|  |
| --- |
| Documentation technique |
| Green IA |
| 1. Architecture du système et présentation du projet |
| 1. Développement et technologies utilisées |
| 1. IA et Data 2. Interface utilisateur et déploiement 3. Sécurité, maintenance et support 4. Tests, validation et améliorations 5. Annexes |

**Sommaire :**

[I. Architecture du système et présentation du projet 4](#_Toc172557638)

[1. Présentation du projet 4](#_Toc172557639)

[Contexte 4](#_Toc172557640)

[Objectifs 4](#_Toc172557641)

[2. Qu’est-ce que l’éco-score ? 4](#_Toc172557642)

[Les impacts sont étudiés tout au long du cycle de vie du produit 5](#_Toc172557643)

[3. Vue d’ensemble 6](#_Toc172557644)

[4. Acteurs du système et interactions (utilisateurs, admins, autres) 7](#_Toc172557645)

[5. Diagramme de sequence UML 7](#_Toc172557646)

[6. Use case des fonctionnalités 7](#_Toc172557647)

[code unique + use case + entrées et sortie 7](#_Toc172557648)

[II. Développement et technologies utilisées 7](#_Toc172557649)

[7. Langages de Programmation et normes de codage 7](#_Toc172557650)

[8. Framework et bibliothèques 9](#_Toc172557651)

[9. Systèmes d’exploitation et environnement de travail 12](#_Toc172557652)

[10. Déploiement de nos solutions 13](#_Toc172557653)

[Stratégies de déploiement (blue-green, canary) 13](#_Toc172557654)

[Environnement de déploiement (dev, test, prod) 13](#_Toc172557655)

[Outils de déploiement (CI/CD pipelines, versions) 13](#_Toc172557656)

[Stratégie de scalabilité (verticale et horizontale) 13](#_Toc172557657)

[11. Contenu du projet, fichiers et exécution 13](#_Toc172557658)

[III. IA et Data 13](#_Toc172557659)

[12. Collecte des données 13](#_Toc172557660)

[13. Préparation des données 13](#_Toc172557661)

[14. Entraînement et validation des modèles 13](#_Toc172557662)

[15. Stockage et mise à disposition des données 14](#_Toc172557663)

[16. Dashboard utilisateur 14](#_Toc172557664)

[17. Dashboard global 14](#_Toc172557665)

[18. Pipeline 14](#_Toc172557666)

[IV. Interface utilisateur 15](#_Toc172557667)

[19. Sécurité et authentification 15](#_Toc172557668)

[20. Intégration avec des services externes 15](#_Toc172557669)

[21. Présentation des maquettes 15](#_Toc172557670)

[Page principale et page d’informations 15](#_Toc172557671)

[Dashboards utilisateur et global 16](#_Toc172557672)

[Informations dépôts et collectes 16](#_Toc172557673)

[22. Accessibilité 16](#_Toc172557674)

[23. Chemins utilisateur 17](#_Toc172557675)

[24. Diagrammes de flux 17](#_Toc172557676)

[V. Sécurité, maintenance et support 17](#_Toc172557677)

[25. Contraintes de sécurité 17](#_Toc172557678)

[26. Gestion des données sensibles 17](#_Toc172557679)

[27. Conformité et réglementations 18](#_Toc172557680)

[28. Plan de maintenance et MCO 18](#_Toc172557681)

[29. Surveillance et monitoring 18](#_Toc172557682)

[Rapport d’état de santé du système 18](#_Toc172557683)

[Rapport d’évolution des données 18](#_Toc172557684)

[Notifications aux administrateurs 18](#_Toc172557685)

[30. Plan des mises à jour et support 18](#_Toc172557686)

[VI. Tests, validation et améliorations 18](#_Toc172557687)

[31. Critères d’acceptation pour chaque fonctionnalité 18](#_Toc172557688)

[32. Stratégie de tests 20](#_Toc172557689)

