

TABLE DES MATIÈRES

Liste des figures	6
Liste des tableaux	8
Liste des abréviations	9
Introduction générale	11
Chapitre 1 : Etude des systèmes existants	15
Introduction	15
1.1: Définition d'un ROC	15
1.2: Avantages de la reconnaissance optique de caractères	15
1.3: Définition d'une plateforme d'impression de mémoire en ligne	16
1.4: Étude de l'existant	16
1.4.1: La plate forme Scribbr.fr	16
1.4.2: La plate forme Corep.fr	17
1.4.3: La plate forme Nuance.com	18
1.5: Étude du besoin sur le marché	19
1.5.1: Utilisation de l'outil « Google trends »	19
1.5.2: La communauté Facebook	21
Conclusion	22
Chapitre 2 : Analyse et conception du système Traixto	24
Introduction	24
2.1: Spécification des besoins du système	24
2.1.1: Identification des acteurs	24
2.1.2: Besoins fonctionnels	25
2.1.3: Besoins non fonctionnels	27
2.2: Modélisation des besoins fonctionnels	27
2.2.1: Diagramme de cas d'utilisation	28
2.2.2: Diagramme de paquetage	30
2.2.3: Diagramme de séquences	32
2.2.3.1: Diagramme de séquences « Créer une commande »	32

2.2.3.1.1: Description textuelle	33
2.2.3.1.2: Diagramme de séquence système « Créer une commande »	34
2.2.3.2: Diagramme de séquence« convertir une image au texte »	36
2.2.3.1.1: Description textuelle	36
2.2.3.1.2: Diagramme de séquence système «Convertir une image au texte»	38
2.2.4: Diagramme de classe	39
Conclusion	39
Chapitre 3 : Implémentation du système	41
Introduction	41
3.1: Structure d'application web moderne	41
3.1.1: Application web moderne	41
3.1.2: Une application Client-serveur	41
3.1.3: Les services web RESTful	42
3.1.4: Une application web monopage (Single page application « SPA »)	43
3.1.4.1: Les avantages d'utiliser une application application web monopage	44
3.1.5: Authentification par JSON Web Tokens « JWT »	44
3.1.5.1: JSON Web Token	44
3.1.5.2: La structure d'un jeton JWT	44
3.1.5.3: L'authentification basée sur le JWT	46
3.1.6: Google Cloud Platform (GCP)	47
3.1.6.1: Cloud Storage	48
3.1.6.2: Cloud Vision API	48
3.2: Développement javascript full-stack (MEAN Stack)	48
3.2.1: Javascript coté client	48
3.2.1.1: Javascript (ES6)	48
3.2.1.2: TypeScript	50
3.2.1.3: Les Framework Javascript côté client	52
3.2.1.3.4: Angular	53
3.2.2: Javascript côté serveur	54
3.2.2.1: Nodejs	54
3.2.2.1.1: Le fonctionnement	54
3.2.2.1.2: Quand utiliser Node.js?	56
3.2.2.2: Node Package Manager (NPM)	56
3.2.2.3: Développer des applications élégantes Node.js en utilisant NestJS	58
3.2.3: MongoDB	59
3.2.3.1: Mongoose	59

3.2.3.2: Mlab (Database-as-a-Service)	60
3.3: L'architecture de l'application	61
3.4: Maquettes du système	62
3.5: L'implémentation de « Traixto »	65
3.5.1: Installation d'Angular	65
3.5.2: Les composants principaux d'Angular	65
3.5.3: API RESTful de Traixto	72
3.5.4: Installation de NestJs	77
3.5.5: Les principaux éléments constitutifs de NestJs	77
Conclusion	79
Conclusion générale	80
Références bibliographiques	82
Résumé	85

Liste des figures

Figure 1.1: page d'accueil Scribbr

Figure 1.2: page d'accueil Corep

Figure 1.3: page d'accueil Nuance

Figure 1.4: image représentant le taux de recherche de la phrase « photo to pdf »

dans le monde

Figure 1.5: image représentant le taux de recherche de la phrase « impression de mémoire » dans le monde

Figure 1.6: une capture d'écran de Facebook qui représente une demande d'opération d'OCR

Figure 2.1: diagramme de cas d'utilisation de notre système « Traixto »

Figure 2.2: diagramme de paquetage de notre système « Traixto »

Figure 2.3: Diagramme de séquence pour l'action « Créer une commande »

Figure 2.4: Diagramme de séquence pour l'action « Convertir une image au texte »

Figure 2.5: diagramme de classe de notre système « Traixto »

Figure 3.1: la structure d'un jeton JWT

Figure 3.2: L'authentification basée sur le JWT

Figure 3.3: la relation entre le Javascript (ES5/ES6) et le Typescript

Figure 3.4: la structure d'une Application Angular

Figure 3.5: Le fonctionnement d'une application Nodejs (mono-thread) par rapport à une application multi-thread.

Figure 3.6: la communication entre une application nodejs et MongoDB à l'aide de mongoose.

Figure 3.7: L'architecture de l'application

Figure 3.8: Création d'une nouvelle commande (1)

Figure 3.9: Création d'une nouvelle commande (2)

Figure 3.10: Création d'une nouvelle commande (3)

Figure 3.11: Création d'une nouvelle commande (4)

Figure 3.12: la page de la liste des commandes

Figure 3.13: la page de connexion

Figure 3.14: créer un nouveau projet Angular avec la CLI Angular

Figure 3.15: l'architecture d'un module Angular (DeliveriesModule)

Figure 3.16: l'architecture d'un Composant (Component) Angular
(AdminCreateRoleComponent)

Figure 3.17: l'architecture d'un Service Angular (DeliveryService)

Figure 3.18: l'architecture d'un Route Guard Angular (AuthGuardService)

Figure 3.19: l'architecture d'un intercepteur HTTP Angular
(TokenInterceptorService)

Figure 3.20: l'architecture d'un Pipe personnalisé Angular
(HttpErrorHandlingPipe)

Figure 3.21: l'architecture d'un contrôleur (Controller) NestJs
(DeliveriesController)

Figure 3.22: schéma mongoose 'USER'

Liste des tableaux

Tableau 2.1: Description textuelle du cas d'utilisation « Créer une commande »

Tableau 2.2: Description textuelle du cas d'utilisation « Convertir une image au texte »

Tableau 3.1: Documentation sur la API UsersMicroService

Tableau 3.2: Documentation sur la API DeliveriesMicroservice

Tableau 3.3: Documentation sur la API AuthMicroService

Tableau 3.4: Documentation sur la API UsersRolesMicroService

Tableau 3.5: Documentation sur la API FinishesMicroService

Tableau 3.6: Documentation sur la API DeliveriesEmsMicroService

Liste des abréviations

ROC: reconnaissance optique de caractères

OCR: Optical Character Recognition

IA: intelligence artificielle

JWT: JSON Web Token

API: application programming interface

GCP: Google Cloud Platform

REST: Representational State Transfer

VPS: virtual private server

UML: Unified Modeling Language

NPM: node package manager

HTML: Hyper Text Markup Language

CSS: Cascading Style Sheets

SCSS: Sassy CSS

Sass: Syntactically awesome style sheets

Less: Leaner Style Sheets

MEAN: MongoDB-Expressjs-Angular-Nodejs

HTTP: Hypertext Transfer Protocol

HTTPS: Hypertext Transfer Protocol Secure

URL: Uniform Resource Locator

SPA: Single Page Application

UX: User experience

RFC: Request for Comments

HMAC: hash-based message authentication code

RSA: Rivest–Shamir–Adleman

ECDSA: Elliptic Curve Digital Signature Algorithm

ES6: ECMAScript 6

ES5: ECMAScript

CLI: Command Line Interface

SGBD: système de gestion de base de données

SQL: Structured Query Language

NoSQL: Not Only SQL

BDD: base de données

JSON: JavaScript Object Notation

BSON: Binary JSON

ODM: Object Data Modeling

DBaaS: Database-as-a-Service

Introduction générale

“Every once in a while, a new technology, an old problem, and a big idea turn into an innovation.”

by Dean Kamen

Je pense que la citation de Dean Kamen ci-dessus résume parfaitement notre relation actuelle avec la technologie, et souligne son importance ainsi que son étendu remarquable sur notre vie. Actuellement, la technologie touche pratiquement tous les domaines à savoir celle de santé, de sécurité, du divertissement et même les emplois les plus traditionnels, ceci revient aux différents avantages fournis à notre société.

La technologie automatise les tâches longues et fastidieuses et améliore également la productivité tout en garantissant une place à l'innovation et le développement personnel.

La première chose à faire est donc de trouver un problème précis et intéressant à résoudre pour notre marché cible, qui, dans notre cas, est le marché algérien. Nous devons ensuite penser d'une part, à une solution pratique, intelligente et facile à utiliser pour nos utilisateurs potentiels, et pourvoir nous adapter au marché algérien d'autre part. Ceci revient exactement à ce que notre client de «Traixto service» a pensé en faisant partie des premiers Solution algérienne de reconnaissance optique de caractères et d'impression documentaire en ligne.

A ce stade tout est conforme, mais qu'en est-il du paiement électronique qui n'est pas encore disponible en Algérie? Pour bien répondre à cette question, nous devons examiner

les plateformes algériennes qui prouvent actuellement une réelle progression même sans paiement électronique, telles que «Tem Tem», «Yassir», «Jumia» et s'adaptent à la situation actuelle.

Contexte

Dans le cadre de la formation de Génie Logiciels (GL) à l'université Abou Bekr Belkaid, nous avons eu l'occasion d'effectuer notre projet de fin d'études pour l'obtention du diplôme de Master en Informatique au sein de l'entreprise « Media Startup ».

Ce stage s'est déroulé du 3 février 2019 au 20 juin, à Alger sur le thème :

«Développement d'une plateforme de reconnaissance de caractères et d'impression documentaire».

Notre client pour ce projet est «Traixto Service», une entreprise d'impression de documents académique établie à Alger et possédant déjà un nombre de clients que nous considérons comme des utilisateurs potentiels de la plateforme. Notre client pense également que la technologie améliorera sûrement ses bénéfices lui permettant ainsi de se développer et progresser comme jamais auparavant.

Problématique

L'un des problèmes les plus courants dont souffrent les étudiants en dernière année est l'impression de leur mémoire, qui peut se compliquer assez rapidement avec toutes les règles, le format papier... Etc, en particulier quand le temps devient insuffisant et compté, ceci m'est Personnellement arrivé et certainement aussi pour d'autres clients de «Traixto Service». Par ailleurs, Un autre problème commun peut survenir à d'autres personnes de différents domaines d'où la nécessité d'extraire et de réutiliser des textes d'anciennes sources imprimées, telles que des images, en les convertissant en formulaires numériques modifiables comme PDF, ce qui permet de gagner du temps et d'accroître la productivité.

Objectifs

Notre objectif ultime est de créer une application Web moderne, Responsive et prête pour la production, capable de s'adapter et de fonctionner correctement dans des conditions d'utilisation réelles. La plateforme sera un croisement entre une reconnaissance optique de caractères et une impression de documents en ligne qui , lors des premières étapes, se concentrera d'avantage sur l'impression des mémoires, en se basant sur le fait que la plupart des clients de «Traixto Service» sont des étudiants. Il sera donc tout à fait logique de commencer modestement avec les élèves comme cible démographique, puis de s'étendre à une cible plus grande.

Notre application fournira à ses utilisateurs les services suivants:

- Un espace utilisateur où chacun de nos utilisateurs peut interagir et accéder à nos services en toute sécurité et librement.
- Possibilité d'extraire le texte d'une photo à l'aide des dernières technologies d'OCR.
- Impression de mémoires en ligne professionnelle et personnalisée avec livraison à domicile.

Plan du mémoire

Ce mémoire aura 03 chapitres:

- **Chapitre 1:** dans celui-ci, nous allons essayer de valider notre problème par rapport aux besoins réels des clients, soit en utilisant la communauté Facebook, soit en utilisant des outils tels que Google Trends.
- **Chapitre 2:** analyse approfondie des besoins de nos utilisateurs et conception de l'architecture globale à l'aide de diagrammes UML
- **Chapitre 3:** dans ce chapitre, nous aborderons la partie technique de notre application et comment les choses fonctionnent réellement en arrière-plan.

Chapitre 1

Etude des systèmes existants

Chapitre 1 : Etude des systèmes existants

Introduction

Dans ce premier chapitre, nous allons exploiter le marché Algérien à travers une étude menée dans le but de trouver les solutions qui fournissent des services similaires ou légèrement différents des nôtres, c'est pourquoi nous allons prospérer avec ce chapitre pour évaluer la crédibilité du problème que nous résolvons et aussi notre valeur ajoutée unique.

1.1: Définition d'un ROC

La reconnaissance optique de caractères (ROC) est l'utilisation de la technologie pour distinguer les caractères textuels imprimés ou manuscrits à l'intérieur des images numériques de documents physiques, tels que des documents papier numérisés (Scannés). Le processus de base de la ROC consiste à examiner le texte d'un document et à traduire les caractères en code pouvant être utilisé pour le traitement des données. La ROC est parfois aussi appelé reconnaissance de texte.

Les logiciels de ROC de nos jours peuvent également tirer parti de l'intelligence artificielle (IA) pour mettre en œuvre des méthodes plus avancées.

Le processus de la ROC est le plus souvent utilisé pour convertir des documents juridiques ou historiques sous forme imprimée en fichiers PDF. Une fois placés dans cette copie logicielle, les utilisateurs peuvent éditer, formater et rechercher dans le document. [1]

1.2: Avantages de la reconnaissance optique de caractères

Avant le développement des programmes de la ROC, les documents papier devaient être convertis en copies numériques à la main. Par conséquent, les principaux avantages de la technologie de la ROC sont le gain de temps, la réduction des erreurs de frappe et la réduction des efforts. Il permet également des actions qui ne sont pas compatibles avec les

versions papiers, telles que la compression dans des fichiers ZIP, la mise en évidence de mots-clés, l'intégration dans un site Web et l'attachement à un courrier électronique.

Tandis que la prise d'images de documents permet leur archivage numérique, la ROC offre la fonctionnalité supplémentaire de pouvoir éditer et rechercher dans ces documents. [1]

1.3: Définition d'une plateforme d'impression de mémoire en ligne

Une plate-forme d'impression de mémoire en ligne est une plateforme qui aide les utilisateurs à automatiser le processus d'impression de leurs thèses de fin d'études.

Elle fournit des services de première qualité en matière de correction d'épreuves et de correction d'erreurs, ainsi que la numérisation, les photocopies, la plastification et reliure. La plate-forme proposera également un paiement en ligne et une livraison à domicile.

1.4: Étude de l'existant

Il existe déjà plusieurs solutions étrangères similaires à la notre qui résolvent les mêmes problèmes. Mais après une étude du marché Algérien, on n'a pu constaté la non existence d'aucune plate-forme algérienne capable d'effectuer les mêmes fonctionnalités.

Vous trouverez, ci-dessous, une liste des solutions existantes les plus intéressantes.

1.4.1: La plate forme Scribbr.fr

Scribbr est une plateforme qui a été créée en 2012 par trois amis étudiants et un correcteur. Aujourd'hui, Scribbr dispose de plus de 600 correcteurs professionnels.

Les différents services fournis par cette plate-forme, en plus de la préparation et de l'impression de votre mémoire, sont les suivants :

- Correction d'épreuves et correction d'erreurs : Assurez-vous que votre document (Projets de thèse ; Candidatures universitaires ; Essais ; Résumés de conférences ; Documents académiques) est parfaitement théorique, sans erreur.

- Vérification du plan et de la structure : Le correcteur s'assurera que votre document est bien structuré avec des intentions claires.
- La mise en page : Aide à la mise en page de votre document en fournissant une numérotation automatique des pages, une liste de référence pour tous vos tableaux et figures.
- Logiciel anti-plagiat : Logiciel pour scanner vos documents et éliminer le plagiat. [2]

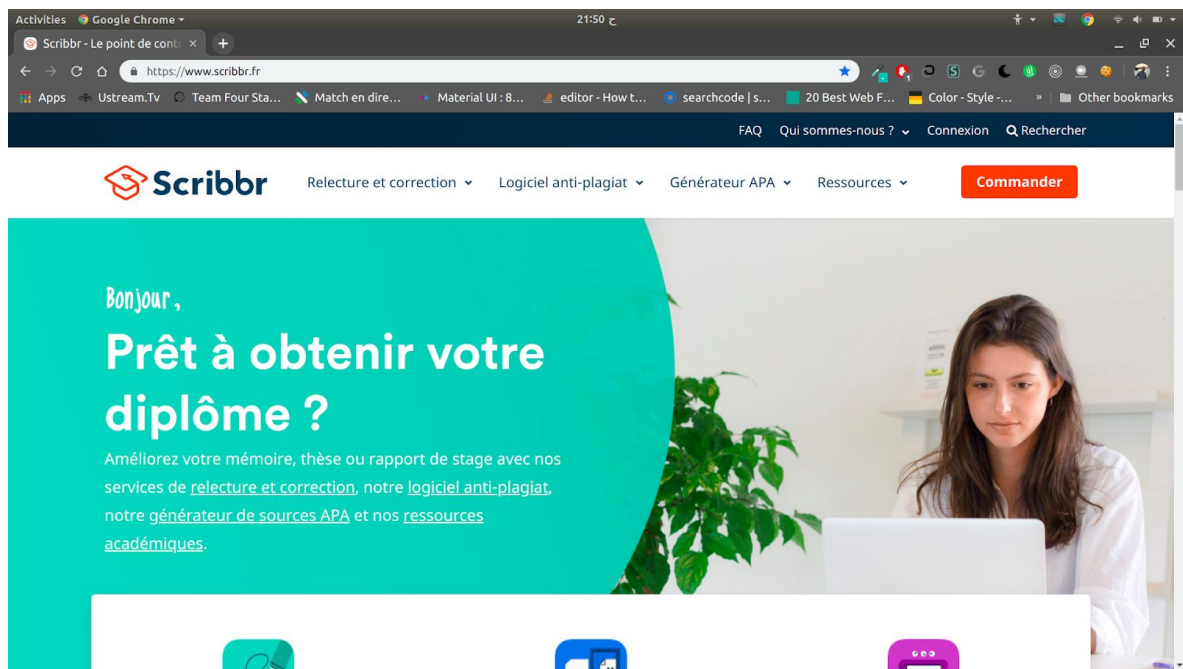


Figure 1.1: page d'accueil Scribbr [2]

1.4.2: La plate forme Corep.fr

Corep est une plate-forme française spécialisée dans l'impression et la correction de thèses, dont les fonctionnalités peuvent être résumées dans les points suivants :

- Haute qualité avec impression de corrections d'erreur.
- Calculateur de prix en ligne fiable. [3]

