**Mémoire de Projet**

*Concepteur développeur d’applications*

Application BlockLumen

Plateforme de simulation de trading de cryptomonnaies.

Table des matières

[1. Présentation 3](#_Toc199704615)

[a. Introduction en français 3](#_Toc199704616)

[b. Introduction et présentation du projet en anglais 3](#_Toc199704617)

[2. Contexte 4](#_Toc199704618)

[a. Objectif 4](#_Toc199704619)

[b. Technologies 4](#_Toc199704620)

[1. Back-end 4](#_Toc199704621)

[2. Front-end 4](#_Toc199704622)

[3. OS / IDE / Utilitaires 4](#_Toc199704623)

[3. Le Design 5](#_Toc199704624)

[a. Maquettage et Conception Graphique 5](#_Toc199704625)

[1. Logo 5](#_Toc199704626)

[2. Palette graphique 5](#_Toc199704627)

[3. La typographie 5](#_Toc199704628)

[4. Maquette & WireFrames 6](#_Toc199704629)

[4. Conception et modélisation des données 7](#_Toc199704630)

[a. Clarification des besoins 7](#_Toc199704631)

[b. Dictionnaire de données 8](#_Toc199704632)

[c. Modèle Conceptuel Des Données ( MCD ) 9](#_Toc199704633)

[d. Modèle Logique Des Données ( MLD ) 9](#_Toc199704634)

[e. Modèle Physique Des Données ( MPD ) 10](#_Toc199704635)

[f. Diagrammes UML 10](#_Toc199704636)

[1.Diagramme de cas d’utilisation 10](#_Toc199704637)

[2. Diagramme de Flux 10](#_Toc199704638)

[3. Diagramme d’entités 10](#_Toc199704639)

[5. Environnement de développement 10](#_Toc199704640)

[6. Développement de l’application 10](#_Toc199704641)

[7. 10](#_Toc199704642)

[15. Conclusion 10](#_Toc199704643)

[a. Remerciements 10](#_Toc199704644)

[b. Axes d’amélioration 10](#_Toc199704645)

[c. Retour d’expérience 10](#_Toc199704646)

# 1. Présentation

## a. Introduction en français

Je m’appelle Nicolas Lopez. En 2022, j’ai intégré la Prépa Concepteur Développeur d’Applications à l’Idem, où j’ai validé ma première année avec succès.

Dans ce cadre, j’ai effectué un stage de fin d’études chez REVIMPORT S.A.S., dont la mission était double : me familiariser avec les process en entreprise et lancer la migration de l’ERP Navision vers Microsoft Business Central. Mon implication m’a valu une gratification de stage et l’offre d’un contrat d’alternance d’un an.

Durant cette alternance, j’ai poursuivi l’analyse fonctionnelle, la conception de la base de données et le développement des premiers modules métiers. Malheureusement, REVIMPORT a été placée en liquidation judiciaire, entraînant un licenciement économique et l’interruption du projet de migration.

Aujourd’hui, dans le cadre de ma deuxième année et pour valider mon titre RNCP 6, je consacre mon mémoire à BlockLumen, une plateforme de simulation de trading de cryptomonnaies. Ce projet représente l’aboutissement de ma formation et de mes expériences.

## b. Introduction et présentation du projet en anglais

# 2. Contexte

Le marché des cryptomonnaies, en forte croissance, reste très volatil et complexe pour les débutants. BlockLumen propose une simulation sécurisée du trading (achats, ventes, gestion de portefeuille) enrichie de modules pédagogiques ~~et d’une veille d’actualités pour guider l’utilisateur pas à pas~~. Ce nom, court et évocateur, associe “Block” (blockchain) et “Lumen” (lumière), pour symboliser l’éclaircissement et la compréhension du trading crypto

## a. Objectif

Permettre aux utilisateurs de tester leurs stratégies dans des conditions de marché réalistes (cours différés et frais inclus) sans risquer leur capital, grâce à une interface claire et responsive sur desktop et mobile.

