Université de la Nouvelle-Calédonie

Institut d'Administration des Entreprises - IAE NC

Master MIAGE 2ème année - Semestre 3
2024 - 2025

Mémoire

Projet de DATAVIZ sur Matrice LED

Projet réalisé par
Raphaël BORDAIS et **José GOUE**

du 19 septembre 2024 au 21 janvier 2025

En partenariat avec l'Office des Postes et Télécommunications de la Nouvelle-Calédonie

Encadré par **Monsieur Adrien SALES**,
Chef du Bureau Génie Logiciel au sein de la DSI

Remerciements

Nous tenons à exprimer toute notre gratitude envers toutes les personnes qui nous ont soutenu et accompagné tout au long de ce projet, et sans qui cette expérience n'aurait pas été aussi enrichissante :

À notre tuteur et Chef du Bureau Génie Logiciel au sein de la DSI de l'OPT-NC, **Monsieur Adrien SALES** pour sa disponibilité, ses conseils avisés, et son immense implication dans notre projet tout au long de la période.

À toute **l'équipe du Bureau Génie Logiciel** de la DSI de l'OPT-NC et en particulier à **Madame Michèle BARRE**, Ingénieure Logiciel et **Monsieur Vinh FAUCHER**, Concepteur Développeur, pour leur accueil chaleureux, leur esprit collaboratif, leurs conseils et le partage de leurs connaissances, qui ont contribué à l'avancement efficace du projet.

À notre directrice de formation du Master MIAGE, **Madame Nazha SELMAOUI** pour son encadrement et ses encouragements constants qui nous ont guidé dans la rédaction de ce rapport.

À notre assistante de formation, **Madame Cynthia COSTE** pour ses encouragements, son aide et sa disponibilité pour répondre à nos questions.

À **l'équipe pédagogique du Master MIAGE** de l'Université de Nouvelle-Calédonie pour la qualité de la formation dispensée, qui nous a permis d'acquérir les compétences nécessaires pour relever les défis professionnels rencontrés.

À **nos familles, nos camarades de classe et nos amis** pour leur soutien moral et leurs encouragements tout au long de cette période.

Et enfin à toutes les personnes qui ont contribué directement ou indirectement, à la réussite de ce projet et à notre apprentissage global.

Figure 1 : Slogan de l'UNC

Table des matières

Introduction

« L'information est belle lorsqu'elle est bien présentée." – Hans ROSLING, éminent statisticien et spécialiste en datavisualisation.

Dans ce contexte technologique où la visualisation des données occupe une place prépondérante dans le processus décisionnel et l'interaction avec l'utilisateur, la compétence à présenter de manière efficace des informations au sein d'espaces limités constitue un défi technique majeur. Ce mémoire s'inscrit dans cette problématique en examinant la possibilité d'exploiter une matrice LED de 32x32, un dispositif compact mais restreint en termes de résolution, afin de concevoir une solution de visualisation dynamique et fonctionnelle.

L'objectif de ce projet consiste à illustrer comment un espace aussi restreint peut se transformer en un outil efficace de datavisualisation, grâce à une conception logicielle soigneusement optimisée et à une gestion réfléchie des ressources matérielles. En exploitant le langage Micro-Python, nous avons réussi à optimiser les fonctionnalités proposées par la matrice LED :

- **Présentation personnalisée et dynamique :** Élaboration de typographies sur mesure et amélioration des fontes existantes afin d'assurer une lisibilité optimale, même dans des espaces restreints.

- **Connectivité de pointe** : Intégration de la technologie Wi-Fi et utilisation d'APIs afin d'enrichir les contenus présentés en temps réel, notamment à travers des messages dynamiques ou des indicateurs clés de performance.

- Administration des ressources matérielles : Optimisation de la mémoire et de la capacité de calcul de la matrice afin de soutenir des fonctionnalités avancées, telles que l'affichage de codes QR interactifs.

La mise en œuvre de ce projet a été effectuée dans un cadre collaboratif, facilitée par l'utilisation de GitHub, ce qui a permis une gestion rigoureuse des versions, un suivi précis des modifications et une documentation exhaustive. Cette sélection d'environnement de travail a également permis d'accroître la modularité du code, facilitant ainsi les évolutions futures.

Cette étude met en exergue la synergie entre les contraintes matérielles et les solutions logicielles novatrices, démontrant de quelle manière une infrastructure élémentaire peut se transformer en un vecteur innovant de datavisualisation. Ce mémoire exposera de manière détaillée les choix technologiques, les défis rencontrés, les solutions mises en œuvre ainsi que les perspectives offertes par cette approche, fournissant ainsi une vision exhaustive du potentiel des matrices LED dans le domaine de la visualisation des données.

Afin de valoriser ce projet et d'en faire bénéficier d'autres professionnels et passionnés de technologie, nous avons publié une description détaillée de notre projet sur la plateforme Hackster.io (https://www.hackster.io/adriensales/post-office-wait-time-api-driven-waiting-time-led-matrix-39c00b). Cette publication permet de partager les enseignements tirés et de promouvoir l'innovation technique au sein d'un large réseau de développeurs.

1. Contexte du projet

1.1. Le master MIAGE et ses objectifs

Le Master MIAGE allie informatique et gestion pour former des experts en systèmes d'information capables de concevoir et piloter des projets IT couvrant divers domaines tels que le développement logiciel, la data science, le business intelligence et la gestion de projet, adaptées aux besoins des entreprises.

Ce diplôme prépare aux métiers d'ingénieur logiciel, chef de projet IT, consultant SI ou expert en data science. Grâce à ses partenariats avec les entreprises, les étudiants réalisent des projets applicatifs concrets, facilitant leur insertion professionnelle dans le secteur numérique.

1.2. L'OPT NC, notre organisme d'accueil

L'OPT NC, établissement public fondé en 1958, est un acteur clé du développement numérique en Nouvelle-Calédonie, opérant dans les télécommunications, services postaux et financiers. Engagé dans l'innovation, il propose aux étudiants des projets applicatifs concrets.

Nous avons été sélectionnés pour travailler sur un projet de datavisualisation sur matrice LED, visant à afficher en agence les temps d'attente des clients grâce aux données d'une API. Ce projet répond aux besoins de l'OPT NC en matière d'amélioration de l'expérience client et d'optimisation des services.

Sous la supervision de M. Adrien SALES, nous avons collaboré avec le bureau Génie Logiciel de la DSI, dont le rôle était de :

- Fournir les données sur les temps d'attente en agence.
- Offrir un encadrement technique sur les processus internes et infrastructures IT.
- Valider et intégrer le projet pour s'assurer de son adéquation aux besoins des agences.
- Optimiser l'expérience client en améliorant l'affichage des informations.

Cette expérience nous a permis de contribuer aux objectifs de l'OPT NC tout en développant nos compétences en gestion de projet IT et développement technologique.

1.3. Les enjeux de la Datavisualisation et son impact potentiel

L'essor des technologies de datavisualisation transforme la manière dont les organisations exploitent et communiquent leurs données. Dans un contexte où la rapidité et la clarté de l'information sont essentielles, la mise en place d'une solution d'affichage dynamique sur matrice LED offre de nouvelles opportunités. L'enjeu principal est d'améliorer la transmission des informations critiques, comme les temps d'attente en agence, tout en optimisant les ressources disponibles. L'impact potentiel d'une telle technologie repose sur plusieurs axes :

Amélioration de la prise de décision : En fournissant des informations en temps réel, la datavisualisation permet aux gestionnaires d'agences d'ajuster leurs effectifs en fonction de l'affluence.

Expérience utilisateur améliorée : Un affichage clair et instantané des temps d'attente réduit l'incertitude et améliore la satisfaction des clients.

Efficacité opérationnelle accrue : L'exploitation des données historiques et en temps réel permet d'optimiser les flux et de mieux anticiper les périodes de forte affluence.

Perspectives d'évolution : Cette technologie pourrait être intégrée à d'autres solutions numériques, comme des applications mobiles ou des systèmes de gestion avancés.

