|  |
| --- |
| Plot that lines |

Table des matières

[1 Analyse préliminaire 3](#_Toc181219899)

[1.1 Introduction 3](#_Toc181219900)

[1.2 Objectifs 3](#_Toc181219901)

[1.3 Gestion de projet 4](#_Toc181219902)

[2 Analyse / Conception 4](#_Toc181219903)

[2.1 Domaine 4](#_Toc181219904)

[2.2 Concept 5](#_Toc181219905)

[2.3 Analyse fonctionnelle 7](#_Toc181219906)

[2.4 Stratégie de test 9](#_Toc181219907)

[3 Réalisation 10](#_Toc181219908)

[3.1 Points de design spécifiques 10](#_Toc181219909)

[3.1.1 Gestion des erreurs lors de l’appel à l’API 10](#_Toc181219910)

[3.1.2 Utilisation de secret pour la clef API 10](#_Toc181219911)

[3.1.3 Traitement des devises non supportés 10](#_Toc181219912)

[3.1.4 Gestion des erreurs 10](#_Toc181219913)

[3.2 Déroulement 11](#_Toc181219914)

[3.2.1 Afficher le graphique 11](#_Toc181219915)

[3.2.2 Effectuer un changement monétaire vers la même monnaie 11](#_Toc181219916)

[3.3 Mise en place de l’environnement de travail 12](#_Toc181219917)

[3.4 Description des tests effectués 13](#_Toc181219918)

[3.5 Erreurs restantes 14](#_Toc181219919)

[4 Conclusions 15](#_Toc181219920)

[5 Annexes 16](#_Toc181219921)

[5.1 Journal de travail 16](#_Toc181219922)

# Analyse préliminaire

## Introduction

Ce projet est réalisé dans le cadre du module I323 afin d’apprendre la programmation fonctionnelle en créant une application console affichant des graphiques.

Dans ce projet l’application console permet d’afficher des graphiques concernant les dépenses militaires de chaque pays dans la devise décidé par l’utilisateur afin de comparer les dépenses effectuées par les différents pays sélectionné.

## Objectifs

* Création d’une interface graphique permettant de filtrer l’affichage des données.
  + Filtre sur les dates
  + Au moins 3 courbes
  + L’axe X correspond à la temporalité

## Gestion de projet

La méthode de projet utilisé est la méthode agile. Le chef de projet est Xavier Carrel.

La méthode Agile est une méthode de gestion de projet centré sur la flexibilité et la collaboration avec les membres de l’équipe et le client.

Elle fonctionne avec des cycles de développement court appelés « sprints ».

Dans le cadre de ce projet un seul sprint est effectué.

# Analyse / Conception

## Domaine

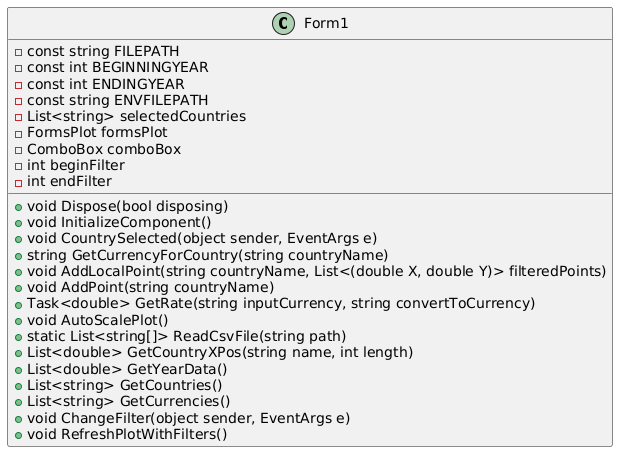
Les données récoltées sont les dépenses militaires de plusieurs pays selon leurs devises locales.

Le domaine des données est militaire et économique.