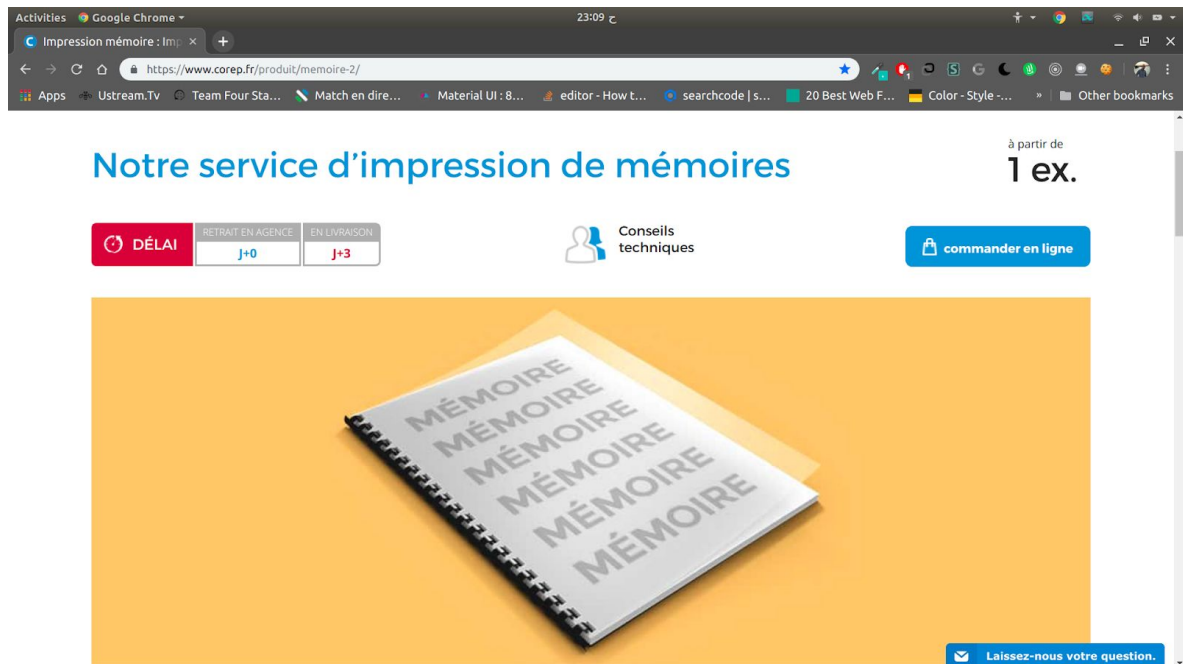


Figure 1.2: page d'accueil Corep [3]

1.4.3: La plate forme Nuance.com

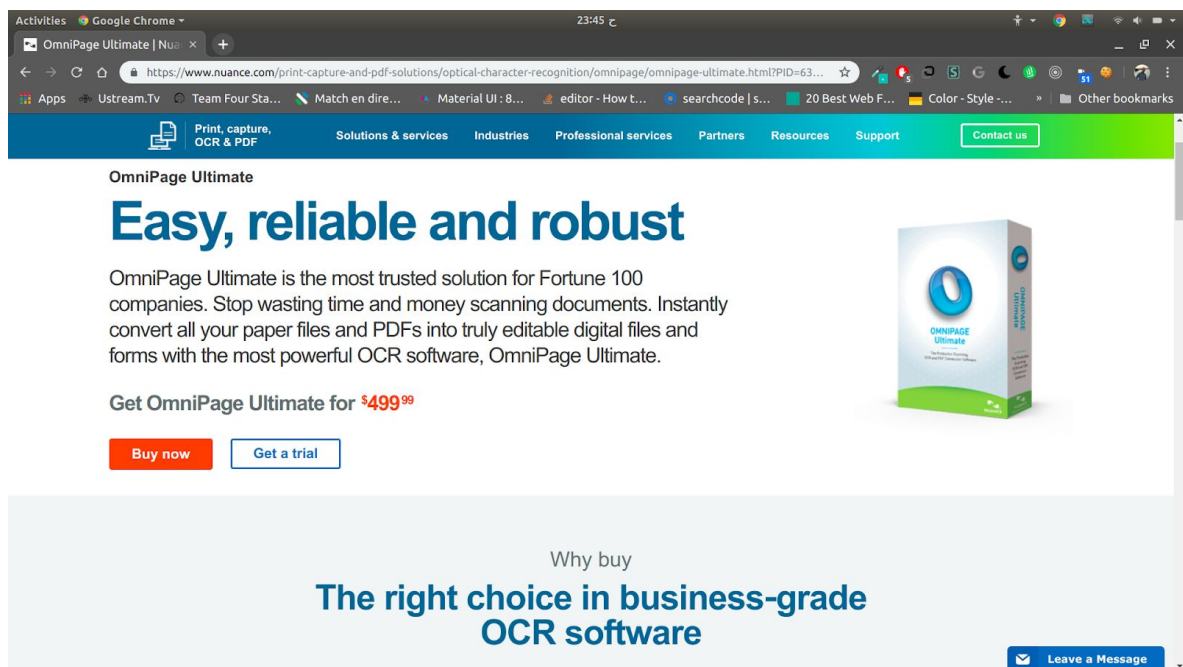


Figure 1.3: page d'accueil Nuance [4]

Pour la ROC, le type de solution le plus répandu, à l'étranger, est le logiciel plutôt que la plate-forme. Nuance est réputé pour sa précision de conversion. Même les plus grandes entreprises multinationales, dont Amazon, font confiance à ce logiciel. Et ça grâce à la magnifique performance de Nuance, qui consiste en :

- La rapide et facilité de conversion des documents papier en n'importe quel type de format éditable (PDF, Word ...).
- Flux de travail fluide et automatisé. [4]

Tandis que toutes les trois solutions citées fournissent de services de bonne qualité et cela pour l'impression en ligne ou les fonctionnalités OCR, cependant aucune de ces solutions ne permet les deux à la fois. La solution que nous allons aborder essaye d'accomplir en étant une plate-forme hybride qui fournit les deux fonctionnalités en même temps.

1.5: Étude du besoin sur le marché

L'étude de besoin sur le marché est une étape nécessaire dans chaque projet, avoir un besoin bien défini et un problème commun est primordial pour nous. Pour cela, il existe des outils permettant de vérifier et valider le besoin et le problème que nous essayons de trouver une solution. Parmi ces outils, nous avons opté pour Google Trend

1.5.1: Utilisation de l'outil « Google trends »

Google Trends est une plateforme qui a pour but servir de source de tendances précises. Google Trends peut également aider à concevoir un plan pour les futures stratégies de marketing.

Nous avons utilisé Google Trends afin d'effectuer une recherche sur quelques-uns de nos principaux problèmes et les résultats ont été les suivants :

- La figure suivante (figure 1.4) montre que l'Algérie est classé en 7ème position mondiale pour le pays où les utilisateurs cherchent le plus le terme “photo au pdf”, nous interprétons ce résultat comme un signification que les Algériens effectivement le problème et la nécessité d'extraire du texte des photos papier ce que notre projet est exactement là pour résoudre.

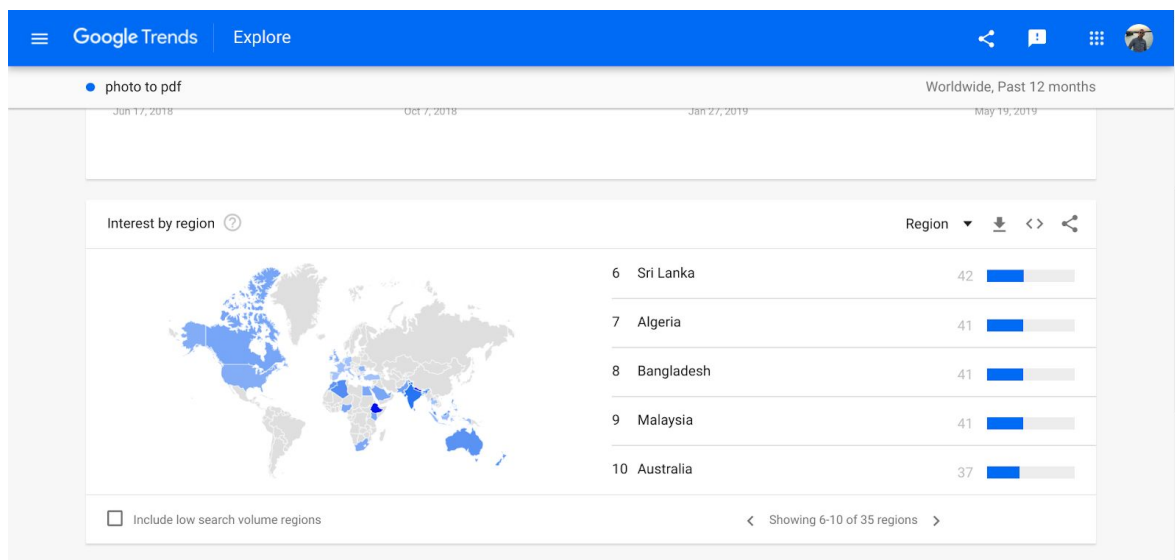


Figure 1.4: image représentant le taux de recherche de la phrase « photo to pdf » dans le monde

- La figure suivante (Figure 1.5) montre l'intérêt pour l'impression de thèses à l'échelle mondiale et encore une fois l'Algérie prend une place plus importante (5ème place).

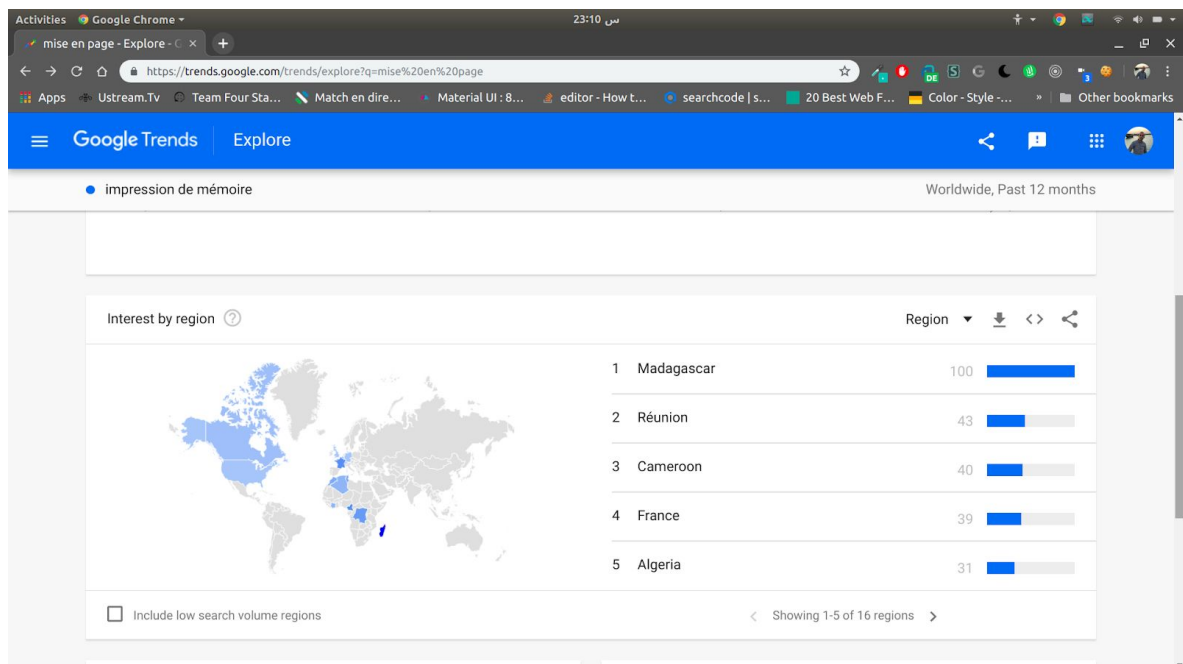


Figure 1.5: image représentant le taux de recherche de la phrase « impression de mémoire » dans le monde

1.5.2: La communauté Facebook

D'après napoleoncat.com [5], Facebook est le réseau social le plus populaire et utilisé en Algérie avec plus de 20M comptes d'utilisateurs actifs, ce qui permet de dire que c'est la source d'informations la plus fiable sur le marché algérien. La figure suivante montre le besoin réel d'une telle plate-forme:



Figure 1.6: une capture d'écran de Facebook qui représente une demande d'opération d'OCR

Conclusion

Dans ce chapitre, nous avons souligné l'importance du problème de notre projet et analysé les solutions similaires qui existaient déjà. Ce chapitre sert également à donner une plus grande visibilité aux principaux objectifs de notre projet (projet de fin d'étude).

Dans le chapitre suivant, nous allons entamer la phase d'analyse et de conception du système.

Chapitre 2

Analyse et conception du système Traixto

Chapitre 2 : Analyse et conception du système

Traixto

Introduction

Ce chapitre nous aidera à préciser les besoins les plus importants auxquels l'application répond, en donnant un regard approfondi sur la façon dont tout est structuré et maintenu ensemble, l'objectif principal étant la bonne identification des exigences fonctionnelles/non fonctionnelles.

2.1: Spécification des besoins du système

L'analyse que nous avons effectuée précédemment sur les plates-formes similaires déjà existantes nous aidera à identifier les acteurs ainsi que les besoins fonctionnels et non fonctionnels de notre système, ce qui est très important, en particulier à ce stade de développement.

2.1.1: Identification des acteurs

Les acteurs que nous avons réussi à identifier sont :

- Admin: contrôler le flux et l'état de l'application, ainsi que le responsable de sa gestion (utilisateurs, commandes, agents...).

- Client: est l'acteur responsable de la demande des commandes d'impression et de traitement de texte.
- Agent d'impression: est l'acteur responsable de l'impression des commandes du client.
- Agent de livraison: est l'acteur responsable de la livraison des commandes du client.

2.1.2: Besoins fonctionnels

Dans cette section, nous allons décrire les exigences fonctionnelles de notre système. Comme chaque acteur a un ensemble d'interactions uniques avec notre application, nous avons décidé de mettre l'accent sur les exigences fonctionnelles propres à chaque acteur, afin de bien montrer cette unicité.

Les exigences fonctionnelles auxquelles répond notre application sont les suivantes:

- L'authentification des utilisateurs en utilisant une authentification sans état (stateless) grâce à JWT pour accéder aux différentes fonctionnalités.
- L'extraction de texte à partir des images chargées par l'utilisateur à l'aide de la technologie de 'Machine Learning' fournie par l'API VISION de Google pour la reconnaissance de texte.
- L'export du texte extrait de la photo comme document texte.
- Le chargement des fichiers vers le cloud à travers le service de stockage RestFull de la plateforme GCP (Google Cloud Platform).
- L'approvisionnement d'une passerelle de paiement électronique pour que les utilisateurs puissent payer en ligne.
- Le calcul du prix de la commande en fonction de plusieurs facteurs.

Le système doit permettre aux administrateurs de :

- Gérer les utilisateurs (Ajout, modification, suppression...).
 - Gérer les rôles d'utilisateurs (Ajout, modification, suppression...).

- Gérer les wilayas de livraison (Ajout, modification, suppression...).
- Gérer les finitions de livraison (Ajout, modification, suppression...).
- Effectuer le suivi de toutes les commandes dans tous leurs états (commande reçue, commande en cours d'impression, commande prête, commande livrée).

Le système doit permettre aux clients de :

- Consulter leurs commandes dans tous leurs états.
- Créer une nouvelle commande avec la possibilité de paiement en ligne + la livraison.
- Charger une photo et en extraire le texte.
- Annuler une commande si elle est encore dans l'état "prête à être implémentée".

Le système doit permettre aux agents d'impression de :

- Consulter toutes les commandes qui ont le statut "commande reçue" et "en cours d'impression" et le type "livraison"
- Marquer une commande comme étant en cours d'impression lorsqu'un traitement est effectué sur elle.
- Marquer une commande comme prête dès que son traitement s'achève.

Le système doit permettre aux agents de livraison de :

- Consulter toutes les commandes qui ont le statut "commande prête" et le type "livraison".
- Marquer une commande comme livrée dès qu'elle est livrée en toute sécurité.

2.1.3. Besoins non fonctionnels

Il s'agit des besoins qui caractérisent le système. Ce sont des besoins en matière de performance, de type de matériel ou de type de conception. Ces besoins peuvent concerner

les contraintes d'implémentation comme le langage de programmation, le type de la plateforme, le type d'hébergement (cloud ou VPS) et le système d'exploitation.

L'ensemble des extensions à réaliser doivent respecter les besoins suivants :

L'ergonomie de l'interface : L'application doit être facile à utiliser, les interfaces utilisateur doivent être conviviales, c'est-à-dire simples, ergonomiques et adaptées à l'utilisateur.

La compatibilité : L'application doit être compatible avec tous les navigateurs ainsi que toutes les plateformes telles que les plateformes mobile et desktop.

La fiabilité : Les informations apportées par l'application doivent être fiables et sûres.

La disponibilité : L'application doit être disponible à tout instant pour être utilisée par n'importe quel utilisateur.

La sécurité : L'application comporte des informations personnelles et sensibles, donc elle doit respecter les règles relatives à la sécurité des systèmes informatiques.

La performance : L'application doit être performante, c'est-à-dire qu'elle doit répondre, à travers ses fonctionnalités et de manière optimale, à toutes les exigences des usagers.

2.2: Modélisation des besoins fonctionnels

La conception et la modélisation du système nous aidera à clarifier la vision sur la façon dont le système fonctionne pour atteindre ses objectifs.

2.2.1: Diagramme de cas d'utilisation

Un diagramme de cas d'utilisation nous aide à donner plus de détail sur les interaction et les activités des utilisateurs de notre système (appelés aussi acteurs). Un diagramme de cas d'utilisation efficace peut discuter et représenter :

- Les scénarios dans lesquels le système ou l'application interagit avec des personnes, des organisations ou d'autres systèmes externes.
- Les objectifs que le système ou l'application permet aux acteurs d'atteindre.
- La portée du système. [6]

Ci-dessous, notre diagramme de cas d'utilisation du système :

Systeme Traixto

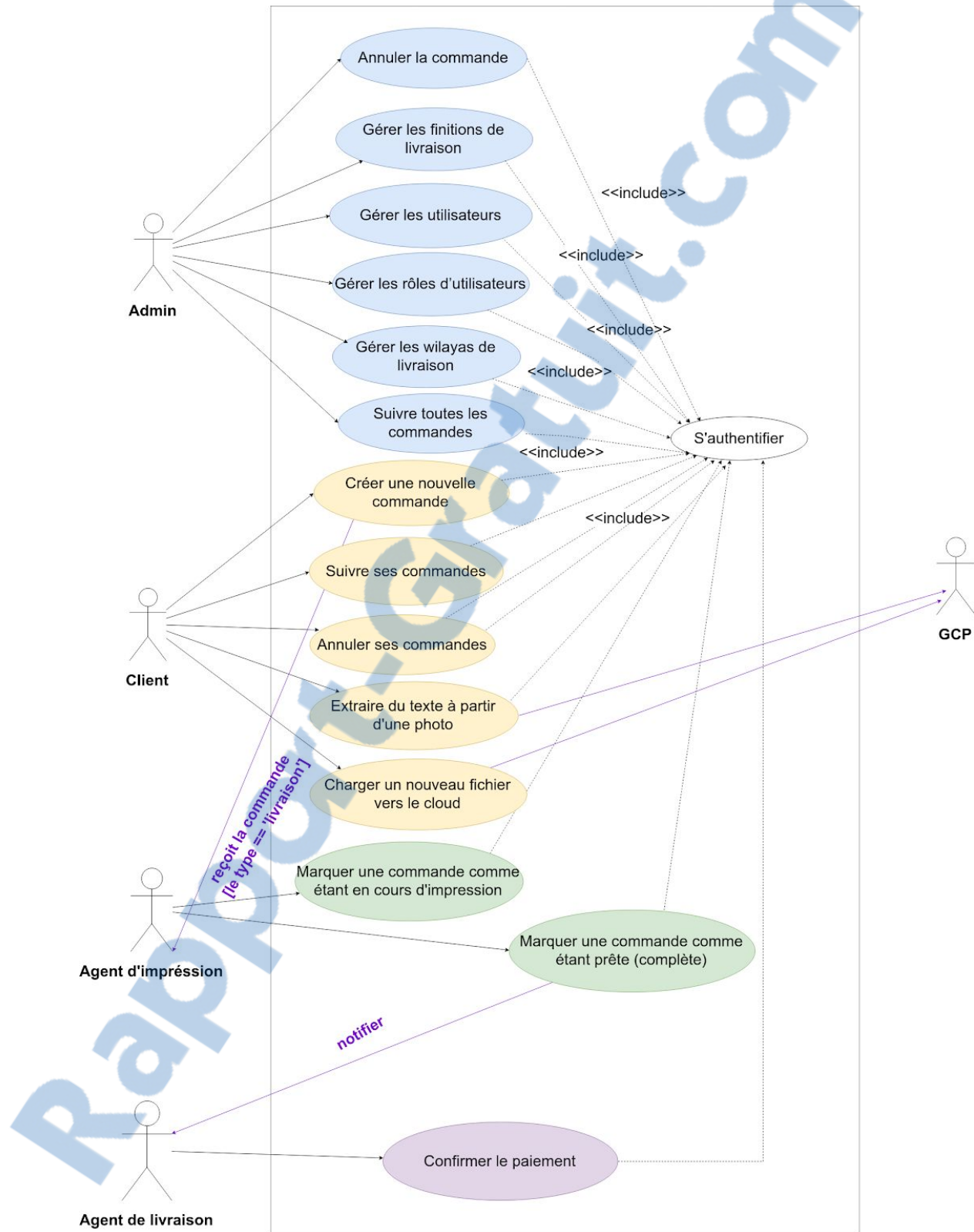


Figure 2.1: diagramme de cas d'utilisation de notre système « Traixto »

2.2.2: Diagramme de paquetage

Etant donné qu'Angular est fondé sur une architecture modulaire (en d'autres termes, une application Angular représente en elle-même un ensemble de modules qui communiquent entre eux), avoir un diagramme de paquetages pourrait nous être d'une grande aide afin d'avoir une vue d'ensemble claire sur la structure de l'application ainsi que la manière dont les différents paquetages intercommuniquent.