## b. Technologies

### 1. Back-end

**Node.js & Express :** Cadre JavaScript léger et performant pour développer les API REST.

**MySQL :** Base de données relationnelle robuste, hébergeant utilisateurs, portefeuilles et historiques de transactions.

**Docker :** Containerisation du serveur et de la base de données pour garantir portabilité et isolation.

**TypeScript :** Typage statique pour renforcer la fiabilité et la maintenabilité du code.

### 2. Front-end

**React :** Bibliothèque JavaScript pour construire une interface dynamique et réactive.

**Vite :** Outil de bundling ultra-rapide, optimisé pour le développement moderne.

**Tailwind CSS :** Framework utilitaire facilitant la mise en forme et le design cohérent.

**React-Chart.js-2 / Chart.js :** Pour afficher graphiques de prix et indicateurs en temps réel.

### 3. OS / IDE / Utilitaires

**Git & GitHub :** Gestion de versions et collaboration.

**VS Code :** Environnement de développement, avec extensions TypeScript et Docker.

**Postman :** Tests et appels d’API.

**Figma :** Prototypage des écrans et design UX/UI.

**app.diagrams.net / Draw.io :** Modélisation du MCD et des architectures applicatives.

# 3. Le Design

## a. Maquettage et Conception Graphique

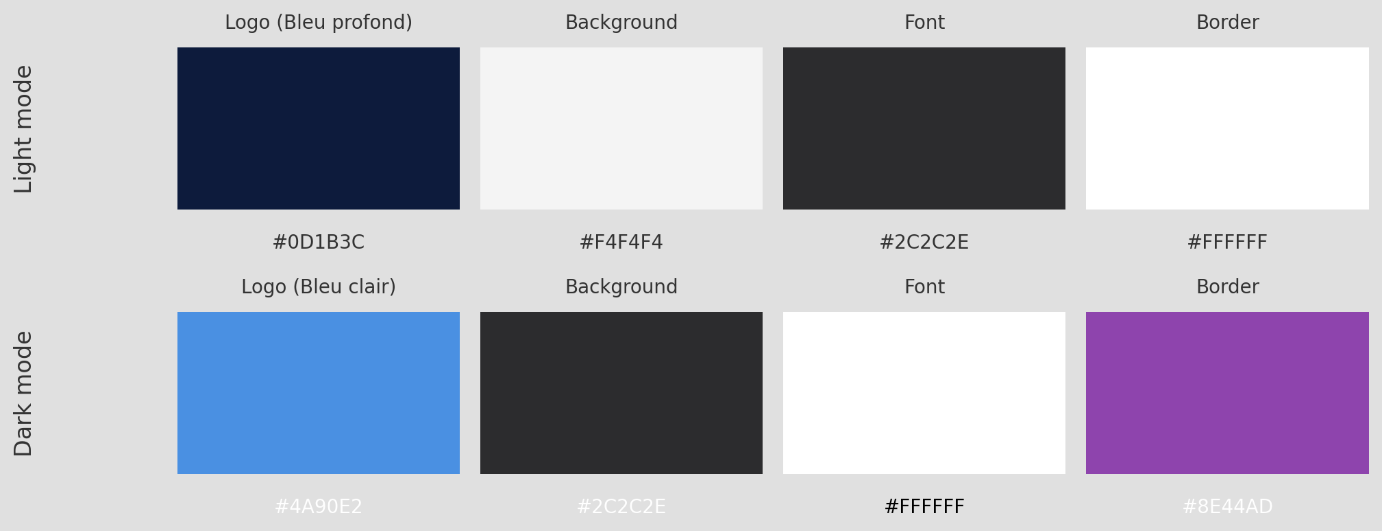
La phase de design a été réalisée à l’aide de **Figma**, outil principal pour l’élaboration des wireframes et des maquettes visuelles de l’application. Après définition de la charte graphique (logo, palette de couleurs, typographies), chaque écran a été prototypé dans Figma pour valider l’ergonomie et la cohérence visuelle avant le développement.

