# Introduction Générale



La facturation et l'informatisation des factures sont des éléments importants de la gestion administrative et financière d'une entreprise.

La reconnaissance des caractères (OCR) est une technologie clé pour l'informatisation des factures et l'automatisation du processus de traitement des factures. Elle permet de scanner les factures papier ou les documents numériques contenant du texte, et d'extraire automatiquement les données pertinentes à partir de ces documents à l'aide de l'intelligence artificielle.

Dans le cadre du processus traditionnel, la gestion des factures dans une entreprise se fait généralement manuellement en saisissant les informations de la facture dans les champs correspondants des formulaires, ce qui entraîne une perte de temps et un risque d'erreurs. Afin d'optimiser la gestion des factures et de simplifier les activités au sein de l'entreprise, notre objectif est de développer une application qui permettra le classement et la comptabilisation automatique des factures, en utilisant des techniques de reconnaissance des caractères (OCR) et d'intelligence artificielle. Nous prévoyons également de développer une application mobile qui permettra de scanner les factures, de les enregistrer dans la base de données de l'entreprise et de consulter la liste des factures téléchargées. Cette approche innovante permettra d'améliorer l'efficacité et la précision du processus de gestion des factures, tout en offrant une solution plus pratique et mobile pour les utilisateurs.

Le présent rapport constitue un compte rendu détaillé de notre activité au sein de l'entreprise pendant la période de stage. Il vise à fournir une vue d'ensemble claire et professionnelle de notre travail et mettre en lumière les activités que nous avons réalisées, les défis que nous avons relevés et les solutions que nous avons mises en place.

Pour ce faire, dans le premier chapitre, nous commençons par une étude préalable dans laquelle nous présentons l’organisme d’accueil et le projet ainsi que le choix de la méthodologie adoptée et les outils de travail.

Le deuxième chapitre présente une étude comparative des différentes solutions OCR qui ont été testées dans le but de déterminer la meilleure approche.

Nous enchaînons par un troisième chapitre intitulé la « planification du backlog produit », suivi de trois chapitres réservés aux trois releases, enfin nous clôturons par une conclusion générale dont laquelle nous mentionnons les différents atouts de ce projet.

# Chapitre I : Cadre du projet

## Introduction

Ce chapitre introductif est consacré à la présentation du contexte général du projet. Nous présenterons en premier lieu l’organisme d’accueil de notre projet. Ensuite, nous donnerons une vue d’ensemble de notre projet en expliquant les problèmes à résoudre et la solution proposée. Enfin, nous détaillerons la méthodologie que nous adoptons pour la réalisation du projet.

## Présentation de l’organisme d’accueil

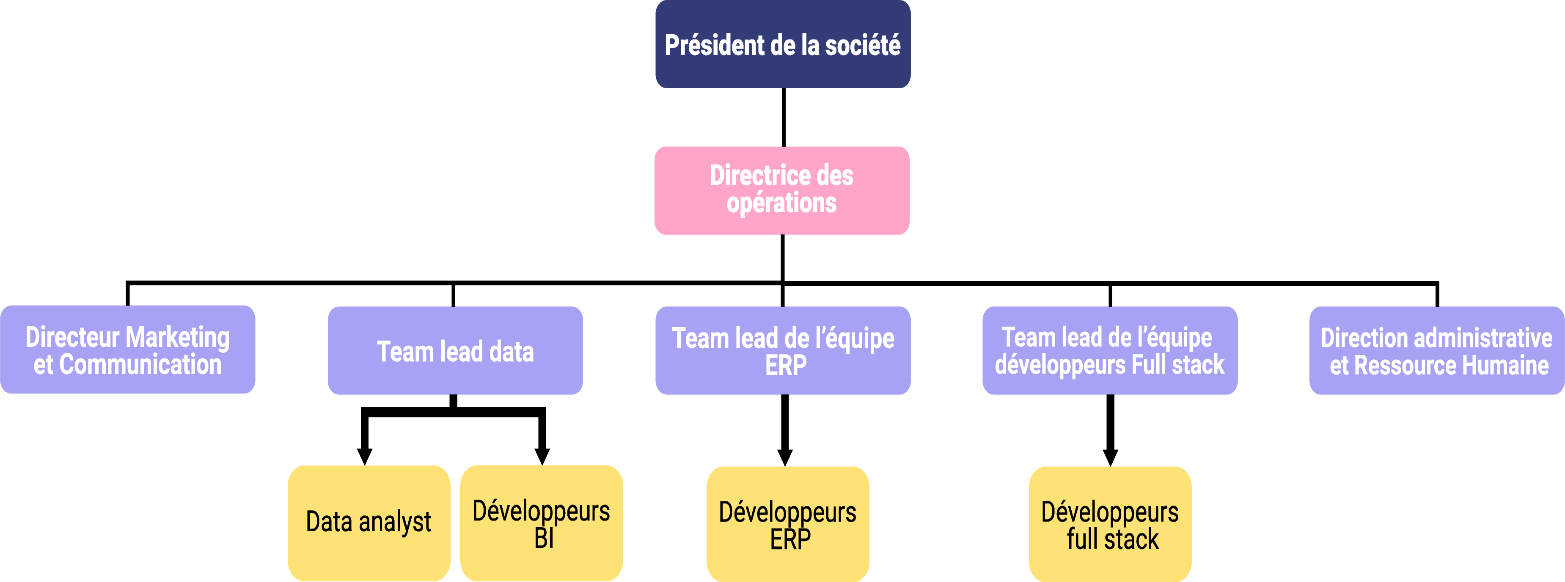
Metam Tunisie est une entreprise passionnée par Ia technologie. Une organisation dynamique résolument ouverte sur le monde avec des bureaux au Canada France et en Tunisie. Spécialisée dans Ia digitalisation des entreprises, implémentation des ERP, intégration ses systèmes d'information et le développement sur mesure.

Logo, company name

Description automatically generated

00

**Figure 1 : Logo de l’entreprise Metam Tech**



**Figure 2: Organigramme de la société Metam**

## Présentation du projet

### Étude de l’existant

Lors de discussions approfondies avec notre superviseur de la société, il a clairement souligné les difficultés rencontrées lors du traitement et de la classification des factures. Ce processus est complexe et prend beaucoup de temps, ce qui peut entraver l'efficacité globale de l'entreprise.

Un des problèmes clés est la comptabilisation manuelle des montants des factures, qui est effectuée par un gestionnaire dédié.

En conséquence, il est clair que pour améliorer l'efficacité et la précision de ce processus, il est crucial de mettre en place un système de traitement automatisé des factures. Cela permettra de minimiser les erreurs humaines et de libérer du temps pour d'autres tâches plus stratégiques.

### Critique de l’existant

Les problèmes constatés au niveau de la gestion des factures et leur comptabilisation de l’entreprise se résument dans les points suivants :

* La perte du temps.
* Un surcroît de travail et de stress pour la gestionnaire
* Des erreurs de saisie et de calcul ce qui peut entraver la précision financière de l'entreprise

### Solution proposée

Le développement d'une application mobile pour numériser les factures et les dépenses de la société pourrait être une stratégie clé pour améliorer les opérations financières. En utilisant la reconnaissance optique de caractères (OCR) ou des API OCR pour convertir les informations sur les factures en données numériques, le processus de traitement des factures pourrait être considérablement accéléré et automatisé.

De plus, en intégrant des algorithmes d'apprentissage automatique pour classer et comptabiliser les factures, l'application pourrait non seulement rationaliser le processus, mais aussi améliorer la précision des informations financières de l'entreprise. Les algorithmes pourraient apprendre à partir des données antérieures et les classer automatiquement en fonction des catégories appropriées, éliminant ainsi les erreurs de saisie manuelle et les retards associés. L'application mobile serait également pratique pour les employés, qui pourraient scanner les factures et les dépenses directement depuis leurs appareils mobiles, sans avoir à les envoyer manuellement à un gestionnaire dédié.

Cela permettrait non seulement d'améliorer la rapidité et l'efficacité du processus, mais aussi de libérer du temps pour les employés pour se concentrer sur des tâches plus stratégiques. En résumé, le développement d'une application mobile pour numériser et automatiser le traitement des factures et des dépenses pourrait être un pas en avant significatif pour améliorer les opérations financières de la société.

## Méthodologies et formalisme adopté

Une méthodologie de développement est un cadre de planification et de structuration du développement d'applications. Pour cela, il est nécessaire de traiter et de modéliser le système avant sa mise en œuvre pour bien comprendre son fonctionnement et assurer sa cohérence.

### Le langage de modélisation (UML)

Pour la réalisation d’un projet informatique, une méthodologie de modélisation et de conception est un procédé qui a pour objectif de permettre de formaliser les étapes préli- minaires du développement d’un système afin de rendre ce développement plus fidèle aux besoins du client. C’est pour cela qu’on a choisi **UML** ou **Unified Modeling Language** pour la conception de notre projet.

UML est un langage de modélisation graphique à base de pictogrammes conçu comme une méthode normalisée de visualisation dans les domaines du développement logiciel et en conception orientée objet.

Une image contenant texte, clipart

Description générée automatiquement

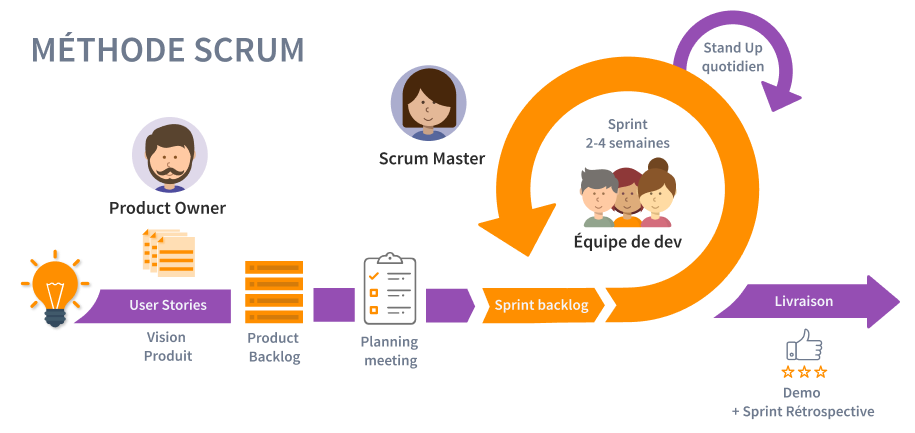
00

**Figure 3: Modèle UML**

### Méthodologie de travail

Nous avons été inspirés par SCRUM. Il représente l’approche agile la plus utilisée des approches agiles existantes et il est simple à comprendre.

Le principe de SCRUM est de diriger l'attention de l'équipe de développement vers un objectif unique qui sera complété en plusieurs étapes, chacune se déroulant sur une période de deux à quatre semaines, ces étapes sont appelées Sprints. Chaque sprint doit aboutir à la sortie d'un produit partiel appelé "incrément".



**Figure 4: Méthode SCRUM**

#### Pourquoi Scrum ?

Nous avons choisi Scrum en raison de ses points forts. Ça se résume à comme suit :

* Plus de flexibilité et de réactivité.
* Forte capacité d'adaptation aux changements dus à des temps d'itération courts.
* La combinaison entre théorie et pratique, très proche de la réalité

#### L’équipe Scrum

Comme pour toute méthode Agile, l’équipe Scrum est auto-organisée et pluridisciplinaire. Elle choisit la meilleure façon d’accomplir son travail et possède toutes les compétences nécessaires à l’accomplissement du projet. La flexibilité, la créativité et la productivité de l’équipe sont ainsi optimisées.[1]

L’équipe Scrum se compose en outre de :

* **Un Scrum Master :** c’est membre de l’équipe qui occupe le rôle de coach pour les équipes de développement. Le Scrum Master dans notre projet est notre encadreur de l’ISET Mme Chalouah Anissa.
* **Un Product Owner :** C'est le membre de l'équipe qui porte la vision du produit à réaliser, Il est aussi l’intermédiaire entre le client et l’équipe de travail, Il est la seule personne responsable de gérer le Product Backlog. Le Product Owner de notre projet est notre encadreur de la société Mr Ghouli Riadh
* **L’équipe de développement :** Il est composé de professionnels qui livrent à chaque sprint un incrément terminé et potentiellement livrable du produit. Seuls les membres de l’équipe de développement créent l’incrément. Ils sont auto- organisés polyvalents. L’équipe de développement de notre projet est Weda Mossab et Mohamed Nour Ben Salah.

