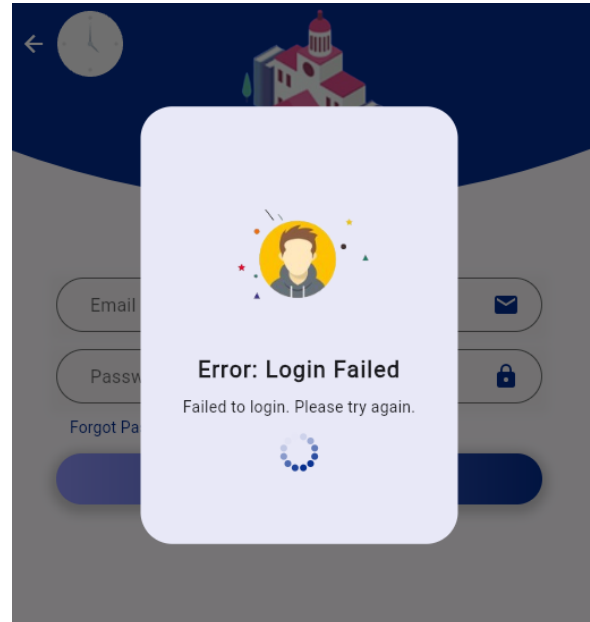


Figure 5.11 – Interface erreur de connexion

2.6. Les interfaces de l'administrateur

Dans cette partie, nous allons traiter de la section consacrée aux interfaces de l'administrateur. L'administrateur est l'acteur responsable de la gestion de l'application, c'est pourquoi nous trouverons plusieurs interfaces de gestion.

2.6.1. Interface pour le tableau de bord

Cette interface est dédiée au tableau de bord où l'administrateur de l'application pourra visualiser des données, telles que les nombres significatifs (comme le total des cours, le total des apprenants), ainsi que des schémas graphiques présentant l'évolution des données au fil des années, par exemple, l'ajout de cours.