[33. Tests unitaires 20](#_Toc172557690)

[34. Tests d'intégration 20](#_Toc172557691)

[35. Tests de performance 20](#_Toc172557692)

[36. Améliorations envisagées et versioning 20](#_Toc172557693)

[37. Confidentialité et accès 20](#_Toc172557694)

[VII. Annexes 20](#_Toc172557695)

[38. Glossaire 20](#_Toc172557696)

[39. Documents applicables 21](#_Toc172557697)

[40. Diffusion du document 21](#_Toc172557698)

[41. Historique des modifications 21](#_Toc172557699)

**Table des figures :**

[Figure 1, cycle de vie d'un produit alimentaire 5](#_Toc172557700)

[Figure 2, vue d'ensemble du projet 6](#_Toc172557701)

[Figure 3, extrait PEP-8 1/2 8](#_Toc172557702)

[Figure 4, extrait PEP-8, 2/2 9](#_Toc172557703)

[Figure 5, pipeline représentation graphique 14](#_Toc172557704)

[Figure 6, chemin utilisateur 17](#_Toc172557705)

# Architecture du système et présentation du projet

## Présentation du projet

### Contexte

​Green IA est née d’un constat simple : l’accès à l’information concernant l’empreinte carbone des produits alimentaires est quasiment inexistant. Quand d’autre appli permette d’informer l’utilisateur sur l’impact d’un produit sur sa santé, nous souhaitons informer l’utilisateur de l’impact d’un produit sur l’environnement.

### Objectifs

​L’objectif de Green IA est de devenir une application indispensable pour les consommateurs souhaitant améliorer leurs habitudes de consommations pour réduire leur emprunte carbone. Green IA les accompagnera dans leur quête en leur donnant accès aux données relatives à l’impact écologique de leurs produits alimentaires. Ses objectifs principaux sont :

* ​Élever la conscience publique sur l’empreinte des produits alimentaires et rendre accessible les données à tous.
* ​Fournir des données fiables et transparentes concernant les produits alimentaires et leurs empreintes carbone, consommation d’eau et plus encore.
* ​Proposer des options plus écologiques aux produits trop polluants.
* ​Mesurer et suivre la réduction de l’empreinte hebdomadaire ou mensuelle de l’utilisateur.
* ​Construire une plate-forme non seulement d’information mais aussi de sensibilisation et d’encouragement.

## Qu’est-ce que l’éco-score ?

L'Eco-score est un indicateur expérimental représentant l'impact environnemental des produits alimentaires. Il classe les produits en 5 catégories (A, B, C, D, E), de l'impact le plus faible, à l'impact le plus élevé.  L'impact environnemental tient compte de plusieurs facteurs, dont en voici la liste exhaustive :

* Émissions de gaz à effet de serre (CO2)
* Destruction de la couche d'ozone
* Émissions de particules fines
* Oxydation photochimique
* Acidification
* Radioactivité
* Épuisement des ressources en eau
* Pollution de l'eau douce
* Épuisement des ressources non renouvelables
* Eutrophisation (terrestre, eau douce & marine)
* Utilisation des terres
* Toxicités (eau douce & humaine)
* Perte de biodiversité