****

**Figure 5 : L'équipe Scrum**

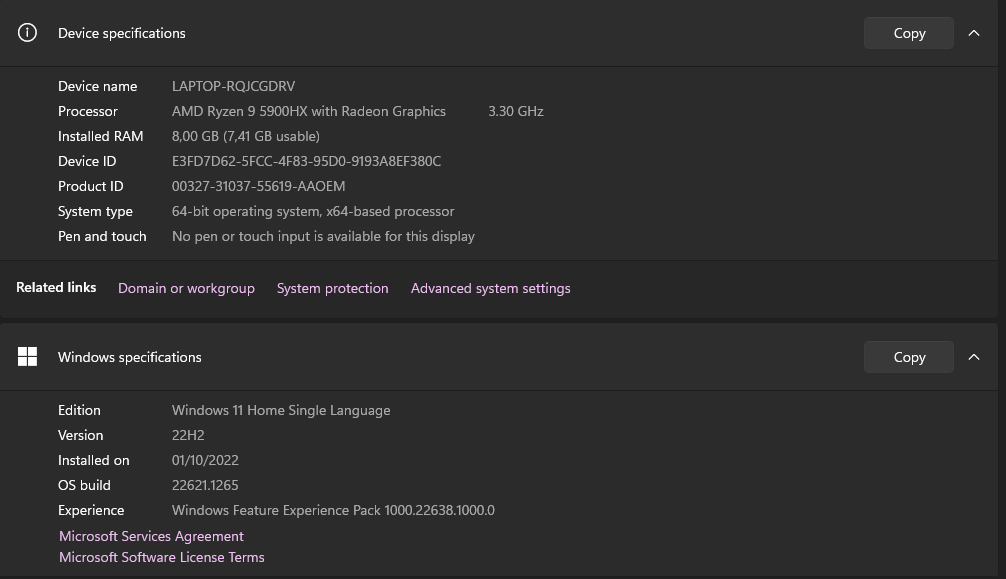
L’organisation d’un projet Scrum est rythmée par un ensemble de réunions définies avec précision et limitées dans le temps appelé :

* **Sprint Planning :** Le sprint planning est la première réunion de chaque sprint qui dure en général de deux semaines à un mois, il consiste à déterminer un sprint goal puis à planifier les fonctionnalités et user stories jugées prioritaires parmi toutes celles répertoriées dans le backlog produit. [2]
* **Sprint Review** : est la réunion destinée à évaluer les résultats obtenus par l’[équipe scrum](https://www.appvizer.fr/magazine/operations/gestion-de-projet/equipe-scrum) après un sprint. Elle permet d’analyser les progrès réalisés par le développement afin d’atteindre l’objectif fixé. [3]
* **Daily Meeting :** cette réunion est journalière de 15 minutes, elle est très importante. Cette réunion a pour but de faire un point sur la progression quotidienne du Sprint. Cette rencontre permet à l’équipe de synchroniser ses activités et de faire un plan pour les prochaines 24 heures. [4]
* **Rétrospective du Sprint** : C’est une réunion qui a lieu à la fin de chaque sprint. La rétrospective est interne à l’équipe Scrum. L’équipe se sert de la rétrospective pour passer en revue le Sprint terminé et déterminer ce qui a bien fonctionné et ce qu’il faut améliorer. [5]

## Technologies Et Outils De Travail

### Environnement matériel

Pour le développement de la solution nous avons utilisé les ordinateurs suivants dont leurs caractéristiques sont illustrées dans la figure ci-dessous :



## 

**Figure 6: Caractéristiques PC Mohamed Nour**

## Une image contenant texte Description générée automatiquement

**Figure 7: Caractéristiques PC Mossab**

### Environnement Logiciel

Nous allons énumérer au cours de cette partie les différents outils utilisés tout au long de ce projet pour l’étude et la mise en place de notre application.

## 

**Figure 8: Visual Studio logo**

**Visual Studio** est un environnement de développement intégré (IDE) extensible, complet et gratuit pour créer des applications modernes pour Android, iOS, Windows, ainsi que des applications web et des services cloud. [6]

## 

**Figure 9: Visual Studio Code logo**

**Visual Studio Code** est un éditeur de code open-source développé par Microsoft supportant un très grand nombre de langages grâce à des extensions. Il supporte l’autocomplétions, la coloration syntaxique, le débogage, et les commandes git. [7]

## 

**Figure 10: GitHub logo**

**GitHub** est un service web d'hébergement et de gestion de développement de logiciels, utilisant le logiciel de gestion de versions Git qui permet de stocker le code source d'un projet et de suivre l'historique complet de toutes les modifications apportées à ce code. GitHub est un site de partage de code, sur lequel on peut publier des projets dont le code est géré avec le système de gestion de version Git. [8]

## 

**Figure 11: Draw io logo**

**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. [9]



**Figure 12: Jira logo**

**Azure DevOps** est un logiciel créé par Microsoft, pour piloter des projets de développement d’application ou de logiciels. Entre autres fonctionnalités, Azure DevOps permet de planifier ses projets et suivre le travail de développement grâce à Git (logiciel de gestion de versions) [10]



**Figure 13: ASP.NET logo**

**ASP.NET** est un framework web open source, côté serveur, créé par Microsoft, qui fonctionne sous Windows. ASP.NET permet aux développeurs de créer des applications web, des services web et des sites web dynamiques. [11]



**Figure 14: C# logo**

**C#** est un langage de programmation moderne, orienté objet et de type sécurisé. C# permet aux développeurs de créer de nombreux types d'applications sécurisées et fiables qui s'exécutent dans .NET [12]

Une image contenant texte

Description générée automatiquement

**Figure 15: Microsoft Azure logo**

**Microsoft Azure** est une plateforme applicative qui comprend un service d’hébergement sur le cloud pour le stockage de données. Azure propose aussi des outils de développement pour la programmation d’applications et de machines virtuelles. L’utilisateur a la possibilité de concevoir des services web XML. [13]



**Figure 16: Blazor logo**

**Blazor** est un framework web open source développé par Microsoft et permettant de créer des applications web d’interface utilisateur web côté client basé sur le langage de programmation C# au lieu du commun JavaScript. Blazor peut s’exécuter directement dans le navigateur via WebAssembly. [14]

## 

**Figure 17: SQL Server logo**

**Microsoft SQL Server** est un système de gestion de base de données en langage SQL incorporant entre autres un SGBDR développé et commercialisé par la société Microsoft.

### Choix techniques

* **Choix du Framework ASP.NET :**

ASP.NET est considérer comme un choix de premier plan en raison de son support solide de Microsoft, de sa grande communauté, de son intégration avec d'autres technologies Microsoft, de sa haute performance, de sa sécurité robuste et de sa compatibilité avec plusieurs langages de programmation.

* **Choix du langage C# :**

C# est également un choix populaire pour le développement de solutions d'entreprise. En raison de son intégration avec d'autres technologies Microsoft telles que .NET Framework, Visual Studio, SQL Server et Azure, les développeurs peuvent créer des solutions d'entreprise puissantes et évolutives qui peuvent répondre aux besoins complexes des organisations.

* **Choix du SQL Server :**

Le Déploiement par un setup, mise en œuvre et adminsitration par des interfaces graphiques intuitives. Programmabilité. Gestion avancée de la sécurité en offrant deux modes d'authentification (Authentification Windows et Authentification S Server).

* **Choix du Blazor :**

Blazor est une option intéressante pour les développeurs cherchant à créer des applications web interactives et dynamiques rapidement, en utilisant des langages familiers et en bénéficiant de performances et de mesures de sécurité avancées.

## Architecture de l’application

Le MVC signifie Model-View-Controller est un modèle architectural qui sépare une application en trois composants logiques principaux : modèle, vue et le contrôleur. Chacun de ces composants est construit pour gérer des aspects de développement spécifiques d’une application. MVC est l’un des frameworks de développement Web standard les plus fréquemment utilisés dans l’industrie pour créer des projets extensibles et évolutifs.

* **Modèle :** Le modèle contient les données manipulées par le programme. Il assure la gestion de ces données et garantit leur intégrité.
  + *Dans notre cas ça correspondant aux objets manipulant les tables de la base de données SQL Server*
* **Vue :** composant graphique de l’interface qui permet de présenter les données du modèle à l’utilisateur.
  + *Dans notre cas elles sont définies en fichiers CSHTML*
* **Contrôleur :** composant responsable des prises de décision, gère la logique du code qui prend des décisions, il est l’intermédiaire entre le modèle et la vue.
* *Dans notre cas, notre contrôleur est les traitements gérés par ASP.NET*

## 

**Figure 18: Architecture MVC**

Une image contenant diagramme

Description générée automatiquement

**Figure 19: Diagramme de déploiement**

## Conclusion

Ce chapitre constitue une présentation générale du cadre du travail. Nous avons présenté en premier lieu la société par la suite nous nous sommes focalisés sur la problématique et le projet pour finir par le Framework du travail utilisé qui est un Framework agile «Scrum» et les outils du travail. Notre sujet sera expliqué plus en détail dans le chapitre suivant, en exposant les besoins et les objectifs à atteindre.

# Chapitre II : Etude de l’art

## Introduction

L'objectif de ce chapitre est de présenter une étude comparative des outils OCR disponibles sur le marché, en vue d'identifier le meilleur choix pour notre projet. Nous commencerons par une présentation générale de la reconnaissance optique de caractères (OCR).

Ensuite Nous nous concentrerons plus particulièrement sur Microsoft Azure OCR, en effectuant des tests sur ses outils pour évaluer leurs précision, rapidité et leurs facilité d'intégration.

Enfin, nous utiliserons les résultats de cette étude comparative pour justifier notre choix d'outil OCR pour notre projet.

## Microsoft Azure Cognitive Services

La solution Azure Cognitive Services correspond à des services d’intelligence artificielle (IA) basés sur le cloud qui aident les développeurs à intégrer une intelligence cognitive dans des applications sans connaissances ni compétences directes en IA ou en science des données. Ils sont disponibles via les API REST et les Kits de développement logiciel (SDK) de bibliothèque de client dans les langages de développement populaires. Azure Cognitive Services permet aux développeurs d’ajouter facilement des fonctionnalités cognitives dans leurs applications à l’aide de solutions cognitives capables de voir, entendre, parler et analyser.

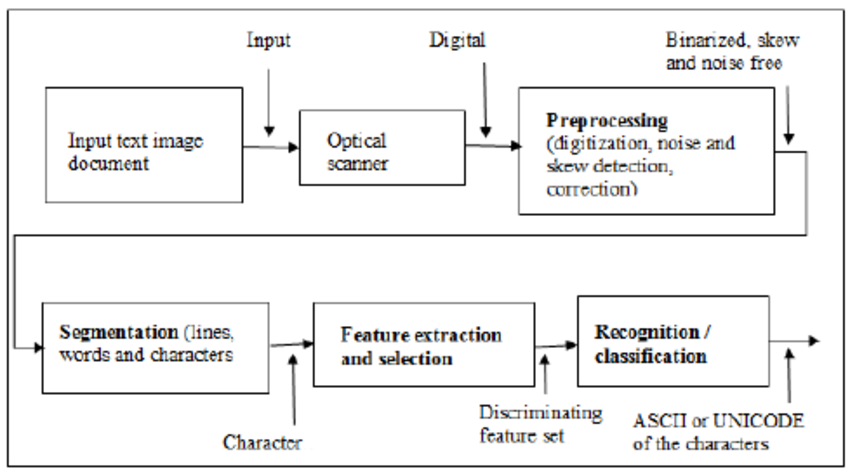
## La reconnaissance optique de caractères (OCR)

La reconnaissance optique de caractères (OCR) est une technologie qui permet aux ordinateurs de traiter des images ou des documents numérisés et d'extraire le texte qu'ils contiennent en utilisant des algorithmes avancés et des techniques d'apprentissage automatique. En utilisant des modèles de reconnaissance de caractères préalablement entraînés, l'OCR identifie et convertit les caractères, les mots et les phrases présents sur l'image en texte éditable.

L'OCR est largement utilisé dans de nombreux secteurs, y compris les services gouvernementaux, les entreprises, les bibliothèques et les archives, pour numériser et traiter des documents tels que des factures, des formulaires de demande, des documents historiques, etc. Cela peut améliorer l'efficacité, la productivité et l'expérience utilisateur, tout en réduisant les coûts et les erreurs liées à la saisie manuelle de données.

L'objectif de notre projet est l'extraction de données intelligente, qui reconnaît et traite le texte extrait par OCR pour produire des données pertinentes ultérieures.

Nous avons essayé les outils OCR pour atteindre notre objectif avant de nous fixer sur le plus réussi.



**Figure 20: Diagramme de reconnaissance optique de caractères**

## Etude Comparative entre les outils OCR Azure

### 3.1. Les outils OCR Microsoft Azure

Azure propose plusieurs services OCR (Reconnaissance Optique de Caractères) pour extraire du texte des images et des documents. On prendra l’exemple de :

1. **Azure Form Recognizer :** Ce service permet d'extraire des données structurées telles que des paires clé-valeur, des tableaux et du texte à partir de formulaires, de factures, de reçus et d'autres types de documents.
2. **Azure Computer Vision** qui comprit la capacité de reconnaître et de lire du texte dans des images, ainsi que la détection d'objets, la reconnaissance faciale, la reconnaissance optique de caractères (OCR) et d'autres capacités.
3. **Azure Cognitive Search** qui permet de rechercher et d'analyser du texte dans des images et des documents, y compris la reconnaissance optique de caractères.