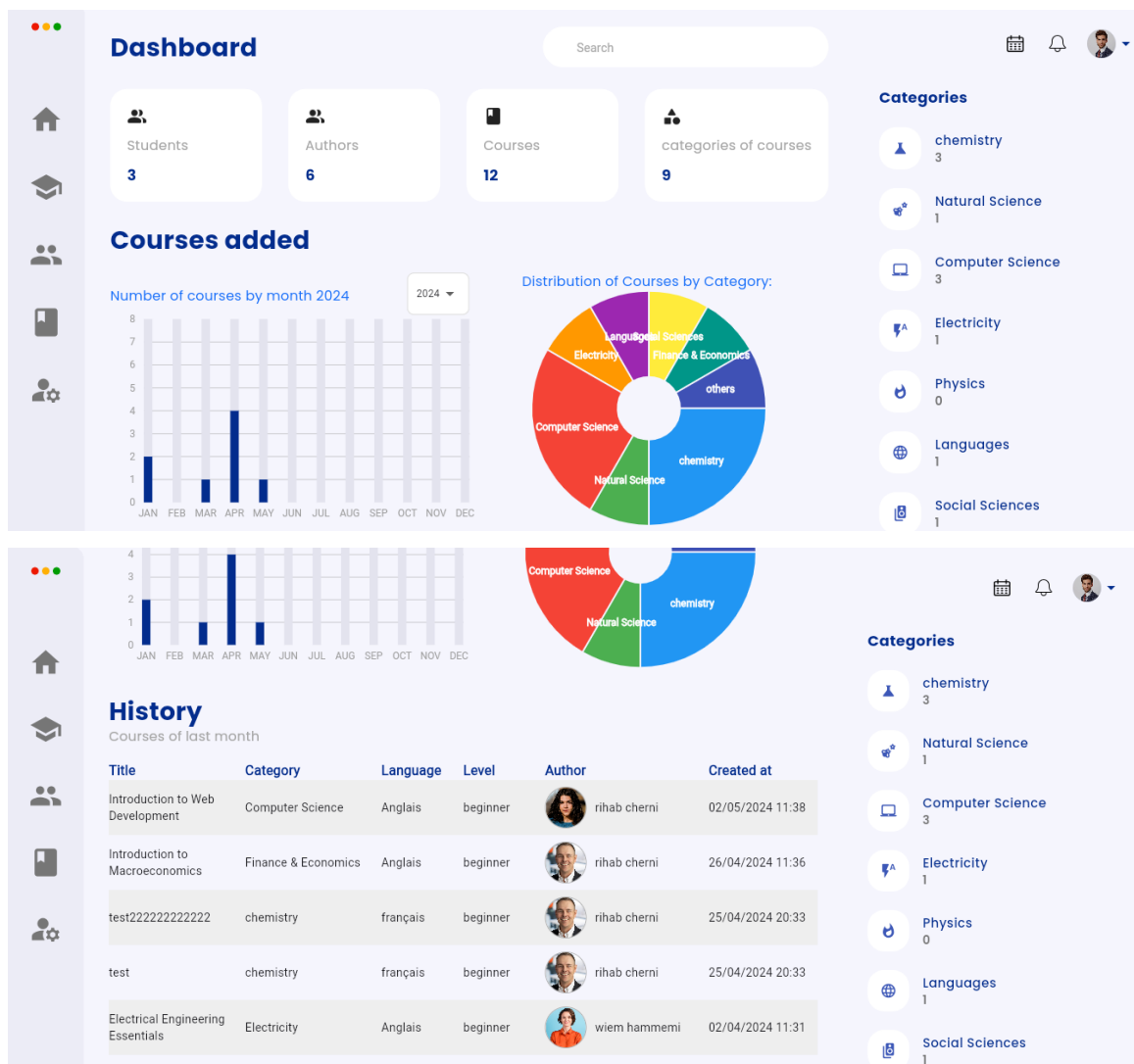


Figure 5.12 – Interfaces tableau de bord pour l'administrateur

2.6.2. Interface de gestion des apprenants

Cette interface de gestion des apprenants permet à l'administrateur d'afficher, rechercher et supprimer des apprenants.

Students

Photo	firstName	LastName	Email	Address	Phone Number	Action
	rania	cherni	rania@gmail.com		2001-06-05	
	ahmed	ben ahmed	ahmed@gmail.com			
	apprenant	apprenant	apprenant@gmail.com			

Figure 5.13 – Interface gestion des apprenants

2.6.3. Interface de gestion des auteurs

Cette interface de gestion des auteurs permet à l'administrateur d'afficher, rechercher et supprimer des auteurs.

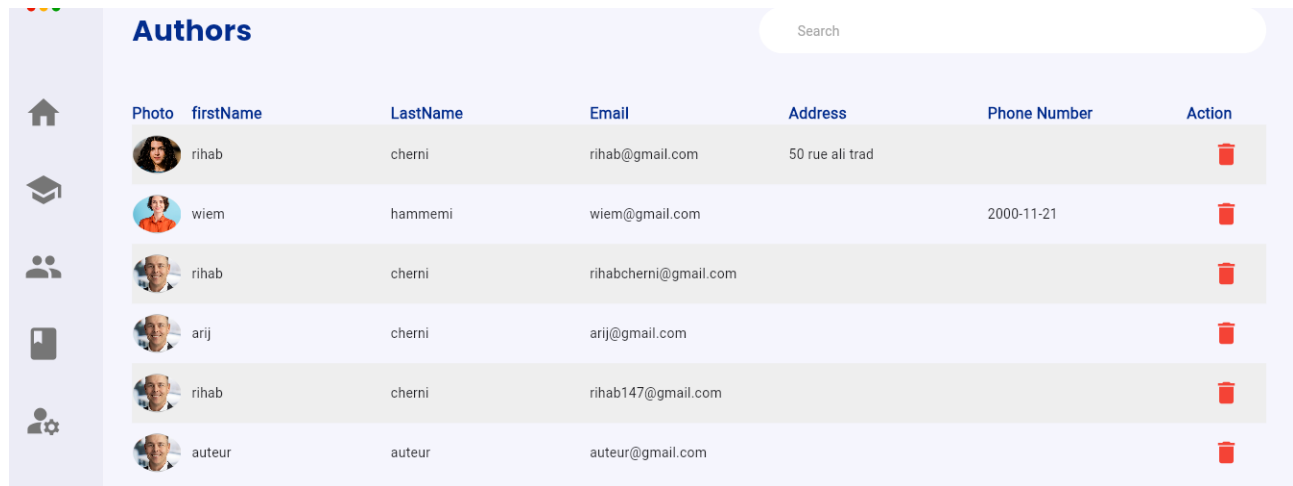


Photo	firstName	LastName	Email	Address	Phone Number	Action
	rihab	cherni	rihab@gmail.com	50 rue ali trad		
	wiem	hammemi	wiem@gmail.com		2000-11-21	
	rihab	cherni	rihabcherni@gmail.com			
	arij	cherni	arij@gmail.com			
	rihab	cherni	rihab147@gmail.com			
	auteur	auteur	auteur@gmail.com			

Figure 5.14 – Interface gestion des auteurs

2.6.4. Interface des cours

Cette interface de gestion des cours permet à l'administrateur d'afficher, rechercher et supprimer des cours.