### 1. Logo

Ce logo associe une forme hexagonale inspirée de la blockchain à un dégradé bleu-pourpre-rose évoquant la lumière et la clarté pédagogique, tandis que la dominante de bleu profond installe une sensation de confiance et de sérieux ;

Le logo a été créé dans Figma au format vectoriel (SVG).

### 2. Palette graphique



### 3. La typographie

Pour les titres et intertitres, la police **Poppins** est utilisée en gras (700) pour les niveaux H1 et H2, et en semi-gras (600) pour H3 et H4 ; leur couleur est le bleu profond (#0D1B3C) sur fond clair ou le blanc (#FFFFFF) sur fond sombre.

Pour le corps de texte, on adopte **Roboto** en regular (400) pour les paragraphes et en medium (500) pour les labels et mentions importantes ; la couleur est le gris anthracite (#2C2C2E) sur fond clair, ou le blanc (#FFFFFF) sur fond bleu profond.

### 4. Maquette & WireFrames

Des wireframes ont été réalisées afin de valider la structure et l’enchaînement des écrans, d’optimiser le parcours utilisateur et de clarifier l’emplacement des éléments.

Une image contenant texte, capture d’écran, conception

Le contenu généré par l’IA peut être incorrect.

Une image contenant texte, capture d’écran, logiciel, Icône d’ordinateur

Le contenu généré par l’IA peut être incorrect.

# 4. Conception et modélisation des données

Pour structurer la base de données de BlockLumen, la méthode Merise a été utilisée, fondée sur la distinction entre modèles conceptuel, logique et physique. Après une phase d’étude préalable des besoins (simulation d’ordres, suivi de portefeuilles, actualisation des cours), un MCD détaillé a été élaboré, puis traduit en MLD et enfin en schéma physique optimisé. Cette démarche assure la cohérence, la performance et l’évolutivité de la solution.

La structuration des données de BlockLumen est déclinée en quatre étapes :

1. Clarification des besoins & dictionnaire de données,
2. Modèle conceptuel de données (MCD),
3. Modèle logique de données (MLD),
4. Modèle physique SQL.

Pour les parties Merise, Base de données et développement, une convention en anglais a été adopté.

## a. Clarification des besoins

La phase de clarification a pour objectif de recenser et formaliser l’ensemble des besoins fonctionnels et des flux d’information de BlockLumen, puis de définir un dictionnaire précisant chaque donnée manipulée par le système. Cette étape garantit une base solide pour la modélisation conceptuelle, en assurant cohérence terminologique et exhaustivité des entités.

L’ensemble des exigences s’articule autour de ~~cinq~~ processus clés :

1. **Gestion des comptes :** inscription, authentification sécurisée (JWT + bcrypt), récupération de mot de passe, et personnalisation des paramètres.
2. **Visualisation des cours :** affichage en temps réel (API CoinGecko ou dataset) de plusieurs cryptomonnaies avec sélection de périodes (1 h, 1 j, 1 sem., 1 mois).
3. **Simulation de trading :** crédit initial de 10 000 $ fictifs, module d’achat/vente avec calcul instantané des gains/pertes et mise à jour du solde.
4. **Modules pédagogiques :** séquence de leçons ordonnées pour découvrir les concepts du trading crypto.
5. **Portefeuille :** Gestion du portefeuille de cryptos.
6. **~~Veille d’actualités :~~** ~~flux d’articles externes (RSS ou API) classés par date et résumé.~~

La plateforme doit rester claire et responsive (desktop, tablette, mobile) et regrouper les sections **Accueil**, **Marchés**, **Trader**, **Apprendre**, **Comptes**. Ces processus et règles de gestion (API, sécurité, responsive) constituent le socle de la future modélisation Merise.

## b. Dictionnaire de données

Afin de créer le dictionnaire de données, on relève chaque information à stocker. Une image contenant texte, capture d’écran

Le contenu généré par l’IA peut être incorrect.