2. Présentation du projet

Avec la digitalisation croissante des services, les attentes des clients en matière d'information en temps réel se sont considérablement accrues. Dans ce contexte, l'OPT souhaite exploiter ses données internes pour offrir un service innovant permettant d'informer les usagers sur les temps d'attente en agence de manière instantanée et accessible. Cependant, l'affichage de ces données en temps réel pose plusieurs défis, notamment en termes de lisibilité, d'actualisation des informations et d'optimisation des supports de communication. L'utilisation d'une matrice LED compacte apparaît alors comme une solution adaptée pour concilier clarté de l'affichage, rapidité de mise à jour des données et encombrement minimal. Ce projet s'inscrit ainsi dans une démarche d'amélioration continue de l'expérience client et d'optimisation du fonctionnement des agences.

2.1. Les objectifs et la problématique

Ce projet vise plusieurs objectifs majeurs :

L'optimisation de l'affichage des temps d'attente en agence

- Exploiter des données actuellement sous-utilisées pour fournir une information en temps réel aux clients.
- Améliorer l'accessibilité et la lisibilité des informations sur un support compact.

La démonstration de la polyvalence d'une matrice LED 32x32

- Transformer un espace restreint en un outil efficace de datavisualisation interactive.
- Développer une gestion innovante des ressources matérielles et logicielles pour maximiser les capacités d'affichage.

Ces objectifs doivent pouvoir répondre à la problématique suivante :

"Comment exploiter les données existantes pour afficher en temps réel les temps d'attente en agence sur une matrice LED compacte, tout en optimisant la gestion des flux clients ?"

Avant d'aborder les motivations et enjeux du projet, il est essentiel de souligner l'importance de la problématique soulevée. La mise en place d'un affichage dynamique des temps d'attente en agence repose sur une analyse fine des données internes de l'OPT et sur l'exploitation de technologies adaptées aux contraintes du terrain. Ce projet ne se limite pas à un simple affichage, mais vise une transformation de la manière dont les informations sont partagées avec les usagers. En répondant aux objectifs définis précédemment, il s'agit également d'anticiper les futurs besoins d'optimisation et d'adaptation des services aux nouvelles attentes des clients.

2.2. Les Motivations et les Enjeux

L'essor des technologies de l'information et la disponibilité croissante des données offrent de nouvelles opportunités pour améliorer la communication et l'expérience utilisateur dans divers secteurs. Dans ce contexte, l'OPT cherche à exploiter ses données internes afin d'optimiser la gestion du temps d'attente en agence. Ce projet vise à répondre à un double enjeu : améliorer le service rendu aux clients en leur fournissant des informations en temps réel et optimiser l'organisation interne des agences en exploitant des données précieuses souvent sous-utilisées. La mise en place d'un affichage sur matrice LED permet ainsi de transformer ces données en un outil d'aide à la décision et en un levier d'amélioration de l'expérience utilisateur. Les éléments qui nous ont motivés à choisir ses projets sont :

L'amélioration de l'expérience client

- Réduire la frustration liée aux longues files d'attente en offrant une meilleure visibilité sur le temps d'attente.
- Informer les clients en temps réel sur la disponibilité des services, améliorant ainsi leur organisation et leur satisfaction.

L'optimisation du fonctionnement des agences

- Permettre une gestion proactive des flux de clients en identifiant les pics d'affluence.
- Aider à la planification des ressources humaines en ajustant les effectifs selon les besoins en temps réel.

La valorisation des données existantes

- Exploiter les données internes de l'OPT pour en extraire un service à forte valeur ajoutée.
- Mettre en place un tableau de bord pour un suivi précis et analytique des performances des agences.

L'innovation et transformation digitale

- Introduire une solution technologique moderne pour améliorer la communication avec les clients.
- Poser les bases d'une évolution vers une application mobile ou un service web permettant de consulter les temps d'attente à distance.

2.3. Les Défis Technologiques et Les Solutions

Le développement d'une solution innovante de datavisualisation sur matrice LED implique plusieurs défis techniques et méthodologiques. Ce projet doit garantir une lisibilité optimale des informations sur un support compact tout en assurant une connectivité fluide et une gestion efficace des ressources matérielles. Il est donc essentiel de relever ces défis en mettant en place des stratégies adaptées pour optimiser l'affichage, la récupération des données en temps réel et l'interaction avec les utilisateurs. Les solutions envisagées pour surmonter ces contraintes et garantir la performance du système sont les suivantes :

La conception d'une solution de visualisation dynamique

- Exploitation optimale de la matrice LED pour garantir une lisibilité et une clarté maximales.
- Développement de typographies adaptées aux contraintes d'un affichage de petite taille.

La connectivité et interactivité avancées

- Intégration du Wi-Fi pour récupérer et afficher des données en temps réel via des API.
- Ajout de QR Codes interactifs permettant aux utilisateurs d'accéder à des informations complémentaires sur leur smartphone.

L'optimisation des ressources matérielles

- Maximiser l'utilisation de la mémoire et de la puissance de calcul pour garantir des performances fluides.
- Expérimenter l'utilisation de MicroPython, une alternative permettant un développement plus souple et une meilleure évolutivité du projet.

La modularité et collaboration

- Utilisation de GitHub pour gérer le code source, faciliter le partage et assurer une documentation claire et accessible.
- Encourager une approche collaborative pour favoriser les évolutions futures et assurer la pérennité du projet.

2.4. Apports attendus pour l'entreprise et pour l'apprentissage académique

Ce projet présente des bénéfices à la fois pour l'entreprise partenaire et pour le cadre académique dans lequel il s'inscrit.

Pour l'entreprise

- L'amélioration de la gestion des flux clients : Grâce à une visualisation en temps réel des temps d'attente, l'OPT pourra mieux gérer l'affluence et optimiser les ressources humaines en conséquence.
- La modernisation des outils de communication : L'affichage dynamique sur matrice LED constitue un moyen innovant et efficace pour informer les clients, réduisant ainsi leur frustration et améliorant leur satisfaction.
- L'exploitation des données sous-utilisées : Ce projet permet d'exploiter les données internes de l'OPT pour leur donner une valeur ajoutée et optimiser le suivi des performances des agences.
- Vers une transformation digitale : En intégrant des outils connectés et des technologies modernes, ce projet contribue à la transition numérique de l'OPT.

Pour l'apprentissage académique

- Application concrète des compétences acquises : Ce projet permet de mettre en pratique les connaissances en datavisualisation, développement embarqué et gestion de projet informatique.
- Travail en conditions réelles : En collaborant avec une entreprise, les étudiants sont confrontés à des problématiques concrètes et doivent s'adapter aux exigences d'un environnement professionnel.
- Expérimentation et innovation : Le développement sur matrice LED offre une opportunité unique d'expérimenter avec des interfaces matérielles et logicielles, renforçant ainsi la capacité à innover dans des contextes technologiques contraints.
- Renforcement des compétences collaboratives : L'utilisation de méthodologies de gestion de projet et d'outils collaboratifs comme GitHub favorise le travail en équipe et la gestion efficace des développements.

3. La datavisualisation sur matrice LED avec l'OPT-NC

3.1. Les technologies contemporaines en matière de datavisualisation

La datavisualisation (DATAVIZ) s'appuie sur la conversion de données complexes en représentations graphiques à la fois claires et esthétiques. De nos jours, un éventail de technologies permet la diffusion d'informations, allant des écrans haute définition aux dispositifs spécialisés tels que les matrices LED.

Les tendances majeures en matière de visualisation des données comprennent :

- Écrans interactifs et tableaux de bord analytiques : Interfaces tactiles et outils analytiques (*Power BI, Tableau, D3.js*) pour une expérience immersive.
- Objets connectés (Internet des objets - IoT) : Affichage de données en temps réel via des dispositifs compacts (panneaux connectés, wearables, tableaux informatifs).
- Matrice à diodes électroluminescentes (Matrice LED) : Solution modulable et claire, utilisée dans les IoT, enseignes et visualisations intégrées.

3.2. Utilisation des matrices LED

La matrice LED, telle que la Pimoroni Cosmic-Unicorn, constitue une technologie de visualisation reposant sur un réseau de diodes électroluminescentes, lesquelles émettent de la lumière afin de représenter des caractères, des animations ou des images. Parmi les bénéfices qu'il présente :

- Compacité : Légères et compactes, elles s'intègrent facilement dans des dispositifs de petite taille.
- Flexibilité : Personnalisation totale des animations et affichages via MicroPython ou C.
- Coût réduit : Alternative économique aux écrans LCD ou OLED.
- Applications IoT : Faible consommation énergétique et affichage en temps réel, idéales pour l'IoT.