### Les impacts sont étudiés tout au long du cycle de vie du produit

Une image contenant diagramme, capture d’écran, ligne, cercle

Description générée automatiquement

Figure 1, cycle de vie d'un produit alimentaire

## Vue d’ensemble

Une image contenant texte, capture d’écran, diagramme, conception

Description générée automatiquement

Figure 2, vue d'ensemble du projet

## Acteurs du système et interactions (utilisateurs, admins, autres)

## Diagramme de sequence UML

## Use case des fonctionnalités

### code unique + use case + entrées et sortie

# Développement et technologies utilisées

## Langages de Programmation et normes de codage

* **JavaScript** : utilisé pour la gestion des messages envoyés à l’utilisateur, les actions des boutons ou encore de la caméra.
* **Python** : utilisé principalement pour les scripts de backend et les modèles d'intelligence artificielle, en raison de ses bibliothèques riches pour le machine learning et la manipulation de données.
* **HTML5 et CSS3** : utilisés pour structurer et styliser les pages web de l'application, assurant une facilité de conception pour nous Data Scientist, sans expérience dans la création d’application mobile ou d’interface web.
* **Json**: pour le stockage de données, tel que le fichier de configuration du pipeline automatique. Facile d’utilisation et léger.
* **CSV**: pour la gestion des données d’Open Food Facts, fichiers faciles à manipuler, stocker et importer comme data frame.

Pour assurer un développement soigné et compréhensible par tous nos pairs, nous avons suivi la convention PEP-8 Python, dont voici un aperçu :

Une image contenant texte, capture d’écran, document, Police

Description générée automatiquement

Figure 3, extrait PEP-8 1/2

Une image contenant texte, capture d’écran, document, Police

Description générée automatiquement

Figure 4, extrait PEP-8, 2/2

## Framework et bibliothèques

Ci-dessous, une liste non exhaustive des librairies et Framework que nous avons utilisé :

absl-py==2.1.0  
asttokens==2.4.1  
astunparse==1.6.3  
attrs==23.2.0  
beautifulsoup4==4.12.3  
bleach==6.1.0  
Bottleneck   
bson==0.5.9  
cachetools==5.3.3  
certifi==2024.7.4  
cffi   
charset-normalizer==3.3.2  
click   
comm==0.2.2  
contourpy   
cryptography   
cycler   
debugpy==1.8.2  
decorator==5.1.1  
defusedxml==0.7.1  
dnspython   
exceptiongroup==1.2.2  
executing==2.0.1  
fastjsonschema==2.20.0  
flatbuffers==24.3.25  
fonttools   
gast==0.4.0  
google-auth==2.32.0  
google-auth-oauthlib==0.4.6  
google-pasta==0.2.0  
grpcio==1.64.1  
h5py==3.11.0  
idna   
ipykernel==6.29.5  
ipython==8.26.0  
jedi==0.19.1  
Jinja2==3.1.4  
joblib   
jsonschema==4.23.0  
jsonschema-specifications==2023.12.1  
jupyter\_client==8.6.2  
jupyter\_core==5.7.2  
jupyterlab\_pygments==0.3.0  
keras==2.10.0  
Keras-Preprocessing==1.1.2  
kiwisolver   
libclang==18.1.1  
Markdown==3.6  
MarkupSafe==2.1.5  
matplotlib   
matplotlib-inline==0.1.7  
mistune==3.0.2  
nbclient==0.10.0  
nbconvert==7.16.4  
nbformat==5.10.4  
nest-asyncio==1.6.0  
nltk   
numexpr   
numpy   
oauthlib==3.2.2  
opt-einsum==3.3.0  
packaging   
pandas   
pandocfilters==1.5.1  
parso==0.8.4  
patsy   
pexpect==4.9.0  
pillow   
platformdirs==4.2.2  
ply==3.11  
prompt\_toolkit==3.0.47  
protobuf==3.19.6  
psutil==6.0.0  
ptyprocess==0.7.0  
pure-eval==0.2.2  
pyasn1==0.6.0  
pyasn1\_modules==0.4.0  
pycparser   
Pygments==2.18.0  
pymongo   
pyparsing   
PyQt5==5.15.10  
PyQt5-sip   
python-dateutil   
pytz   
pyzmq==26.0.3  
referencing==0.35.1  
regex   
requests==2.32.3  
requests-oauthlib==2.0.0  
rpds-py==0.19.0  
rsa==4.9  
scikit-learn   
scipy   
seaborn   
sip   
six   
soupsieve==2.5  
stack-data==0.6.3  
statsmodels   
tensorboard==2.10.1  
tensorboard-data-server==0.6.1  
tensorboard-plugin-wit==1.8.1  
tensorflow==2.10.1  
tensorflow-estimator==2.10.0  
tensorflow-io-gcs-filesystem==0.37.1  
termcolor==2.4.0  
threadpoolctl   
tinycss2==1.3.0  
tomli   
tornado   
tqdm   
traitlets==5.14.3  
typing\_extensions==4.12.2  
tzdata   
unicodedata2   
urllib3==2.2.2  
wcwidth==0.2.13  
webencodings==0.5.1  
Werkzeug==3.0.3  
wrapt==1.16.0