## c. Modèle Conceptuel Des Données ( MCD )

Une image contenant capture d’écran, diagramme, conception

Le contenu généré par l’IA peut être incorrect.Après avoir établi le dictionnaire de données et en fonction des informations recueillies, j’ai identifié les entités, leurs attributs et les relations entre celles-ci. Cette étape m’a permis de créer une représentation des données et ainsi définir les relations fonctionnelles, pour cela on utilise la notion de cardinalité au sein du mcd, cela représente le nombre d’occurrences ou d’entités qui peuvent être associées à une autre entité au travers d’une relation. Elle permet ainsi de délimiter et de définir les contraintes liées aux associations entre les entités.

## 

## d. Modèle Logique Des Données ( MLD )

En déterminant la structure pour stocker les données, j’ai élaboré un MLD.

**User** ( #user\_id, username, email, password\_hash, created\_at, last\_login )

**Wallet** ( #wallet\_id, created\_at, initial\_balance, #user\_id )

**Wallet\_Holding** ( #holding\_id, crypto\_symbol, quantity, average\_price, #wallet\_id )

**Trade** ( #trade\_id, crypto\_symbol, type, amount, price, fee, timestamp, #holding\_id )

**Preference** ( #preference\_id, key, value, #user\_id )

**Price** ( #price\_id, crypto\_symbol, value, recorded\_at )

**Learn** ( #learn\_id, title, content, order\_index )

**User\_Learn** ( #user\_id, #learn\_id, is\_completed, completed\_at )

## e. Modèle Physique Des Données ( MPD )

Le MPD est la traduction en SQL du MLD : on y définit les tables, types, clés primaires, clés étrangères et contraintes d’intégrité pour un SGBD (ici MySQL).

Une image contenant texte, capture d’écran

Le contenu généré par l’IA peut être incorrect.

Une image contenant texte, capture d’écran

Le contenu généré par l’IA peut être incorrect.

## f. Relations

**User → Wallet : One-to-One**

* Un utilisateur peut posséder plusieurs portefeuilles, et chaque portefeuille appartient à un seul utilisateur.
* **Implémentation :** dans la table **Wallet**, user\_id est NOT NULL et référencé par une FK → User(user\_id).

**Wallet → Wallet\_Holding : One-to-Many**

* Un portefeuille peut enregistrer plusieurs lignes de détention, chaque ligne de détention appartenant à un unique portefeuille.
* **Implémentation :** dans la table **Wallet\_Holding**, wallet\_id est NOT NULL et référencé par une FK → Wallet(wallet\_id).

**Wallet\_Holding → Trade : One-to-Many**

* Une ligne de détention (pour une cryptomonnaie précise) peut enregistrer plusieurs transactions, chaque transaction appartenant à une seule détention.
* Implémentation : dans la table **Trade**, holding\_id est NOT NULL et référencé par une FK → Wallet\_Holding(holding\_id).

**User → Preference : One-to-Many**

* Un utilisateur peut avoir plusieurs préférences, chaque préférence rattachée à un seul utilisateur.
* **Implémentation :** dans la table Preference, user\_id est NOT NULL et référencé par une FK → User(user\_id).

**User ↔ Learn (Module Pédagogique) : Many-to-Many**

* Un utilisateur peut suivre plusieurs modules, et un module peut être suivi par plusieurs utilisateurs.
* **Implémentation :** table User\_Learn avec (user\_id, learn\_id) en PK composite, chacune référencée par une FK → User(user\_id) et Learn(learn\_id). Les champs is\_completed et completed\_at enregistrent la progression.

**Price : Entité autonome**

* Stocke l’historique des cours sans relation directe en V1.
* **Implémentation :** pas de FK ; un index (crypto\_symbol, recorded\_at) optimise les recherches temporelles.

**Résumé des cardinalités (notation Merise) :**

* **Utilisateur (0..1) – (1..1) Wallet**
* **Wallet (1..1) – (0..n) Wallet\_Holding**
* **Wallet\_Holding (1..1) – (0..n) Trade**
* **Utilisateur (1..1) – (0..n) Preference**
* **Utilisateur (0..n) – (0..n) Module\_Pedagogique** (via User\_Learn)
* **Price :** entité autonome sans relation directe en V1.