### 3.2. Test et Analyse des outils (les solutions)

Le choix du tel type d’outil OCR nécessite le test de tous les outils pour avoir quel est le meilleur choix qui répondre à vos besoins de point de vue fiabilité, performance et sécurité. C’est pour cela on à prendre en considération ces facteurs et on a tester ces services (sauf le service Azure Cognitive Search car son rôle n’est pas compatible avec notre sujet) en réalisant les deux applications suivantes :

### Pourquoi Azure Form Recognizer ?

Après la phase du test on a revient à choisir le Form Recognizer comme service car il est plus adapté pour notre projet et plus avantageux par rapport au Computer Vision Service dans les points suivants :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Performance** | **Temps de réponse** | **Précision** | **Document** | **Non-document** |
| Azure Form Recognizer | X | X | X | X |  |
| Azure Computer Vision |  | X |  |  | X |

**Tableau 1: Comparaison entre les outils OCR**

* **Performance :** dans la phase test l’application celle de computer vision n’a pas extraire toutes les données alors que l’application Form Recognizer as fait ça.
* **Temps de réponse :** les deux applications ont le même temps de réponse qui est 6s
* **Précision :** d’après la figure 31 et la figure 21 en conclure qu’en terme précision l’outil Form Recognizer est plus mieux
* **Document :** Azure Form Recognizer est un service OCR spécialisé dans la reconnaissance de caractères optiques pour les documents tels que les factures, les reçus et les formulaires, en format PDF ou en image.
* **Non-document :** Azure Computer Vision OCR est un service de reconnaissance de caractères optiques (OCR) qui est conçu pour extraire du texte à partir d'images et de documents en texte brut, de manière générale.

## Consommation d’API From Recognizer

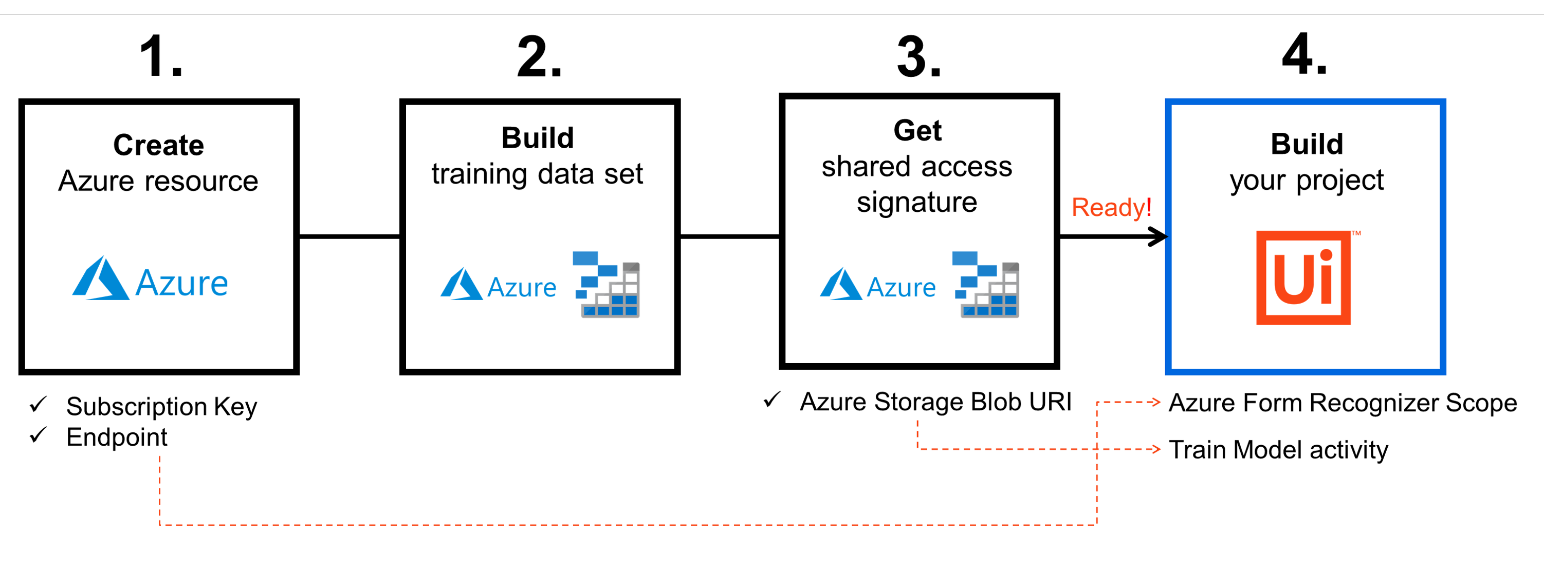
Pour établir une connexion et envoyer des requêtes à l’API Form Recognizer on doit fournir une clé (subscription key) qui valide votre abonnement au service Azure Form Recognizer et suivre les étapes suivantes :

En premier lieu, nous avons créé le ressource Azure Form Recognizer dans le portail Azure et on a vérifié les valeurs de Key1 et d’Endpoint

En deuxième lieu, on a collecté les factures dont nous allons l’entrainer (Dataset) et les télécharger dans le Blob container qu’on a déjà créé

La troisième étape consiste à obtenir une signature d'accès partagéà partir du Blob Container et la valeur de l'URI pour partager l'accès aux ressources avec l’application.

Et enfin, on a créé un projet Ui et on a installé le package UiPath.AzureFormRecognizer.Activities pour l’utiliser dans la création des workflows d'automatisation liés à Form Recognizer d'Azure.



**Figure 21: Les étapes de consommation de l'API Form Recognizer**

## 

**Figure 22: Architecture Microsoft Azure Form Recognizer**

## C’est que Neural Network ?

Un réseau de neurones est une technique d'apprentissage automatique inspirée du fonctionnement du cerveau humain. Ils sont utilisés pour analyser des données complexes et trouver des modèles dans ces données.

Un réseau de neurones est constitué de couches de neurones interconnectés. Chaque neurone est un nœud qui reçoit une entrée, effectue des calculs et envoie une sortie à d'autres neurones de la couche suivante.

Le processus d'apprentissage d'un réseau de neurones est accompli en ajustant les poids des connexions entre les neurones. Ce processus est généralement effectué à l'aide d'algorithmes d'optimisation tels que la descente de gradient. L'objectif est d'ajuster les pondérations pour minimiser une fonction de coût qui mesure l'écart entre la sortie prévue du réseau et la sortie réelle.

Un exemple d'utilisation des réseaux de neurones est la reconnaissance d'images. Dans ce cas, le réseau de neurones peut être formé sur un ensemble de données d'images avec des étiquettes correspondantes. Le réseau ajuste ensuite les poids de ses connexions pour minimiser l'erreur de prédiction sur les données d'apprentissage. Après la formation, le réseau peut être utilisé pour prédire les étiquettes de nouvelles images qu'il n'a jamais vues auparavant.

En résumé, un réseau de neurones est une technique d'apprentissage automatique qui utilise des neurones interconnectés pour analyser des données complexes et rechercher des modèles. Ils sont largement utilisés dans des domaines tels que la reconnaissance vocale, la reconnaissance d'images, la traduction automatique et la prévision de séries chronologiques.

## Neural Network Microsoft Azure

Azure Form Recognizer utilise un type de réseau neuronal récurrent (RNN) appelé réseau de mémoire longue à court terme (LSTM). Les LSTM sont un type de RNN conçu pour résoudre le problème de gradient de fuite qui peut survenir lors de la formation de RNN traditionnels sur de longues séquences de données.

Dans le contexte de l'analyse de documents et de la reconnaissance de formulaires, les LSTM sont bien adaptés au traitement de données séquentielles telles que du texte. Ils peuvent apprendre à capturer à distance les dépendances et les modèles dans le texte, ce qui est important pour une identification et une extraction précises des champs et des valeurs des formulaires.

Azure Form Recognizer combine les LSTM avec les réseaux de neurones convolutifs (CNN) pour créer des architectures d'apprentissage en profondeur afin de gérer la complexité et la variabilité des mises en page de formulaires et des modèles de texte. Microsoft n'a pas divulgué publiquement l'architecture spécifique de la combinaison RNN/CNN utilisée par Form Recognizer car elle est considérée comme une technologie propriétaire.

Dans l'ensemble, l'utilisation de LSTM dans Azure Form Recognizer permet à la plateforme d'identifier et d'extraire avec précision des données structurées à partir de documents non structurés, une tâche complexe qui nécessite un apprentissage en profondeur et un traitement du langage naturel.

## Conclusion

En conclusion, cette étude comparative nous a permis d'identifier le meilleur outil OCR pour notre projet, en fonction de critères tels que la précision, la performance et la rapidité. À travers cette analyse, nous avons exploré différentes bibliothèques et APIs OCR, en examinant leurs avantages et leurs limites.

# Chapitre III : Planification du Backlog Produit

## Introduction

Ce chapitre est essentiel dans la réalisation de tout projet, car il permet de déterminer les fonctionnalités de chaque utilisateur, ainsi que les différentes fonctionnalités qui peuvent exister dans notre projet.

## Identification des profils utilisateurs

Dans notre projet, les utilisateurs sont des entités externes au système de modélisation qui interagissent directement avec des personnes ou des appareils. Chaque utilisateur a un ensemble d'actions qui correspondent aux fonctions dont il a besoin. Notre projet comporte deux acteurs différents, chacun avec leur propre rôle correspondant.

L’acteur **Utilisateur** **:** Désigne l’employé de la société. Il a comme fonctionnalités :

* + Scanner des factures, peu importe leur format (PDF ou image), pour les traiter
  + Voir les documents déjà importés dans l'application.
  + Voir les résultats de l'analyse et leur niveau de confiance.
  + Modifier les résultats s'ils sont erronés.
  + Sauvegarder les résultats de la prédiction dans la base de données.
  + Sauvegarder les fichiers importés dans un stockage cloud.
  + Catégoriser les types de factures.
  + Personnaliser les modèles d'analyse pour mieux répondre à ses besoins spécifiques.
  + Exporter les résultats de l'analyse sous forme de rapport ou de fichier Excel.
  + Visualiser l'historique complet de chaque facture, y compris toutes les modifications et les commentaires.

L’acteur **Admin :** Il a comme fonctionnalités :

* Entraîner des modèles de factures.
* Tester les modèles.
* Modifier les modèles existants.
* Supprimer des modèles.
* Voir les modèles existants.
* Voir les fichiers importés sur le cloud et les utiliser pour l'entraînement des modèles.

## Les besoins non fonctionnels

Notre application doit contenir certaines contraintes indispensables pour son bon fonctionnement.

* **Maintenabilité :** Le code de l’application développé doit être lisible et compréhensible afin d’assurer son état extensible et évolutif par les autres développeurs de l’ERP.
* **La convivialité :** les interfaces de l’application doivent être simples et facile à manipuler pour les utilisateurs.
* **Fiabilité :** L’application doit être fonctionnelle en minimisant les erreurs d'extraction de données, ce qui peut entraîner des retards et des pertes de temps.
* **La Performance :** Notre application doit être capable de lire rapidement les informations des factures scannées et d'extraire avec précision les données.

## Les User stories

Une User story, ou récit utilisateur, sont des **descriptions d’exigences** pour toutes fonctionnalités ou “tâches” nécessaires au fonctionnement du produit ou du service en développement. Ils sont écrits par[**le Product Owner**](https://blog-gestion-de-projet.com/product-owner/) (Mr Ridha Ghouli) et aident à faciliter l'exécution et la compréhension des tâches par les équipes. [3]

Le Backlog du produit est constitué des US priorisé par rapport à leurs valeurs métier L’histoire précise essentiellement :

* **Qui** **:** pour qui on fait la fonctionnalité
* **Quoi :** qu’est-ce que l’utilisateur veut faire
* **Pourquoi :** pourquoi il veut faire ceci Elle est confirmée par des critères d’acceptation rédigés au même momentque l’histoire. En tant que, je veux pour atteindre

En tant que « **Qui** », je veux « **Quoi** », afin de « **Pourquoi »**

**Les User Stories de l’utilisateur**

En tant qu'utilisateur, je souhaite que l'application puisse numériser et extraire les données de mes reçus et factures, afin de ne pas avoir à les saisir manuellement.

.

.

En tant qu’utilisateur je veux sauvegarder les fichiers importés dans un stockage cloud.

.

En tant qu’utilisateur je veux scanner des factures pour les traiter.

En tant qu’utilisateur je veux

Consulter la liste des factures téléchargées.

.

En tant qu’utilisateur je veux catégoriser les types de factures.

.

En tant qu’utilisateur je veux sélectionner les modèles d'analyse pour mieux répondre à ses besoins spécifiques.

.

En tant qu’utilisateur je veux exporter les résultats de l'analyse sous forme de rapport ou de fichier Excel.

.

En tant qu’utilisateur je veux modifier les résultats s'ils sont erronés.

.

En tant qu’utilisateur je veux sauvegarder les résultats de la prédiction dans la base de données.

.

En tant qu’utilisateur je veux voir les résultats de l'analyse et leur niveau de confiance.

.

## 

## 

**Les User Stories de l’administrateur**

En tant qu’administrateur je veux voir les modèles existants.

En tant qu’administrateur je veux modifier les modèles existants.

En tant qu’administrateur je veux supprimer des modèles.

En tant qu’administrateur je veux gérer les fichiers importés.

En tant qu’administrateur je veux tester les modèles.

En tant qu’administrateur je veux entraîner des modèles de factures.