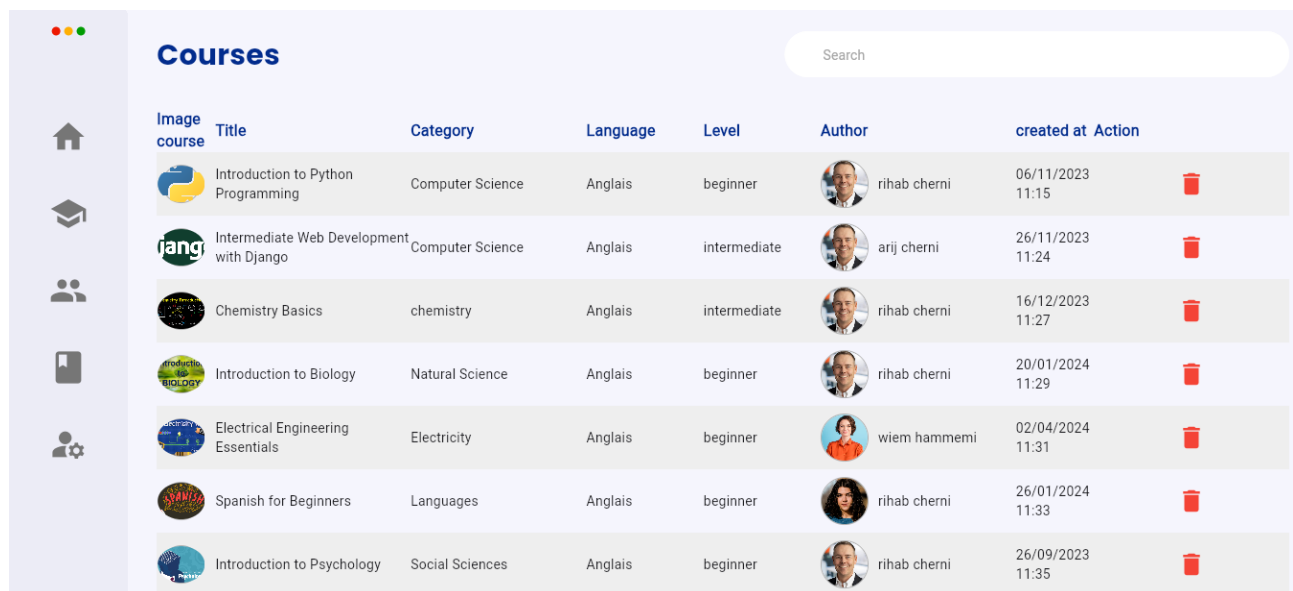


Image course	Title	Category	Language	Level	Author	created at	Action
	Introduction to Python Programming	Computer Science	Anglais	beginner	rihab cherni	06/11/2023 11:15	
	Intermediate Web Development with Django	Computer Science	Anglais	intermediate	arij cherni	26/11/2023 11:24	
	Chemistry Basics	chemistry	Anglais	intermediate	rihab cherni	16/12/2023 11:27	
	Introduction to Biology	Natural Science	Anglais	beginner	rihab cherni	20/01/2024 11:29	
	Electrical Engineering Essentials	Electricity	Anglais	beginner	wiem hammemi	02/04/2024 11:31	
	Spanish for Beginners	Languages	Anglais	beginner	rihab cherni	26/01/2024 11:33	
	Introduction to Psychology	Social Sciences	Anglais	beginner	rihab cherni	26/09/2023 11:35	

Figure 5.15 – Interface gestion des cours

2.6.5. Interface profil administrateur

Cette interface offre à l'administrateur la possibilité de gérer son profil et de modifier ses informations personnelles.

2.7. Les interfaces de l'auteur

Maintenant, abordons les interfaces de l'auteur. Ensemble, ces interfaces représentent la partie de l'application spécifiquement dédiée à l'auteur.