Dans la partie qui suit, nous allons détailler chaque paquetage (module) avec ses sous-paquetages:

1. AppModule

Il s'agit du paquetage de plus haut niveau qui inclut tous les autres modules, y compris les modules spécifiques au framework Angular et en outre, diverses dépendances logicielles, telles que les dépendances d'NPM.

1.1. SharedModule

Nous y trouvons tous les modules réutilisables qui peuvent être requis durant tout le développement de l'application. Trois sous-modules constituent principalement ce module, et qui sont:

1.1.1: ServicesModule

Il contient la logique et les fonctionnalités communes à plusieurs composants (plus d'informations sur les services à Angular plus tard)

1.1.2. NgxFlowModule

Ce module englobe le paquetage NgxFlow fourni par NPM. Par conséquent, il contient des composants réutilisables qui facilitent et raffinent le chargement des fichiers.

1.1.3. PipesModule

Similaires aux services, les pipes font partie des caractéristiques uniques d'Angular. Ils représentent des classes spéciales qui permettent de rédiger des transformations display-value qui peuvent être déclarées au niveau HTML (plus d'informations sur les services à Angular plus tard)

1.2. DashboardModule

L'intérêt de ce module est de regrouper les composants et les dépendances des utilisateurs Dashboard (Admin, Agent et Client) dans un seul module.

1.3. SessionsModule

Ce module contient tous les composants relatifs aux sessions des utilisateurs tels que les composants d'authentification, d'inscription, d'oubli et de réinitialisation du mot de passe (SignInComponent, SignUpComponent, ForgotPasswordComponent et ResetPasswordComponent).

1.4. DeliveriesModule

L'intérêt de ce module est de faciliter l'accès et la manipulation des composants spécifiques à la livraison, tels que les formulaires de produit et de contact.

1.4. CoreModule

Ce module inclut tous les composants du framework prêts à l'emploi qui facilitent notamment le développement de ce projet.

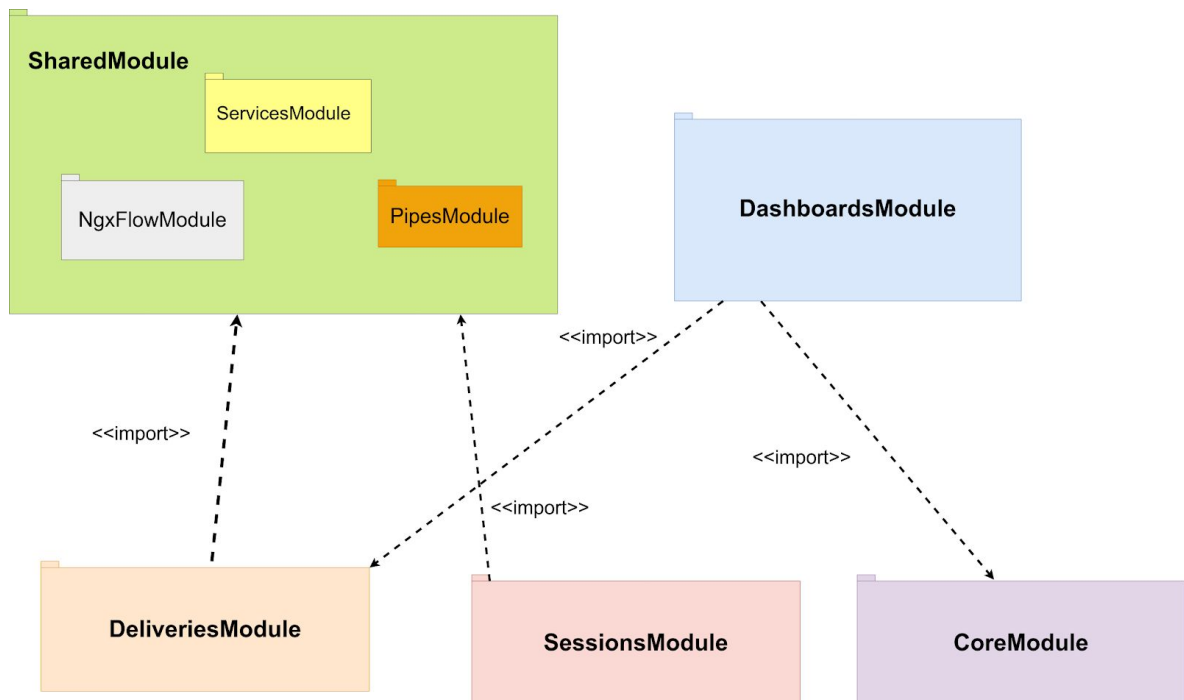


Figure 2.2: diagramme de paquetage de notre système « Traixto »

2.2.3: Diagramme de séquences

Ayant visualisé la structure modulaire du projet, nous allons nous intéresser à l'exploration de la structure interne de chaque module en adoptant une approche par scénarios. Une application Angular repose principalement sur l'utilisation des événements et des callbacks. De ce fait, il est parfois difficile de déduire la séquence d'actions ayant lieu dans un tel type d'applications, à partir de son code source.

2.2.3.1: Diagramme de séquences « Créer une commande »

L'une des fonctionnalités principales de notre logiciel consiste à donner au client la possibilité de créer des demandes d'impression en ligne personnalisées. Dans ce qui suit, nous allons éclaircir la manière selon laquelle nous allons procéder.

2.2.3.1.1: Description textuelle

Le tableau 2.1 représente la description textuelle du cas d'utilisation « Créer une commande»

Scénario 1: Créer une commande
Objectif : Créer une commande
Acteurs principaux: client, GCP
Préconditions: le client doit être connecté
Postcondition: enregistrement de la commande ou un message d'erreur
Scénario nominal: 1. Le client se connecte. 2. Le client clique sur le bouton <créer une commande> 3. Le système redirectionne le client vers la route de <créer une commande> 4. Le client remplit le formulaire de produit 5. Le système vérifie si le formulaire est valide et active le bouton de <ajouter un produit au commande> 6. Le client clique sur le bouton <ajouter un produit au commande> 7. Le client clique sur le bouton <confirmer commande> 8. Le système présente le POPUP de confirmation 9. Le client confirme l'action. 10. Le système charge les fichiers au cloud (GCP storage) 11. Le système enregistre la commande au base de données. 12. Le système affiche un message de succès.
Scénarios alternatifs:

Scénario1:

9.a: client clique sur le bouton <annuler>.

9.a.1: Le système redirectionne le client vers la route de <créer une commande>

Scénario2:

10.a: erreur dans le chargement des fichiers

10.a.1: Le système affiche un message d'erreur

Scénario3:

11.a: erreur dans l'enregistrement de commande.

11.a.1: Le système affiche un message d'erreur

Tableau 2.1: Description textuelle du cas d'utilisation « Créer une commande »

2.2.3.1.2: Diagramme de séquence système « Créer une commande »

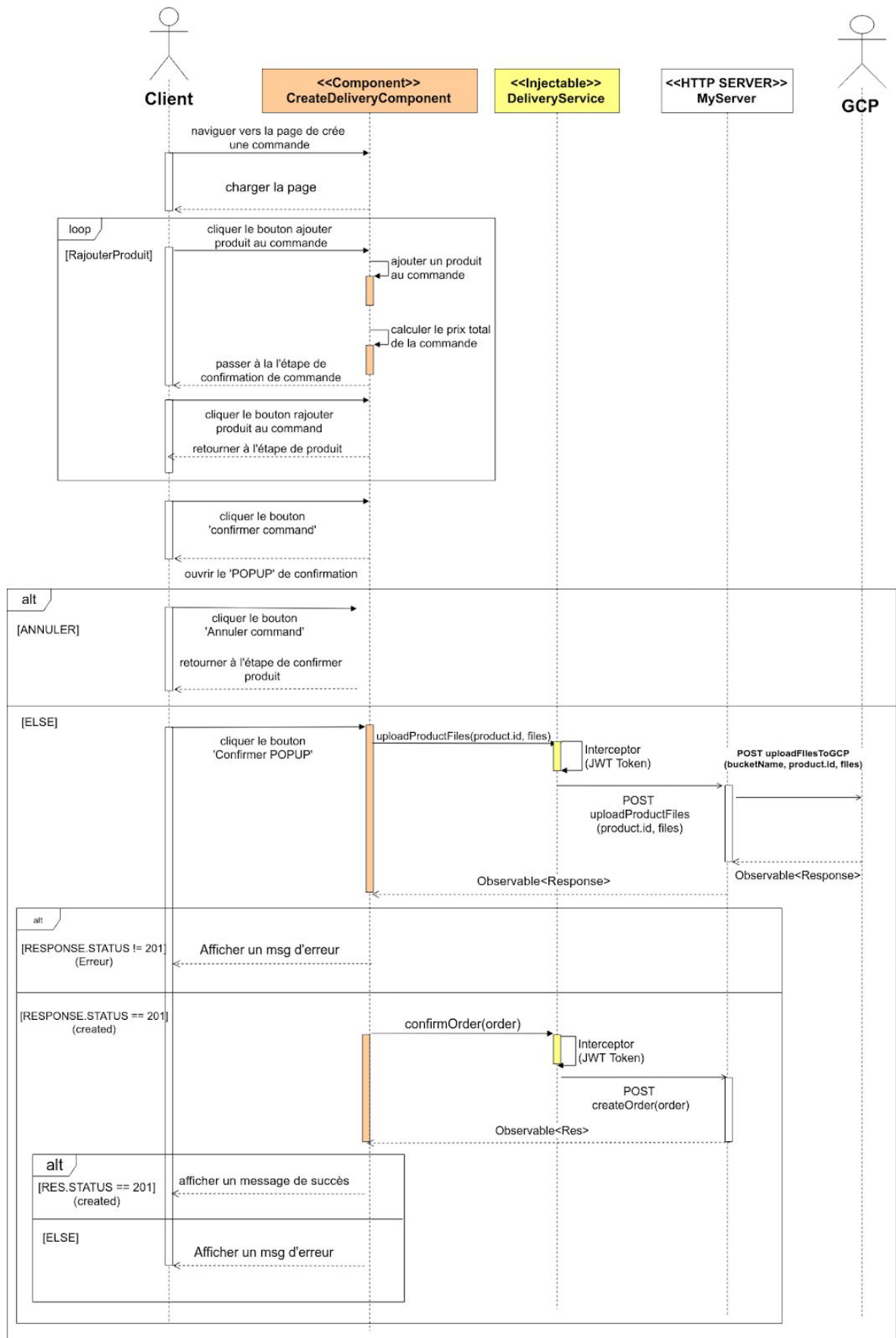


Figure 2.3: Diagramme de séquence pour l'action « Créer une commande »

2.2.3.2: Diagramme de séquence« convertir une image au texte »

Parmi les autres fonctionnalités principales de notre logiciel, nous pouvons citer l'extraction de texte d'une photo à l'aide de mécanismes et de techniques de reconnaissance optique de caractères. Nous tenterons ensuite de donner un aperçu détaillé de la manière dont nous y parvenons.

2.2.3.1.1: Description textuelle

Le tableau 2.2 représente la description textuelle du cas d'utilisation « convertir une image au texte»

Scénario 1: convertir une image au texte
Objectif : convertir une image au texte
Acteurs principaux: client, GCP<VISION_API>
Préconditions: le client doit être connecté
Postcondition: l'extraction de texte de l'image et le transformer au texte ou un message d'erreur
Scénario nominal: 1. Le client se connecte. 2. Le client clique sur le bouton <créer une commande> 3. Le système redirectionne le client vers la route de <traitement de texte> 4. Le client glisse une photo pour la traiter. 5. Le système vérifie s'il y a une photo et active le bouton de <ajouter un produit au commande>

6. Le client clique sur le bouton <convertir photo>
7. Le système charge les fichiers au cloud (GCP storage)
8. Le système fait un appel au VISION_API pour faire l'extraction du texte.
9. Le système export le texte en pdf
10. Le système charge les fichiers au cloud (GCP bucket)
11. Le système télécharge automatiquement le fichier pdf

Scénarios alternatifs:

Scénario2:

- 7.a: erreur dans le chargement des fichiers
- 7.a.1: Le système affiche un message d'erreur

Tableau 2.2: Description textuelle du cas d'utilisation « Convertir une image au texte »

2.2.3.1.2: Diagramme de séquence système «Convertir une image au texte»

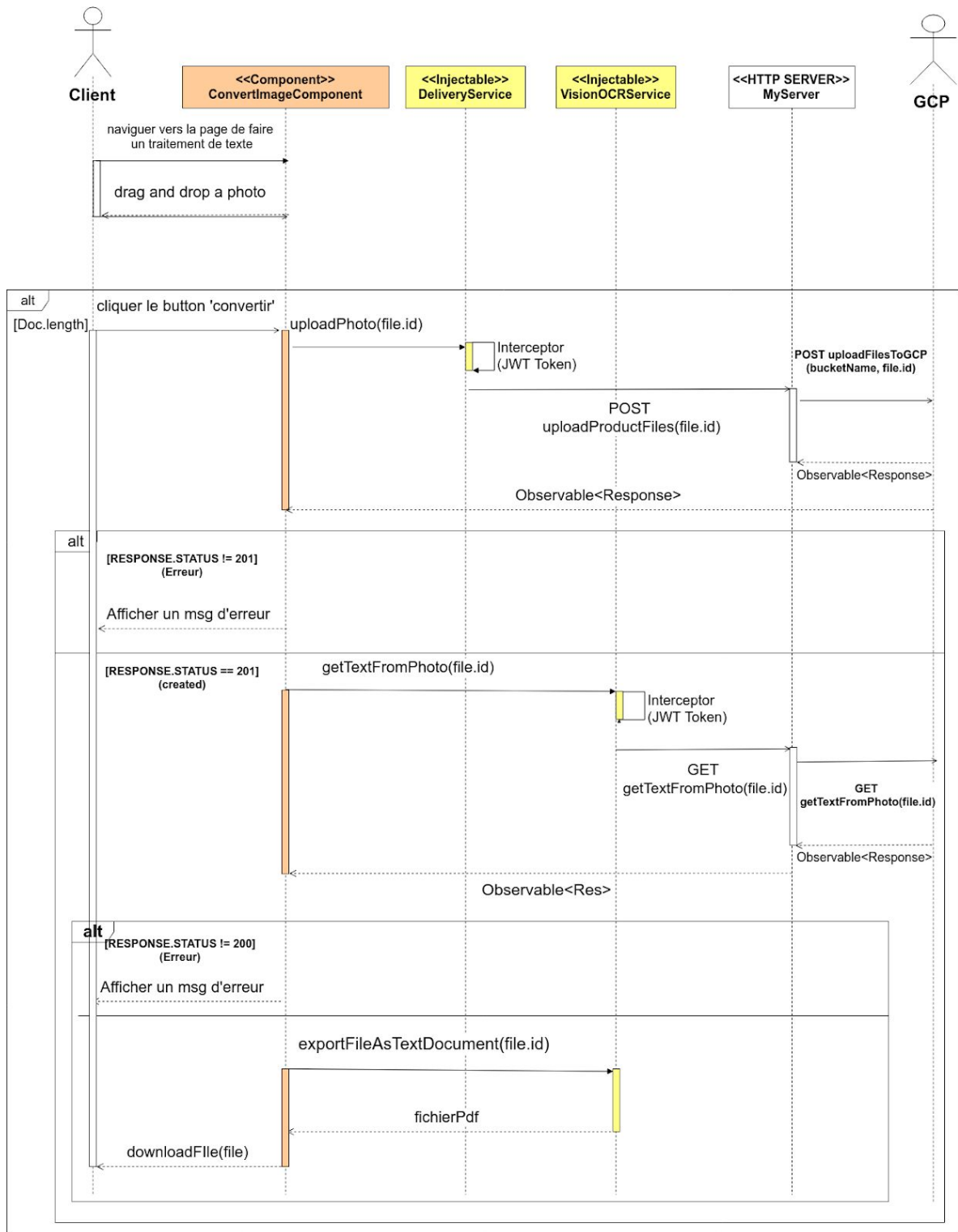


Figure 2.4: Diagramme de séquence pour l'action « Convertir une image au texte »

2.2.4: Diagramme de classe

Un diagramme de classe est un diagramme de structure, vu qu'il décrit ce qui doit être présent dans le système modélisé.