## Systèmes d’exploitation et environnement de travail

Pour ce projet nous utilisons trois systèmes d’exploitation différents, nous permettant de tester le fonctionnement de nos infrastructures sur différentes machines ; une Ubuntu 24, un MacOs Sonoma ainsi que deux Windows 11. L’autre atout de travailler sur différents OS, est la diminution du risque de mises à jour comportant des failles, ce qui rendrait la machine concernée inutilisable quelques temps.

Pour éviter les problèmes de déploiement et les conflits durant l’installation sur une nouvelle machine, nous avons exclusivement travaillé sur un environnement virtuel. Sa configuration aura été scrupuleusement mise à jour au fur et à mesure par chaque membre de l’équipe, grâce au fichier requirements.txt. Cet environnement virtuel est une reproduction à l’identique de notre environnement de déploiement.

## Déploiement de nos solutions

### Stratégies de déploiement (blue-green, canary)

### Environnement de déploiement (dev, test, prod)

### Outils de déploiement (CI/CD pipelines, versions)

### Stratégie de scalabilité (verticale et horizontale)

## Contenu du projet, fichiers et exécution

# IA et Data

## Collecte des données

Ici, trois sources de données ont été utilisées ; OpenFoodFacts pour la prédiction de l’éco score, le portail métropolitain ainsi que M data pour la gestion des points de collecte de déchets dans le département des Bouches du Rhône. Nous avons utilisé des techniques de scraping sur ces sites internet, notamment grâce à la librairie Beautiful Soup. Beautiful Soup permet de parser des documents HTML et XML, facilitant l'extraction de données à partir de pages web.

Les données pour la prédiction de l’éco score auront été plus complexes à traiter, en effet seulement deux fichiers compressés contiennent la totalité des articles d’Open Food Facts, un au format jsonl, l’autre au format Mongo DB Dump. Nous avons développé un script qui permet de télécharger le fichier compressé, avec la capacité de reprendre le téléchargement là où il s’était arrêté en cas de coupure réseau.

## Préparation des données

Une fois le fichier compressé téléchargé, notre script le décompressera en fichier jsonl, puis le parcourra en écrivant au fur et à mesure chaque ligne dans un fichier au format csv. Chacun de ces fichiers csv contiendra un maximum de 10 000 lignes, afin de les rendre exploitables sur des machines de particuliers. Par la suite, chaque csv sera ouvert et se verra amputer des colonnes jugées inutiles, il passera de 720 colonnes par ligne à une petite quinzaine.

Enfin, toutes les techniques de préparation classiques seront réalisées sur chacun des csv, avec une normalisation des valeurs numériques, une tokenisation du contenu textuel, ainsi que la fusion de plusieurs colonnes.

## Entraînement et validation des modèles

Régulièrement, un nouveau jeu de données provenant d’Open Food Facts sera téléchargé. Systématiquement un nouveau modèle sera entrainé sur ces nouvelles données, en parallèle du meilleur modèle que nous avons pu entrainer par le passé. Ainsi, un algorithme pourra comparer les résultats sur notre nouveau jeu de données de nos deux modèles, le meilleur sera mis de côté pour la prochaine mise à jour des données. Le csv avec le plus de bonnes réponses sera déployé.