2.7.1. Interface des cours

L'interface des cours est celle où l'on trouve l'affichage de l'ensemble des cours enseignés par l'auteur. Le cours n'est pas détaillé; on y trouve des informations générales telles que le titre, la date d'ajout, etc.

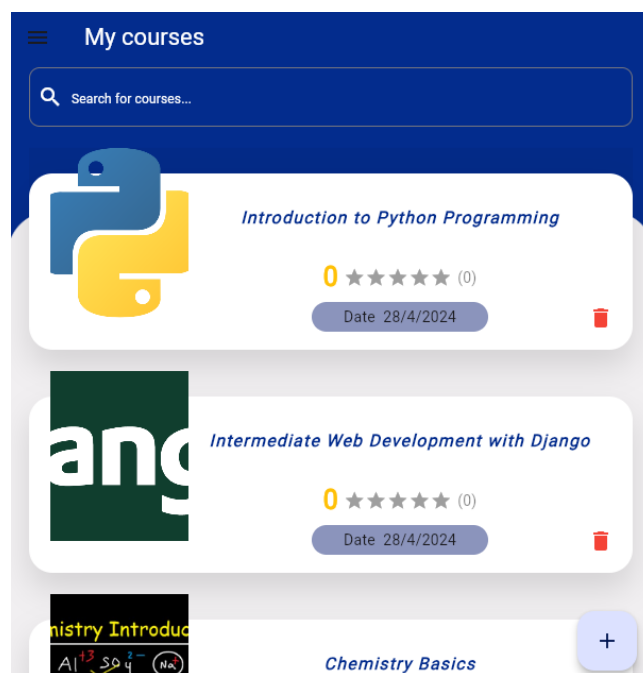


Figure 5.16 – Interface des cours

2.7.2. Interface des détails d'un cours

L'interface détaillée du cours est celle qui affiche les détails du cours (accessible suite à un clic sur un cours dans l'interface précédente). Par détails du cours, on entend d'autres informations en plus du contenu du cours.

2.7.3. Interface d'ajout d'un cours

Cette interface représente le moyen par lequel un auteur peut ajouter son cours. L'ajout doit se faire selon des conditions spécifiques pour assurer l'accessibilité. L'auteur doit joindre chaque image ou vidéo à un texte explicatif.

2.7.4. Interface de profil

Cette interface offre à l'auteur la possibilité de gérer son profil et de modifier ses informations personnelles.

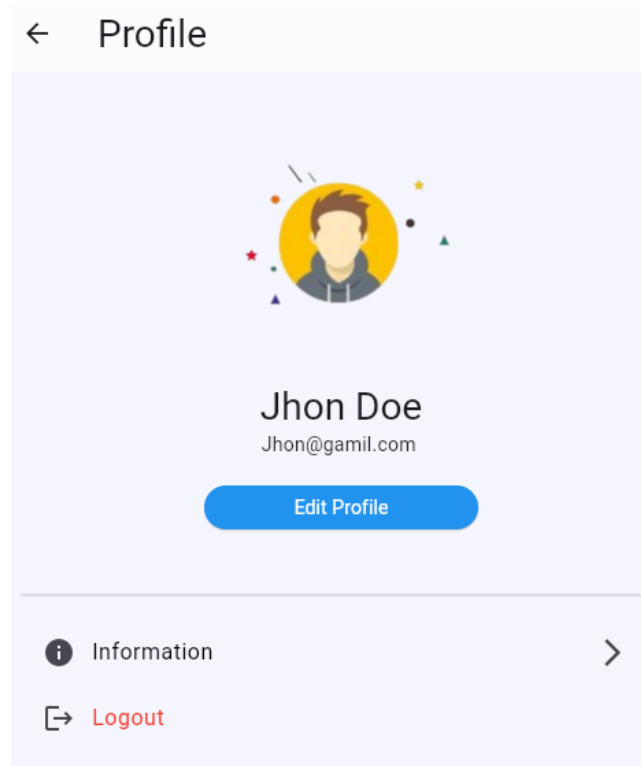


Figure 5.17 – Interface de profil

2.7.5. Interface pour l'affichage des apprenants

Cette interface présente la vue qui affiche des informations sur les apprenants actuellement en train d'étudier à travers les cours de l'auteur.

