Journal de bord – Projet Data Visualisation : Gestion de portefeuille

Étudiant : Bernier Aude
Master : Systèmes d'information économiques et financiers
Logiciels utilisés : Python, Streamlit, Git / GitHub, VS Code, Selenium WebDriver, Firefox + GeckoDriver
Période : Novembre 2025

Semaine 1 — Mise en place de l'architecture et fondations techniques du projet

Objectifs et Tâches réalisées

Définition de l'architecture applicative :

- Création de la structure modulaire du projet : `stock/`, `calculs_financiers/`, `pages/`, `utils/`.
- Séparation explicite entre logique métier, accès aux données et interface.

Mise en place du pipeline de données financières :

- Intégration initiale de yfinance.
- Développement de la fonction de récupération avec mécanisme de cache (`fetch_data_with_cache`).
- Début du resolver de tickers pour améliorer l'expérience utilisateur.

Calculs fondamentaux rendement–risque :

- Création des fonctions de calcul de rendements, volatilité, covariance.
- Construction des matrices nécessaires à la théorie moyenne–variance.

Première implémentation du module Markowitz :

- Développement des fonctions GMV, portefeuille tangent et frontière efficiente.
- Tests unitaires simples pour valider le comportement des solveurs.
- Correction des premières erreurs liées aux contraintes et aux bounds.

Ébauche de l'interface utilisateur Streamlit :

- Champs de saisie des tickers.
- Premiers composants interactifs (boutons, affichage conditionnel).
- Ajout des premiers graphiques : cours, corrélation.

Intégration des données macro-économiques :

- Interrogation Eurostat pour l'inflation HICP YoY (début).
- Début du module taux sans risque via ECB/FRED.

Initialisation Git et synchronisation GitHub :

- Création du dépôt Git local.
- Configuration du remote GitHub.
- Premier commit structurant : architecture du projet et modules initiaux.

Définition de la feuille de route fonctionnelle :

- Identification des modules à développer : optimisation, analyse technique, visualisations avancées.
- Définition des contraintes et hypothèses de l'application (nombre minimal d'actifs, cohérence des données, gestion du risque).

Problématiques rencontrées

- Difficulté à définir une architecture modulaire cohérente entre les modules financiers, les APIs et l'interface Streamlit.
- Incohérences de formats entre tickers, colonnes manquantes, données incomplètes ou séries de longueurs différentes nécessitant un nettoyage systématique.
- Retours irréguliers, soucis de disponibilité, besoin de mettre en place un mécanisme de cache pour limiter les appels.
- Erreurs liées aux contraintes et à la configuration des solveurs (bounds, poids négatifs, somme ≠ 1), covariance non inversible dans certains cas tests.
- Gestion de l'état (`st.session_state`) encore instable, affichages conditionnels mal synchronisés avec les données chargées.
- Structure complexe des réponses Eurostat, nécessité de naviguer dans les dimensions JSON et d'harmoniser les séries temporelles.

Semaine 2 — Structuration fonctionnelle et consolidation des calculs financiers

Objectifs et Tâches réalisées

Incohérences dans les optimisations Markowitz :

- Correction des problèmes liés aux contraintes de pondération (w_min, w_max).
- Unification des signatures des fonctions GMV, Tangency et Efficient Frontier.
- Gestion explicite des erreurs d'optimisation et des cas dégénérés (portefeuilles impossibles).

Amélioration du pipeline rendement–risque :

- Ajout des fonctions de diagnostics de données (détection NaN, volumes insuffisants, index non alignés).
- Normalisation rigoureuse des données en entrée pour éviter les erreurs silencieuses dans les calculs.

Renforcement de l'interface Streamlit :

- Ajout du système de sélection dynamique des tickers avec suppression en un clic.
- Mise en place d'un affichage conditionnel exigeant au moins deux actifs avant calcul.
- Amélioration de la gestion d'état (st.session_state) pour empêcher les recalculs inutiles.

Début de la gestion avancée des visualisations :

- Stabilisation des courbes de cours.
- Consolidation de la heatmap de corrélation.
- Premières intégrations de la frontière efficiente et du portefeuille tangent en visuel.

Développement du module d'indicateurs techniques :

- Implémentation du RSI, MACD et moyennes mobiles.
- Préparation du pipeline pour l'intégration dans la page dédiée (séparation propre logique/UX).

Travail sur l'intégration macro-économique :

- Création du label inflation via HICP des pays européens.
- Vérification de la cohérence des clés Eurostat.
- Premiers tests d'intégration dans l'interface (affichage rendement vs inflation).

Amélioration du module taux sans risque :

- Récupération automatisée avec fallback FRED → ECB.
- Harmonisation du passage en décimal (0.03 et non 3).
- Sécurisation des cas où les données ne sont pas disponibles.

Consolidation Git/GitHub :

- Nettoyage du dépôt : .gitignore, structure claire des modules.
- Documentation minimale du README (objectifs, installation, premiers tests).
- Premier push complet du projet opérationnel.

Clarification des objectifs fonctionnels :

- Définition des exigences UI : presets de contraintes, sliders dynamiques, messages d'erreur explicites.
- Priorisation des prochaines fonctionnalités : CML, CAPM, ratios de performance, optimisation avancée.

Problématiques rencontrées :

- Difficultés liées aux bornes de pondération (w_min, w_max) générant des solutions non valides.
- Détection de NaN persistants dans certaines séries, imposant un nettoyage plus strict.
- Problèmes liés à la persistance de certains composants, ralentissant l'interface.
- Difficulté à synchroniser les messages conditionnels et les composants interactifs.
- Courbes instables ou non mises à jour correctement après modifications des tickers.
- Premières intégrations de la frontière efficiente nécessitant un contrôle renforcé des données d'entrée.
- Retours irréguliers sur les séries inflation nécessitant une meilleure normalisation.
- Besoin de tester différents pays pour fiabiliser le pipeline.
- Valeurs incohérentes selon la source nécessitant une conversion systématique en décimal.
- Risque d'absence temporaire de données FRED ou ECB obligeant à sécuriser le fallback.
- Difficulté à harmoniser `rf` dans tous les modules de calcul.