Ci-dessous, le diagramme de classes de notre système. [7]

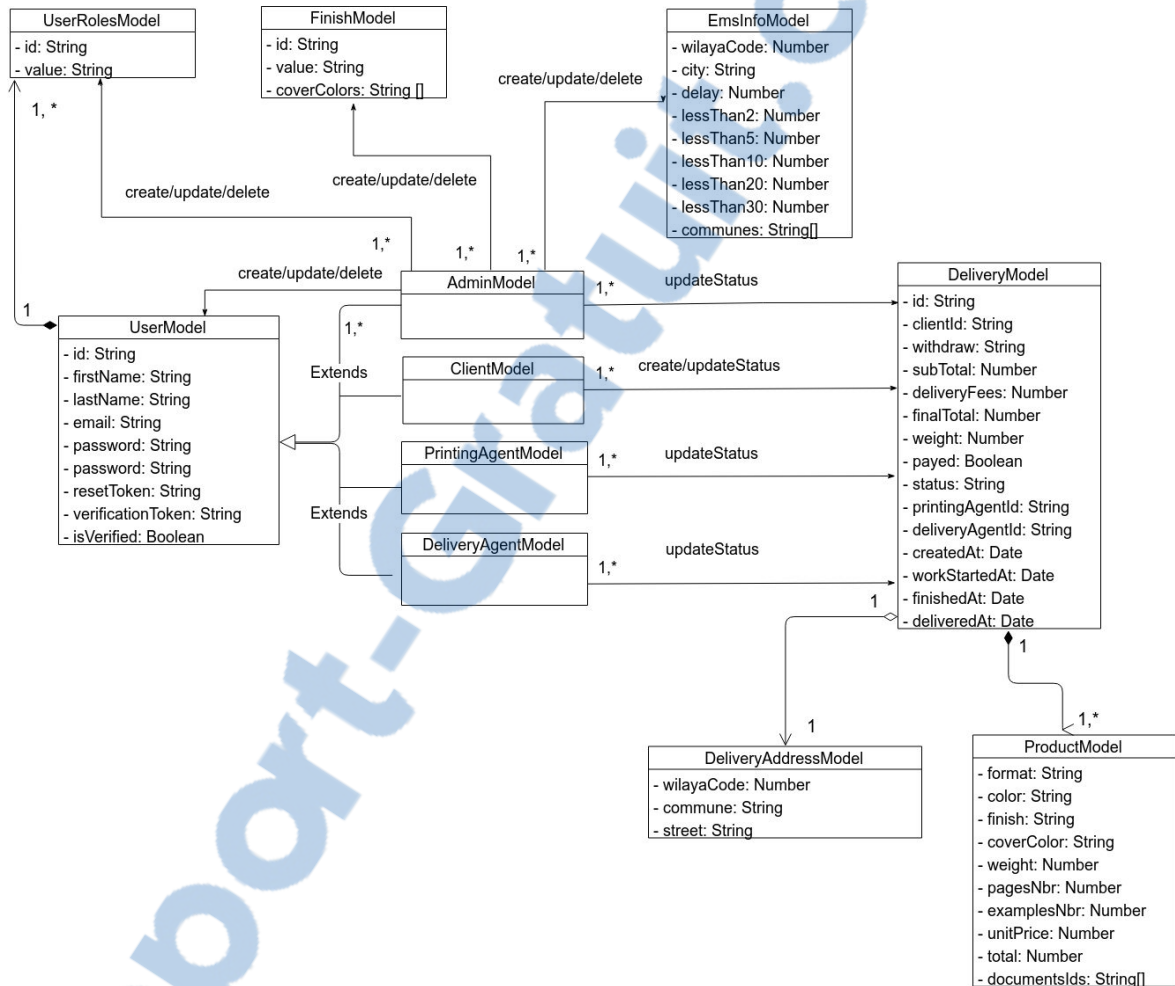


Figure 2.5: diagramme de classe de notre système « Traixto »

Conclusion

Ce chapitre nous a permis de déterminer avec succès les parties les plus importantes de notre application et d'expliquer en détails comment certaines de nos fonctionnalités les plus importantes fonctionnent théoriquement.

Chapitre 3

Implémentation du système « Traixto »

Chapitre 3 : Implémentation du système

Introduction

Dans ce chapitre, nous aborderons l'aspect technique de l'application. Nous verrons en premier lieu la manière dont une application web moderne doit être structurée. Par la suite, nous détaillerons les technologies utilisées par les applications de type **MEAN stack**. Au final, nous nous intéresserons au code source même (implémentation).

3.1: Structure d'application web moderne

3.1.1: Application web moderne

Une application web est une application client-serveur dont le navigateur joue le rôle d'un client et le serveur web comme serveur.

La logique d'une application web est distribuée entre le client et le serveur. Les données sont échangées sur le réseau avec le protocole de transfert hypertexte (HTTP). L'un des avantages de cette approche est le fait que les utilisateurs ne dépendent pas d'un système d'exploitation ou d'une configuration de matériel spécifique. Donc les application web sont des services multi plateforme (cross-platform services).

3.1.2: Une application Client-serveur

Nous pouvons la décrire comme étant un réseau client-serveur du fait que toutes les requêtes sont traitées au niveau du réseau, typiquement par l'un des protocoles HTTP et HTTPS. Cette application s'agit d'un modèle informatique dans lequel le serveur héberge des ressources qui font l'objet des requêtes lancées par les clients. Ce type d'architectures admet plusieurs clients connectés à un serveur central par le biais d'un réseau ou d'une

connexion internet.[8]

Ce type d'architectures a plusieurs avantages, parmi lesquels nous pouvons citer :

La centralisation

Il s'agit de l'une des raisons principales pour lesquelles un développeur opterait pour cette architecture, puisqu'elle offre un contrôle centralisé des ressources et des droits d'accès au serveur.

La sauvegarde et la récupération des données

Comme toutes les données résident au niveau du serveur, leur sauvegarde est très simple puisqu'elle revient à sauvegarder les données du serveur uniquement, ce qui n'est pas le cas d'autres architectures telles que l'architecture pair à pair dans laquelle les données de chaque poste de travail doivent être sauvegardées séparément.

La scalabilité

Ce qui a été dit à propos de la sauvegarde s'applique aux mises à jour aussi. Le serveur et ses ressources sont mis à jour sans se soucier des clients.

L'accessibilité

L'accès au serveur peut se faire à distance depuis n'importe quelle plateforme dans le réseau.

La sécurité

Toutes les règles relatives à la sécurité et aux droits d'accès peuvent être définies au moment de la configuration du serveur.[9]

3.1.3: Les services web RESTful

REST (Representational State Transfer) est un modèle architectural qui sert à la création de

services Web. D'autre part, les services Web qui implémentent une telle architecture sont appelés **RESTful**.

Dans un service Web RESTful, la réponse d'un serveur dépend de 02 éléments principaux:

- L'URL (Uniform Resource Locator) de la ressource, également appelé **endpoint**.
- L'opération à effectuer sous forme d'une méthode HTTP (GET, POST, DELETE, PUT, PATCH ...).

Une API RESTful doit satisfaire les 03 contraintes principales suivantes :

- **Stateless** : les données client ne sont pas stockées sur le serveur pendant une intervention et une autre et la session est stockée du côté client (généralement dans la mémoire de session ou locale <<local storage>>).
- **Client <-> Serveur** : le client et le serveur doivent être séparés et ne doivent interagir entre eux qu'avec des requêtes.
- **Cache** : les données du serveur peuvent être mises en cache du côté client, ce qui peut améliorer les performances de l'application.[10][11]

3.1.4: Une application web monopage (Single page application « SPA »)

Une application "Single Page" (SPA) est une application qui fonctionne dans un navigateur et qui ne nécessite pas un rechargement de la page pendant son utilisation. Par exemple: Gmail, Google Maps, Facebook ou GitHub.

les SPA ont pour but de servir une bonne expérience d'utilisation (UX) en imitant l'environnement "naturel" dans le navigateur sans aucun rechargement de pages, sans aucun temps d'attente supplémentaire. C'est juste une page web qui, après être visitée, charge tout le contenu en utilisant Javascript dont ils dépendent fortement.

Les SPA demandent le markup et les données indépendamment et rend les pages directement dans le navigateur. Nous pouvons le faire grâce aux frameworks avancées de Javascripts, notamment AngularJS, Ember.js, Meteor.js, Knockout.js . [12]

3.1.4.1: Les avantages d'utiliser une application web monopage

- Les SPA sont rapides, car la plupart des ressources (HTML, CSS, les scripts) sont chargées une seule fois pendant le cycle de vie de l'application et seulement les données qui sont transmises dans les deux sens.
- Les SPA sont faciles à déboguer sur le navigateur Google Chrome.
- Il est plus facile de développer une application mobile car l'utilisateur réutilise le même code backend pour les applications web et les applications mobiles natives.
- Les SPA peuvent mettre en cache tout stockage local de manière efficace. Une application n'envoie qu'une seule demande, stocke toutes les données, puis peut utiliser ces données et travailler même hors ligne. [13]

3.1.5: Authentification par JSON Web Tokens « JWT »

3.1.5.1: JSON Web Token

JSON Web Token (JWT) est un standard (RFC 7519) qui définit un moyen compact et autonome pour la transmission sécurisée de l'information entre les parties sous forme d'objet JSON.

Cette information peut être vérifiée et fiable vu qu'elle est signée numériquement. Les JWT peuvent être signés en utilisant une clé secrète (avec l'algorithme HMAC) ou d'une paire publique/privée en utilisant RSA ou ECDSA. [14]

3.1.5.2: La structure d'un jeton JWT

JWT est une chaîne de caractères avec trois parties principales: l'en tête (Header), la Charge utile (payload) et la signature et séparées par des points (.) comme indiquée ci-dessous.

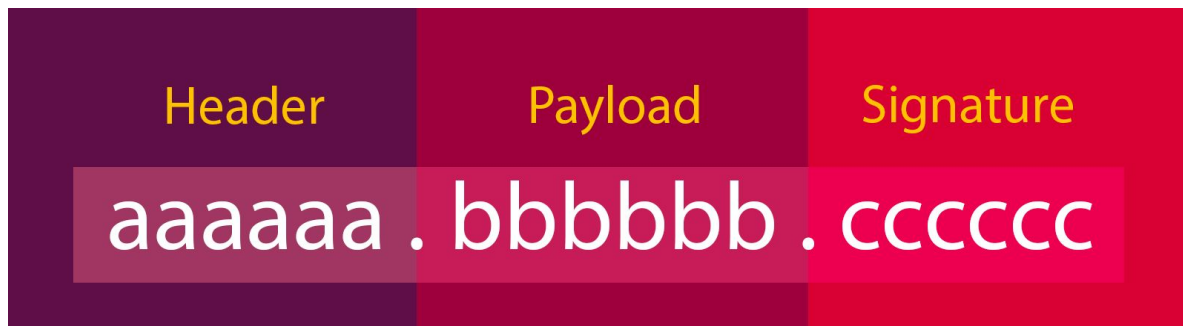


Figure 3.1: la structure d'un jeton JWT

L'entête

Il contient des informations sur la façon dont la signature doit être calculée, et a généralement le format suivant:

```
{ "alg": "HS256",  
  "typ": "JWT" }
```

Généralement il est composé de deux parties: le type du jeton (token) qui est le JWT et l'algorithme de signature utilisé comme: HMAC SHA256 ou RSA.

Ensuite, ce JSON est Base64Url encodé pour former la première partie du JWT.

Charge utile (Payload)

C'est les données stockées dans le JWT. Nous pouvons aussi avoir d'autres charges utiles prédéfinies qui ne sont pas obligatoires mais recommandées pour les bonnes pratiques comme: iss(issuer qui est l'émetteur), exp(expiration time qui est le temps de péremption, sub(subject qui est le sujet), aud (audience) et autres.

Un exemple simple de payload peut être:

```
{ "sub": "1234567890",  
  "name": "John Doe",  
  "admin": true }
```

Le payload est donc Base64Url encodé pour former la seconde partie du JWT.

Signature

La signature est utilisée pour vérifier si le message n'a pas été altéré en cours de route et

dans le cas des jetons signés avec une clé privée que l'émetteur est bien celui qu'il prétend être (le cas d'autorisation)

La signature est calculée en utilisant le pseudo code suivant:

```
/* encodé en utilisant Base64url et en joignant l'entête et le payload */  
data = base64urlEncode( header ) + "." + base64urlEncode( payload )  
/* Hacher les données obtenus en utilisant le type d'algorithme prédéfinis dans l'entête et la  
clé secrète */  
hashedData = hash( data, secret )  
/* et finalement les données hachées sont encodée en bae64url pour générer la signature  
JWT */  
signature = base64urlEncode( hashedData )
```

Il en résulte une sortie comme celle-ci :

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKxwRJSMeKKF2QT4fwpMeJf36P  
Ok6yJV_adQssw5c
```

3.1.5.3: L'authentification basée sur le JWT

L'une des utilisations les plus connues du JWT est l'autorisation qui va de pair avec l'authentification et qui a un scénario comme ceci:

Modern Token-Based Auth

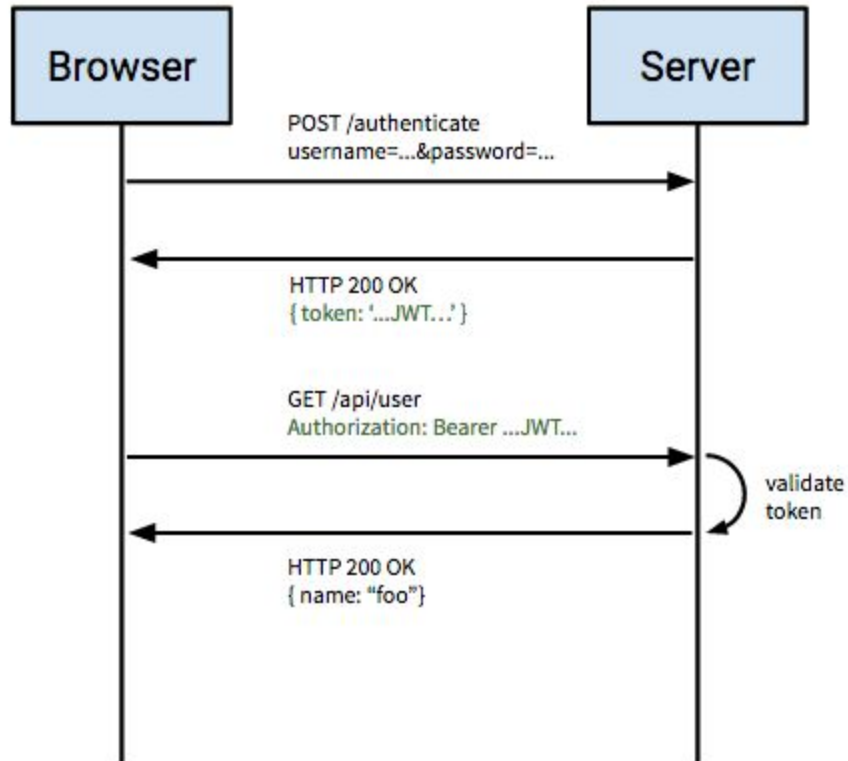


Figure 3.2: L'authentification basée sur le JWT [15]

Une fois l'utilisateur est authentifié, chaque demande ultérieure comprendra le JWT permettant à l'utilisateur d'accéder aux routes, services, et aux ressources qui lui sont permises avec ce jeton. [15]

3.1.6: Google Cloud Platform (GCP)

Google Cloud Platform (GCP) est un ensemble de services cloud hébergés sur l'infrastructure de Google. Cela peut aller de stockage, analyse de données, machine learning, et les réseaux, GCP offre une grande variété de services et APIs qui peuvent être intégrés avec une application ou un projet cloud, à une échelle personnelle ou professionnelle. [16]

3.1.6.1: Cloud Storage

Google Cloud Storage est un service Web de stockage de fichiers en ligne RESTful pour le stockage et l'accès aux données sur l'infrastructure de la plate-forme Google (GCP). Il fournit un stockage d'objets unifié pour les données. Les objets stockés dans Google Cloud Storage sont regroupés dans des **buckets**. Les buckets sont des conteneurs dans le nuage(cloud) qui peuvent être affectés individuellement aux classes de stockage. [17]

3.1.6.2: Cloud Vision API

Cloud Vision API est une API REST qui peut être utilisée pour comprendre le contenu d'une image en catégories, détecter les objets et visages individuels dans les images, et trouver et lire les mots imprimés dans les images. [17]

3.2: Développement javascript full-stack (MEAN Stack)

3.2.1: Javascript coté client

Javascript est un langage de script ou de programmation qui permet d'implémenter des programmes complexes sur les pages web, ces scripts peuvent être écrits directement sur une page web HTML et exécutés automatiquement quand la page se charge.

3.2.1.1: Javascript (ES6)

Es6 fait référence à la version javascript 2015 et est considéré comme une amélioration majeure du langage Javascript et ajoute plusieurs fonctionnalités destinées pour rendre le développement plus facile.

Les principales caractéristiques d'ES6:

Les mots clés 'let' et 'const'

Avant en javascript, la seule façon pour définir une variable c'est à travers le mot clé "var", qui est une fonction de portée (scope function) et qui a beaucoup d'effets indésirables pour le développeur, où 'let' est un bloc de portée et est utilisé pour se débarrasser des effets d'utilisation.

Exemple de l'une des différences notables entre 'let' et 'var':

- **Redeclaration:**

'var' permet de redéclarer la même variable dans la même portée, 'let' ne le permet pas.

En utilisant 'var'

```
var me = 'foo';
```

```
var me = 'bar'; // la variable me est remplacée
```

En utilisant 'let'

```
let me = 'foo';
```

```
let me = 'bar'; // Une erreur de syntaxe: l'identifiant 'me' a été déjà déclaré
```

- **Global:**

Les variables globales définies par 'let' ne seront pas ajoutées comme propriétés sur l'objet global 'window' comme celles définies par 'var'

```
let me = 'go'; // portée globale
```

```
var i = 'able'; // portée globale
```

```
console.log(window.me); // indéfinie
```

```
console.log(window.i); // 'able'
```

La même chose pour 'const' avec le fait que ce n'est pas possible de ré-affecter une variable qui est 'const'

Les fonctions ‘Fat arrow’

Une fonction Fat arrow a pour objectif de réduire le code pour créer des fonctions javascript, par exemple:

L’ancienne méthode:

```
function greetings (name) { return 'hello ' + name }
```

En utilisant ES6:

```
const greetings = name => `hello ${name}`;
```

Les opérateurs des fonctions ‘REST’

Avec les opérateurs REST, il est maintenant possible de créer des fonctions qui prennent un nombre de paramètres variable, ces paramètres sont stockés dans un tableau et utilisée dans la fonction.

```
const howMany = (...args) => args.length;  
console.log(howmany(2,2,1));           // retourne 3
```

Le mot clé ‘class’

Avant ES6, il n'y avait pas de façon élégante pour créer des classes, avec ES6, javascript a obtenu une façon correcte et facile de le faire, il s'agit de syntaxe sur les classes déjà disponibles à base de prototypes, ce qui signifie que nous n'ajoutons aucun nouveau modèle d'objet. [18]

3.2.1.2: TypeScript

[19] TypeScript est un ensemble de ES6 qui est lui-même un ensemble de ES5 (Javascript normal) et est considéré comme le résultat d'une collaboration entre Google et Microsoft.

Typescript ne peut pas être utilisé directement sur le navigateur puisqu'il a une syntaxe différente que celle du javascript régulier, il a besoin de ce que nous appelons 'transpiler' qui traduit le fichier typescript (.ts) au fichier javascript régulier (.js) connu par le

navigateur.

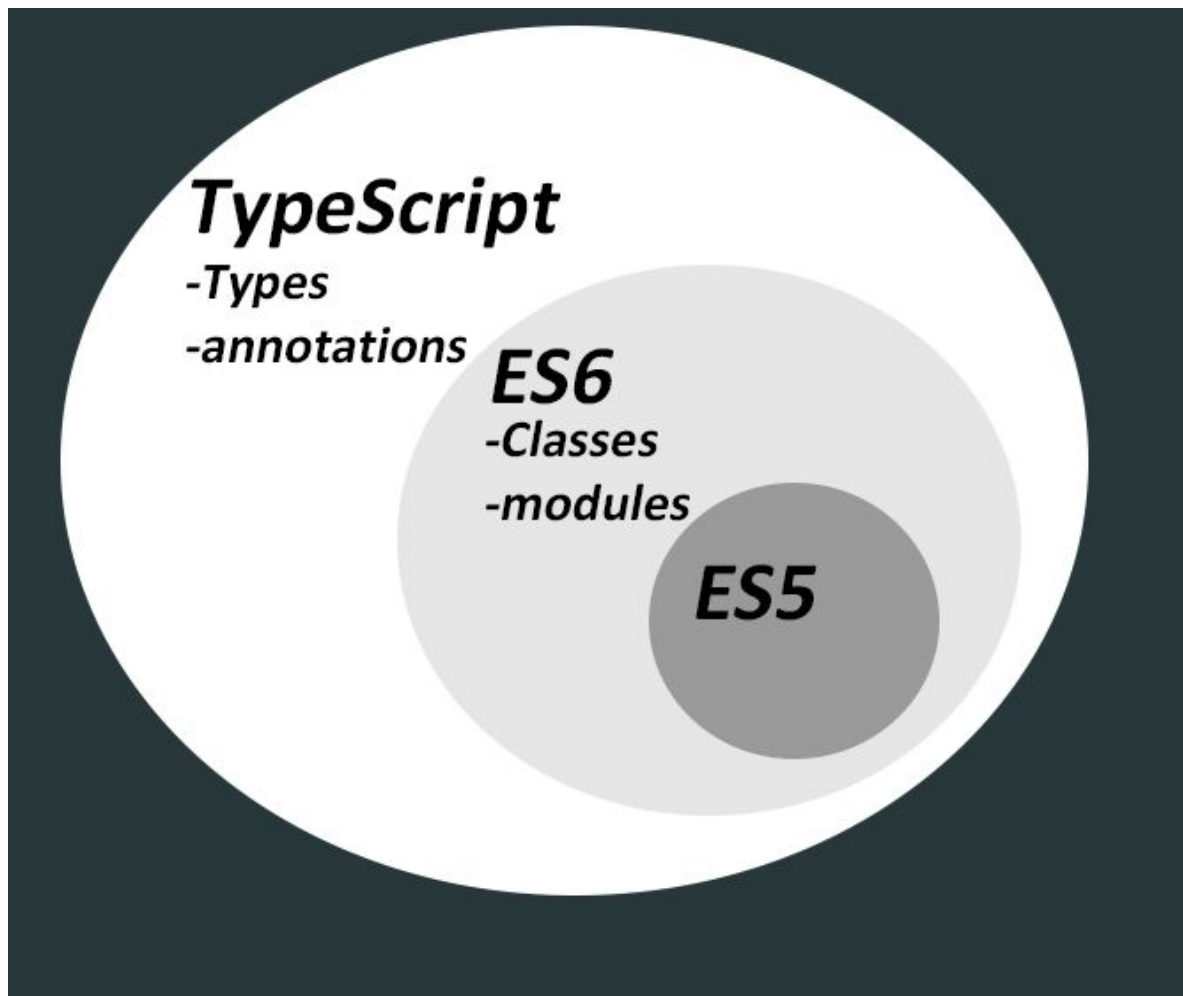


Figure 3.3: la relation entre le Javascript (ES5/ES6) et le Typescript [19]

Les caractéristiques les plus notables de Typescript:

1. Un sur-ensemble typé de JavaScript

[20] Comme nous le savons, javascript est un langage dynamiquement typé, ce qui pose un problème pour certains développeurs, donc avoir un mécanisme de vérification de type peut aider à:

- Prévenir les bugs dans la phase de compilation ce qui aide au débogage.
- Avoir un code propre et clair.

L'exemple suivant donne un aperçu sur la façon dont typescript prévient les bugs lors de l'exécution::

Javascript (ES6)

```
const sum = (a,b) => `result: ${a+b}`;  
console.log(sum('10', 2));    // result: 102
```

Typescript

```
const sum = (a: number, b: number) => `result: ${a+b}`;  
console.log(sum('10', 2));    // runtime error
```

2. Les Interfaces

Dans typescript, les interfaces remplissent le rôle de nommer des nouveaux types, et sont un moyen puissant de définitions des contrats dans le code ainsi que des contrats hors le code du projet.