2.8. Les interfaces de l'apprenant

Après la connexion, l'apprenant aura son propre compte à partir duquel il commencera son expérience éducative

2.8.1. Interface visualisant le compte de l'apprenant

Cette interface met en évidence que l'apprenant est maintenant dans son compte à partir d'une barre latérale affichant son adresse, son image, ainsi que les autres pages où l'apprenant peut naviguer.

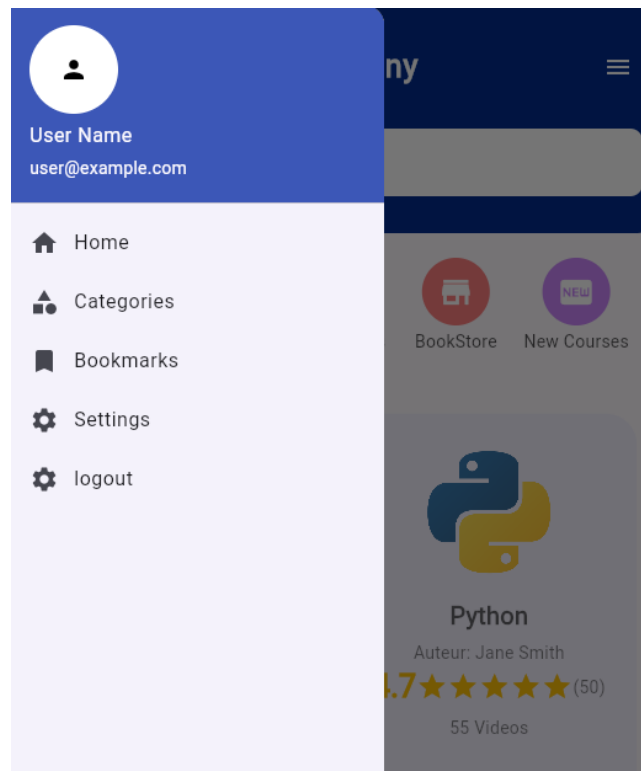


Figure 5.18 – Interface visualisant le compte de l'apprenant

2.8.2. Interface de Consultation des Cours

Cette interface permet à l'apprenant de parcourir et de rechercher les différents cours disponibles. Elle fournit des informations telles que les titres des cours, les descriptions ect ...

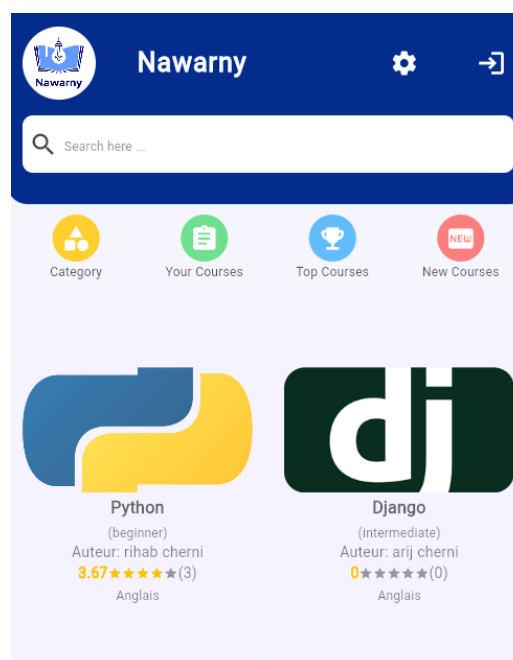


Figure 5.19 – Interface de Consultation des cours

2.8.3. Interface de Consultation du Contenu des Cours

Cette interface offre un accès aux différents contenus pédagogiques à l'intérieur d'un cours sélectionné.

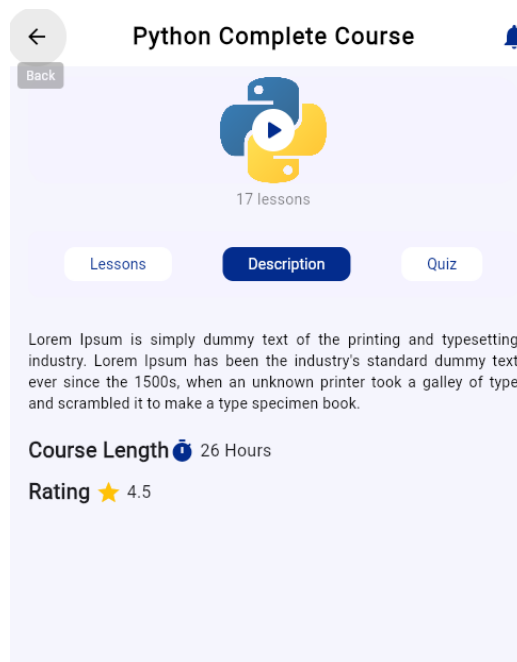


Figure 5.20 – Interface de consultation des détails des cours.