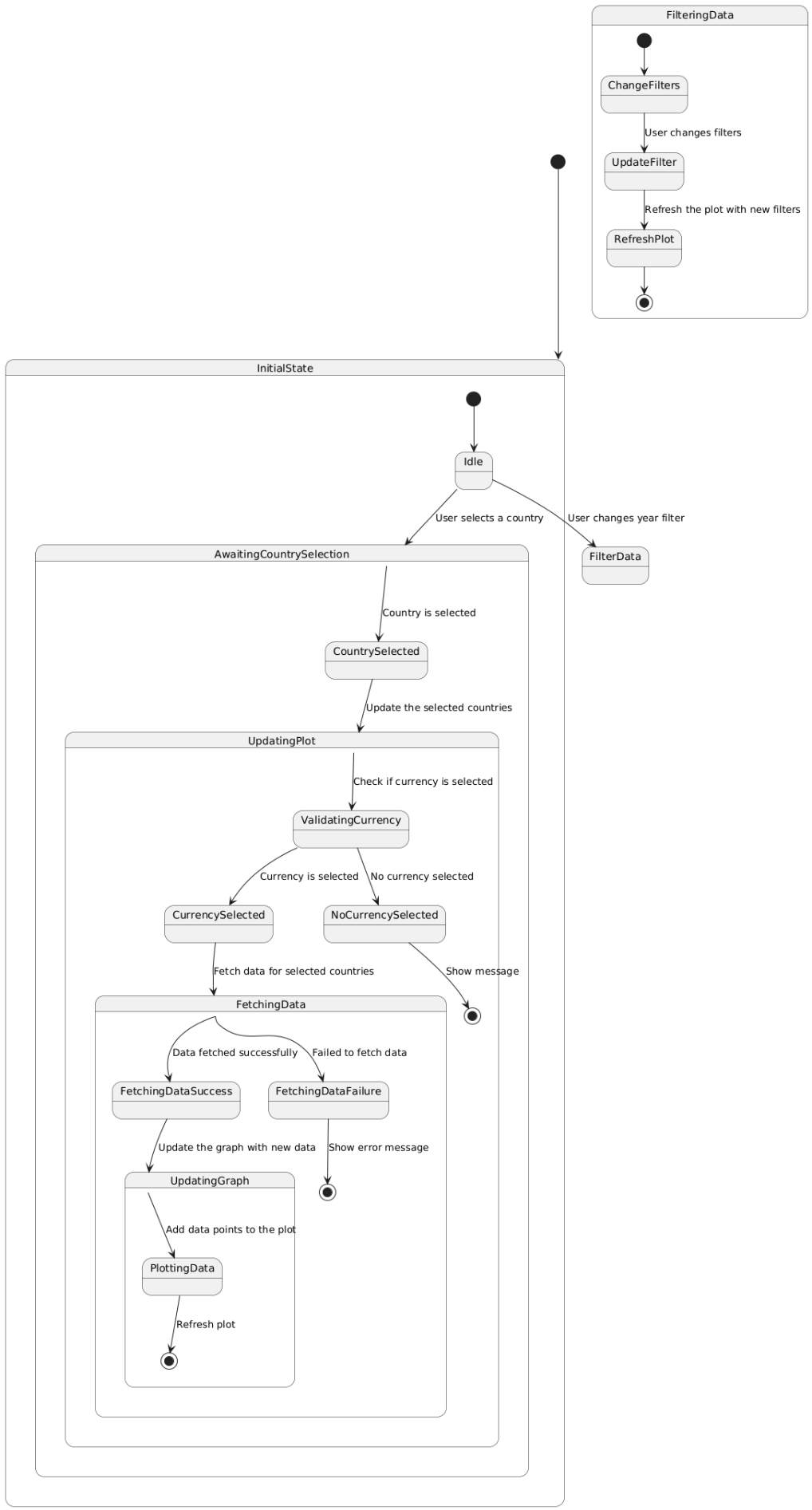
L’échelle de temps utilisé est d’une année en une année.

Le public destiné contient des économistes mais aussi n’importe quelle personne curieuse de connaître les dépenses militaires d’un pays en particulier et de les comparer à d’autres pays.

## Concept



1 - Diagramme de classe



2 - Diagramme d'état

## Analyse fonctionnelle

Afficher le graphique

|  |
| --- |
| En tant que utilisateur Je veux sélectionner le pays Pour afficher le graphique de ce pays |
| Tests d'acceptance:   |  |  | | --- | --- | | Modifier le pays sélectionné | Dans le menu d'accueil de l'application (voir maquette) Quand je clique sur le nom d'un pays Alors le graphique du pays concerné apparaît dans la devise sélectionné | | Lancer l'application | Au démarrage de l'application La page d'accueil apparaît (voir maquette) | | Afficher l'année et l'argent | Dans un graphique Quand je passe ma souris sur un point du graphique Alors les données de ce point sont affichés (année et argent) | | Filtrer la date de début | Dans la page des graphiques Quand je sélectionne une date de départ Alors le graphique s'actualise et commence à cette date | | Filtrer la date de fin | Dans la page des graphiques Quand je sélectionne une date de fin Alors le graphique s'actualise et fini à cette date | | Filtrer la date de début et de fin | Dans la page des graphiques Quand je sélectionne une date de départ et une date de fin Alors le graphique s'actualise et commence à la date de départ et fini à la date de fin | | Exception de filtrage | Dans la page des graphiques Quand je défini une date de départ et/ou une date de fin invalide (chiffre, date illogique, commencer après la fin) Alors le graphique ne se met pas à jour et un message apparaît pour aider l'utilisateur | |