Semaine 3 — Stabilisation des modules d'optimisation et intégration des composants avancés

Objectifs et Tâches réalisées

Stabilisation avancée des solveurs d'optimisation :

- Correction des erreurs liées au retour d'objet `Perf` et gestion explicite des poids.
- Normalisation des fonctions internes `bounds`, contraintes d'égalité et reshape des vecteurs.
- Validation du comportement sur portefeuilles simples et cas limites.

Implémentation de la Capital Market Line (CML) :

- Calcul complet du portefeuille tangent intégré au taux sans risque dynamique.
- Génération de la droite CML dans les visualisations.
- Gestion des échecs d'optimisation (covariance singulière, Sharpe non calculable).

Intégration avancée du taux sans risque :

- Harmonisation de la propagation du taux dans tous les modules.
- Correction des conversions d'unités (pour éviter les confusions % → décimal).
- Gestion des indisponibilités API.

Amélioration UX Streamlit :

- Mise en place des presets de contraintes de pondération (10 %, 20 %, 35 %, 50 %, 100 %).
- Transformation en slider intelligent pour les poids maximums personnalisés.
- Refonte de l'alignement et de l'organisation des boutons et colonnes.

Renforcement des modules de performance :

- Finalisation des fonctions Sharpe, Sortino et Treynor dans `resume_portefeuille`.
- Vérification de la cohérence avec les résultats de Markowitz.
- Sécurisation des cas où la volatilité est nulle.

Débogage visuel et logique Streamlit :

- Résolution d'erreurs DeltaGenerator causées par des composants mal positionnés.
- Stabilisation des conditions d'arrêt (`st.stop()`).
- Correction des affichages conditionnels tels que « rendement > inflation ».

Travail approfondi sur les APIs macro :

- Débogage de la structure JSON Eurostat.
- Correction du mapping temporel pour les séries HICP YoY.
- Gestion robuste des pays non valides.

Refonte du module Selenium / Proxies :

- Gestion automatique de disponibilité Selenium (local vs cloud).
- Mise en place de la rotation headers/proxies.
- Test de connectivité via `httpbin.org`.

Problématiques rencontrées :

- Solveurs d'optimisation fragiles aux données légèrement corrélées.
- Difficultés d'ajustement de l'interface lors de mises à jour successives.
- Incohérences dans certains retours Eurostat nécessitant un parsing plus robuste.

- Gestion complexe du taux sans risque entre différents modules.

Semaine 4 — Renforcement des modules macro, optimisation de l'interface et fiabilisation du système

Objectifs et Tâches réalisées

Consolidation de l'architecture de l'application :

- Revue complète des modules `calculs_financiers/` pour assurer une cohérence.
- Séparation stricte entre logique métier, visualisations et gestion des erreurs.
- Standardisation des signatures de fonctions pour faciliter leur réutilisation dans Streamlit.

Amélioration du module inflation (Eurostat – HICP YoY) :

- Refonte de la fonction `get_hicp_yoy` avec gestion robuste des pays valides.
- Correction de la lecture du JSON Eurostat (dimensions, index, valeurs).
- Gestion des erreurs réseau et fallback en cas d'absence de données.
- Uniformisation du format de sortie en DataFrame exploitable dans l'interface.

Finalisation des visuels avancés dans Streamlit :

- Stabilisation des graphiques de performance, corrélation et frontière efficiente.
- Ajout d'indicateurs clairs pour guider l'utilisateur (exigence de deux actifs minimum).
- Amélioration de la réactivité des composants avec `st.session_state`.
- Ajustement des boutons (taille, alignement, lisibilité).

Renforcement de la logique de cohérence du portefeuille :

- Détection des incohérences (covariance singulière, actifs du même indice, séries trop courtes).
- Intégration des fonctions `diagnostiquer_donnees` et `verifier_coherence_tangent`.
- Messages utilisateurs explicites pour éviter les erreurs silencieuses.

Refonte du module Proxies / Selenium :

- Adaptation du code pour fonctionnement local vs Streamlit Cloud.
- Gestion dynamique des headers navigateurs via fichiers YAML.
- Mise en place d'un test automatique de proxy via `httpbin.org/ip`.
- Préparation à l'intégration future en environnement dockerisé.

Optimisation de la gestion du risque et des ratios :

- Intégration complète des ratios Sharpe, Sortino et Treynor dans les résumés de portefeuille.
- Harmonisation du calcul annualisé et sécurisation des cas `sigma = 0`.
- Validation croisée avec les résultats issus de Markowitz.

Débogage avancé et gestion d'erreurs Streamlit :

- Résolution d'erreurs DeltaGenerator liées aux colonnes imbriquées.
- Suppression d'éléments persistants non désirés dans l'interface.
- Mise en place de conditions d'arrêt lorsque les données sont insuffisantes.

Amélioration progressive de la documentation technique :

- Clarification du fonctionnement des modules internes.
- Nettoyage de portions de code obsolètes, imports inutilisés.
- Début de rédaction d'une documentation orientée "développeur" pour garantir la maintenabilité.

Problématiques rencontrées :

- Parsing Eurostat plus complexe que prévu.
- Interactions Streamlit parfois instables avec les colonnes imbriquées.
- Gestion multi-modules du taux sans risque nécessitant une coordination globale.
- Approche Selenium demandant une adaptation selon l'environnement.