2.8.4. Interface de catégories de cours

Cette interface affiche les catégories de cours disponibles pour l'apprenant.

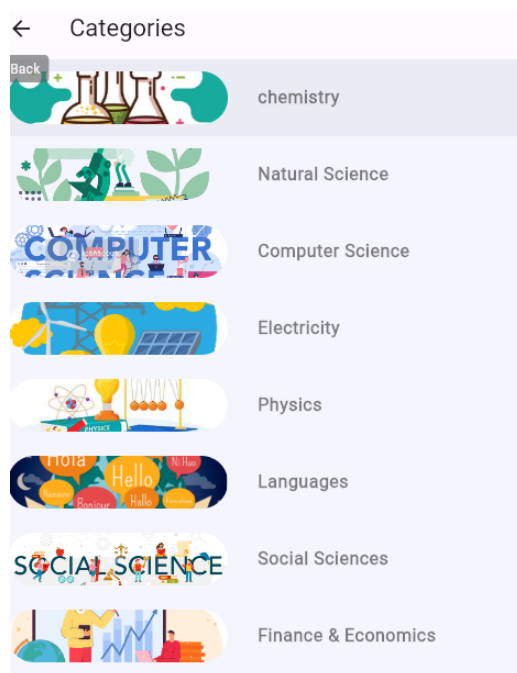


Figure 5.21 – Interface de catégories de cours

2.9. Les interfaces mettant en avant l'accessibilité

Ces interfaces mettent en avant le lecteur d'écran, un outil essentiel pour garantir l'accessibilité en lisant les détails de l'interface lors de la navigation à travers les pages.

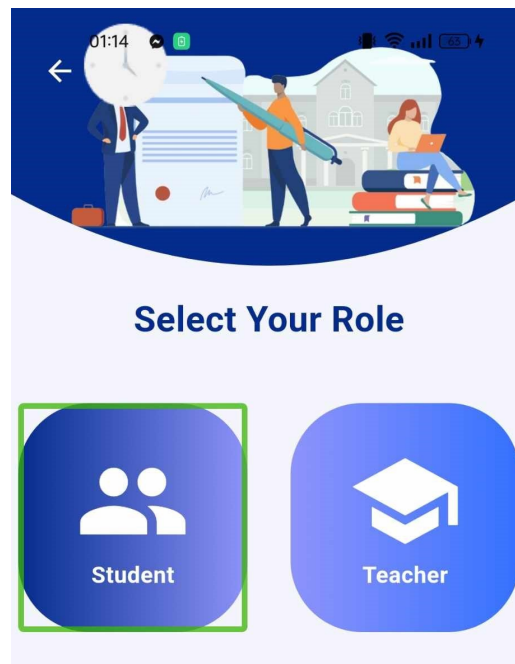


Figure 5.22 – L'interface mettant en avant le lecteur d'écran pour la page de sélection des rôles