## Product Backlog

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nom du Sprint** | **Num** | **User Story** | **Description** | **Priorité** | **Dod** |
| **Sprint 1-**  **Etude de l'environnement** | TS 1 | Installer l'environnement du travail | En tant que Scrum team, nous voulons initialiser l'environnement de développement, afin de pouvoir commencer à travailler sur le projet et livrer efficacement les histoires utilisateur. | Haut | **L'équipe a installé et configuré les logiciels et outils nécessaires, y compris mais sans s'y limiter l'IDE, le système de contrôle de version, les outils de construction et les bibliothèques de développement.** |
| TS 2 | [Maitriser le développement avec les frameworks blazor, ASP.NET et Microsoft Azure](http://asp.net/) | En tant qu'équipe Scrum, nous voulons maîtriser l'utilisation d'Azure Cognitive Services et développer une application web Blazor qui s'y intègre. Notre équipe va acquérir les connaissances et les compétences nécessaires pour développer un module à l'aide du framework ASP.NET Blazor . | Haut | **L'équipe a acquis une connaissance approfondie d'Azure Cognitive Services et du cadre ASP.NET Blazor.** |
| **Sprint 2 –**  **Choix de la solution optimale** | TS 3 | Etude de la solution optimale | En tant qu'équipe de scrum, je veux étudier les différentes techniques pour connaître la solution optimale pour détecter les articles d'une facture. | Haut | **L'équipe a mené des recherches et des analyses approfondies sur diverses techniques de détection des sections d'une facture, y compris, mais sans s'y limiter, machine learning, rule-based systems, and deep learning.** |
| **Sprint 3 –**  **Gestion et numérisation des factures** | TS 4 | Etude de détection des factures et développement de solution intégrée | En tant que Scrum team, nous voulons étudier différentes techniques pour détecter les factures, collecter un ensemble de données diversifié et représentatif pour les entrainements, et développer une solution qui intègre la technique sélectionnée pour détecter les sections sur une facture. | Haut | **L'équipe a collecté un ensemble de données diversifié et représentatif pour les entrainements, y compris des factures de différents formats et langues, et a nettoyé et préparé les données pour l'apprentissage automatique.** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | US 5 | Numérisation des factures pour traitement | En tant qu’utilisateur je veux scanner des factures pour les traiter. | Haut | **Les factures doivent être scannées et enregistrées dans un format numérique.** |
| US 6 | Extraction des données à partir de factures et de reçus | En tant qu'utilisateur, je souhaite que l'application puisse numériser et extraire les données de mes reçus et factures, afin de ne pas avoir à les saisir manuellement. | Haut | **L'application dispose d'une intégration OCR et/ou API pour l'extraction de données, avec une précision et une gestion des erreurs appropriées** |
| US 7 | Catégorisation des factures dans l'application | En tant qu'utilisateur, je souhaite que l'application puisse catégoriser mes factures | Moyenne | **L'application dispose d'algorithmes de catégorisation basés sur l'apprentissage automatique, avec une précision appropriée et des mécanismes de rétroaction des utilisateurs** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | US 8 | Stockage et organisation des données dans l'application | En tant qu'utilisateur, je souhaite que l'application puisse stocker et organiser mes données financières, afin de pouvoir y accéder ultérieurement. | Haut | **L'application dispose d'algorithmes de catégorisation basés sur l'apprentissage automatique, avec une précision appropriée et des mécanismes de rétroaction des utilisateurs** |
| **Sprint 4 - Analyse et Entraînement de Modèles de Factures** | US 9 | Visualisation des résultats d'analyse avec niveau de confiance | En tant qu'utilisateur, j'aimerais pouvoir visualiser les résultats de l'analyse accompagnés de leur niveau de confiance pour prendre des décisions éclairées en fonction de ces informations | Haut | **L'utilisateur peut voir les résultats d'analyse de manière claire et concise, accompagnés de leur niveau de confiance** |
| US 10 | Modification des résultats d'analyse | En tant qu'utilisateur, je souhaite avoir la possibilité de modifier les résultats de l'analyse si je découvre des erreurs ou des incohérences dans les données présentées | Haut | **L'utilisateur a la possibilité de modifier les résultats d'analyse s'il découvre des erreurs ou des incohérences dans les données présentées** |
| US 11 | Entraînement de modèles de factures | En tant qu'admin, je souhaite avoir la capacité d'entraîner des modèles de factures pour améliorer la précision et la pertinence des résultats de l'analyse. | Haut | **L'administrateur a la capacité d'entraîner des modèles de factures personnalisés pour améliorer la précision et la pertinence des résultats d'analyse** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | US 12 | Test des modèles de factures | En tant qu'admin, je souhaite pouvoir tester les modèles de factures pour évaluer leur précision et leur fiabilité | Haut | **L'administrateur peut tester les modèles de factures pour évaluer leur précision et leur fiabilité** |
| **Sprint 5 –**  **Gestion des modèles et des données d'entraînement.** | US 13 | Fusion des modèles existants | En tant qu'administrateur, je veux composer les modèles existants pour faire la fusion entre eux. | Moyenne | **L'administrateur est en mesure de fusionner les modèles existants** |
| US 14 | Suppression sélective des modèles | En tant qu'administrateur, je souhaite pouvoir supprimer des modèles de manière sélective. | Haut | **L'administrateur peut supprimer des modèles de manière sélective** |
| US 15 | Visualisation des modèles existants | En tant qu'administrateur, je veux visualiser une liste des modèles existants pour avoir une vue d'ensemble de l'état de l'application. | Haut | **L'administrateur peut visualiser une liste complète des modèles existants, y compris leur nom, leur date de création et leur état (actif ou inactif)** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | US 16 | Gestion des fichiers importés sur le cloud | En tant qu'utilisateur, je veux gérer les fichiers importés sur le cloud en les visualisant, en les supprimant si nécessaire et en les utilisant pour l'entraînement des modèles, afin de pouvoir gérer efficacement mes données d'entraînement. | Haut | **L'administrateur peut visualiser les fichiers importés sur le cloud, les supprimer si nécessaire et les utiliser pour l'entraînement des modèles** |
| **Sprint 6 - Optimiser l'expérience des utilisateurs** | US 17 | Ajouter des notes aux modèles de facture | En tant qu'utilisateur, je souhaite avoir la possibilité de noter le modèle de facture utilisé pour mes factures, afin de pouvoir les différencier facilement. | Moyenne | **L'utilisateur doit pouvoir ajouter des notes ou des commentaires personnalisés pour chaque modèle de facture utilisé** |
| US 18 | Annoter les factures avec des commentaires | En tant qu'utilisateur, je souhaite pouvoir ajouter des commentaires ou des notes aux factures afin de pouvoir les annoter et les organiser de manière plus efficace. | Faible | **L'utilisateur doit pouvoir ajouter des commentaires ou des notes personnalisées à chaque facture** |
| US 19 | Recevoir des notifications par e-mail | En tant qu'utilisateur, je souhaite pouvoir recevoir des notifications par e-mail lorsqu'une facture est en attente de traitement ou a été payée, afin d'être informé rapidement. | Faible | **L'utilisateur doit avoir la possibilité de recevoir des notifications par e-mail pour être informé des factures en attente de traitement ou payées** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | Options d'analyse sélective pour une interprétation plus efficace des données | En tant qu'utilisateur, je veux pouvoir choisir des modèles d'analyse adaptés à mes besoins spécifiques afin que mon analyse de données produise des résultats plus précis et plus pertinents. | Haut | **L'utilisateur est en mesure de sélectionner et d'utiliser avec succès des modèles d'analyse adaptés à ses besoins spécifiques.** |

**Tableau 2: Product Backlog**

Nous avons choisi de répartir les Sprints que nous avons identifiés en trois Release se présentant comme suit :

## Une image contenant diagramme Description générée automatiquement

## 

**Figure 23: Organisation des releases**

## Conclusion

En conclusion, après la spécification des différentes fonctionnalités qui existent dans notre projet, nous pouvant commencer par la réalisation du Release l‘objet du chapitre suivant.

# Chapitre IV : Release 1

## Introduction

Ce chapitre présente en détail les sprints de premier release commençant par le backlog, suivi des différents diagrammes élaborés pour la réalisation du premier release, incluant le diagramme des cas d'utilisation, le diagramme d'analyse, le diagramme des classes et les diagrammes des séquences. Enfin, les interfaces graphiques de chaque sprint.

## Organisation des sprints

Une image contenant texte, carte de visite, capture d’écran

Description générée automatiquement

**Figure 24: Release 1**

## 2. Sprint 1 « Collecte et catégorisation des données »

### 2.1. Sprint Goal

Ce sprint vise à faire étudier une étude approfondie de l'environnement de travail et de tester les différentes solutions envisageables afin de trouver l’outil le plus adapté pour garantir la réussite de notre projet.

### 2.2. Sprint Backlog

Le tableau ci-dessous présente le Backlog de notre preer sprint :

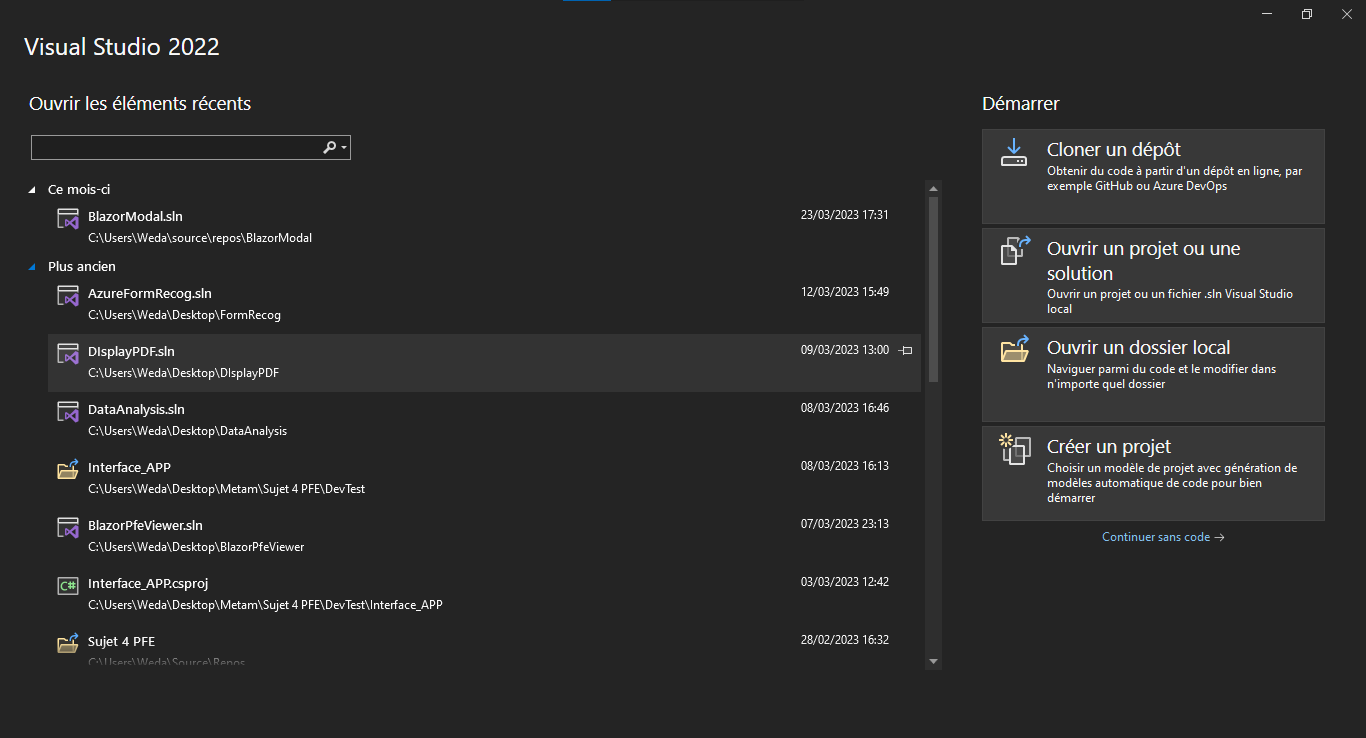
|  |  |  |  |
| --- | --- | --- | --- |
| **User Stories** | | **Les taches** | **Période** |
| **Sprint 1** | **Installer l'environnement du travail** | Installer Visual Studio 2022 | 1j |
| Installer le .NET Framework |
| Microsoft SQL Server Management Studio 18 |
| **Maitriser le développement avec les frameworks blazor, ASP.NET et Microsoft Azure** | Recherche sur : • la notion des composants • render-fragment • les call-back functions • OCR - API Azure | 13j |
|  |
|  |
|  |
|  |
| Préparer une présentation sur l’outil OCR   Microsoft Azure et les comparer |  |
| Développer des mini-applications :  • Upload File • Render Fragment |  |
|  |
|  |
| Faire l'entrainement des quelques factures avec Microsoft Azure afin de comprendre mieux comment fonctionne ce dernier |  |

**Tableau 3 : Sprint Backlog du sprint 1**

### 2.3. Implémentation du sprint

Cette partie contient des captures écrans pour nos taches réalisés dans ce sprint :