Cette section met en évidence, pour chaque association, le sens et le fonctionnement concret des relations dans le schéma physique de BlockLumen.

## d. Diagrammes UML

### 1.Diagramme de cas d’utilisation

Dans ces diagrammes de cas d’utilisation, l’utilisateur est l’acteur principal et interagit avec le système BlockLumen. Les cas d’utilisation représentant les principales fonctionnalités et interactions que les utilisateurs peuvent avoir avec l’application sont les suivants :

**Créer un compte** :  
Permet à l’utilisateur de créer un compte sur la plateforme BlockLumen en fournissant un nom d’utilisateur, une adresse e-mail et un mot de passe.

**Se connecter** :  
Permet à l’utilisateur de s’authentifier sur BlockLumen pour accéder aux fonctionnalités protégées.

**Gérer profil et préférences** :  
Permet à l’utilisateur authentifié de consulter et modifier ses informations de profil ainsi que ses préférences (thème, notifications, etc.).

**Créer un portefeuille** :  
Permet à l’utilisateur de générer un nouveau portefeuille virtuel avec un solde initial (10 000 $ par défaut).

**Lister mes portefeuilles** :  
Permet à l’utilisateur de visualiser la liste de tous ses portefeuilles existants.

**Voir détail d’un portefeuille** :  
Permet à l’utilisateur de sélectionner un portefeuille pour afficher ses retenues (chaque crypto détenue, quantité, prix moyen) ainsi que la valeur calculée.

➢ Voir mes holdings : Affiche toutes les lignes de détention (`Wallet\_Holding`) associées à un portefeuille donné.

➢ Consulter cours actuel : Récupère et affiche les derniers prix des cryptomonnaies concernées à partir de la table `Price`.

**Effectuer un trade (achat/vente)** :  
Permet à l’utilisateur d’acheter ou de vendre une cryptomonnaie pour un portefeuille donné. Lorsqu’un trade est validé :  
 – Si **achat**, créer ou mettre à jour la ligne correspondante dans Wallet\_Holding (quantité + average\_price).  
 – Si **vente**, diminuer la quantité dans Wallet\_Holding, et supprimer la ligne si la quantité devient nulle.

➢ Mettre à jour/inserer un holding : Après le trade, mettre à jour la `quantity` et le `average\_price` dans `Wallet\_Holding`.

➢ Vérifier solde disponible : S’assurer que le portefeuille dispose d’assez de fonds fictifs pour réaliser l’opération.

➢ Calculer frais : Déterminer et appliquer les frais simulés à la transaction.

**Consulter historique des transactions** :  
Permet à l’utilisateur de voir la liste de tous ses trades (filtrable par portefeuille ou par cryptomonnaie), avec date, type (achat/vente), quantité, prix et frais.

**Suivre les modules pédagogiques (Learn)** :  
Permet à l’utilisateur de naviguer dans les modules d’apprentissage disponibles. Pour chaque module :  
– **Voir le détail du module** (titre, contenu)  
– **Marquer comme complété** (<<include>>) : créer ou mettre à jour la ligne dans User\_Learn (is\_completed, completed\_at).

**Se déconnecter** :  
Permet à l’utilisateur de terminer la session en invalidant son token JWT.

### 

### 2. Diagramme de Flux

Les diagrammes de Flux sonts … outil essentiel pour la modélisation … de cette application.

Ce diagramme représente les étapes de connexion et création de compte.

Ce deuxième diagramme illustre les étapes de la gestion du trade ou portefeuille

### 3. Diagramme d’entités

A partir de la modélisation, j’ai developpé un diagramme d’entités dans le cadre de la conception UML avec cardinalité et relations d’agrégation entre classes.

# 5. Environnement de développement

# 6. Développement de l’application

# 7.

# 15. Conclusion

## a. Remerciements

## b. Axes d’amélioration

## c. Retour d’expérience