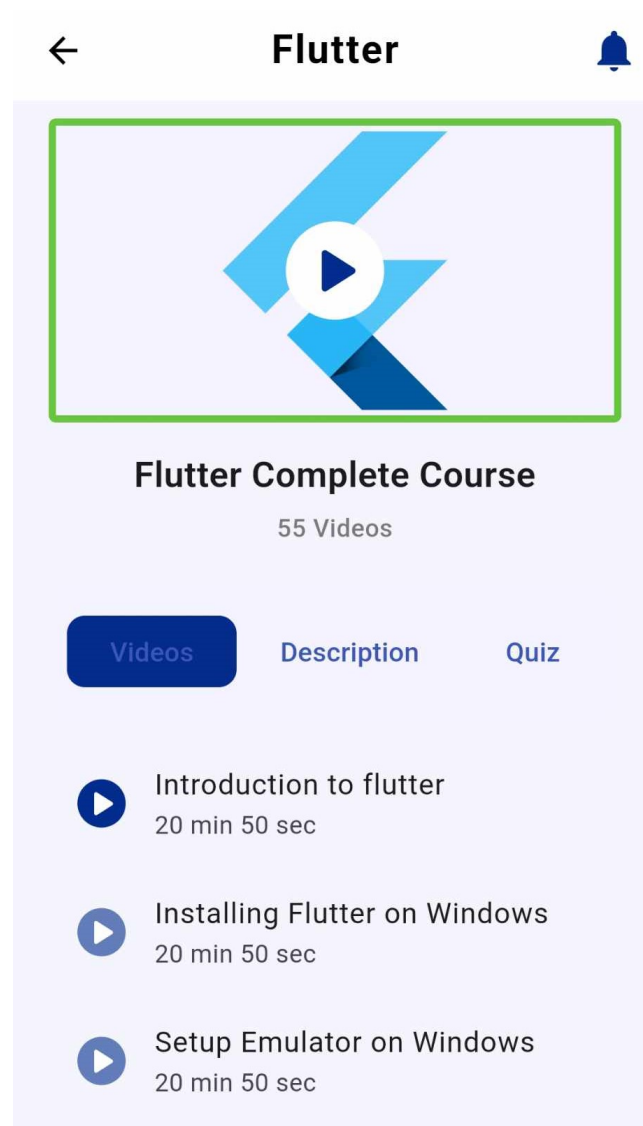


Figure 5.23 – L'interface mettant en avant le lecteur d'écran pour la consultation du contenu des cours

Conclusion

Pour conclure ce chapitre, nous avons examiné l'architecture globale de notre application. Nous avons également présenté des interfaces de l'application pour offrir une vue d'ensemble.

Conclusion générale

Le développement d'une application d'e-learning accessible représente un défi essentiel dans le domaine de l'éducation en ligne. Notre projet vise à répondre à cette exigence en proposant une solution centrée sur l'accessibilité, simplifiant ainsi l'expérience d'apprentissage pour les personnes non voyantes.

Notre approche se distingue par sa simplicité et son accessibilité intégrée, offrant une expérience fluide à tous, sans nécessiter de paramétrage complexe. En mettant l'accent sur une conception intuitive et des fonctionnalités spécifiquement adaptées aux besoins des utilisateurs non voyants, nous visons à garantir une expérience d'apprentissage agréable pour tous. À l'avenir, nous prévoyons d'explorer les possibilités offertes par l'intelligence artificielle, notamment en intégrant des fonctionnalités telles que des assistants virtuels et des analyses avancées des progrès des apprenants, afin d'innover constamment notre application pour répondre aux besoins évolutifs du marché de l'éducation en ligne.

Ce projet revêt une importance particulière pour nous, car il nous permet d'approfondir notre expertise dans de nouvelles technologies telles que Flutter, MongoDB et Django. En contribuant à l'amélioration continue de l'accessibilité et de l'efficacité des applications d'e-learning, notre projet soutient une éducation inclusive et accessible à tous.

Bibliographie

- [1] “Canvas LMS, The Premier Teaching and Learning Software Worldwide,” Consulté le 18 mars 2024. adresse : <https://www.instructure.com/canvas>.
- [2] “Canvas Student,” Consulté le 20 mars 2024. adresse : <https://canvas.fr.aptoide.com/app>.
- [3] “Fix Accessibility Issues with Canvas LMS,” Consulté le 20 mars 2024. adresse : <https://www.instructure.com/resources/blog/fix-accessibility-issues-canvas-lms-popetech>.
- [4] “<https://play.google.com/store/apps/details?id=com.blackboard.android.bbstudent>,” Consulté le 21 mars 2024. adresse : <https://www.softwaresuggest.com/blog/blackboard-vs-moodle-vs-canvas/>.
- [5] “<https://help.blackboard.com/Learn/Student/Ultra/Accessibility>,” Consulté le 22 mars 2024. adresse : <https://www.softwaresuggest.com/blog/blackboard-vs-moodle-vs-canvas/>.
- [6] “Emportez l’apprentissage en ligne avec vous, peu importe où tu vas,” Consulté le 22 mars 2024. adresse : <https://moodle.com/fr/solutions/application-moodle/>.
- [7] “Moodle app,” Consulté le 22 mars 2024. adresse : <https://download.moodle.org/mobile/>.
- [8] “Usability Problems on Desktop and Mobile Interfaces of the Moodle Learning Management System (LMS),” Consulté le 22 mars 2024. adresse : <https://research.moodle.org/396/>.
- [9] “Blackboard Vs Moodle Vs Canvas : The Better LMS,” Consulté le 21 mars 2024. adresse : <https://www.softwaresuggest.com/blog/blackboard-vs-moodle-vs-canvas/>.