Technologie	Avantages principaux	Limites principales
LCD	Haute résolution, couleurs vives, grand format	Consommation énergétique plus élevée, coût souvent plus important
OLED	Contrastes élevés, finesse des écrans	Sensibilité aux burn-in, coût plus élevé
Matrice LED	Compacte, économique, adaptable, faible consommation	Résolution limitée, nécessite un codage optimisé
E-Ink (papier électronique)	Faible consommation, lisibilité en plein jour	Vitesse d'affichage lente, coût élevé pour des écrans couleur

Tableau 1 : Comparaison avec d'autres technologies d'affichage

La matrice LED a été sélectionnée pour ce projet pour diverses raisons :

- Restriction d'espace : Idéale pour les environnements restreints où un écran classique est inadapté.
- Présentation dynamique des informations : Personnalisation avancée pour présenter des informations clés (temps d'attente, QR codes).
- Coûts et maintenance : Solution abordable, durable et facile à intégrer dans les systèmes IoT.
- Amélioration de l'utilisation des ressources : Faible consommation mémoire, parfaite pour microcontrôleurs et systèmes embarqués.

3.3. Conception et architecture du système

3.3.1. Les choix technologiques

La Pimoroni Cosmic-Unicorn

Elle a été sélectionnée en raison de ses caractéristiques distinctives et de sa pertinence par rapport aux exigences du projet :

- Format compact et densité élevée des LEDs :1024 LED (32x32), idéale pour afficher textes, animations et QR Codes, tout en restant légère et facile à intégrer.
- Compatibilité avec les environnements Python et MicroPython : Prise en charge native via les bibliothèques Pimoroni, facilitant le développement rapide et la gestion efficace des ressources.
- Simplicité d'alimentation : Fonctionne via USB-C, compatible avec les systèmes portables et fixes.
- Support de fonctionnalités avancées : Contrôle précis de la luminosité et effets dynamiques intégrés.
- Communauté et documentation technique : Documentation complète et assistance active de la communauté Pimoroni pour une prise en main rapide.

Des informations précises concernant la matrice LED Pimoroni sont disponible en annexe.

Python/MicroPython

- Flexibilité : Python est largement utilisé en IoT, avec de nombreuses bibliothèques adaptées aux matrices LED. MicroPython, plus léger, est idéal pour les systèmes embarqués à ressources limitées.
- Facilité et rapidité de développement : Développement rapide et évolutif grâce à Python et MicroPython.
- Administration de la matrice : Des bibliothèques comme cosmic_unicorn simplifient la gestion des LEDs, animations et interactions utilisateur.

APIs

Établissement d'une connexion aux données dynamiques : Les APIs permettent d'extraire et d'afficher en temps réel des données comme les temps d'attente sur la matrice LED.

Format normalisé : Les API REST en JSON s'intègrent facilement avec Python pour une interaction fluide avec les services externes.

Stratégies pour la gestion des animations (algorithmes et bibliothèques) : Dans le cadre de la gestion des animations et de l'affichage, diverses stratégies ont été mises en œuvre afin d'optimiser les performances et d'assurer une expérience utilisateur harmonieuse :

Algorithmes d'affichage optimisés : Prétraitement des données pour limiter l'usage mémoire, avec affichage en 32x32 et animations optimisées (défilement, transitions).

Bibliothèques destinées à la matrice : Python offre des outils prêts à l'emploi pour le dessin, le texte et l'ajustement de la luminosité. Pillow est utilisé pour adapter les polices et garantir la lisibilité.

Animations sur mesure : Effets visuels (fondu, défilement), affichage de QR Codes via qrcode, clignotements et variations de couleur pour attirer l'attention.

La plateforme APIGEE

APIGEE est une solution de gestion d'API développée par Google qui permet aux entreprises de créer, sécuriser, surveiller et optimiser leurs interfaces de programmation. Dans notre projet, APIGEE joue un rôle central en tant qu'API Gateway pour l'OPT NC. Concrètement, elle assure l'authentification des requêtes (grâce à OAuth 2.0 et JWT), la transformation des données et la gestion des accès, garantissant ainsi une communication sécurisée entre l'OPT NC et nos dispositifs d'affichage.

L'OPT NC utilise APIGEE pour exposer ses API, notamment celles fournissant en temps réel les temps d'attente des agences, permettant ainsi une intégration fluide dans notre système d'affichage.

Le lien d'accès à la plateforme APIGEE de l'OPT NC est le suivant : https://apigee-optnc-prd-api.apigee.io/

3.4. Le schéma d'architecture de la solution

Dans le cadre du projet visant à améliorer l'expérience client des agences de l'OPT NC, un système d'affichage en temps réel des temps d'attente a été mis en place. Ce système repose sur l'exploitation des API publiques de l'OPT NC, exposées via la plateforme Apigee, afin de récupérer et afficher dynamiquement les informations de chaque agence.

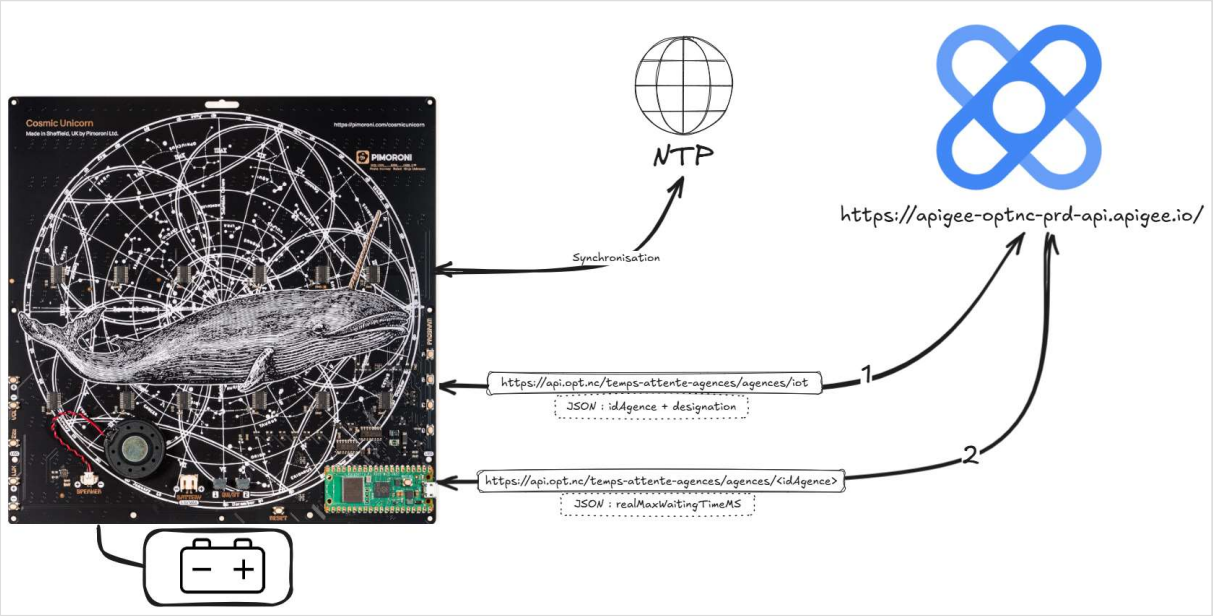


Figure 2 : Le schéma d'architecture

Le schéma représente l'architecture du système d'affichage de temps d'attente des agences de l'OPT NC mis en place dans le cadre du projet et ses différentes étapes d'architectures.

3.4.1. Connexion WiFi à l'OPT

La matrice LED se connecte au réseau WiFi de l'OPT pour récupérer les informations nécessaires.

3.4.2. Synchronisation NTP (Network Time Protocol)

Elle synchronise son horloge avec un serveur NTP sur Internet pour garantir une heure précise.