1. **La première Technical Story :**

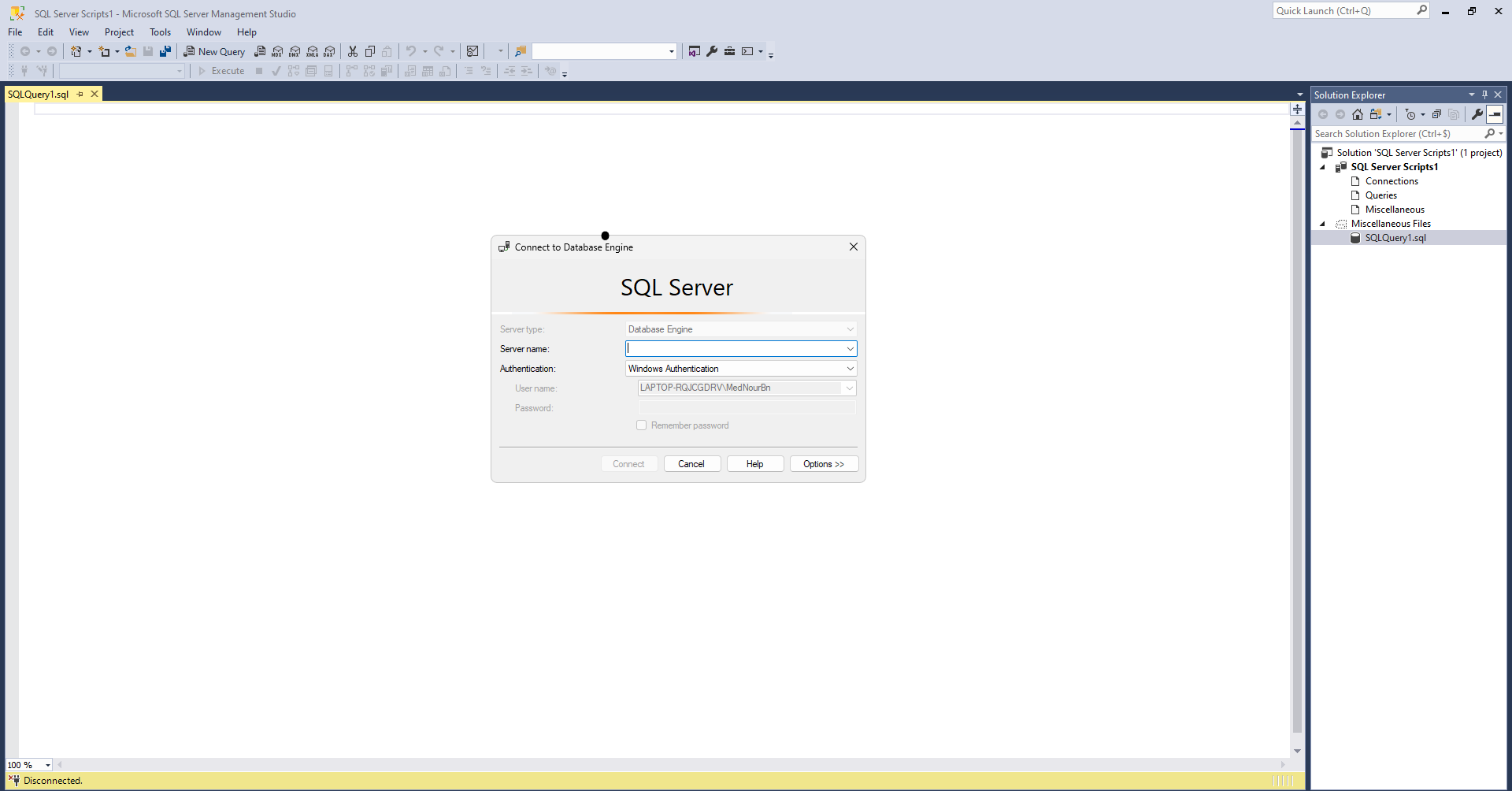


**Figure 25 : Installé Visual Studio 2022**

Une image contenant texte

Description générée automatiquement

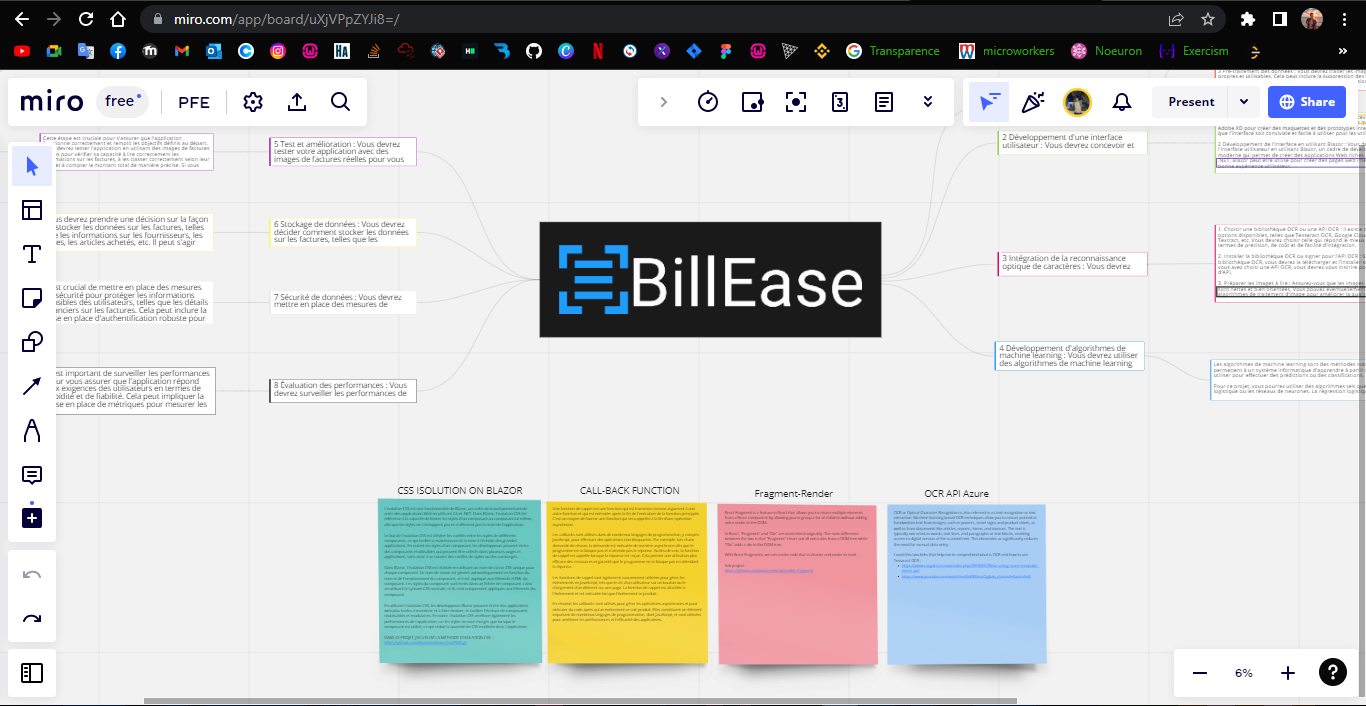
**Figure 26 : installé .NET**



**Figure 27 : Microsoft SQL Server Management Studio 18**

1. **La deuxième Technical Story**

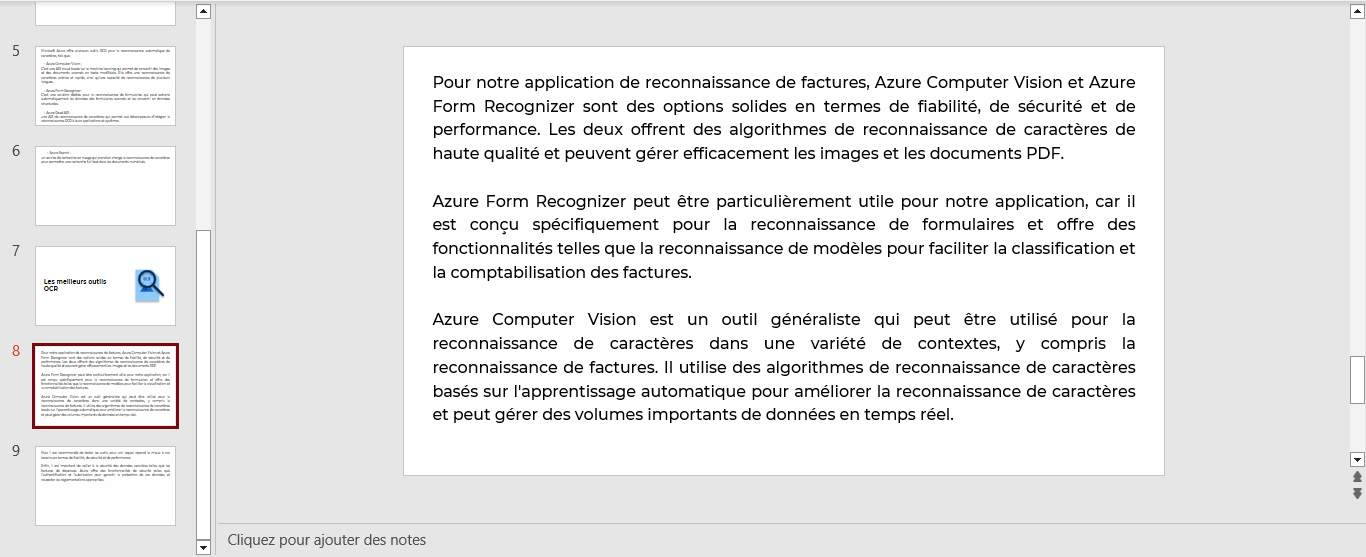
Cette partie renferme une carte mentale [3] élaborée pour notre projet, des recherches sur les différentes notions requises, une présentation détaillée des outils OCR Azure, ainsi que des captures d'écran des mini-applications réalisées au cours de cette phase et l'entraînement des quelques factures.



**Figure 28: Les recherches demandées**



**Figure 29 : Présentation sur les outils OCR Microsoft Azure 1**



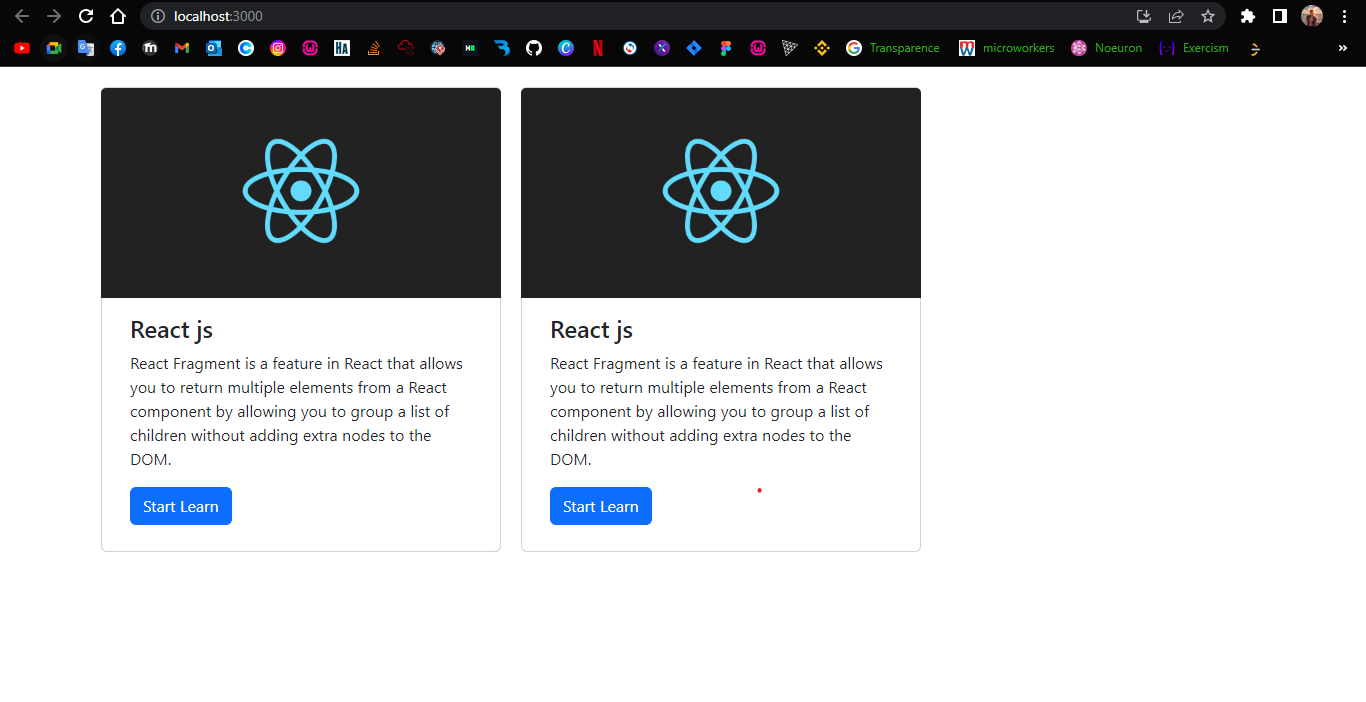
**Figure 30 : Présentation sur les outils OCR Microsoft Azure 2**

Une image contenant texte

Description générée automatiquement

**Figure 31 : L'application "Upload File"**

* **Les fragments** nous permettent de grouper une liste d'enfants sans ajouter de nœud supplémentaire au DOM. [4]



**Figure 32 : l'application "Render-Fragment"**

Une image contenant texte

Description générée automatiquement

**Figure 33 : Entrainement des factures**

### 2.4. Sprint Review

Après avoir terminé ce sprint notre environnement de travail devient prés afin de pouvoir achever les sprints suivants avec succès.