```
interface User {  
    name: string;  
    birthday?: string; // optional property  
}
```

3.2.1.3: Les Framework Javascript côté client

Le framework est une technologie qui assure les bons outils pour développer une application web tout en définissant sa conception et comment le code doit être organisé. Il faut aussi mentionner que la majorité des frameworks modernes de Javascript ont leur propre philosophie quand il s'agit de la façon dont les application doivent être développées, d'où il est nécessaire d'apprendre les concepts de base.

Pour l'utilisation d'un framework:

- La majorité des frameworks javascripts ont leur propre philosophie.
- Aider à écrire un code propre bien entretenu et évolutif.
- Éliminer le risque de réinventer les choses

3.2.1.3.4: Angular

Angular est un framework de javascript pour développer des applications web monopage que nous appelons aussi SPA (Single Page Applications). Il est désigné pour des applications web, desktop et les plateformes mobiles.

En réalité, une application angular est un arbre de composants dont le composant le plus haut est l'application elle-même, que le navigateur la rend quand l'application est lancée.[21]

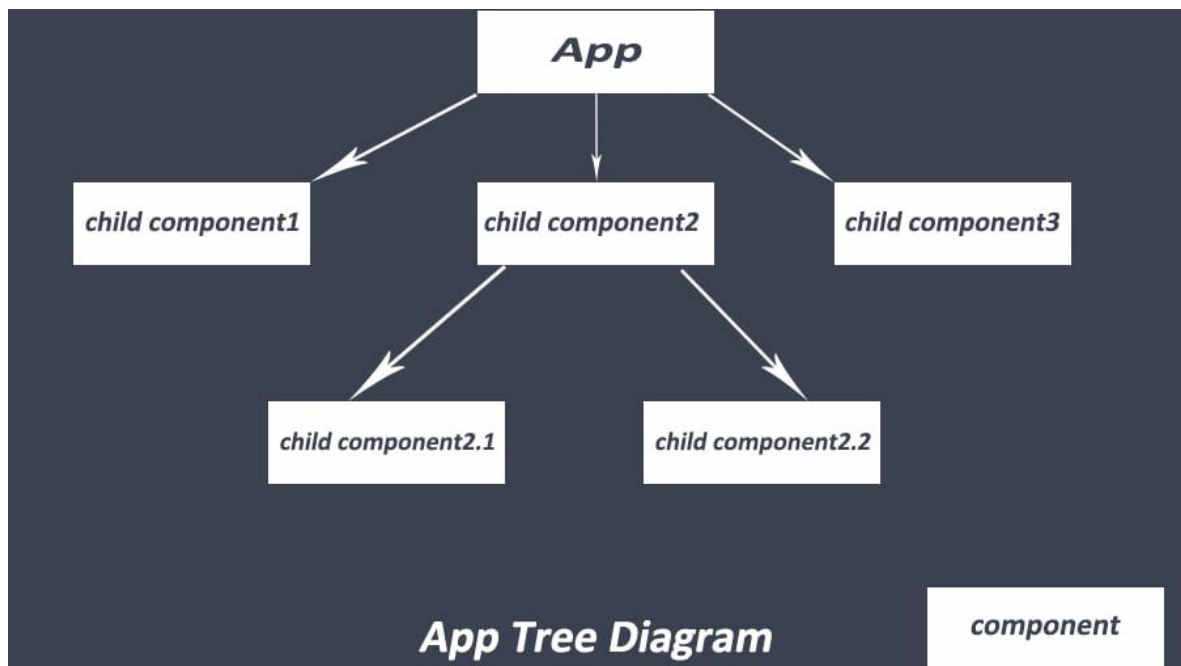


Figure 3.4: la structure d'une Application Angular [21]

Les fonctionnalités d'Angular

- **Multi Plateformes:** Angular est utilisé sur le web, mobile et desktop.
- **Vitesse et performance:** Angular utilise la génération du code, et la division du code à l'aide de son architecture basée sur les composants.
- **La productivité:** en utilisant les gabarits personnalisés, et l'interface de ligne de code Angular CLI. [22]

3.2.2: Javascript côté serveur

3.2.2.1: Nodejs

Nodejs est un environnement d'exécution javascript open source, multi plateformes utilisé pour donner aux applications javascript la possibilité pour s'exécuter sur le serveur, pour cela Nodejs prend l'avantage du moteur d'exécution Javascript V8 (**V8 JavaScript runtime engine**)

Le moteur V8 est écrit en C++ et utilisé dans Chrome et avec Node.js, V8 joue le rôle d'un pont qui prend le code et le convertit à un code machine plus rapide qui peut être exécuté nativement sans interprétation. [23]

3.2.2.1.1: Le fonctionnement

JavaScript est un langage **single-thread**, donc Node.js opère sur un seul thread, en utilisant des appels d'E/S non bloquants, ce qui lui permet de prendre en charge des dizaines de milliers de connexions simultanées détenues dans la boucle d'événements (**event loop**).

Ainsi, la boucle d'événements est une boucle qui attend des événements et les répartit lorsqu'ils surviennent. Les développeurs de Node écrivent donc des fonctions de rappel qui s'exécutent lorsqu'un événement Node se produit.

De ce fait, au lieu de créer un nouveau thread pour chaque requête, comme le font les langages multithread, Node.js prend toutes les requêtes, puis attribue la majeure partie du travail à d'autres utilisateurs du système (**system workers**). Cela fonctionne parfaitement avec les opérations quotidiennes telles que la manipulation du système de fichiers, les requêtes réseau ou les périphériques d'accès.

Donc, si vous avez suivi à la lettre, vous aurez remarqué que Node.js n'est pas idéal pour les tâches gourmandes liées au processeur et qui peuvent surcharger le thread principal (et l'unique) qui ne pourra pas exécuter ses tâches convenablement par la suite.

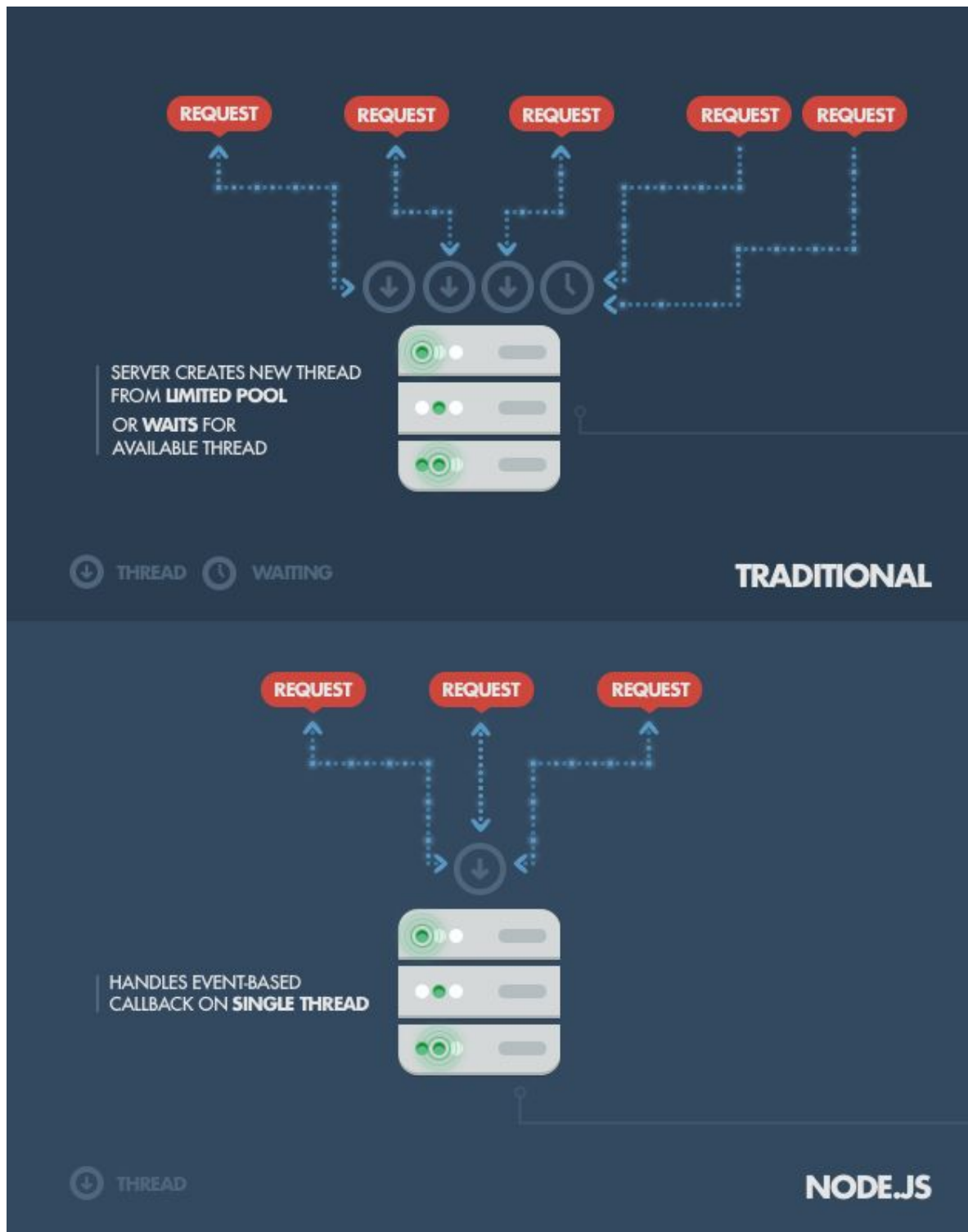


Figure 3.5: Le fonctionnement d'une application Nodejs (mono-thread) par rapport à une application multi-thread [23]

3.2.2.1.2: Quand utiliser Node.js?

Node.js prend l'avantage dans le développement d'applications en temps réel et en streaming, telles que :

- Les applications de messagerie instantanée.
- Les serveurs de jeux qui doivent gérer des milliers de requêtes simultanées tout en offrant des performances élevées à bas coûts.
- Les serveurs streaming qui doivent exécuter des milliers de requêtes multimédia.

3.2.2.2: Node Package Manager (NPM)

NPM est le gestionnaire de paquets spécifique à Node. Il permet d'installer divers paquets et de résoudre leurs différentes dépendances. Cela signifie qu'il est possible de spécifier toutes les dépendances d'un projet dans un seul fichier appelé **package.json**.

NPM est fourni avec une excellente interface de ligne de commande (CLI) qui permet de gérer ces dépendances. [24]

Un simple fichier **package.json** pourrait ressembler à ceci:

```
{
  "name":
  "sample-project",
  "version": "1.0.0",
  "main": "app.js",
  "scripts": {
    "start": "ng serve"
  },
  "description": "sample
project",
  "dependencies": {
    "@angular/animations":
```