3.4.3. Requête 1 : Listing des agences

https://api.opt.nc/temps-attente-agences/agences/iot

Cette requête est utilisée pour récupérer la liste de toutes les agences disponibles.

Retour : Un objet JSON contenant pour chaque agence au moins les informations suivantes.

- idAgence : l'identifiant unique de l'agence.
- designation : le nom de l'agence.

Traitement : Le JSON est parcouru et chaque agence est ajoutée à un tableau (array) sous la forme d'une structure [idAgence, designation, 0] (le temps d'attente étant initialisé à 0).

3.4.4. Requête 2 : Temps d'attente d'une agence

https://api.opt.nc/temps-attente-agences/agences/<idAgence>

(où <idAgence> est remplacé par l'identifiant de l'agence à interroger)

Cette requête vise à récupérer le temps d'attente pour une agence spécifique.

Retour : Un objet JSON contenant une clé principale.

- realMaxWaitingTimeMs : qui indique le temps d'attente maximal en millisecondes.

Traitement : Ce temps est ensuite utilisé pour mettre à jour l'affichage de l'agence dans le tableau, et déterminer l'humeur (happy, neutral, ou sad) du smiley affiché.

3.4.5. Demande vers l'OPT pour nouvel EndPoint

Lors des premiers tests, nous avons constaté que l'endpoint initial de l'OPT, destiné à fournir un listing complet des agences, renvoyait un volume trop important d'informations. Cette surcharge entraînait une saturation de la mémoire du microcontrôleur de la matrice LED, qui est, en tant qu'appareil IoT, limité en ressources.

Pour remédier à cette contrainte, une demande a été faite à l'OPT afin de créer un nouvel endpoint simplifié. Ce nouvel endpoint renvoie uniquement les données essentielles : le nom de l'agence et son identifiant. Ce choix permet de construire dynamiquement, dans notre script, un tableau (array) référençant toutes les agences sans avoir à stocker en local un volume de données trop important.

Ce tableau sert ensuite de base pour la deuxième requête, qui interroge, au cas par cas, l'API dédiée afin de récupérer le temps d'attente pour chaque agence. Ainsi, l'architecture en deux temps assure une utilisation optimisée des ressources de la matrice LED tout en garantissant que les informations affichées sont à jour et pertinentes.

3.4.6. Synthèse

L'ensemble de ces étapes permettent à la matrice LED d'afficher différentes informations telles que :

- L'heure synchronisée via le NTP (HEURE)
- Les différents smileys indiquant le temps d'attente (SMILEY)
- L'affichage dynamique du nom des agences (SCROLLING NOM)

L'ensemble de ces étapes a été formalisé dans un script développé en MicroPython.

4. L'implémentation du code

Pour assurer le bon fonctionnement de la matrice LED comme souhaité, nous avons structuré notre projet en créant plusieurs fichiers, chacun dédié à une fonction spécifique. Cette approche modulaire a permis d'optimiser le développement, la maintenance et l'évolutivité du système. Dans les paragraphes suivants, nous détaillerons le rôle et les contributions de chaque fichier, afin de mettre en lumière comment ils s'articulent pour offrir une expérience utilisateur fluide et dynamique.

L'ensemble des scripts et des fichiers sources sont stockés et versionnés sur GitHub, accessible via le lien suivant : https://github.com/adriens/temps-attente-matrix-led/

4.1. Le fichier HOW_TO_INSTALL.md

Le document HOW_TO_INSTALL.md fournit des instructions essentielles pour l'installation et la configuration du projet Cosmic Unicorn. Il décrit les étapes nécessaires à la mise en place du système, en couvrant à la fois les aspects matériels et logiciels.

La procédure d'installation comprend :

- Préparation du matériel : assemblage de la matrice LED et du Raspberry Pi Pico W.
- Configuration logicielle : installation de Thonny et modification du fichier information.env pour intégrer les identifiants WiFi et clés API.
- Clonage et transfert des fichiers : récupération du code depuis GitHub et copie sur le Raspberry Pi.
- Exécution des scripts : lancement automatique du programme et validation des paramètres.

Comme indiqué dans le document HOW_TO_INSTALL.md, la solution repose sur plusieurs fichiers pour garantir son bon fonctionnement :

- Boot.py : initialise les paramètres au démarrage.
- Main.py : exécute le programme principal.
- Information.env : contient les configurations réseau et API.

Le guide inclut également des solutions de dépannage et des instructions pour la mise à jour des scripts et des paramètres (luminosité, volume, etc.). Il assure ainsi une installation et une maintenance simplifiées du projet.

Le contenu du fichier est disponible en annexe 1.

4.2. Le fichier boot.py

boot.py est un script crucial sur les microcontrôleurs comme le Raspberry Pi Pico W. Il est exécuté automatiquement lors du démarrage de l'appareil, avant tout autre script (y compris main.py). Son but est de :

- Préparer l'environnement :
  Initialiser le matériel et les modules nécessaires (réseau, GPIO, gestion des erreurs, etc.).
  Charger les fichiers de configuration.
- Gérer les paramètres essentiels au système :
  Activer le WiFi.
  Vérifier que les fichiers nécessaires (comme main.py) sont présents.
- Lancer automatiquement main.py : Si tout est configuré correctement, boot.py exécute main.py pour démarrer l'application principale.

Le contenu du fichier est disponible en annexe 2.

4.3. Le fichier information.env

Le fichier information.env contient des paramètres essentiels pour le fonctionnement du projet. Il sert à stocker des informations de configuration sans avoir à les écrire directement dans le code.

- *API_KEY : Clé d'accès pour utiliser une API externe.*
- *SSID : Nom du réseau WiFi auquel l'appareil doit se connecter.*
- *WIFI_PASSWORD : Mot de passe du réseau WiFi.*

Ce fichier permet au programme de récupérer ces informations automatiquement au démarrage, sans modifier le code source. Il est utilisé principalement pour gagner du temps et éviter d'exposer des informations sensibles dans le programme.

Le contenu du fichier est disponible en annexe 3.

4.4. Le fichier main.py

Le fichier main.py constitue le noyau du système. Il renferme l'ensemble des logiques requises pour la mise en œuvre des fonctionnalités essentielles du projet. Dans les paragraphes suivants, nous vous donnons une explication approfondie des éléments essentiels de ce fichier.

Le fichier de code source principal est particulièrement volumineux, ce qui le rend difficile à intégrer en annexe de ce rapport. Toutefois, ce fichier est disponible et accessible en ligne à l'adresse suivante :

https://github.com/adriens/temps-attente-matrix-led/blob/main/src/main.py

4.4.1. Le développement du script

Le script a été conçu de façon organisée et modulaire afin d'atteindre divers buts techniques et fonctionnels, tout en prenant en considération les spécificités propres au dispositif Cosmic Unicorn. Voici les motifs clés qui justifient cette démarche :

Objectif principal : Afficher des informations dynamiques sur un écran LED

L'objectif est de montrer des informations en temps réel (comme les temps d'attente des agences) de manière visuelle et interactive. Le script a été conçu pour :

- Extraire les données depuis une API.
- Les transformer en un format compréhensible et attrayant pour l'utilisateur.
- Utiliser des animations et des graphiques pour rendre l'expérience plus engageante.

Appareil ciblé : Cosmic Unicorn

L'appareil Cosmic Unicorn est un écran LED programmable avec des capacités audio et lumineuses. Le script est conçu pour exploiter au maximum ses fonctionnalités :

- LEDs et graphique : Le script utilise des dessins pixelisés (lettres, chiffres, smileys) adaptés à un affichage à basse résolution.
- Son : Le son est utilisé pour des notifications interactives (activation, erreurs, etc.).
- Boutons : L'appareil possède des boutons qui permettent de naviguer entre différents écrans et de modifier les paramètres.

Gestion de la complexité avec une structure modulaire

Le script est découpé en sections et fonctions pour faciliter :

- La lisibilité : Chaque fonction joue un rôle défini, simplifiant la compréhension du script.
- La maintenance : il est facile de localiser et de corriger le problème.
- L'évolutivité : De nouvelles fonctionnalités peuvent être ajoutées.