- [10] K. KEITA, “Qu’est-ce que Flutter?,” Consulté le 18 avril 2024. adresse : <https://ibracilinks.com/blog/quest-ce-que-flutter>.
- [11] “Découvrir Visual Studio Code,” Consulté le 18 avril 2024. adresse : <https://www.blogdumoderateur.com/tools/visual-studio-code/>.
- [12] “Draw.io : un outil pour dessiner des diagrammes en ligne,” Consulté le 17 avril 2024. adresse : <https://www.tice-education.fr/tous-les-articles-et-ressources/articles-internet/819-draw-io-un-outil-pour-dessiner-des-diagrammes-en-ligne>.
- [13] “qu’est-ce que c’est et comment apprendre à l’utiliser?,” Consulté le 18 avril 2024. adresse : <https://datascientest.com/github-tout-savoir>.
- [14] “Documentation,” Consulté le 18 avril 2024. adresse : <https://fr.overleaf.com/learn>.
- [15] M. BRUCHER, “Présentation de StarUML,” Consulté le 17 avril 2024. adresse : <https://matthieu-brucher.developpez.com/tutoriels/conception/staruml/>.
- [16] L. DARRAGI, “MangoDB,” Consulté le 18 avril 2024. adresse : <https://www.rbktunisia.com/post/mangodb>.
- [17] “Introduction à Django,” Consulté le 17 avril 2024. adresse : <https://developer.mozilla.org/fr/docs/Learn/Server-side/Django/Introduction>.
- [18] “Outils Agile destinés aux équipes de développement,” Consulté le 18 avril 2024. adresse : <https://www.atlassian.com/fr/software/jira/agile>.
- [19] “Comparing React Native vs Flutter vs Ionic vs Kotlin for Mobile App Development,” Consulté le 30 mars 2024. adresse : <https://bluewhaleapps.com/blog/comparing-react-native-vs-flutter-vs-ionic-vs-kotlin-for-mobile-app-development>.
- [20] “Google Trends,” Consulté le 30 mars 2024. adresse : <https://trends.google.fr/trends/>.
- [21] “Node.js vs Spring Boot vs Django : Which One Is Best for Beginners?,” Consulté le 30 mars 2024. adresse : <https://sandydev.medium.com/node-js-vs-spring-boot-vs-django-which-one-is-best-for-beginners-8782d3be54>.
- [22] “What’s the Difference Between MongoDB and PostgreSQL?,” Consulté le 30 mars 2024. adresse : <https://aws.amazon.com/compare/the-difference-between-mongodb-and-postgresql/>.

- [23] “Flutter architectural overview,” Consulté le 25 mars 2024. adresse : <https://docs.flutter.dev/resources/architectural-overview>.

Résumé

Ce projet a pour objectif le développement d'une application d'e-learning accessible afin d'améliorer l'expérience d'apprentissage des personnes non voyantes. En combinant les technologies Flutter pour le frontend, Django pour le backend, et MongoDB pour la base de données, notre application propose une solution orientée vers l'accessibilité, assurant une expérience d'apprentissage fluide et plaisante pour tous les utilisateurs.

Mots clés : Application d'e-learning , Accessible , Personnes non voyantes ,Flutter , Django ,Mongo DB

Abstract

This project aims to develop an accessible e-learning application to enhance the learning experience of visually impaired individuals. By combining Flutter technology for the frontend, Django for the backend, and MongoDB for the database, our application offers an accessibility-focused solution, ensuring a smooth and enjoyable learning experience for all users.

Keywords :E-learning application, Accessible, Visually impaired individuals, Flutter, Django, MongoDB

الملخص

هدف هذا المشروع هو تطوير تطبيق تعليم إلكتروني لتعزيز تجربة التعلم للأشخاص ذوي الإعاقة البصرية. من خلال دمج تقنية flutter للواجهة الأمامية، و Django للواجهة الخلفية، و MongoDB لقاعدة البيانات، يقدم تطبيقنا حلاً موجهاً نحو الوصولية، مما يضمن تجربة تعلم سلسة وممتعة لجميع المستخدمين.

الكلمات المفتاحية : تطبيق تعلم إلكتروني، الأشخاص ذوي الإعاقة البصرية، flutter -

MongoDB - Django