Pour comparer les performances de nos réseaux de neurones, nous avons choisi les cinq métriques suivantes :

* L’erreur quadratique moyenne (MSE)
* L’erreur absolue moyenne (MAE) et médiane
* L’erreur quadratique moyenne racine (RMSE)
* Le coefficient de détermination (R2)

Ces métriques d’évaluation sont adaptées à notre volonté de prédire une variable continue à partir d’un ensemble de caractéristiques, c’est-à-dire une tache de régression.

## Stockage et mise à disposition des données

## Dashboard utilisateur

## Dashboard global

## Pipeline

Une image contenant diagramme, ligne, dessin, croquis

Description générée automatiquement

Figure 5, pipeline représentation graphique

# Interface utilisateur

## Sécurité et authentification

A ce stade du projet, aucune authentification n’est nécessaire pour utiliser l’application, il n’est d’ailleurs tout simplement pas possible de se créer un compte. Les données personnelles de l’utilisateur ainsi que se historique d’articles scannés sont sauvegardé dans des fichiers json en local dans son navigateur. Aucune donné ne nous est transmise, et stockée sur nos serveurs, le navigateur assurant de plus un niveau de sécurité que nous serions capable d’assurer sans un expert du domaine.

## Intégration avec des services externes

Ne cherchant pas à systématiquement réinventer la roue, de nombreuses solutions développées par d’autres entreprises ont été importés dans notre application web. Pour la carte nous avons fait le choix d’utiliser [OpenStreetMap](https://www.openstreetmap.fr/). Gratuite, Open Source et facilement implémentable, cette solution nous a semblé être la meilleur pour afficher des cartes à nos utilisateurs, bien loin des prix exorbitants de Google. Pour scanner les articles de l’utilisateur par le biais de sa caméra, nous avons utilisé [Quagga](https://cdnjs.cloudflare.com/ajax/libs/quagga/0.12.1/quagga.min.js) dans sa version 0.12.1, une libraire Open Source écrite en Java Script. Ces solutions ne sont utilisables que pour des applications déployées en https, le cas échant, ces librairies sont inaccessibles.

## Présentation des maquettes

### Une image contenant texte, capture d’écran, Police, document Description générée automatiquementUne image contenant texte, capture d’écran, Téléphone mobile, Appareil de communication Description générée automatiquementPage principale et page d’informations

Permet à l’utilisateur de scanner des articles alimentaires, télécharger sa liste de produits scannés et la vider ainsi que visualiser le nombre d’articles, hors doublons comptabilisés par l’application. La page d’information sert de manuel utilisateur ligne, tous les conseils d’utilisation et toutes les informations sur le traitement des données utilisateur y figures.

**Input :** Images de la caméra de l’utilisateur.

**Output :** Liste d’historique des produits téléchargés, informations sur l’impact environnemental du produit scanné.

### Une image contenant texte, capture d’écran, Téléphone mobile, gadget Description générée automatiquementUne image contenant texte, capture d’écran, Téléphone mobile, gadget Description générée automatiquementDashboards utilisateur et global

Un bouton permet à l’utilisateur de passer de son dashboard personnel à un dashboard global, ces dashboards ne sont pas hébergés sur la même machine que l’application web elle-même. Les graphiques s’adaptent à la taille de l’écran.

**Input :** Position du bouton.

**Output :** Affiche le dashboard associé à la position du bouton.

### Informations dépôts et collectes

Une image contenant texte, capture d’écran, Police, nombre

Description générée automatiquementUne image contenant texte, capture d’écran, Police, Téléphone mobile

Description générée automatiquementLa page de gauche affiche sur une carte les heures de passage ainsi que les déchets traités, une fois le code postal de la ville où se trouve l’utilisateur entré. Celle de droite affiche directement l’endroit précis où se trouve l’utilisateur, affichant les lieux recyclant les types de déchets sélectionnés par l’utilisateur.