Conception centrée sur l'utilisateur

Le script intègre des éléments visuels et sonores pour améliorer l'expérience utilisateur :

- Feedback visuel : Des smileys et des LEDs indiquent rapidement l'état (heureux, neutre, triste, WiFi connecté ou non, etc.).
- Feedback sonore : Les bips confirment des actions ou signalent des erreurs.
- Navigation : Les boutons permettent de changer d'écran ou d'ajuster des paramètres comme la luminosité ou le volume.

Fonctionnalités spécifiques requises

Affichage des agences et de leurs temps d'attente :

- Les agences sont chargées depuis une API pour garantir des données à jour.
- Les temps d'attente sont affichés sous forme de smileys et de chiffres.

Connexion WiFi : L'appareil doit se connecter au WiFi pour accéder aux données.

Le script gère les erreurs de connexion avec des LEDs rouges et des messages.

Synchronisation de l'heure : L'heure est essentielle pour afficher des données temporelles correctes (temps d'attente, horloge).

Animations et transitions : Les animations (comme le cœur explosant ou le texte défilant) rendent l'expérience utilisateur plus attrayante.

Robustesse et gestion des erreurs

Le script est conçu pour gérer des scénarios inattendus :

- Erreurs de connexion WiFi : Le script affiche un message et une animation pour alerter l'utilisateur.
- Problèmes API : Si l'API est inaccessible, un message explicatif est affiché.
- Gestion des redémarrages : Le bouton D permet de redémarrer proprement l'appareil en cas de problème.

Optimisation pour un environnement embarqué

L'appareil Cosmic Unicorn a des ressources limitées (mémoire, puissance de calcul). Le script est optimisé pour cet environnement :

- Mémoire : Le module gc est utilisé pour libérer la mémoire à chaque requête.
- Temps de réponse : Les animations et mises à jour sont conçues pour être rapides.
- Fiabilité : Les fonctions critiques (comme la mise à jour des agences) sont encapsulées dans des blocs avec gestion des exceptions.

Contexte pratique et adaptabilité

Scénarios d'utilisation :

- OPT NC : Les agences peuvent afficher des temps d'attente en temps réel pour informer les clients.
- Personnalisation : L'écran pourrait être utilisé pour d'autres scénarios, comme afficher des messages promotionnels ou des statistiques.
- Adaptabilité : Le script est conçu pour être modifié facilement (ajout de nouvelles agences, personnalisation des animations, etc.).

4.4.2. La structuration du script

Le script main.py est structuré en plusieurs éléments qui remplissent des fonctions spécifiques pour l'affichage et la gestion des informations sur l'écran Cosmic Unicorn.

4.4.2.1. Les Imports

Ils permettent d'inclure et d'utiliser des modules ou des bibliothèques externes au script.

Ils fournissent des fonctionnalités pré-écrites (gestion du temps, de la mémoire, de la connexion réseau, etc.) que vous n'avez pas besoin de réécrire vous-même.

4.4.2.2. Les classes

Ce sont des structures qui regroupent des données (attributs) et des comportements (méthodes) au sein d'un même objet.

Elles servent à modéliser des entités réelles ou abstraites du programme. Dans ce script, la classe CosmicUnicornDisplay encapsule toutes les fonctionnalités liées à l'affichage et à l'interaction avec la matrice LED, facilitant ainsi la réutilisation et l'organisation du code.

La classe CosmicUnicornDisplay emploie plusieurs méthodes liées à l'afficha et l'interaction avec la matrice LED :

4.4.2.3. Les définitions de fonctions (def)

Une fonction est un bloc de code qui réalise une tâche spécifique et qui peut être appelé depuis d'autres parties du programme.

Elles permettent de décomposer le programme en petites unités logiques et réutilisables, ce qui améliore la lisibilité et la maintenance du code.

4.4.2.4. Variables globales

Les variables globales sont définies en dehors de toute fonction ou classe. Elles permettent de stocker des informations accessibles depuis n'importe quelle partie du script.

4.4.2.5. Letter-Map (LETTER_MAP_3 et LETTER_MAP_4)

Ces dictionnaires servent de tables de correspondance pour dessiner des lettres sur la matrice LED. Chaque clé (une lettre) est associée à une liste de coordonnées (x, y) qui indiquent quelles LED allumer pour représenter cette lettre.

4.4.2.6. handle_button_press

Cette fonction est dédiée à la gestion immédiate des boutons dans le contexte global du programme.

4.4.2.7. Les boucles du script

Boucle main

La fonction main() constitue le point d'entrée du programme.

- Initialisation : Crée une instance de CosmicUnicornDisplay et configure les paramètres initiaux.
- Démarrage du thread pour le bouton D : Lance un thread dédié pour surveiller le bouton D (redémarrage).
- Chargement et configuration : Affiche un écran de chargement, lit les informations de connexion (WiFi et API), synchronise l'heure, et charge les données des agences.
- Gestion des modes d'affichage : Définit une liste de fonctions d'affichage (accueil, info, légende, agences, QR Code) et affiche le premier écran, puis gère la navigation entre les écrans via le bouton C.

Elle orchestre l'ensemble du programme en établissant l'ordre d'exécution et en appelant les autres fonctions de façon séquentielle.

Boucle main_loop

La fonction main_loop() est spécifique au mode d'affichage des agences.

- Mise à jour en continu : Dans une boucle (avec 100 itérations à l'intérieur d'une boucle principale), elle met à jour l'affichage pour une agence donnée.
- Gestion dynamique : Elle vérifie en permanence l'état du WiFi, récupère le temps d'attente, détermine le "mood" (happy, neutral, sad), et affiche le smiley ainsi que le nom de l'agence défilant.
- Vérification des boutons : Elle permet de réagir aux appuis sur les boutons A (son), B (pause de la boucle), C (changement d'écran) et D (redémarrage) de façon réactive grâce à sa fréquence de sondage rapide.

C'est le cœur de l'affichage en mode agences, où le système doit être extrêmement réactif et mettre à jour en temps réel l'information affichée.

Un tableau simplifié détaillant les grandes fonctions est disponible en annexe 4.

Le processus illustrant le principe du script est disponible en annexe 5.

5. Difficultés rencontrées

5.1. Le DEvOps et la complexité liée au Langage MicroPython

Le choix de MicroPython comme langage de programmation a apporté des avantages en termes de simplicité syntaxique et de rapidité de prototypage, mais également plusieurs contraintes techniques :

Limitations matérielles et gestion de la mémoire

Sur microcontrôleur, la mémoire est limitée. Il a fallu optimiser la gestion des ressources lors de l'utilisation de bibliothèques tierces (ex : urequests pour les appels API) et veiller à la libération régulière de la mémoire (utilisation de gc.collect()).

Complexité du code graphique et des animations

La création de polices personnalisées pour l'affichage sur la matrice Cosmic Unicorn implique la manipulation directe des coordonnées LED. Le code doit gérer précisément le défilement de texte, les animations (ex. animation de chargement ou d'explosion de cœur) et la synchronisation avec l'affichage.

Asynchronisme et gestion des threads

Le recours aux threads via le module _thread pour exécuter certaines tâches en parallèle (ex : contrôle des boutons ou l'ajustement du volume en temps réel) a posé des problèmes de synchronisation et de concurrence, nécessitant une attention particulière à l'ordre d'exécution des tâches.

Monitoring en temps réel et réaction aux défaillances

Le dispositif doit surveiller l'état de la connexion WiFi, la synchronisation de l'heure et l'accessibilité de l'API. La mise en œuvre d'un système de de gestion des erreurs (via stop_script), a permis de détecter et de réagir rapidement aux dysfonctionnements, garantissant la continuité du service en conditions réelles.

5.2. Utilisation de GitHub et Standards de Développement

La gestion collaborative du projet via GitHub a été une étape déterminante, mais non dénuée de difficultés.

Respect des standards de contribution

Au départ, l'adaptation aux standards GitHub (conventions de nommage, messages de commit explicites, gestion des branches, utilisation des pull requests pour la revue de code) s'est révélée complexe.

Gestion des retours et communication

L'utilisation de GitHub pour centraliser les échanges (issues, discussions, pull requests) a favorisé une communication fluide au sein de l'équipe, mais a également impliqué une adaptation à un mode de collaboration asynchrone.