### 2.5. Sprint Rétrospective

|  |  |
| --- | --- |
| **Ce qui a bien fonctionné** | **Ce qui n’est pas bien passé** |
| - Nous avons installé l'’environnement avec succès.  - Les taches des recherches, le développement des mini-application et l’entraînement des modèles. | - Difficulté au cours de l’entrainement du modèle |

**Tableau 4:Plan d'amélioration du sprint 1**

## 3. Sprint 2 « Choix de la solution optimale »

### 3.1. Sprint Goal

L'objectif de ce sprint est d'identifier la meilleure solution possible pour notre projet en se concentrant sur les besoins spécifiques et les contraintes qui y sont associées.

### 3.2. Sprint Backlog

|  |  |  |  |
| --- | --- | --- | --- |
| **User Stories** | | **Les taches** | **Période** |
| **Sprint 2** | **Etude de la solution optimale** | **• Azure form Recognizer (1ère solution)** Nous avons testé l'extraction du texte des factures à travers une application qui nous permettons de créer notre modèle form recognizer, l'entrainer et faire la phase labelling  **• Azure Computer Vision (2ème solution)** Nous avons développé une application qui consomme l'api de Azure Computer Vision et testé cet outil avec des factures de type image et PDF | 14 j |

**Tableau 5: Sprint Backlog du sprint 2**

### 3.3. Implémentation du sprint

La réalisation de ce sprint néssecite une comparaison entre les outils pour déterminer la solution optimale :

1. **Computer Vision App (Première Solution) :**

C’est une application qui consomme l’API de computer vision et qui sert à entrer un fichier et qui retourne une description de ce dernier. Cette méthode est plus générale en ce sens qu’il ne fournit pas une contextualisation aussi robuste des paires clé/valeur que l’outil de reconnaissance des formules. Le service offre également une fonctionnalité d’IA de niveau supérieur pour le traitement des images et des vidéos afin d’identifier les personnes/célébrités, les points de repère et les objets communs (entre autres)

* **Les étapes :**
* Créer une ressource de service de vision Azure
* Générer une clé d'API
* Créer un projet Blazor
* Ajouter le package NuGet Microsoft.Azure.CognitiveServices.Vision.ComputerVision
* Ajouter les clés d'API dans le fichier de configuration
* Consommer l'API Azure Computer Vision