**Input :** Code postal de l’utilisateur, activation du bouton de soumission, chaque type de déchet pris en charge par l’application.

**Output :** Affiche les informations sur les points de dépôts et les horaires de collecte des déchets.

## Accessibilité

Ce choix des technologies web, nous permet d’éviter le développement d’une application Android et IOS, environnements avec lesquels nous ne sommes absolument pas familiers, impliquant de nombreuses heures de formations et de développement, pour un résultat surement médiocre. Avec le web nous avons pu développer une application responsive, aussi bien disponible sur ordinateur, que sur mobile. Il suffit de cliquer se rendre sur le lien suivant : [Green Ia](https://caroluscharlemagne.github.io/Green_IA_website/). Il n’est pas non plus nécessaire de créer un compte pour utiliser notre application.

L’unique prérequis pour accéder à cette application web est un accès un internet (wifi, 4G, 5G, etc) ainsi qu’un navigateur web. Attention cependant à certains navigateurs trop restrictifs qui pourraient bloquer l’accès à la caméra où aux notifications.

## Chemins utilisateur

Une image contenant texte, capture d’écran, diagramme, ligne

Description générée automatiquement

Figure 6, chemin utilisateur

## Diagrammes de flux

# Sécurité, maintenance et support

## Contraintes de sécurité

## Gestion des données sensibles

Nous ne traitons pas de données sensibles dans ce projet, aucune donnée personnelle ou médicale. A terme, une base de données sera créée et hébergée chez AWS ou Google, la sécurité des données leur sera ainsi confiée. Nous ferons en revanche auditer par un expert de la sécurité, la robustesse de nos sites web et applications qui afficheront et réaccueillerons ces données, notamment des données utilisateur.

## Conformité et réglementations

## Plan de maintenance et MCO

## Surveillance et monitoring

### Rapport d’état de santé du système

### Rapport d’évolution des données

### Notifications aux administrateurs

## Plan des mises à jour et support

# Tests, validation et améliorations

## Critères d’acceptation pour chaque fonctionnalité

C : conforme, EC : en cours, NC : non conforme.

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifiant fonctionnalité** | **Résultat attendu** | **Constatations** | **Validation** |
| **Site web** | | | |
| 01102023SW05 | Le menu tous les boutons nécessaires, tous cliquables et redirigeant vers la bonne page | Ok | C |
| 01102023SW06\_00 | Possibilité de scanner en boucle le code barre de produits alimentaires. Il faut afficher un message d’erreur dans le cas où le produit est inconnu. Pour ne pas empiéter sur la vie privée de nos utilisateurs, il sera également obligatoire de valider l’utilisation de sa caméra à chaque ouverture d’application | ~~Impossible de scanner plusieurs articles sans fermer et ouvrir l’application de nouveau~~ | C |
| 01102023SW06\_01 | Le bouton doit permettre le téléchargement de la liste d’historique des produits scannés par l’utilisateur | Ok | C |
| 01102023SW06\_02 | Bouton permettant à l’utilisateur de vider son historique | Ok | C |
| 01102023SW07\_00 | L’utilisateur doit avoir un champ pour entrer son adresse postale, ainsi qu’un bouton | Ok | C |
| 01102023SW07\_01 | Le code postal renseigné doit permettre d’afficher la position de la ville sur la carte, avec les informations de collecte des déchets associés, stockés dans le fichier json | Ok | C |
| 01102023SW08\_00 | L’utilisateur doit pouvoir filtrer les types de déchets qu’il souhaite jeter | Ok | C |
| 01102023SW08\_01 | Seuls les points de dépôt appartenant à une catégorie choisie par l’utilisateur et autour de lui, doivent apparaitre sur une carte. De plus, au clic sur l’un de ses points, toutes les informations sur le lieu doivent apparaitre (horaires, types, jours d’ouverture) | ~~Différencier la couleur du pin de l’utilisateur de celui des boutiques~~ | C |
| 01102023SW11\_00 | Cette page doit contenir toutes les informations utiles à l’utilisateur pour comprendre le projet, ses objectifs et répondre à toutes ces questions courantes | ~~Le bouton du menu associé à la page n’est pas mis en évidence comme les autres lorsqu’il est cliqué~~ | C |
| 01102023SW12 | Aucun code inutile ne doit être présent, il doit être compréhensible par tous, respecter la PEP-8 | Aucun soucis | C |
| 01102023SW13 | Le site web doit être déployer en https pour faire fonctionner tous les composants externes, disponible depuis n’importe quel navigateur | Ok | C |
| **Dashboard** | | | |
| 01102023DBrd02 |  |  |  |
| 01102023DBrd03 |  |  |  |
| 01102023DBrd04 |  |  |  |
| 01102023DBrd05 |  |  |  |
| 01102023DBrd06 |  |  |  |
| **Prédiction éco score** | | | |
| 01102023PES01 | Ne produisant pas de données et n’ayant pas la possibilité de payer, nous devons nous tourner vers une ou plusieurs sources gratuites | Ok | C |
| 01102023PES07 | Étant donné la complexité de prédire l’éco score sur des produits alimentaires, un objectif de 80% de positifs sur des données de validation serait un bon résultat |  | EC |