Présentation du projet via Reveal.JS

La création d'une présentation interactive avec Reveal.JS a permis de présenter le projet de manière claire et professionnelle. L'implémentation a nécessité de maîtriser l'outil, de préparer des slides cohérents, et d'intégrer des notes et une télécommande pour le présentateur afin de faciliter cette présentation.

6. Les perspectives d'évolutions

La matrice LED déployée dans le cadre du projet pour l'OPT NC offre plusieurs perspectives d'évolution afin d'optimiser son utilisation et d'enrichir l'expérience client :

1. **Affichage d'informations contextuelles** : Outre les temps d'attente, la matrice pourrait afficher d'autres informations utiles telles que les heures de pointe, les services disponibles dans chaque agence, ou encore des conseils pour optimiser la visite des clients.
2. **Personnalisation de l'affichage** : La matrice pourrait être configurée pour adapter automatiquement le contenu affiché selon l'heure de la journée ou le type de clientèle (ex. affichage en plusieurs langues pour une meilleure accessibilité).
3. **Interactivité** : L'ajout de fonctionnalités interactives, telles que des boutons tactiles ou un capteur de présence, permettrait aux usagers de sélectionner des informations spécifiques (ex. temps d'attente pour un service particulier).
4. **Visualisation de tendances** : Afficher des graphiques ou des tendances sur les temps d'attente des derniers jours permettrait aux clients de mieux planifier leur visite en fonction des périodes les plus calmes.
5. **Intégration avec des systèmes de file d'attente virtuels** : Les clients pourraient réserver une place dans la file d'attente depuis leur smartphone, et la matrice LED pourrait afficher les numéros de passage en temps réel.
6. **Notifications et alertes** : La matrice pourrait être utilisée pour diffuser des alertes importantes (ex. fermeture exceptionnelle de l'agence, problème technique en cours) ou pour notifier les clients lorsque leur tour approche.
7. **Affichage d'indicateurs de satisfaction** : La matrice pourrait afficher des indicateurs de satisfaction clients (ex. pourcentage de clients satisfaits, temps moyen d'attente sur la journée) afin de renforcer la transparence et la confiance envers le service.
8. **Connexion avec d'autres systèmes** : L'intégration avec d'autres systèmes d'information de l'OPT NC (ex. CRM, ERP) permettrait d'enrichir les données affichées et d'automatiser les mises à jour des informations.
9. **Évolutivité technique** : La matrice pourrait évoluer vers un modèle connecté au cloud pour permettre une gestion centralisée et à distance de tous les écrans déployés dans différentes agences.
10. **Support de contenus multimédias** : La matrice pourrait afficher des vidéos explicatives sur les services proposés, des messages institutionnels ou des campagnes de sensibilisation.
11. **Amélioration de la visibilité extérieure** : Installation de matrices LED visibles depuis l'extérieur des agences pour informer les clients avant même qu'ils ne se rendent à l'intérieur.
12. **Évaluation de la performance en temps réel** : La matrice pourrait afficher des KPIs liés à la performance de l'agence, tels que le nombre de clients servis ou le temps d'attente moyen.
13. **Optimisation énergétique** : La mise en place d'un système d'extinction automatique ou d'économie d'énergie lorsque la matrice LED n'est pas utilisée permettrait de réduire les coûts d'exploitation.

Ces perspectives d'évolution visent à maximiser l'utilité de la matrice LED tout en contribuant à l'amélioration continue des services offerts par l'OPT NC.

7. Conclusion

Le projet de déploiement d'une matrice LED pour l'affichage des temps d'attente en agence, réalisé pour le compte de l'OPT NC, s'est inscrit dans une démarche globale d'amélioration de la qualité de service et de modernisation des processus internes. Ce projet a permis de répondre à plusieurs enjeux clés, tels que la réduction de la frustration des clients liée aux files d'attente, la transparence sur les délais d'attente et l'optimisation de la gestion des flux d'usagers dans les agences.

Grâce à l'utilisation des données existantes et à leur valorisation au travers d'une datavisualisation dynamique, ce projet pourra apporter une valeur ajoutée significative à la fois pour les clients de l'OPT NC et pour l'organisation elle-même. En offrant une meilleure visibilité sur les temps d'attente et en permettant une prise de décision éclairée quant à la gestion des ressources humaines dans les agences, cette solution pourra contribuer à l'efficacité opérationnelle de l'OPT NC.

Sur le plan technique, ce projet a été l'occasion de mettre en pratique des compétences avancées en gestion de projet IT, en datavisualisation, et en exploitation de données en temps réel. Il a également permis d'appliquer des méthodologies agiles favorisant une adaptation rapide aux retours des utilisateurs et une amélioration continue de la solution déployée. Par ailleurs, cette expérience a offert l'opportunité d'aborder des problématiques réelles d'intégration de technologies numériques, de gestion des données et de satisfaction client.

Ce projet ouvre également la voie à plusieurs perspectives d'évolution qui pourraient encore renforcer l'utilité et l'impact de la matrice LED. Parmi ces évolutions, on peut citer l'ajout de fonctionnalités interactives, telles que la consultation de services spécifiques, l'affichage de messages personnalisés, ou encore la possibilité pour les clients de réserver une place dans la file d'attente via une application web ou mobile. De plus, l'intégration d'indicateurs de satisfaction client, l'affichage de tendances historiques ou encore la connexion à un système centralisé de gestion des agences permettraient d'enrichir la solution et d'en maximiser les bénéfices.

En conclusion, ce projet constitue une première étape réussie vers une gestion plus fluide et transparente des agences de l'OPT NC, tout en répondant à des besoins d'innovation technologique et de modernisation des services publics. Il démontre l'importance d'une approche orientée données pour améliorer la performance opérationnelle et renforcer la satisfaction des usagers. À terme, cette initiative pourrait servir de modèle pour d'autres projets visant à utiliser la technologie au service d'une meilleure expérience client et d'une gestion optimisée des ressources. Ce projet marque ainsi le début d'une dynamique d'amélioration continue, au bénéfice des clients et de l'organisation dans son ensemble.

8. Webographie

MicroPython Garbage Collector (gc) Module
Description : Documentation du module gc utilisé pour la gestion de la mémoire dans le script.
Lien : https://docs.micropython.org/en/latest/library/gc.html

MicroPython Network Module
Description : Documentation officielle du module network de MicroPython utilisé pour la gestion des connexions réseau.
Lien : https://docs.micropython.org/en/latest/library/network.html

MicroPython NTP Time (ntptime) Module
Description : Documentation du module ntptime utilisé pour synchroniser l'heure via un serveur NTP.
Lien : https://docs.micropython.org/en/latest/library/ntptime.html

MicroPython OS Module
Description : Documentation sur le module os utilisé pour la gestion des fichiers et des chemins.
Lien : https://docs.micropython.org/en/latest/library/os.html

MicroPython uRequests Module
Description : Documentation sur le module urequests utilisé pour effectuer des requêtes HTTP et récupérer des données via des APIs.
Lien : https://docs.micropython.org/en/latest/library/urequests.html

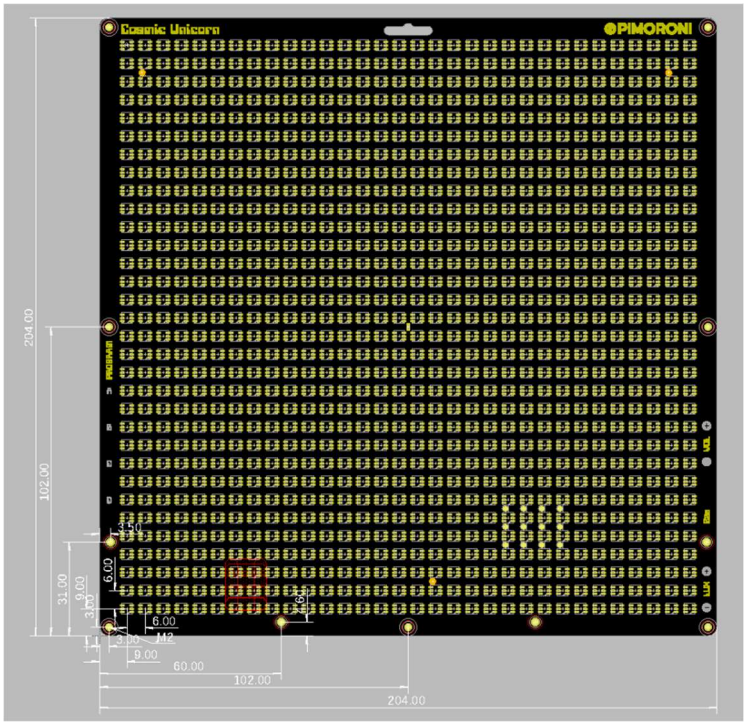
Pimoroni CosmicUnicorn Library
Description : Documentation pour la bibliothèque CosmicUnicorn utilisée pour gérer l'affichage LED du projet.
Lien : https://pimoroni.com/docs/CosmicUnicorn