Une image contenant texte

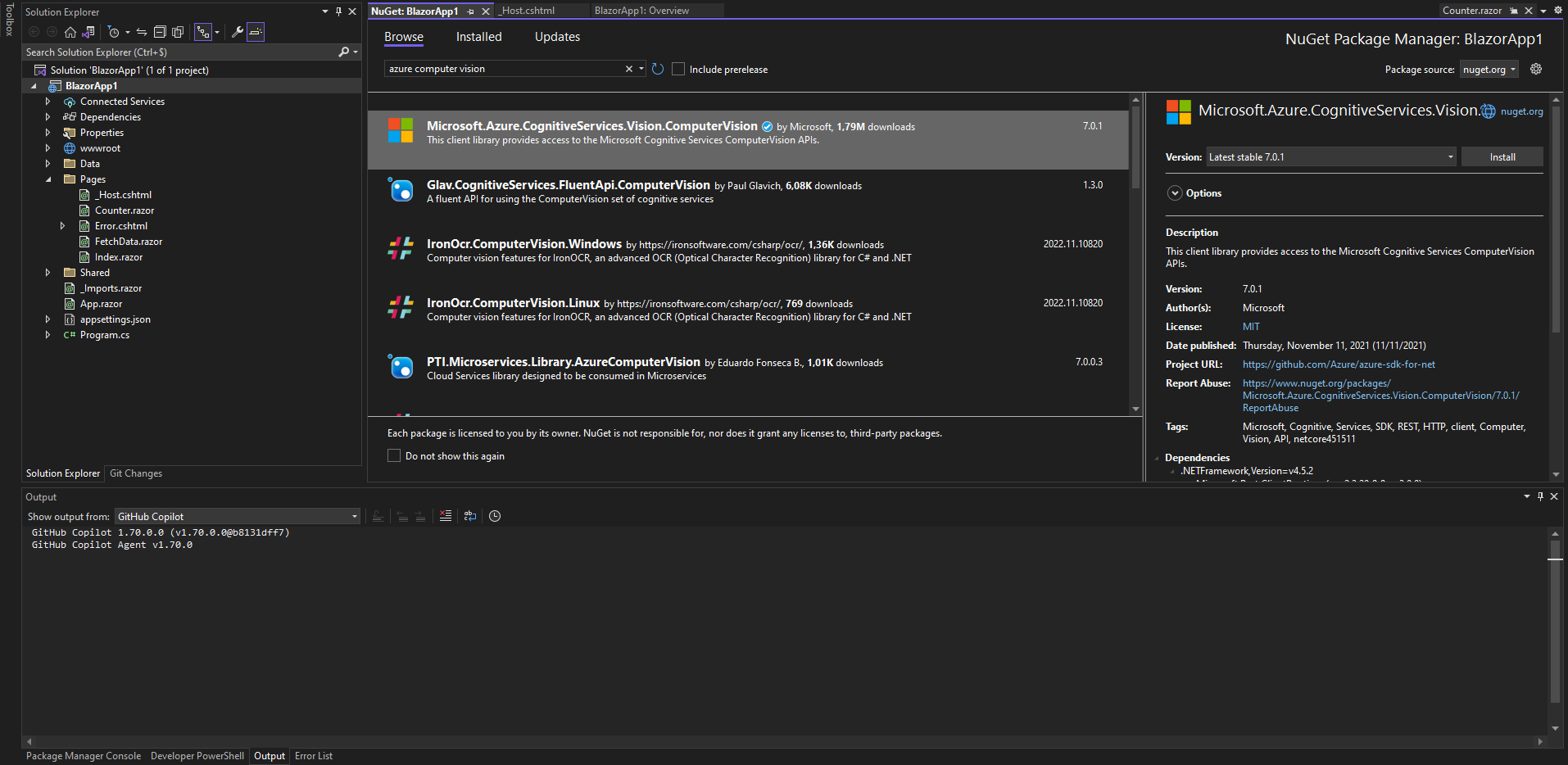
Description générée automatiquement

**Figure 34: Création du service computer vision**

Une image contenant texte

Description générée automatiquement

**Figure 35: Génération de la clé d'API**



**Figure 36: Importation du package Computer Vision**

Une image contenant texte

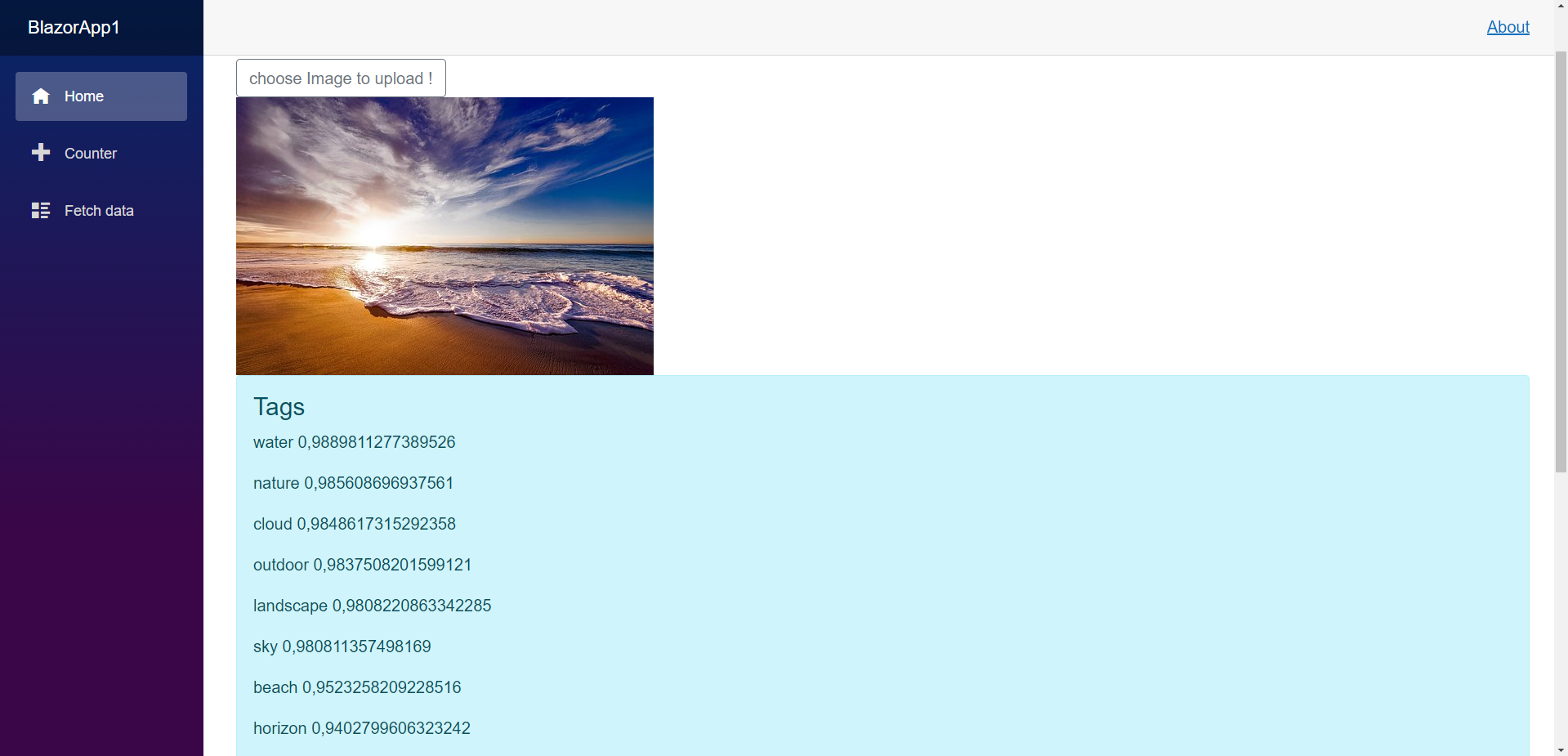
Description générée automatiquement

**Figure 37: Consommation de l’API**

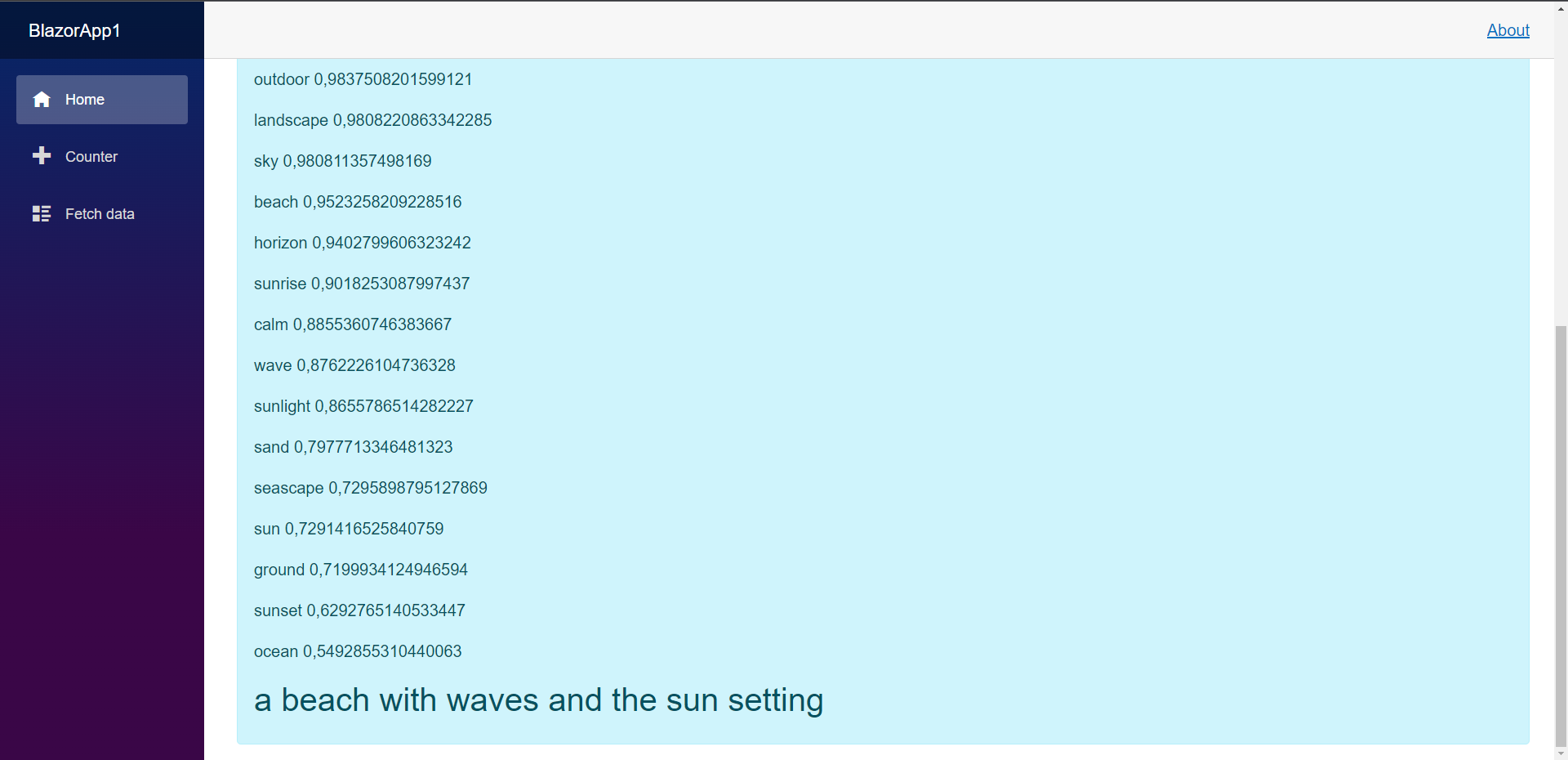
Une image contenant texte

Description générée automatiquement

**Figure 38: Extraction des données d'un document**



**Figure 39: Extraction des données d'une image 1**



**Figure 40: Extraction des données d'une image 2**

1. **Form Recognizer App (Deuxième Solution) :**

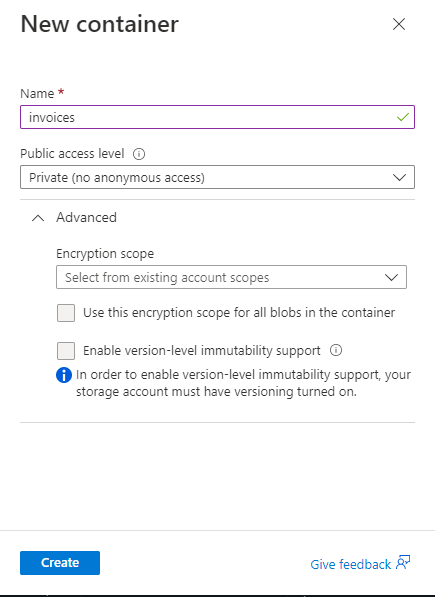
C’est une méthode de test en ligne dans laquelle on a créé un modèle IA qui sert à détecter les informations d’un type de facture (on a pris la facture Amazon comme exemple) et l’entrainé avec 5 factures pour qu’il soit plus précis à la reconnaissance des caractères puis on a testé ce modèle sur une facture de même type.

* **Les étapes :**
* Création d’un Storage Account pour stocker les fichiers à analyser
* Création d’un conteneur de stockage de fichiers utilisés « Blob Container »
* Importer les factures à analyser dans le container
* Faire la connexion entre le Customer et le blob container
* Créer un projet dans lequel on insère le clé API et le URI de service Form Recognizer
* Indiquer les données à extraire
* Entrainer le modelé
* Analyser la facture

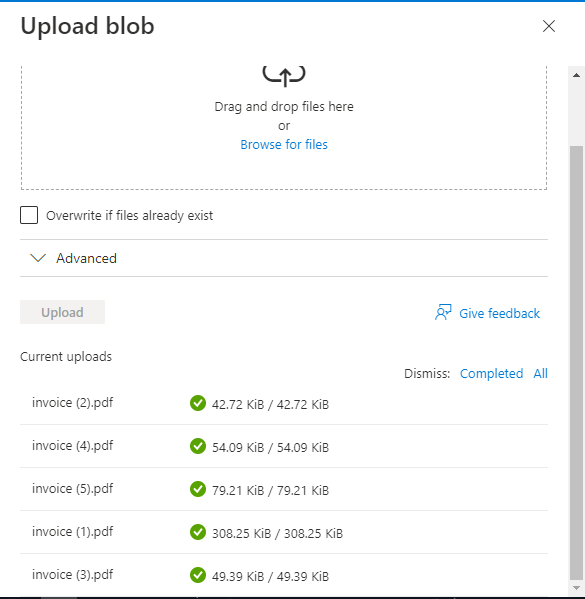
**Une image contenant texte

Description générée automatiquement**

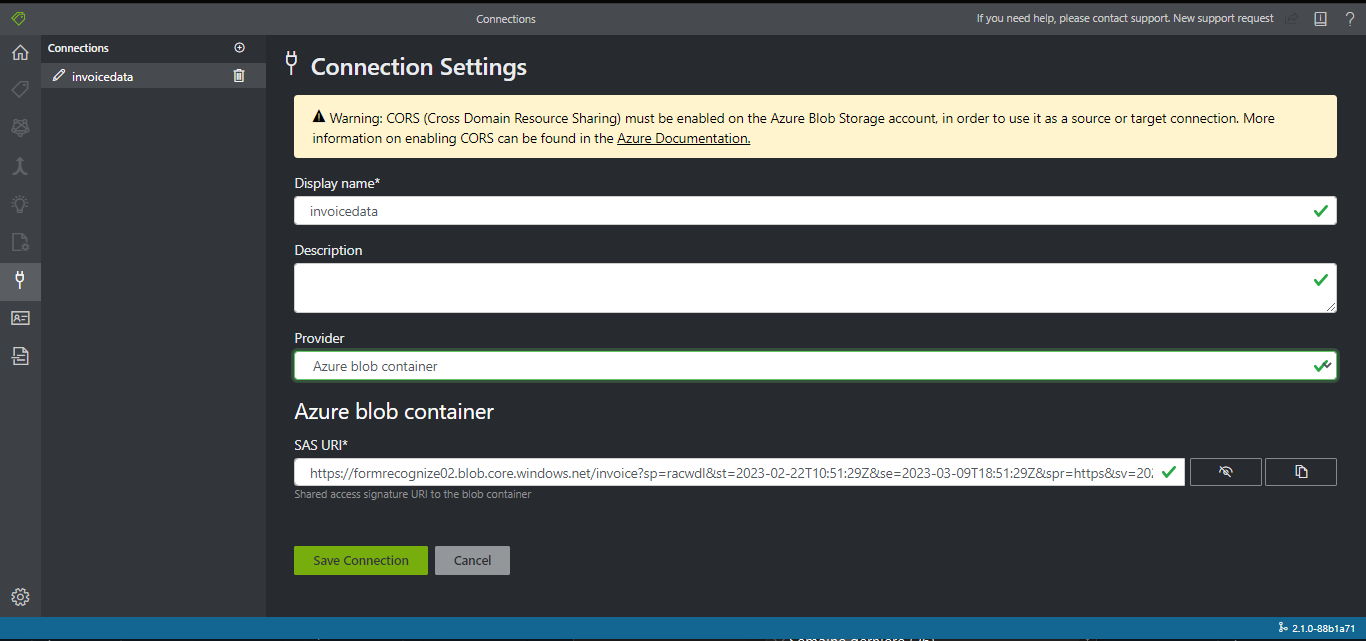
**Figure 41: Création d’un Storage Account**

****

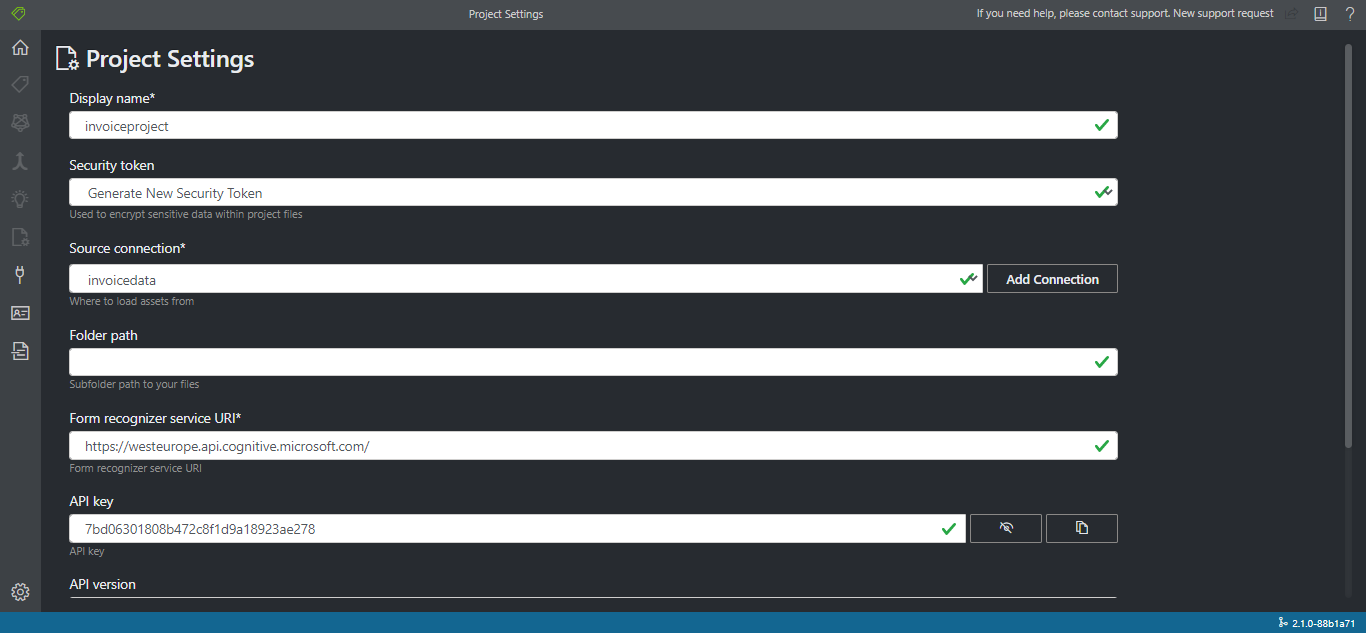
**Figure 42: Création d’un conteneur de stockage**