```
"7.2.13"  
},  
"devDependencies": {  
  "@angular/cli":  
    "7.3.8"  
},  
}
```

name: est le nom du projet.

version: la version du projet.

main: définit le point d'entrée de l'application.

dependencies: contient toutes les dépendances du projet.

devDependencies: contient toutes les dépendances de développement du projet.

NPM CLI:

- **npm init**
Initialiser un nouveau fichier package.json.
- **npm install**
Extraire du référentiel public de NPM et installer toutes les dépendances spécifiées dans le fichier package.json de notre projet.
- **npm install [-g | -s | --save-dev] package_name**
Extraire et installer le paquet nommé 'package_name' à partir du référentiel public de NPM.
-g: installer le paquet globalement.
-s: l'ajouter à l'objet de dépendances du fichier package.json.
--save-dev: l'ajouter à l'objet devDependencies du fichier package.json.

- **npm start**

Rechercher l'objet **scripts** dans le fichier package.json et si trouvé, rechercher un script dont la clé est **start** et exécuter la commande spécifiée comme valeur. Sinon, exécuter la commande **node server.js** .

3.2.2.3: Développer des applications élégantes Node.js en utilisant NestJS

NestJs est une plate-forme Node.js côté serveur permettant de créer des applications efficaces, fiables et évolutives. Fortement inspiré par **Angular**, NestJs a été introduit pour résoudre le problème architectural de Node.js en donnant aux applications serveur une structure modulaire pour organiser le code en modules séparés.

NestJs est basé sur ExpressJs et fournit un niveau d'abstraction supérieur à ce dernier, ce qui permet une utilisation facile des modules tiers disponibles pour chaque plate-forme.

Cela semble très intéressant. Cependant, afin de constater tous les avantages de NestJs et de mettre les valeurs ajoutées par ce dernier en évidence, une introduction d'ExpressJs s'impose. [25]

Expressjs

[26] Express est un framework d'applications Web, minimal et flexible, basé sur Node.js. Il fournit un ensemble fonctionnalités robustes pour les applications Web et mobiles.

Nous pouvons considérer Express comme étant une couche développée en se basant sur Node.js qui permet de gérer un serveur, des modules HTTP et des routes.

Cela étant dit, nous pouvons à présent identifier clairement les avantages de NestJs, et qui sont :

- Envelopper les gestionnaires de routes avec des blocs try ... catch.
- Rendre les gestionnaires de routes asynchrones.
- Créer une route distincte pour chaque contrôleur.
- Faire la liaison des middlewares de gestion des erreurs.

3.2.3: MongoDB

[27] MongoDB est un gestionnaire de bases de données multi-plateforme orienté documents. Il est actuellement considéré comme étant le SGBD de type NoSQL le plus répandu pour diverses raisons, la première étant le fait qu'il fasse partie des composants de la pile logicielle MEAN, à code source ouvert. Il possède également des fonctionnalités intégrées impressionnantes qui en font un excellent choix pour les entreprises qui ont besoin d'un accès rapide et flexible à leurs données, que ce soit pour prendre des décisions en temps réel, à la volée, ou pour permettre une utilisation adaptée aux besoins des utilisateurs, en se basant sur certaines données. Mongo a plusieurs atouts, parmi lesquels nous pouvons citer :

- L'enregistrement et le stockage de quantités énormes de données qui peuvent avoir une structure minimale ou ne pas en avoir carrément.
- Mongo est un SGBD qui ne nécessite pas de schéma des données. Ceci dit, il n'entraîne aucun problème de migration, car le schéma d'une base de données Mongo est représenté par son code.
- Il est très facile à intégrer et à utiliser, en particulier grâce à la manière dont il stocke les données en format BSON (couple clé/valeur). Il est aussi très similaire au format JSON.
- Il a une riche documentation ainsi qu'une communauté importante.
- L'extension des BDD Mongo est facile et peu coûteuse.

3.2.3.1: Mongoose

Mongoose est une bibliothèque ODM (Object Data Modeling) qui sert de passerelle entre MongoDB et Node.js. Elle permet de définir des objets javascript (JSON) avec un schéma fortement typé lié à un document MongoDB au format BSON. Elle s'occupe aussi de la gestion des relations entre les données.

Schéma Mongoose: C'est une structure qui est liée à une collection Mongo.

Modèle Mongoose : Il englobe le schéma et fournit une interface permettant de créer, interroger, mettre à jour, supprimer des enregistrements, etc., de la base de données. [28]

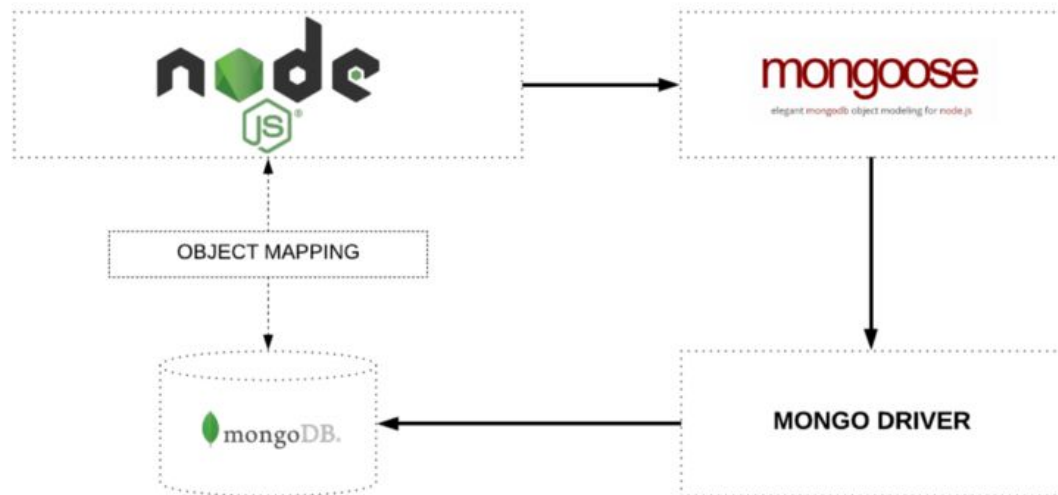


Figure 3.6: la communication entre une application nodejs et Mongodb à l'aide de mongoose [28]

3.2.3.2: Mlab (Database-as-a-Service)

Database-as-a-Service (DBaaS) est un modèle de service de l'informatique en nuage qui offre aux utilisateurs un accès aux bases de données sans qu'ils aient besoin d'installer ou de configurer quoi que ce soit, car tout est géré par le fournisseur DBaaS.

Mlab est un DBaaS MongoDB qui offre un grand nombre de fonctionnalités, telles que des plans tarifaires exceptionnels et une capacité de stockage de 500 Mo gratuite, une interface Web simple à utiliser pour gérer la base de données, une sécurité avancée et une restauration des sauvegardes Simple. [29]

3.3: L'architecture de l'application

Notre application est une application client-serveur. L'application Angular (client) demande des données (json) de l'API NestJs (serveur) en utilisant le protocole HTTP et, si authentifié (à l'aide de JWT), le serveur récupérera les données de MongoDB à l'aide d'un ODM (Mongoose dans notre cas) et le renverra au client via le protocole HTTP. Le serveur peut également extraire des données de l'API GCP (Google Cloud Platform) si nécessaire.

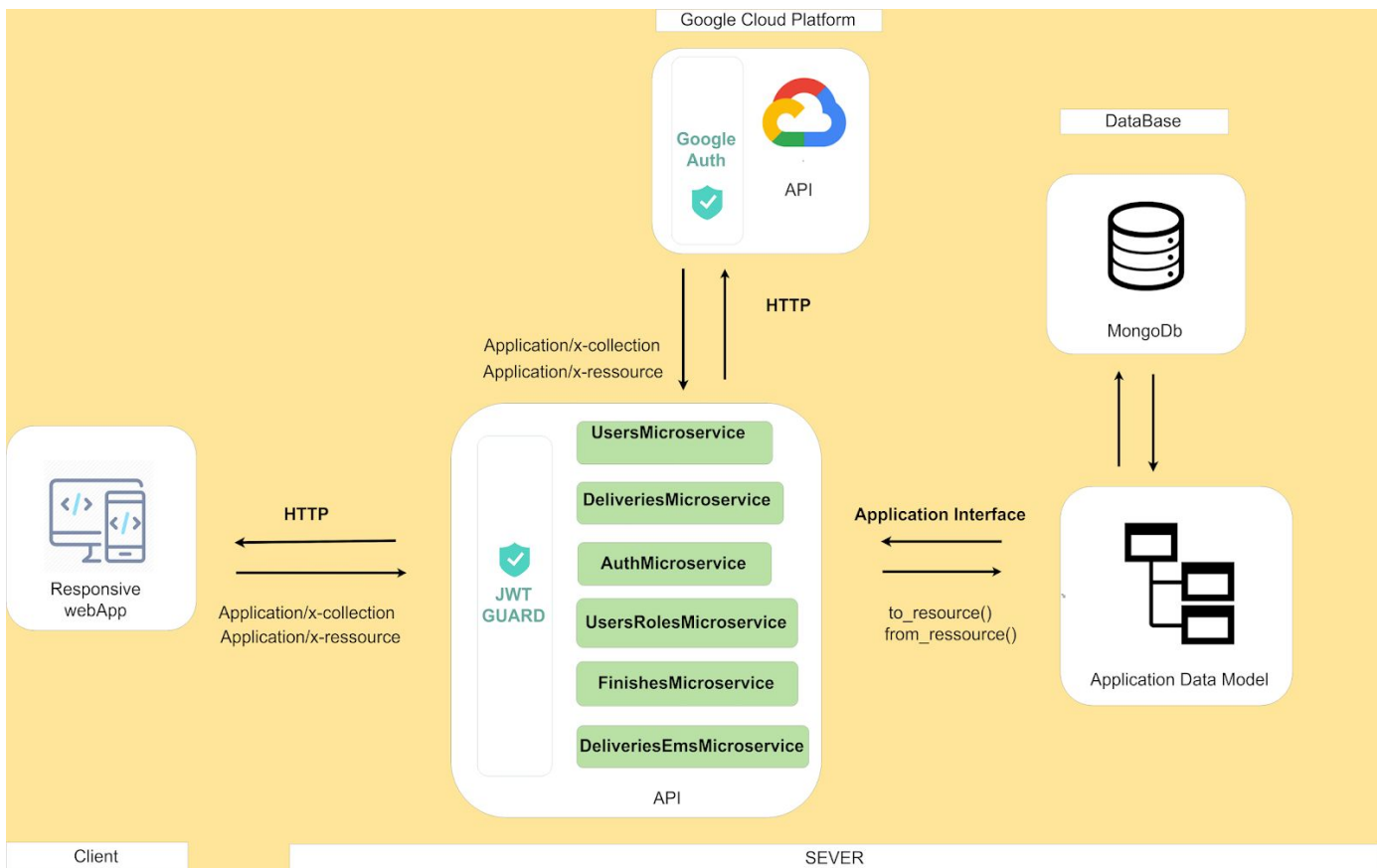


Figure 3.7: L'architecture de l'application

3.4: Maquettes du système

Les maquettes suivantes représentent la page de création d'une nouvelle commande :

The screenshot shows the 'Traixto' dashboard in Google Chrome. The left sidebar contains navigation links: 'Gerer les commandes', 'Liste des commandes', 'Effectuer une commande', 'Factures', and 'Traitement de documents'. The main content area is titled '1 Information du produit' and '2 Validation de la commande'. The 'Information du produit' section includes several selection options: 'Format de votre tirage' (A4, A3, A5), 'Face d'impression' (Recto, Recto-verso), 'Tirage' (Noir, Couleur), and 'Finition' (Sans Finition, Agrafe, Reliure spirale plastique, Pliée et agrafée (brochure), Reliure collée simple, Couverture rigide). Below these options is a field for 'Nombre total de pages de votre original'.

Figure 3.8: Création d'une nouvelle commande (1)

The screenshot shows the 'Traixto' dashboard in Google Chrome. The left sidebar is the same as in the previous image. The main content area is titled '1 Information du produit' and '2 Validation de la commande'. The 'Information du produit' section is now a large gray box with the text 'Déposer vos fichier ici * ou bien utiliser le bouton d'upload'. Below this box is a file selection area showing 'chapitre2: analyse et conception du système <<Traixto>>.pdf'. Below the file selection area are buttons for 'Cancel', 'Pause', and 'Resume'. At the bottom, there is an 'upload *' button and a progress bar showing 'progress: 100%'. A 'Suivant' button is located at the bottom right.

Figure 3.9: Création d'une nouvelle commande (2)

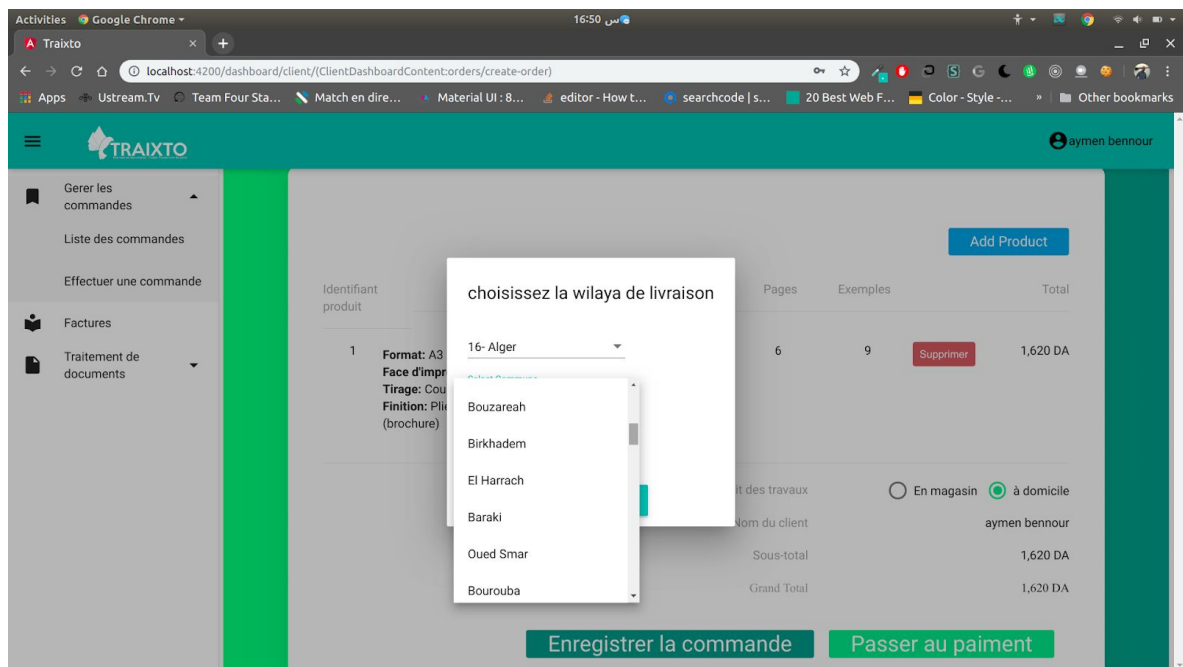


Figure 3.10: Création d'une nouvelle commande (3)

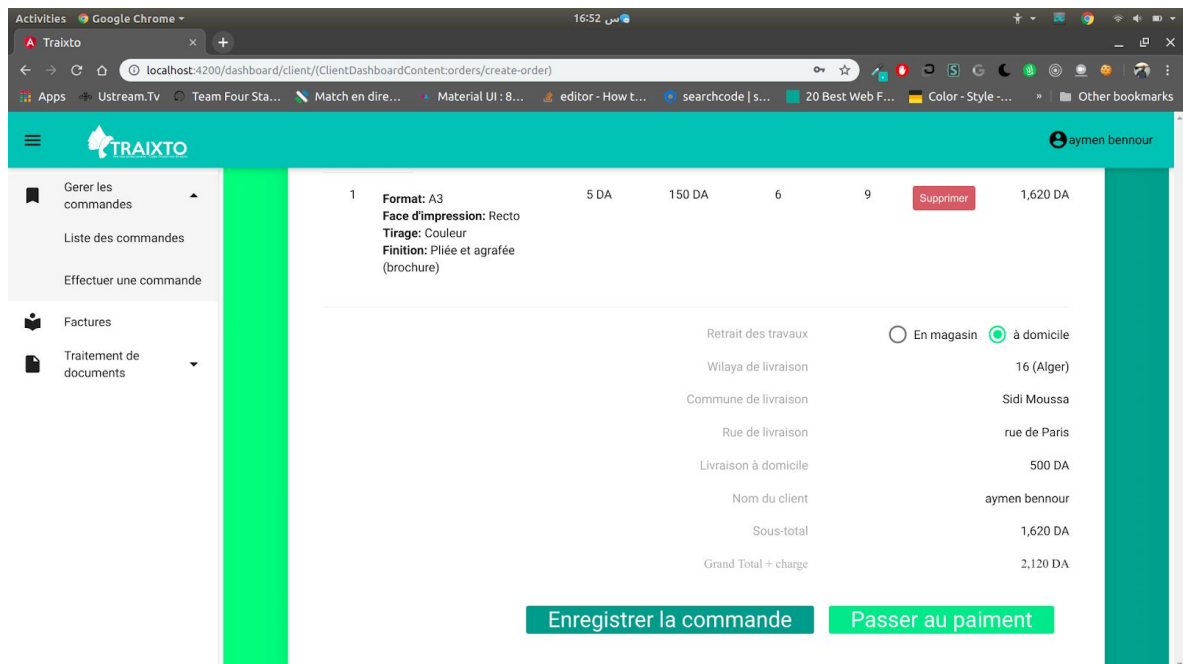


Figure 3.11: Création d'une nouvelle commande (4)

La maquette suivante (Figure 3.12) représente la page de la liste des commandes :

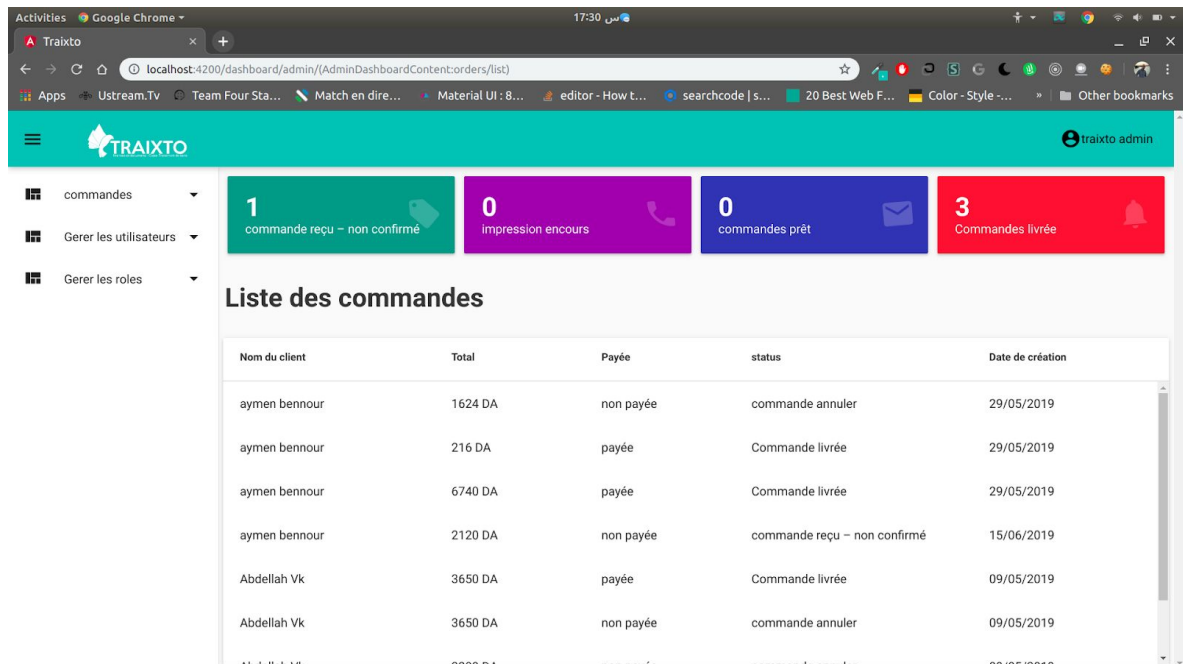


Figure 3.12: la page de la liste des commandes

La maquette suivante représente la page de connexion :

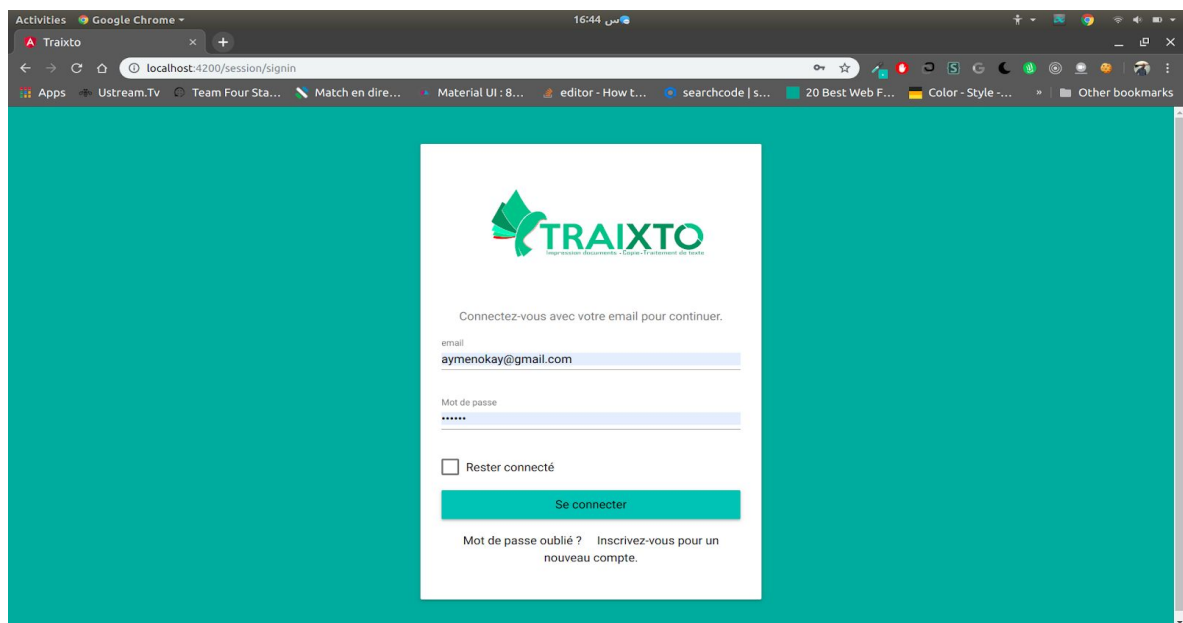


Figure 3.13: la page de connexion

3.5: L'implémentation de « Traixito »

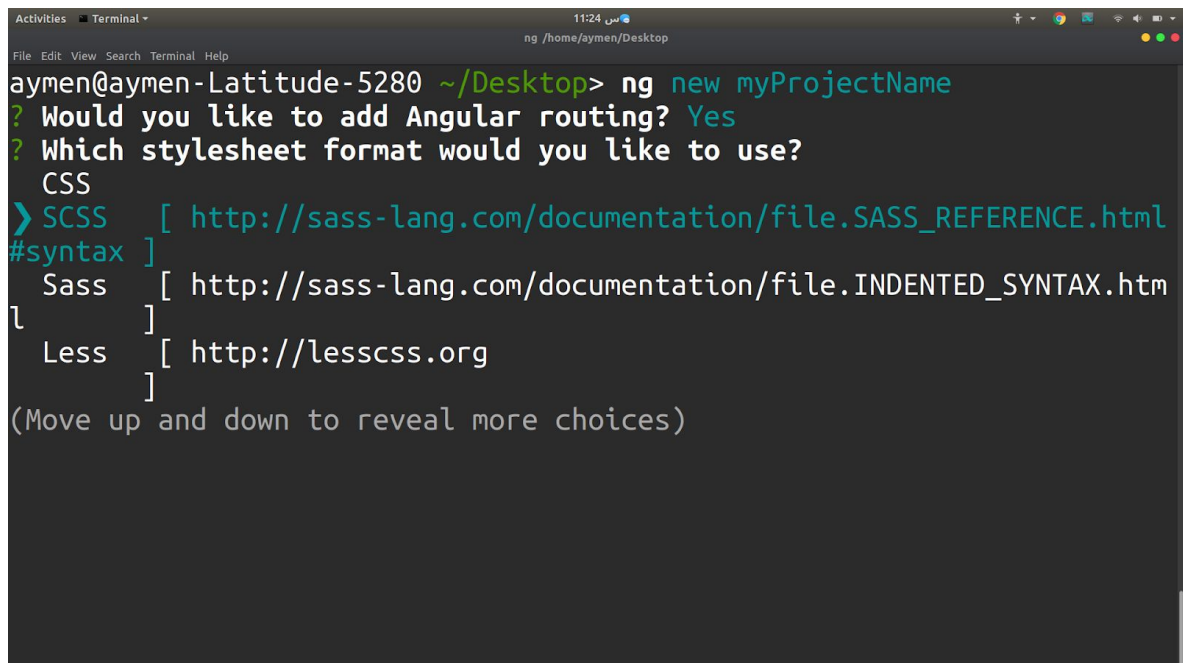
3.5.1: Installation d'Angular

Nous pouvons utiliser NPM pour installer l'interface en ligne de commande (CLI) d'Angular comme tout autre paquet Npm à l'aide de la commande suivante :

```
npm install -g @angular/cli
```

Ensuite, nous pouvons utiliser la CLI d'Angular pour créer un nouveau projet Angular:

```
ng new myProjectName
```



```
aymen@aymen-Latitude-5280 ~/Desktop> ng new myProjectName
? Would you like to add Angular routing? Yes
? Which stylesheet format would you like to use?
  CSS
  > SCSS [ http://sass-lang.com/documentation/file.SASS_REFERENCE.html
#syntax ]
  Sass [ http://sass-lang.com/documentation/file.INDENTED_SYNTAX.htm
[
  Less [ http://lesscss.org
]
(Move up and down to reveal more choices)
```

Figure 3.14: créer un nouveau projet Angular avec la CLI Angular

Nous devons choisir si nous souhaitons ajouter un routage Angular à notre projet ou non, ainsi que le format des feuilles de style à utiliser: CSS, SCSS, Sass ou Less.

3.5.2: Les composants principaux d'Angular

NgModules

Ils sont considérés comme étant l'unité qui lie une application ou des fonctionnalités à une autre application. Toutes les applications Angular doivent avoir une racine NgModule. Il est également recommandé de décomposer les fonctionnalités individuelles d'une application dans leurs propres modules.

Pour générer un NgModule, nous utilisons la commande suivante:

```
ng generate module moduleName
```

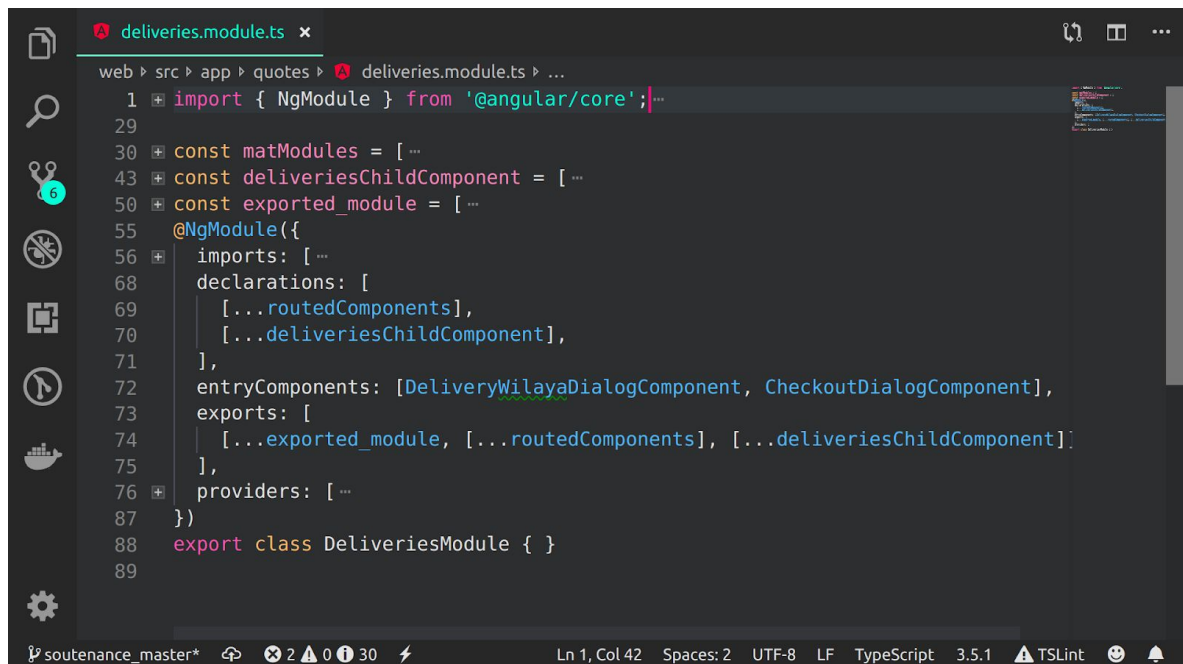


Figure 3.15: l'architecture d'un module Angular (DeliveriesModule)

Dans ce qui suit, nous allons donner une brève explication de chaque entrée du décorateur **@NgModule** :

imports : Doivent être utilisés pour importer d'autres modules qui peuvent être utilisés par les composants faisant partie de ce module.

declarations (Déclarations): Ils spécifient les composants (components), les pipes et les directives de ce module.

entryComponents (Entry Components): Ils représentent les composants qui ne peuvent pas être trouvés par le compilateur Angular pendant la compilation, puisqu'ils sont créés dynamiquement. Un bon exemple serait les Material Dialogs d'Angular (voir l'exemple

cité ci-dessus).

exports : Ils servent à exporter les membres de ce module à être utilisés par les composants d'autres modules.

providers (Fournisseurs) : où les injectables tels que **Services**, Intercepteurs Http (**HttpInterceptors**) et Route Guards (**RouteGuards**) doivent être déclarés.

bootstrap : C'est là où le composant racine, souvent appelé **AppComponent**, doit être défini. bootstrap ne doit contenir qu'un seul membre et il ne doit être défini que dans le module principal de l'application.

Components (Les Composants)

Le bloc fondamental qui constitue une application Angular se compose de 02 parties principales :

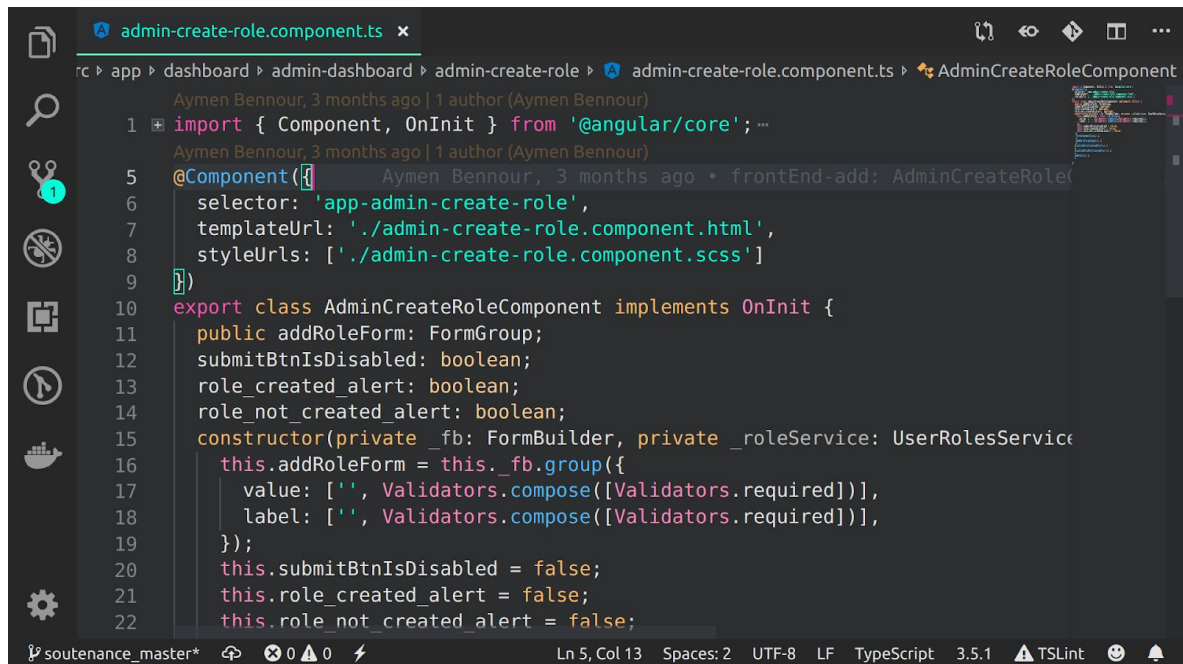
1. @Component

Un décorateur qui marque une classe comme étant un composant Angular et fournit des métadonnées de configuration qui déterminent la manière dont le composant doit être traité, instancié et utilisé au moment de l'exécution.

2. Controller

Une classe qui contrôle la logique du composant. Pour générer un composant, nous utilisons la commande suivante :

```
ng generate component componentName
```



```
1 import { Component, OnInit } from '@angular/core';
2
3 @Component({
4   selector: 'app-admin-create-role',
5   templateUrl: './admin-create-role.component.html',
6   styleUrls: ['./admin-create-role.component.scss']
7 })
8
9 export class AdminCreateRoleComponent implements OnInit {
10   public addRoleForm: FormGroup;
11   submitBtnIsDisabled: boolean;
12   role_created_alert: boolean;
13   role_not_created_alert: boolean;
14   constructor(private _fb: FormBuilder, private _roleService: UserRolesService) {
15     this.addRoleForm = this._fb.group({
16       value: ['', Validators.compose([Validators.required])],
17       label: ['', Validators.compose([Validators.required])],
18     });
19     this.submitBtnIsDisabled = false;
20     this.role_created_alert = false;
21     this.role_not_created_alert = false;
22   }
23 }
```

Figure 3.16: l'architecture d'un Composant (Component) Angular
(AdminCreateRoleComponent)

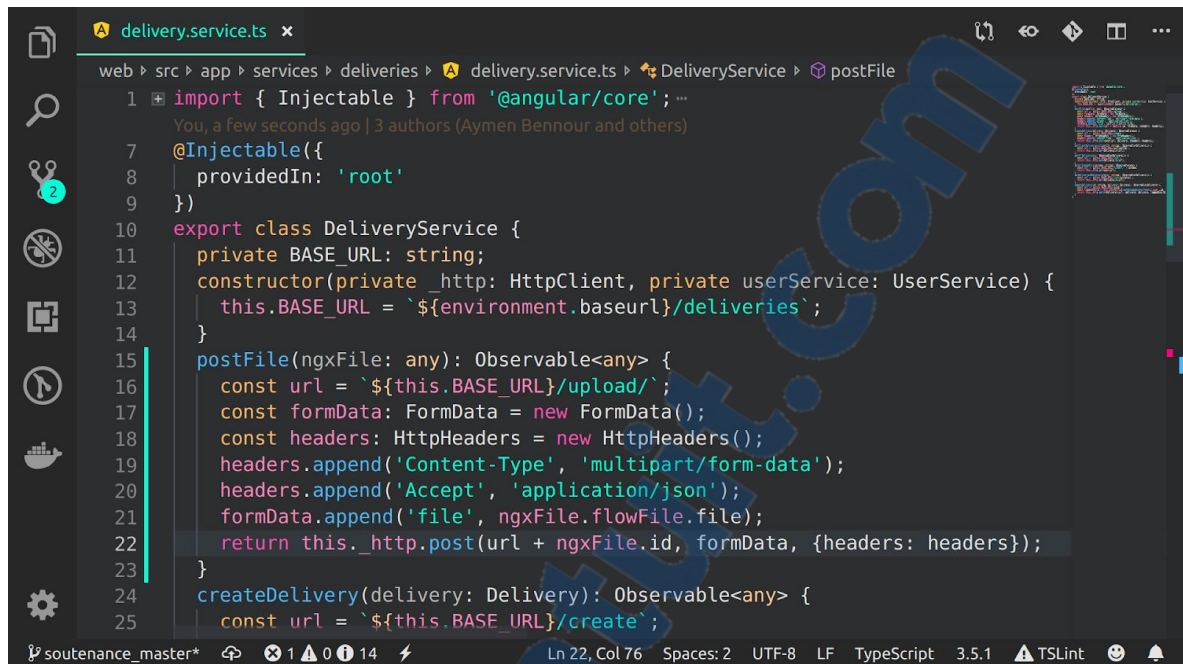
Les services

Un service est une classe dont le rôle est bien déterminé, elle est utilisée pour des fonctionnalités qui :

- Sont indépendantes de tout autre composant.
- Fournissent une logique ou des données partagées entre les composants.
- Encapsulent les interactions externes (telles que les appels HTTP d'une API).

Pour générer un service, nous utilisons la commande suivante :

```
ng generate service serviceName
```

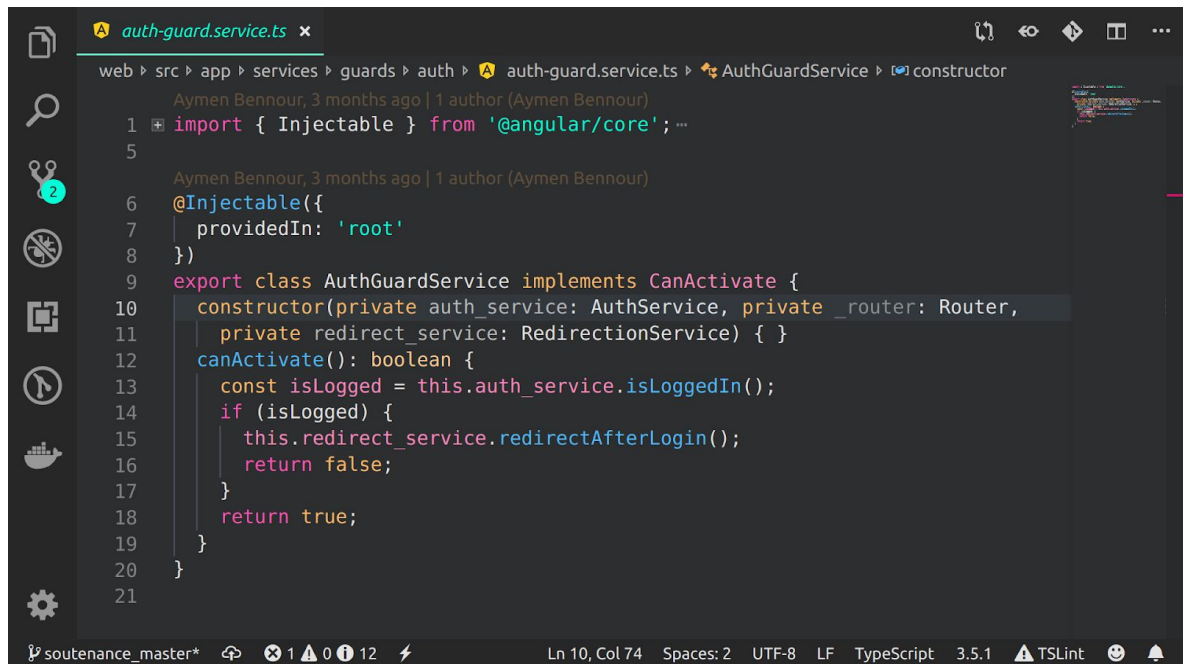


```
1 import { Injectable } from '@angular/core'; ...
2
3 You, a few seconds ago | 3 authors (Ayman Bennour and others)
4
5
6
7 @Injectable({
8   providedIn: 'root'
9 })
10 export class DeliveryService {
11   private BASE_URL: string;
12   constructor(private _http: HttpClient, private userService: UserService) {
13     this.BASE_URL = `${environment.baseurl}/deliveries`;
14   }
15   postFile(ngxFile: any): Observable<any> {
16     const url = `${this.BASE_URL}/upload/`;
17     const formData: FormData = new FormData();
18     const headers: HttpHeaders = new HttpHeaders();
19     headers.append('Content-Type', 'multipart/form-data');
20     headers.append('Accept', 'application/json');
21     formData.append('file', ngxFile.flowFile.file);
22     return this._http.post(url + ngxFile.id, formData, {headers: headers});
23   }
24   createDelivery(delivery: Delivery): Observable<any> {
25     const url = `${this.BASE_URL}/create`;
```

Figure 3.17: l'architecture d'un Service Angular (DeliveryService)

Les Route Guards

Les route guards d'Angular sont des interfaces qui implémentent l'interface **CanActivate** et qui peuvent indiquer au routeur s'il doit autoriser ou non la navigation vers une route demandée en fonction d'une expression booléenne.



```
1 import { Injectable } from '@angular/core'; ...
5
6 @Injectable({
7   providedIn: 'root'
8 })
9 export class AuthGuardService implements CanActivate {
10   constructor(private auth_service: AuthService, private _router: Router,
11     private redirect_service: RedirectionService) { }
12   canActivate(): boolean {
13     const isLoggedIn = this.auth_service.isLoggedIn();
14     if (isLoggedIn) {
15       this.redirect_service.redirectAfterLogin();
16       return false;
17     }
18     return true;
19   }
20 }
21
```

Figure 3.18: l'architecture d'un Route Guard Angular (AuthGuardService)

Pour générer un route guard, nous utilisons la commande suivante :

ng generate guard guardName

Les intercepteurs HTTP

Un intercepteur HTTP permet d'intercepter des requêtes et des réponses HTTP afin de pouvoir les modifier. Un des cas d'utilisation des intercepteurs consiste à ajouter un en-tête d'autorisation (en utilisant JWT) pour l'authentification à toutes les requêtes.

```

token-interceptor.service.ts
web › src › app › services › interceptors › token-interceptor.service.ts › TokenInterceptorService
4  @Injectable({
5    providedIn: 'root'
6  })
7  export class TokenInterceptorService implements HttpInterceptor {
8    constructor(private _auth_service: AuthService) { }
9    intercept(req: HttpRequest<any>, next: HttpHandler) {
10      const login: Boolean = this._auth_service.isLoggedIn();
11      if (login) {
12        /**
13         * if login we add the 'authorization' key with the value of
14         * 'Bearer token' to the req headers */
15        return next.handle(
16          req.clone({
17            headers: req.headers.append('Authorization',
18              `Bearer ${this._auth_service.getToken()}`)
19          })
20        );
21      }
22      // if not we don't add anything
23      return next.handle(req);
24    }
  }

```

Figure 3.19: l'architecture d'un intercepteur HTTP Angular (TokenInterceptorService)

Les Pipes personnalisé

Les Pipes personnalisé nous permettent essentiellement de créer une fonction, qui accepte une entrée et retourne une sortie différente en passant par plusieurs transformations. La logique du pipe est définie dans une fonction appelée **transform**.

Un pipe peut être intégré à un code HTML de la manière suivante :

```
<span>{{403 | httpErrorHandling}}</span>
```

```
<!--Ceci va afficher 'merci de vérifier votre email et essayez à nouveau!' -->
```