Pimoroni PicoGraphics Library
Description : Documentation pour la bibliothèque PicoGraphics, utilisée pour la gestion graphique sur la matrice LED.
Lien : https://pimoroni.com/docs/PicoGraphics

Raspberry Pi Pico SDK:
Raspberry Pi Foundation. (n.d.). Raspberry Pi Pico C/C++ SDK. Raspberry Pi Documentation.
Lien : https://datasheets.raspberrypi.com/pico/raspberry-pi-pico-c-sdk.pdf

9. Annexes

Annexe 0. Matrice LED Pimoroni Cosmic Unicorn



Affichage et LEDs :

- 1024 LEDs RGB organisées en une grille de 32x32.
- Contrôle individuel de la couleur et de la luminosité de chaque LED.
- LED avec diffusion intégrée pour un rendu homogène et attrayant.
- 14-bit de précision dans le traitement de l'image, garantissant une luminosité fluide, même à bas niveau.

Connectivité et programmation :

- Basée sur le microcontrôleur RP2040 (Raspberry Pi Pico W intégré) :
- Dual-core ARM Cortex M0+ (jusqu'à 133 MHz).
- Connexion Wi-Fi 2.4 GHz pour des applications IoT.
- Prise en charge de MicroPython et C/C++, avec des bibliothèques dédiées (PicoGraphics).
- Compatible avec des interfaces Qwiic/STEMMA QT pour connecter facilement des modules supplémentaires.

Fonctionnalités intégrées :

- Haut-parleur mono 1W avec amplificateur pour générer des alertes sonores.
- Capteur de lumière (phototransistor) pour ajuster la luminosité en fonction de l'environnement.
- 9 boutons tactiles pour une interaction utilisateur directe.

Compatibilité et accessoires :

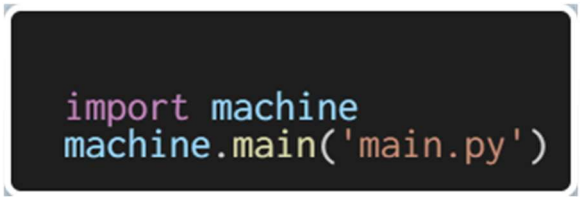
- Préchargée avec MicroPython et des exemples pour faciliter la prise en main.
- Possibilité d'ajouter une batterie (jusqu'à 5.5V) pour des applications mobiles.

Annexe 1. Contenu du fichier « HOW_TO_INSTALL »

```
# Cosmic Unicorn Project Setup

This guide explains how to set up the Cosmic Unicorn project, including configuring WiFi, copying
necessary files, and deploying the display scenes. Follow these steps to get started.

# Prerequisites

**Hardware**: Pimoroni Cosmic-Unicorn 32x32 LED matrix with Raspberry Pi Pico W.

**Software**: Thonny IDE (required for file uploads and troubleshooting), GitHub access, WiFi
credentials.

**Files Needed**:
  - `boot.py`: The boot file to ensure the script runs automatically.
  - `main.py`: The main Python script.
  - `information.env`: File containing WiFi credentials and the API key.

# 1. Install Thonny IDE

Download and install Thonny IDE from [thonny.org](https://thonny.org/). Thonny is **required** for this
project, but only to manage file uploads and necessary modifications (such as customizing the
`information.env` file). Thonny can also be used to troubleshoot issues.

## Using Thonny for the project:
- **File Upload**: Use Thonny to upload `boot.py`, `main.py`, and `information.env` to the Pimoroni
Cosmic-Unicorn.
- **Modifying `information.env`**: Customize the WiFi credentials and API key in the `information.env`
file.
- **Troubleshooting**: Thonny is used to diagnose issues and test scripts on the LED matrice.


# 2. Clone the GitHub Repository

Clone the repository to your local machine:
  git clone https://github.com/adriens/temps-attente-matrix-led.git

Navigate to the project directory
  cd temps-attente-matrix-led

# 3. Configure WiFi Credentials and API Key

**Edit the .env file** to provide your **WiFi credentials** :
SSID=<your-SSID>
WIFI_PASSWORD=<your-password>

Add your **API key** in the format :
  API_KEY=<your-api-key>

# 4. Copy Files to Raspberry Pi Pico

Connect your Raspberry Pi Pico W to your computer.

Open Thonny and select the correct interpreter for Raspberry Pi Pico W.
Save the following files to the Pico's root directory:
- boot.py
- main.py
- information.env
Use Thonny's file browser to ensure these files are saved correctly on the Pico.

# 6. Running the Script

Once the files are uploaded, the Pimoroni Cosmic-Unicorn will run autonomously as soon as it is powered
on. No further intervention via Thonny is required for final usage.

# 7. Troubleshooting

* **WiFi Not Connecting**: Ensure correct SSID/password in the .env file and check network availability.
* **API Key Issues**: Make sure the API key in the .env file is correct and the file is in the correct
directory.
* **Button Response Not Working**: Ensure the Cosmic Unicorn device buttons are functioning correctly
and properly mapped.

# 8. Updating the Script

If there are any changes to the script, make sure to copy the updated version to the Raspberry Pi Pico.
Use Thonny IDE to overwrite the old script on the Pico's filesystem.

# 9. Volume and Brightness Adjustment

* **Volume**: Use the buttons on the Cosmic Unicorn to adjust the sound volume during runtime.
* **Brightness**: Adjust the brightness using the appropriate buttons on the Cosmic Unicorn.

# 10. Button Functionalities
* **Button A "SOUND"**: Toggles the sound of the LED matrix on and off.
* **Button B "LOOP"**: During the agency display, if you wish to remain on the currently shown agency,
pressing this button locks the display loop to prevent automatic changes.
* **Button C "DISPLAY"**: During the display phase, pressing this button cycles through the different
screens (Home, Information, Legend, Agencies, QR_Code).
* **Button D "RESTART"**: Restarts the LED matrix.
* **Volume +/- Buttons**: Adjust the sound intensity of the LED matrix.
* **Brightness +/- Buttons**: Adjust the brightness of the LED matrix.

# Additional Notes
Feedback and Contributions: If you'd like to contribute or provide feedback on this guide, please open
an issue or submit a pull request on GitHub.
```

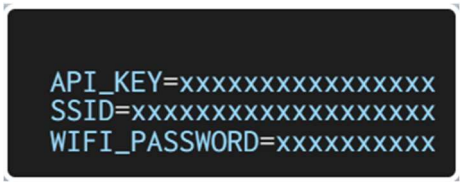
https://github.com/adriens/temps-attente-matrix-led/blob/main/HOW_TO_INSTALL.md

Annexe 2. Contenu du fichier « Boot.py »



https://github.com/adriens/temps-attente-matrix-led/blob/main/src/boot.py

Annexe 3. Contenu du fichier « information.env »



https://github.com/adriens/temps-attente-matrix-led/blob/main/src/information.env

Annexe 4. Les grands modules du script

Détail du fonctionnement de chaque élément du script

Les Imports

- time : Gère les délais, mesures de temps et temporisations.
- network : Permet de gérer la connexion WiFi.
- ntptime : Utilisé pour synchroniser l'heure via des serveurs NTP.
- urequests as requests : Pour effectuer des requêtes HTTP (accès à des APIs).
- os : Pour la gestion des fichiers (par exemple, lecture du fichier de configuration).
- gc : Gestion du ramassage de mémoire (garbage collector) pour libérer la mémoire.
- _thread : Pour lancer des threads et exécuter des tâches en parallèle.
- machine : Fournit un accès aux fonctions matérielles du microcontrôleur (réinitialisation, etc.).
- cosmic (CosmicUnicorn) : Module spécifique pour piloter la matrice LED Cosmic Unicorn.
- picographics (PicoGraphics, DISPLAY_COSMIC_UNICORN) : Pour gérer les graphiques et dessiner sur la matrice LED.