****

**Figure 43: Importer les factures**

****

**Figure 44: la connexion entre le customer et le blob container**

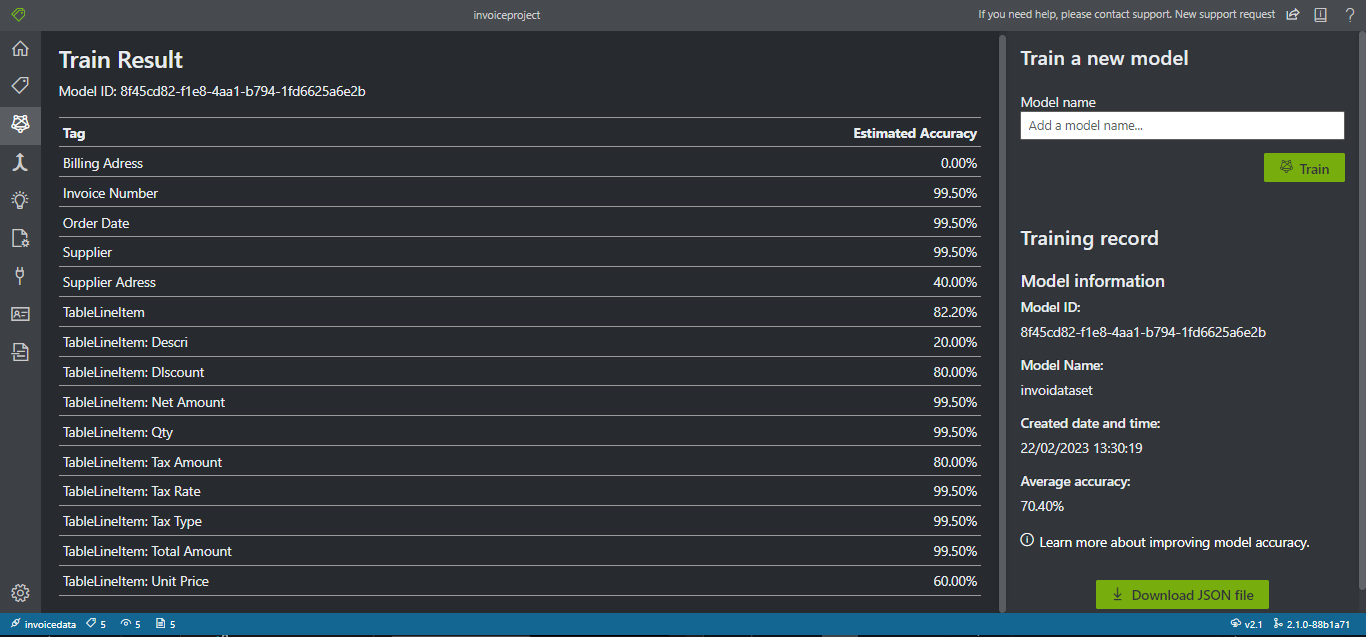
****

**Figure 45: Création du projet**

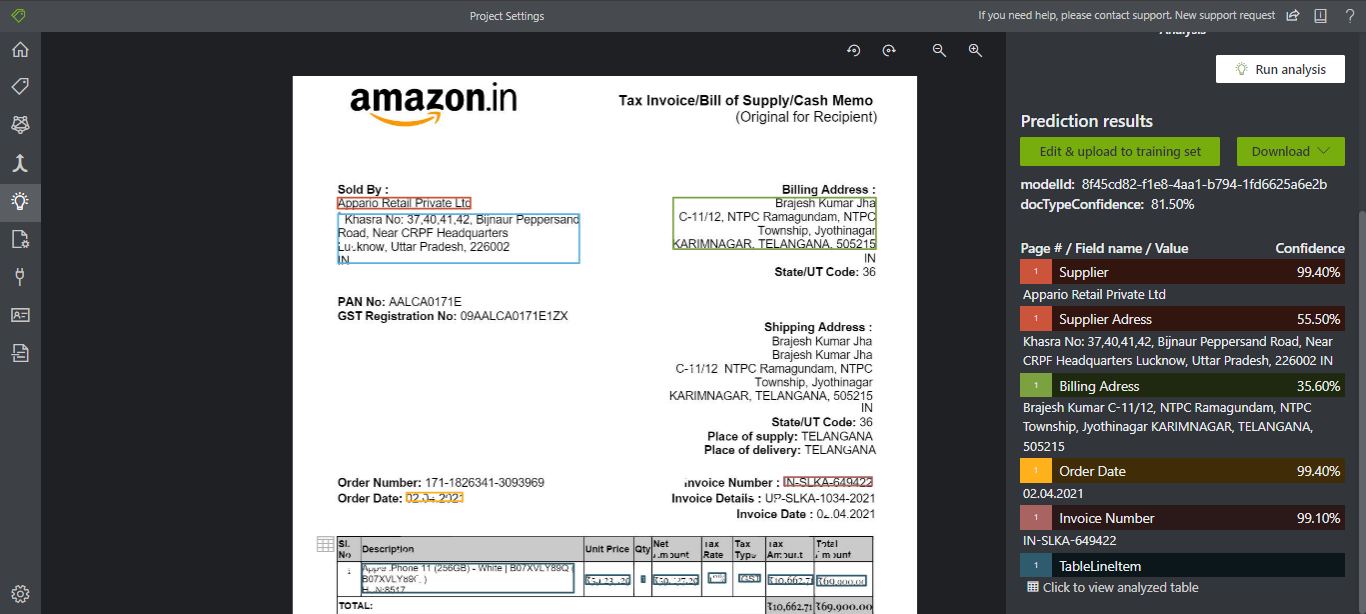
Une image contenant texte

Description générée automatiquement

**Figure 46: Indication des champs**



**Figure 47: Entrainement du modelé**



**Figure 48: Test avec Form Recognizer**

**Azure Cognitive Search (Solution non compatible) :**

A propos cet outils azure il se concentre sur la recherche du texte et qui est non compatible avec le but de notre application pour cette raison là on n’a pas testé cet outil.

### 3.4. Sprint Review

Nous avons tenu la réunion de Sprint Review à distance, au cours de laquelle nous avons expliqué à notre encadrant pour quoi on a choisi la solution du Form Recognizer afin de présenter les avantages et les inconvénients de chaque application.

### 3.5. Sprint Rétrospective

|  |  |
| --- | --- |
| **Ce qui a bien fonctionné** | **Ce qui n’est pas bien passé** |
| - La consommation de l’API Azure Computer Vision | - la consommation de L’API Azure Form Recognizer (on n’a pas trouver une API gratuit) |

**Tableau 6:Plan d'amélioration du sprint 2**

## Conclusion

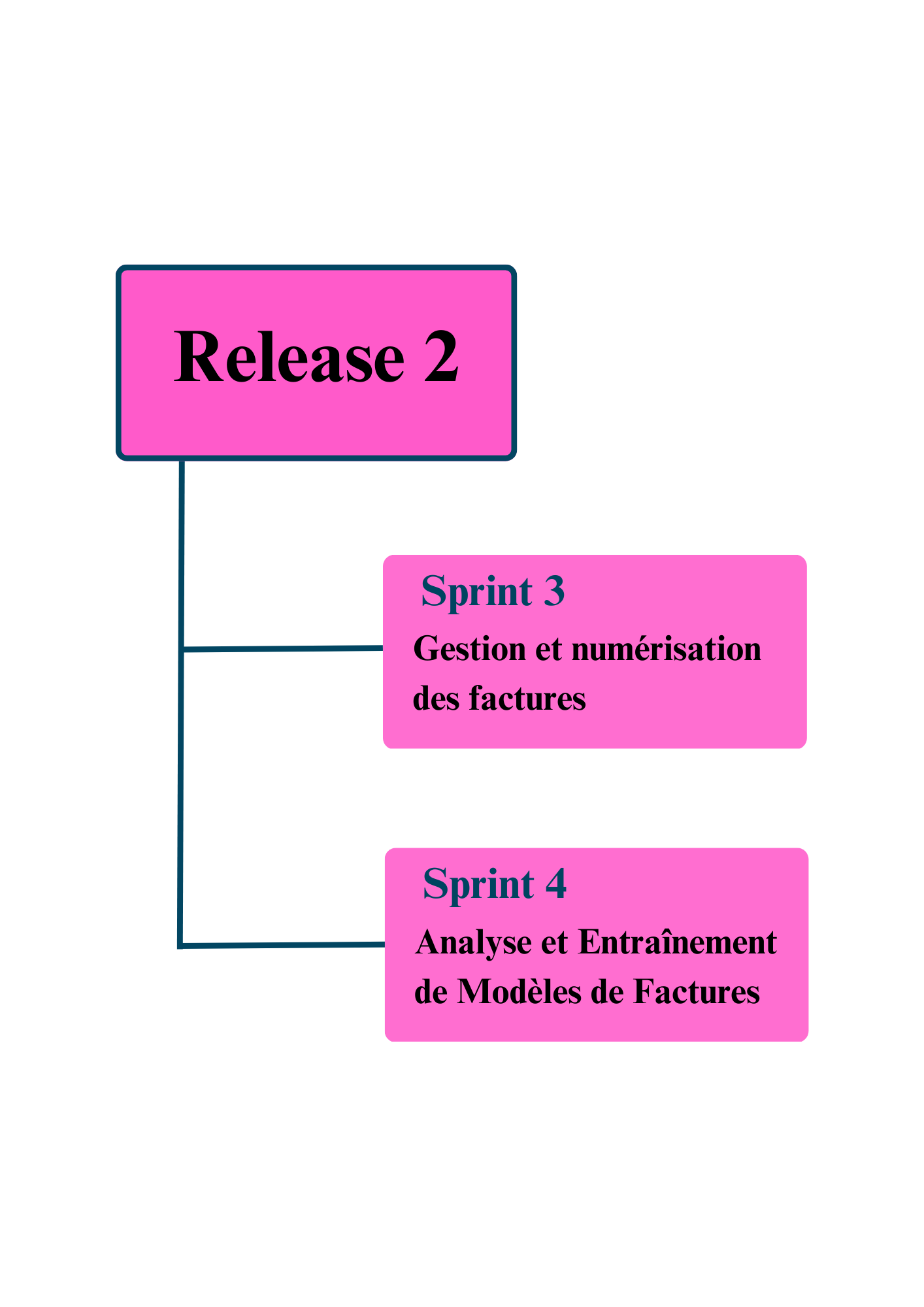
Ce chapitre a été dédié à l'acquisition de compétences sur les nouveaux outils et frameworks de travail, ainsi qu'à l'étude approfondie de la solution qui sera mise en œuvre dans notre projet.

# Chapitre V : Release 2

## Introduction

Une fois que nous avons achevé le premier Release de notre application, nous sommes prêts à entreprendre les tâches nécessaires pour créer la deuxième Release, qui comportera le Product backlog, les différents diagrammes et ses interfaces graphiques.

## Organisation des sprints



**Figure 49: Release 2**

## Sprint 3 « Gestion et numérisation des factures »

### Sprint Goal

L’objectif de ce sprint consiste à combiner entre la détection des factures, la numérisation, l'extraction des données, la catégorisation et le stockage des informations financières dans l'application.

### Sprint Backlog

|  |  |  |  |
| --- | --- | --- | --- |
| **User Stories** | | **Les taches** | **Période** |
| **Sprint 3** | **Etude de détection des factures et développement de solution intégrée** | Collecte du dataset (factures) | 2j |
| Recherche dans les différentes techniques de détection avec Form recognizer : • Prebuilt-Model • Template • Neural Network |
|  |
| **Numérisation des factures pour traitement** | Création de l'interface du scannage | 4j |  |
| Implémenter la fonction upload file qui nous permettons de télécharger un fichier de type image ou pdf |  |
| Implémenter la fonction show file qui permet d'afficher le fichier |  |
| **Extraction des données à partir de factures et de reçus** | Consommation de L'API Form Recognizer qui nous permet d'extraire les données nécessaires du facture | 4j |  |
| Afficher les données dans un tableau |  |
| **Stockage et organisation des données dans l'application** | Stocker les factures dans le cloud et leurs données dans la base des données local | 4j |  |

**Tableau 7: Sprint Backlog du sprint 3**

### Implémentation du sprint 3

#### Expression des besoins du sprint

Pour la mise en évidence des besoins de notre application nous avons utilisées les diagrammes de cas d’utilisations.

Les diagrammes de cas d'utilisation sont des diagrammes UML utilisés pour une représentation du comportement fonctionnel d'un système logiciel Nous avons complété ce diagramme par une description textuelle afin de détailler les interactions entre les acteurs et le système.

**Diagramme de cas d’utilisation détaillé de « Gestion et numérisation des factures »**

Une image contenant diagramme

Description générée automatiquement

**Figure 50: Diagramme de cas d’utilisation de « Gestion et numérisation des factures »**

**Description textuelle « Scanner une facture »**

|  |  |
| --- | --- |
| **Cas d’utilisation** | **L'utilisateur demande de Scanner une facture** |
| **Acteur** | Utilisateur |
| **Pré-condition** | L’utilisateur est authentifié |
| **Post-condition** | La facture est scannée |
| **Scénario Nominal** | 1 - L’utilisateur accèd à la page de scannage |
| 2 - L’utilisateur clique sur le bouton "UploadFile" ou "Scan File" |
| 3 - L’utilisateur séléctionne la facture qu'il va scanner |
| 4 - Le system détecte la facture et l'afficher |
| **Scénario Alternatif** | 4.1 - Le système n’affiche pas la facture téléchargée |

**Tableau 8 : Description textuelle « Scanner une facture »**

**Description textuelle « Extraire et stocker les données des factures »**

|  |  |
| --- | --- |
| **Cas d’utilisation** | **L'utilisateur demande de stocker les données des factures** |
| **Acteur** | Utilisateur |
| **Pré-condition** | L’utilisateur est authentifié / Facture importé ou scanné |
| **Post-condition** | Les données sont extraites et stockées dans la base des données |
| **Scénario Nominal** | 1 - L’utilisateur importe une facture |
| 2- L’utilisateur clique sur le bouton "Run Analysis" |
| 3 - Le système affiche les données de la facture |
| 4 - L’utilisateur clique sur le bouton "Save" |
| **Scénario Alternatif** | 3.1 - Le système n’affiche pas les données de la facture |

**Tableau 9 : Description textuelle « Extraire les données des factures »**

#### Analyse de sprint 3

**Modèle du domaine :**

Dans cette partie nous avons présenté la structure interne de l’application via le modèle du domaine de la partie gérer les factures.

Une image contenant diagramme

Description générée automatiquement

**Figure 51 : Diagramme de classe de la « Gestion et numérisation des factures »**

**Diagramme de classe participante :**

Une classe d’analyse définit ou fournit rarement une interface en termes d’opérations et de signature. Son comportement est défini par des responsabilités à un niveau plus élevé, moins formel

* **Diagramme de classes participante de « Scanner une facture » :**

Une image contenant diagramme

Description générée automatiquement

**Figure 52 : Diagramme de classes participante de « Scanner une facture »**

* **Diagramme de classe participante de « Extraire les données des factures » :**

Une image contenant diagramme

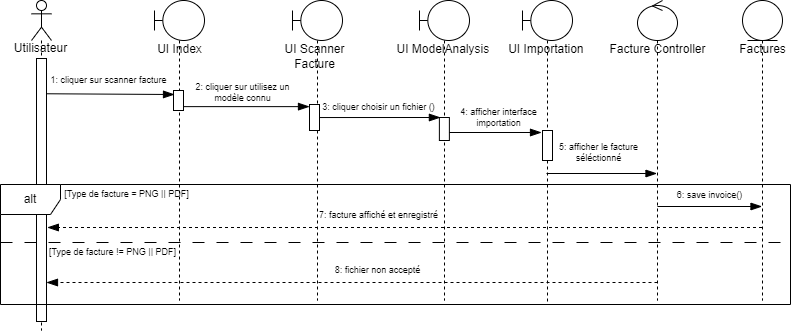
Description générée automatiquement

**Figure 53 : Diagramme de classe participante de « Extraire les données des factures »**

#### Conception du sprint 3

Le diagramme de séquence permet de montrer les interactions d'objets dans le cadre d'un scénario d'un Diagramme des cas d'utilisation

* **Diagrammes de séquence de conception de « Scanner une facture »**



**Figure 54: Diagramme de séquence "Scanner facture"**