```

1 import { Pipe, PipeTransform } from '@angular/core';
2
3 @Pipe({name: 'httpErrorHandling'})
4 export class HttpErrorHandlingPipe implements PipeTransform {
5   transform(error_status: number): string {
6     switch (error_status) {
7       case 409:
8         return 'L\'utilisateur existe déjà! svp essayer avec un autre!';
9       case 401:
10        return 'mauvaise combinaison email / mot de passe veuillez réessayer!';
11       case 403:
12        return 'merci de vérifier votre email et essayez à nouveau!';
13       default:
14        return `error status code is: ${error_status}`;
15     }
16   }
17 }

```

Figure 3.20: l'architecture d'un Pipe personnalisé Angular (HttpErrorHandlingPipe)

3.5.3: API RESTful de Traixto

Notre API est composée de plusieurs microservices qui sont:

1. Les microservices utilisateurs (UsersMicroservice)

Point D'entrée: '/users'

Module: usersModule

Méthode	Route	Action	Réponse
GET	''	getUsers()	Tous les utilisateurs et le code de statut 200
POST	''	createUser(User)	L'utilisateur créé et le code de statut 201
GET	':id'	getUser(id)	L'utilisateur avec l'identifiant donné et

			le code de statut 200
PATCH	'id'	updateUser(id, User)	L'utilisateur mis à jour et le code de statut 200
DELETE	'id'	deleteUser(id)	L'utilisateur supprimé et le code de statut 200

Tableau 3.1: Documentation sur la API UsersMicroService

2. Le microservice de distribution (DeliveriesMicroservice)

Point D'entrée: '/deliveries

Module: DeliveriesModule

Méthode	Route	Action	Réponse
GET	''	getDeliveries()	Toutes les distributions et le code de statut 201
GET	'id'	getDelivery(id)	La distribution avec l'identifiant donné et le code de statut 200
PATCH	'id'	updateDelivery(id, Delivery)	La distribution mise à jour et le code de statut 200

DELETE	'id'	deleteDelivery(id)	La distribution supprimée et le code de statut 200
POST	''	createDelivery(User)	La distribution créée et le code de statut 201
POST	'upload/:id'	uploadFile(bucketName, file, id)	Le fichier chargé avec le code de statut 201
GET	('fileUrl/:photoId')	getPhotoUrl(bucketName, id)	L'url de la photo avec le code de statut 200
GET	('ocr/:photoId')	extractTextFromPhoto(id)	Le texte de la photo donnée par un identifiant avec le code de statut 200

Tableau 3.2: Documentation sur la API DeliveriesMicroservice

3. Le microservice d'authentification (AuthMicroService)

Point D'entrée: '/auth'

Module: AuthModule

Méthode	Route	Action	Réponse
POST	'login'	login(LoggedUser)	JWT token et le code de statut 200 si l'accréditation est valide

POST	'forget-password'	forgetPwd(email)	Le code de statut 200 avec le message ' Un email a été envoyé pour récupérer le mot de passe'
POST	'reset-password'	resetPwd(resetToken, pwd)	Le code de statut 200 avec le message 'votre mot de passe a été mis à jour avec succès'
POST	'confirm-email'	confirmEmail(confirmationToken)	Le code de statut 200 avec le message 'Votre email est validé'

Tableau 3.3: Documentation sur la API AuthMicroService

4. Microservice pour Les rôles des utilisateurs (UsersRolesMicroService)

Point D'entrée: '/user-roles'

Module: UsersRolesModule

Méthode	Route	Action	Réponse
GET	''	getRoles()	Tous les rôles et le code de statut 200
POST	''	createRole(Role)	Le rôle créé et le statut 201
GET	':id'	getRole(id)	Le rôle avec l'identifiant donné et

			le code de statut 200
PATCH	:id	updateRole(id, Role)	Le rôle mis à jour et le code de statut 200
DELETE	:id	deleteRole(id)	Le rôle supprimé et le code de statut 200

Tableau 3.4: Documentation sur la API UsersRolesMicroService

5. Le microservice de finitions (FinishesMicroService)

Point d'entrée: '/finishes'

Module: FinishesModule

Méthod	Route	Action	Réponse
GET	' '	getFinishes()	Toutes les finitions et le code de statut 200
POST	' '	createFinishes(Finish)	La finition créée et le code de statut 201
GET	:id	getFinish(id)	La finition avec l'identifiant donné et le code de statut 200
PATCH	:id	updateFinish(id, Finish)	La finition mise à jour et le code de statut 200
DELETE	:id	deleteFinish(id)	La finition supprimée et le code de statut 200

Tableau 3.5: Documentation sur la API FinishesMicroService

6. Le microservice de livraison Ems (DeliveriesEmsMicroService)

Point d'entrée: '/deliveries-ems'

Module: DeliveriesEmsModule

Method	Route	Action	Response
GET	' '	getDeliveriesEms()	Toutes les DeliveriesEms et le code de statut 200
POST	' '	createDeliveryEms(DeliveryEms)	DeliveriesEms créées et le code de statut 201
GET	':id'	getDeliveryEms(id)	DeliveriesEms de l'identifiant donné et le code de statut 200
PATCH	':id'	updateDeliveryEms(id, DeliveryEms)	DeliveriesEms mise à jour et le code de statut 200
DELETE	':id'	deleteDeliveryEms(id)	DeliveriesEms supprimée et le code de statut 200

Tableau 3.6: Documentation sur la API DeliveriesEmsMicroService

3.5.4: Installation de NestJs

nous pouvons installer NestJs de la même manière que nous avons installé Angular, en utilisant la commande NPM suivante:

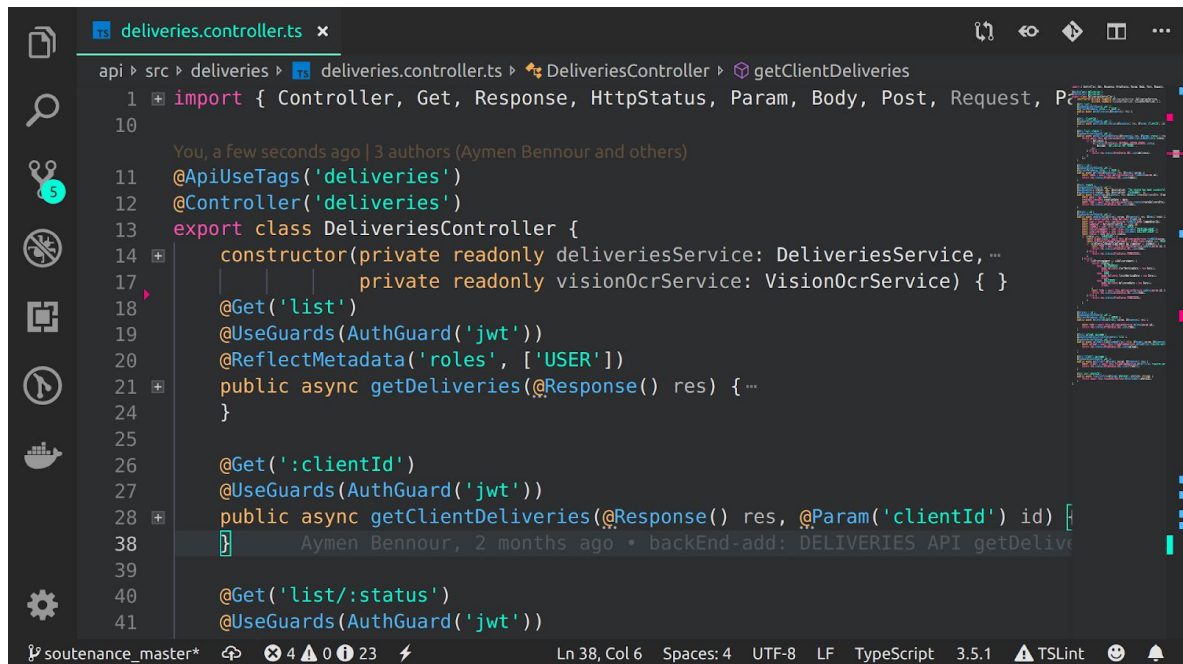
```
npm install -g @nestjs/core
```

3.5.5: Les principaux éléments constitutifs de NestJs

NestJs utilise à peu près les mêmes éléments (services, pipes, modules ...) qu'Angular avec quelques fonctionnalités supplémentaires qui sont :

Les contrôleurs

Les contrôleurs sont responsables du traitement des demandes entrantes et du renvoi des réponses au client.



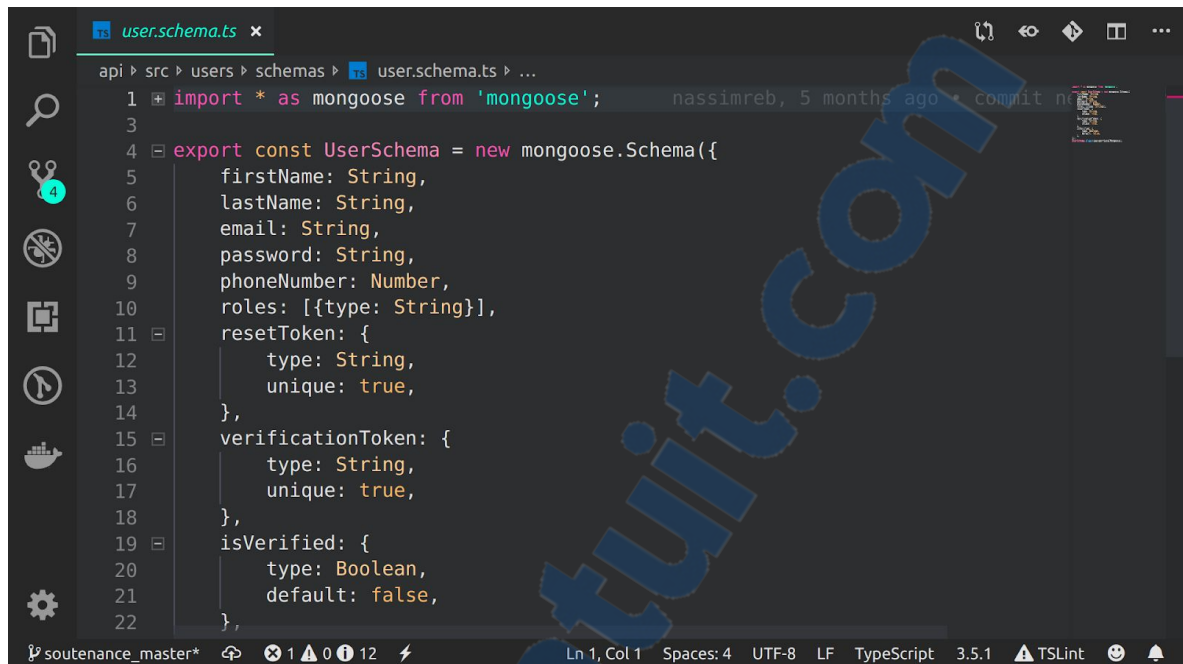
```
1  import { Controller, Get, Response, HttpStatus, Param, Body, Post, Request, Response } from '@nestjs/common';
10
11  @ApiUseTags('deliveries')
12  @Controller('deliveries')
13  export class DeliveriesController {
14      constructor(private readonly deliveriesService: DeliveriesService, ...
15                  private readonly vision0crService: Vision0crService) { }
16
17      @Get('list')
18      @UseGuards(AuthGuard('jwt'))
19      @ReflectMetadata('roles', ['USER'])
20      public async getDeliveries(@Response() res) { ...
21      }
22
23      @Get('/:clientId')
24      @UseGuards(AuthGuard('jwt'))
25      public async getClientDeliveries(@Response() res, @Param('clientId') id) {
26      }
27
28      @Get('list/:status')
29      @UseGuards(AuthGuard('jwt'))
```

Figure 3.21: l'architecture d'un contrôleur (Controller) NestJs (DeliveriesController)

Le but d'un contrôleur est de recevoir des demandes spécifiques pour l'application. Le mécanisme de routage contrôle quel contrôleur reçoit quelles demandes. Fréquemment, chaque contrôleur dispose de plusieurs routes et différentes routes peuvent effectuer différentes actions.

User schema

Notre schéma mongoose 'USER' ressemble au suivant:



```
1 import * as mongoose from 'mongoose';
2
3
4 export const UserSchema = new mongoose.Schema({
5   firstName: String,
6   lastName: String,
7   email: String,
8   password: String,
9   phoneNumber: Number,
10  roles: [{type: String}],
11  resetToken: {
12    type: String,
13    unique: true,
14  },
15  verificationToken: {
16    type: String,
17    unique: true,
18  },
19  isVerified: {
20    type: Boolean,
21    default: false,
22  },
23 });
```

Figure 3.22: schéma mongoose 'USER'

Conclusion

Le but de ce chapitre était de clarifier l'architecture et les aspects techniques tels que l'interface frontend / backend et l'API RESTful de notre application.

Conclusion générale

Ce projet de fin d'études «Traixito» est une plate-forme d'impression et de reconnaissance optique de caractères en ligne qui a été créée dans le but de permettre la conversion de documents sous format papier (photos, images) en format électronique modifiable (pdf, word) ainsi que l'impression de mémoires de projets de fin d'études en ligne. L'une des raisons qui ont motivé et encouragé l'élaboration de ce projet est l'absence de telles plates-formes et d'autres innovations technologiques sur le marché algérien.

Afin d'avoir créé cette plate-forme, nous sommes en premier lieu passés par la phase de validation du problème, qui consiste à rechercher des plates-formes existantes similaires et à les étudier pour mieux comprendre ce qui est déjà fait et comment nous pouvons les améliorer de telle sorte à les personnaliser en fonction des besoins de nos clients. Ensuite, nous sommes passés à la phase d'analyse et de conception où nous avons discuté en détails des acteurs et des exigences fonctionnelles et non fonctionnelles de notre système. Durant cette phase, nous avons également donné un aperçu détaillé de la structure de notre application. En ce qui concerne la phase finale, nous avons abordé les aspects techniques de notre travail. Ce dernier a été implémenté en utilisant les dernières technologies, telles que la **pile MEAN** pour le développement en full-stack en JavaScript, **Git** en tant que système de gestion de versions et **GitLab** en tant qu'hébergeur, **GCP** pour les services de **stockage en nuage** et les fonctionnalités de ROC (Reconnaissance Optique de Caractères) grâce à l'**API VISION**.

Pour finir, nous souhaitons mettre en évidence l'énorme expérience pratique que nous avons acquise durant la réalisation de ce projet de fin d'études, qui nous a également donné l'occasion de mettre à profit nos connaissances en génie logiciel.

Perspectives

Avec cette première version, nous voulions commencer modestement, mais nous ne comptons pas nous arrêter à ce niveau, bien évidemment. Ceci dit, ce projet sera prochainement complété par les fonctionnalités suivantes :

- **Relecture et correction:** Faites relire et corriger vos documents (mémoires, thèses, rapports de stage...) par un expert académique.
- **Logiciel anti-plagiat:** Scanner vos documents et éliminer le plagiat.
- **Générateur APA:** La manière la plus simple pour générer une liste de références suivant les normes APA.

Références bibliographiques

[1]

<https://searchcontentmanagement.techtarget.com/definition/OCR-optical-character-recognition> :: What is OCR (optical character recognition)? - Definition from WhatIs.com :: consulté le 15/06/2019 à 23 h 07

[2] <https://www.scribbr.fr> :: Scribbr - Le point de contrôle de votre mémoire :: consulté le 15/06/2019 à 23 h 07

[3] <https://www.corep.fr/> :: Imprimer en ligne : impression en ligne, copie et reproduction :: consulté le 15/06/2019 à 23 h 07

[4] <https://www.nuance.com/index.html> :: Nuance - PDF, Customer Service, HIM, and Speech Recognition Solutions | Nuance :: consulté le 15/06/2019 à 23 h 07

[5] <https://napoleoncat.com/stats/facebook-users-in-algeria/2018/09> :: Social Media Demographics - Facebook :: consulté le 16/06/2019 à 20 h 32

[6] <https://www.lucidchart.com/pages/uml-use-case-diagram> :: use case diagram definition :: consulté le 10/06/2019 à 19 h 04

[7] <https://www.lucidchart.com/pages/uml-class-diagram> :: class diagram definition :: consulté le 10/06/2019 à 19 h 58

[8] <https://www.techopedia.com/definition/438/clientserver-architecture> : what is a client-server architecture :: consulté le 10/06/2019 à 21 h 04

[9]

<https://www.quora.com/What-are-some-advantages-and-disadvantages-of-client-server-networks> : Client Server Network : Advantages and Disadvantages :: consulté le 10/06/2019 à 21 h 35

[10] <https://dev.to/benhayehudi/a-brief-introduction-to-rest-apis-172> : A Brief Introduction to REST APIs :: consulté le 15/06/2019 à 00 h 37

[11]

<https://medium.com/extend/what-is-rest-a-simple-explanation-for-beginners-part-1-introduction-b4a072f8740f> : What is REST — A Simple Explanation for Beginners, Part 1: Introduction :: consulté le 15/06/2019 à 00 h 37

[12]

<https://www.quora.com/What-is-SPA-single-page-application/answer/Damien-Filiatrault> : What is SPA (single page application)? :: consulté le 15/06/2019 à 01 h 30

[13]

<https://medium.com/@NeotericEU/single-page-application-vs-multiple-page-application-2591588efe58> : Single-page application vs. multiple-page application :: consulté le 15/06/2019 à 01 h 30

[14] <https://jwt.io/introduction/> : introduction to jwt :: consulté le 11/06/2019 à 23 h 59

[15]

<https://medium.com/vandium-software/5-easy-steps-to-understanding-json-web-tokens-jwt-1164c0adfcec> : 5 Easy Steps to Understanding JSON Web Tokens (JWT) :: consulté le 11/06/2019 à 23 h 59

[16]

<https://medium.com/vandium-software/5-easy-steps-to-understanding-json-web-tokens-jwt-1164c0adfcec> : Google Cloud Platform (GCP) basics in its own words, tutorials or documentations :: consulté le 24/06/2019 à 20 h 05

[17]

<https://medium.com/google-cloud/what-are-the-google-cloud-platform-gcp-services-285f1988957a> : What are the Google Cloud Platform (GCP) Services? :: consulté le 24/06/2019 à 20 h 05

[18] <https://medium.com/recraftrelic/es5-vs-es6-with-example-code-9901fa0136fc> : ES5 vs ES6 (With example code) :: consulté le 13/06/2019 à 18 h 15

[19] <https://www.techtalko.com/2018/03/17/typescript-the-future-of-javascript/> : TypeScript the Future Of JavaScript !! :: consulté le 13/06/2019 à 23 h 04

[20] <https://medium.com/@wittydeveloper/typescript-learn-the-basics-2f56eb9b02eb> : TypeScript — Learn the basics :: consulté le 13/06/2019 à 23 h 04

- [21] <https://www.techtalko.com/2018/06/10/angular-for-beginners-how-it-works/> : Angular For Beginners || How Angular Actually Works :: consulté le 14/06/2019 à 22 h 23
- [22] <https://www.edureka.co/blog/angular-tutorial/> : Angular Tutorial: Getting Started With Angular 4 :: consulté le 14/06/2019 à 22 h 23
- [23] <https://www.toptal.com/nodejs/why-the-hell-would-i-use-node-js> : Why The Hell Would I Use Node.js? A Case-by-Case Tutorial :: consulté le 14/06/2019 à 12 h 16
- [24] <https://stackoverflow.com/questions/31930370/what-is-npm-and-why-do-i-need-it> : What is NPM and why do I need it? :: consulté le 14/06/2019 à 15 h 48
- [25] <https://docs.nestjs.com/> : Documentation | NestJS - A progressive Node.js web framework :: consulté le 14/06/2019 à 17 h 32
- [26] <https://www.guru99.com/node-js-express.html> : Node.js Express FrameWork Tutorial - Learn in 10 Minutes :: consulté le 14/06/2019 à 17 h 32
- [27] <https://www.upwork.com/hiring/data/should-you-use-mongodb-a-look-at-the-leading-nosql-database/> : Should You Use MongoDB? A Look at the Leading NoSQL Database :: consulté le 12/06/2019 à 20 h 39
- [28] <https://www.freecodecamp.org/news/introduction-to-mongoose-for-mongodb-d2a7aa593c57/> : Introduction to Mongoose for MongoDB :: consulté le 12/06/2019 à 20 h 39
- [29] <https://mlab.com/> : mlab official site :: consulté le 12/06/2019 à 23 h 21

Résumé

Fournir des solutions intelligentes et facile à utiliser, mais aussi de manière innovante, c'est exactement ce que nous avons envisagé avec notre projet de fin d'étude qui prend en compte des besoins réels tels que la transformation des ressources dures et d'ancien format en fichiers numériques équivalents, cela à l'aide des dernières technologies, telles que la reconnaissance optique de caractères (ROC), machine Learning et en fournissant également aux utilisateurs une plate-forme d'impression en ligne professionnelle.

Mots clés

Reconnaissance optique de caractères, ROC, plateformes en ligne, impression documentaire.

Abstract

providing a smart, easy to use solution for a common problem in a new innovative and technological way is exactly what we do vision with our end-of-study project which takes real need such as the need for transforming old and hard format resources to their digital counterparts with the help of the latest technologies the likes of OCR (Optical character recognition) and machine Learning and also providing users with a professional online printing platform.

Keywords

Optical character recognition, OCR, online platforms, documents printing.

ملخص

توفير حل ذكي وسهل الاستخدام ليس فقط بالتقليد ولكن أيضا بطريقة ابتكارية هو بالضبط رؤيتنا من خلال مشروع تخرجنا، والذي يتمحور حول مشاكل واحتياجات حقيقية، مثل تحويل الموارد القديمة والصعبة إلى موارد مطابقة رقمية و حديثة، وذلك بمساعدة أحدث التقنيات مثل التعرف الضوئي على الحروف (التعرف الضوئي على الحروف) والتعلم الآلي ، وأيضا تزويد المستخدمين بمنصة طباعة محترفة على الانترنت.

الكلمات الدالة

التعرف الضوئي على الحروف , المنصات الإلكترونية , طباعة الوثائق.