Variables Globales

- attempts : Compteur pour suivre les tentatives de connexion WiFi.
- COLORS : Dictionnaire associant des noms de couleurs à leurs valeurs RGB pour l'affichage.

La classe CosmicUnicornDisplay

Cette classe encapsule toutes les fonctionnalités liées à l'affichage et à l'interaction avec la matrice LED.

- Méthode __init__ : Initialise l'instance du Cosmic Unicorn, les outils graphiques, les paramètres (luminosité, volume, mode d'affichage), crée les « pens » pour chaque couleur et lance la configuration initiale (affichage, LED, etc.).
- Méthode clear : Efface l'écran tout en préservant certains indicateurs (par exemple, la LED de pause) et met à jour l'état des LED (son, wifi).
- Méthode update : Rafraîchit l'affichage en envoyant le contenu graphique à la matrice.
- Méthode set_pen : Change la couleur du « stylo » utilisé pour dessiner.
- Méthode scroll_text : Gère le défilement d'un texte sur l'écran, en utilisant un décalage et un timing précis.
- Méthode draw_frame : Dessine un cadre autour d'un élément (par exemple, autour du smiley) en traçant des lignes aux bords de l'écran.
- Méthode draw_text_opt : Affiche un texte fixe ("OPT") en utilisant des coordonnées prédéfinies pour dessiner chaque lettre.
- Méthode draw_smiley : Dessine un smiley dont l'expression (happy, neutral, sad) dépend d'une valeur d'attente. Intègre la logique pour dessiner des LED indiquant le temps d'attente.
- Méthode play_bip : Joue un bip sonore via le canal synthétique si le son est activé.
- Méthode adjust_brightness : Ajuste la luminosité de l'écran en fonction des appuis sur les boutons dédiés.
- Méthode adjust_volume : Ajuste la fréquence (volume) du bip sonore selon les boutons de volume.
- Méthode display_digit et display_clock : Affichent respectivement un chiffre et une horloge en décomposant l'heure en chiffres et en dessinant chaque chiffre à l'aide d'une matrice LED.
- Méthode set_transition_variable : Stocke le texte à faire défiler.

- Méthode display_message_frame_2 : Affiche un message centré dans un rectangle défini sur l'écran.
- Méthode toggle_sound et update_led_sound_status : Gèrent l'activation/désactivation du son et mettent à jour les LED associées.
- Méthode clear_sound_leds : Efface l'affichage des LED relatives au son.
- Méthode toggle_loop_pause : Permet de mettre en pause ou de reprendre la boucle d'affichage des agences, en indiquant visuellement cet état.
- Méthode update_led_wifi_status et check_wifi_status : Vérifient la connexion WiFi et mettent à jour l'affichage des LED correspondantes.

Lettres et affichage de texte

- LETTER_MAP_3 et LETTER_MAP_4 : Tables de correspondance pour dessiner des lettres sur la matrice LED (en tailles 3x5 et 4x5).
- Fonctions draw_letter_3, draw_word_3, draw_letter_4, draw_word_4 : Utilisées pour dessiner des lettres ou des mots entiers sur l'écran à l'aide des matrices de lettres.

Fonctions d'affichage et d'animation

- show_loading_screen et loading_animation_step : Affichent une animation de chargement (texte "WAIT" et blocs animés) pendant les étapes critiques.
- display_welcome_screen : Affiche une animation d'accueil avec défilement de textes "UNC" et "OPT".
- exploding_heart_animation : Réalise une animation d'un cœur qui "explose" sur l'écran.
- display_info_screen : Affiche l'état du WiFi, de la clé API et du fichier de configuration (agences.env) en utilisant des codes couleurs.
- display_legend_screen : Affiche une légende indiquant l'état des LED (son, WiFi, etc.).
- display_qr_code_screen : Affiche un QR code composé de positions LED, avec une boucle pour ajuster la luminosité en temps réel.
- wait_for_start : Attend la pression du bouton C pour démarrer le script principal.

Fonctions utilitaires

- normalize_name : Supprime les accents et convertit le texte en majuscules.
- stop_script : Affiche un message d'erreur et attend la pression du bouton D pour redémarrer - (avec machine.reset()).
- poll_button_D : Fonction exécutée dans un thread pour surveiller en permanence le bouton D et redémarrer la carte dès qu'il est pressé.

Fonctions de communication et de synchronisation

- load_credentials : lit le fichier "information.env" pour charger le SSID, le mot de passe WiFi et la clé API.
- connect_wifi : Tente de se connecter au WiFi en effectuant plusieurs tentatives et en affichant une animation de chargement.
- sync_time : Synchronise l'heure via des serveurs NTP.
- load_agencies_from_api et update_single_agency/update_agency_waiting_time : Appellent l'API pour récupérer et mettre à jour les données d'attente des agences.
- initialize_agencies et initialize_agency_wait_times : Boucles pour initialiser ou mettre à jour les temps d'attente pour l'ensemble des agences.

Gestion de la boucle d'affichage et des interactions

- handle_button_press : Gère la détection des appuis sur les boutons A, B, C et D (le bouton D est aussi géré par le thread dédié) pour ajuster le son et le volume.

main - Point d'entrée du programme.

Initialise l'affichage, démarre le thread de sondage du bouton D, affiche l'écran de chargement, charge les informations de connexion, connecte au WiFi, synchronise l'heure, charge les données des agences, configure les différents modes d'affichage et gère la navigation entre ces modes via les boutons (notamment le bouton C pour changer d'écran).

- Initialisation : Crée et configure l'affichage, lance un thread de surveillance du bouton D.
- Configuration et connexion : Affiche des écrans de chargement, lit le fichier de configuration, connecte au WiFi, synchronise l'heure.
- Chargement des données : Récupère les informations des agences via l'API.
- Lancement des modes d'affichage : Définition d'une liste de modes d'affichage et affichage du premier écran.
- Boucle principale :

  Surveille en continu les boutons pour activer/désactiver le son, changer de mode d'affichage, et gérer les interactions (via handle_button_press).

  Elle permet ainsi une mise à jour dynamique de l'affichage et assure la réactivité du système.

main_loop - Boucle dédiée au mode d'affichage des agences.

Pour chaque agence, le script met à jour l'affichage (vérification du WiFi, choix du smiley selon le temps d'attente, affichage du nom, etc.) et vérifie en permanence les boutons pour permettre le passage à l'écran suivant ou le redémarrage.

Elle repose sur une séquence d'opérations qui se répète pour chaque agence :

- Préparation : Effacer l'écran, afficher un message "WAIT", mettre à jour l'affichage initial.
- Mise à jour des données : Vérifier la connexion WiFi, récupérer et mettre à jour les informations d'une agence via l'API.
- Détermination du "mood" : Selon le temps d'attente, choisir entre happy, neutral ou sad pour le smiley.
- Affichage dynamique :

  Effacer l'écran.

  Afficher le sigle "OPT" (draw_text_opt()).

  Afficher le smiley (draw_smiley(mood)).

  Configurer le texte défilant avec le nom de l'agence (set_transition_variable(name)).

  Mettre à jour les LED de son (update_led_sound_status()).
- Boucle d'interaction (100 itérations) :

  Vérifier les boutons (A, B, C, D) pour activer le son, mettre en pause la boucle, changer d'écran ou redémarrer.

  Actualiser en continu le texte défilant (scroll_text()) et l'horloge (display_clock()).

  Ajuster la luminosité et le volume.
- Passage à l'agence suivante : Mettre à jour l'agence et recommencer le cycle.

Ces fonctions et boucles se combinent pour offrir une mise à jour en temps réel de l'affichage des agences et une réponse immédiate aux interactions via les boutons.

Annexe 5. Le principe du code