## Stratégie de tests

## Tests unitaires

## Tests d'intégration

## Tests de performance

## Améliorations envisagées et versioning

Pour cette version 1.0 qui n’est qu’un prototype, les objectifs nous semble atteints, pour une version 2.0, de nombreux points seront cependant à améliorer, si nous voulons la rendre accessible à tous :

* Une base de données dans le cloud avec redondance dans un autre data center.
* La possibilité de créer un compte utilisateur.
* Changer les technologies de l’application web, pour de l’Angular.
* Améliorer l’UI et l’UX de l’application.
* Discuter avec Open Food Fact pour rendre disponible les prédictions d’éco score directement via leur API.

## Confidentialité et accès

Dans notre groupe de travail restreint, chacun d’entre nous a les mêmes responsabilités. La perte de fichiers, la fuite de données ou une éventuelle régression des performances sera directement imputée au collaborateur n’ayant pas validé lui-même, puis fait valider son travail par un pair avant de le mettre en production.

# Annexes

## Glossaire

|  |  |
| --- | --- |
| Abréviation | Signification |
| IA et Data Science | |
| IA | Intelligence Artificielle |
| Développement | |
| Framework | Environnement de travail facilitant le développement d’une solution technique. |
| Responsivité | Possibilité d’adapter la taille du logiciel à la taille de l’écran de l’utilisateur |
| Front | Développement des aspects visuels du logiciel |
| Back | Développement de la partie logique du logiciel (caché à l’utilisateur) |
| API REST | API Representational State Transfer Application Program Interface est un style architectural qui permet aux logiciels de communiquer entre eux sur un réseau ou sur un même appareil. Le plus souvent les développeurs utilisent des API REST pour créer des services web. Souvent appelés services web RESTful, REST utilise des méthodes HTTP pour récupérer et publier des données entre un périphérique client et un serveur. |

## Documents applicables

|  |  |
| --- | --- |
| Description | Identification |
| 23-24 Modalités Évaluations Titre EISI N7 Étudiants – AYC Pour les M2 | REF [0] |
| Dépôt Moodle filière informatique M2 | REF [1] |

## Diffusion du document

|  |  |  |  |
| --- | --- | --- | --- |
| Diffusion | Statut | Nom | Emis le |
| Edition | Charlemagne | 05/07/2024 |
|  |  |  |
|  |  |  |

## Historique des modifications

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Pages | Description de la modification - Auteur | Date |
| 0.3 |  | Architecture globale du document | 05/07/2024 |
|  |  |  |  |
|  |  |  |  |