Effectuer un changement monétaire vers la même monnaie

(Auteur: Joachim Berchel)

|  |
| --- |
| En tant qu'utilisateur Je veux afficher les données dans la même devise Afin de pouvoir les comparer |
| Tests d'acceptance:   |  |  | | --- | --- | | Choisir la monnaie de conversion | Dans la fenêtre des graphiques Quand je sélectionne une monnaie dans laquelle afficher mon graphique Alors mon graphique s'actualise avec les données converti, la légende est aussi mise à jour | | Choisir la même monnaie de conversion que celle actuel | Dans la fenêtre des graphiques Quand je sélectionne la monnaie actuellement utilisé pour la convertir Alors rien ne se passe | |

Maquette :

Une image contenant texte, écriture manuscrite, papier, Produit en papier

Description générée automatiquement

3 - Maquette papier de l'interface utilisateur

## Stratégie de test

Des tests unitaires sont effectués avec MSTest, ils vérifient que les méthodes utilisées renvoient des valeurs correctes. Les méthodes faisant appel à une API ne sont pas testées ainsi que celles ne retournant aucune valeur.

**Ordre des tests :**

1. Tests unitaires :

Les tests unitaires vérifieront que les méthodes fonctionnent indépendamment du reste.

1. Tests d’intégration :

Les tests d’intégration sont effectués pour vérifier que l’appel à l’API fonctionne correctement.

1. Tests de validation :

Finalement les tests de validation seront effectués pour vérifier que le programme correspond bien aux exigences qui ont été spécifié dans les User Story.

**Moyen à mettre en œuvre :**

Des données de test pertinentes doivent être préparées.

Par exemple des noms de pays, des dates ou un fichier comportant des données.

Pour mettre en place ces données il faudra modifier les valeurs des constantes dans le code ou mettre en place un fichier .env.

Les constantes actuelles sont :  
FILEPATH qui contient l’emplacement du fichier contenant les données.

BEGINNINGYEAR qui contient la date de départ des données récoltés.

ENDINGYEAR qui contient la date de fin des données récoltés.

Afin de réduire le temps et l’effort nécessaire il serait préférable d’automatiser les tests dans un outil de CI/CD.

**Testeurs :**

Les testeurs éventuels de l’application seraient des utilisateurs finaux ou des camarades de classe, donc une communauté de développeur.

**Sprint Review :**

La sprint review peut se dérouler en présentiel comme en visioconférence.

Elle concernera toutes les personnes impliquées dans ce projet.

# Réalisation

## Points de design spécifiques

### Gestion des erreurs lors de l’appel à l’API

Au moment de faire appel à l’API dans le cas où une erreur avec l’API survient le graphique est quand même affiché avec les données locales.

À noter que pour utiliser l’API il faut se créer un compte sur <https://getgeoapi.com/account/> puis « whitelist » l’adresse IP sur laquelle le programme est lancé.

### Utilisation de secret pour la clef API

Le jeton API est stocké dans un fichier .env. Dans le cas ou le fichier n’est pas trouvé les données locales sont utilisé.

À noter que l’emplacement du fichier ".env" est écrit dans une constante, l’emplacement du fichier est de base dans « bin\Debug\net8.0-windows ».

L’API a une limite de requête quotidienne avec un compte gratuit.

### Traitement des devises non supportés

Affichage d’un message d’erreur si impossibilité d’utiliser l’API. L’application fonctionnera toujours avec les données locales.

### Gestion des erreurs

Dans le cas ou une valeur du fichier csv est vide elle est remplacée par 0 afin d’avoir le même nombre de variable dans la variable stockant ces données que dans celle stockant les années. Les deux variables doivent avoir le même nombre de donnée pour ne pas provoquer d’erreur. Ensuite les valeurs étant égales à 0 sont filtrés pour ne pas être affiché.

## Déroulement

### Afficher le graphique

*En tant qu’utilisateur*

*Je veux sélectionner le pays*

*Pour afficher le graphique de ce pays*

Afin de réaliser cette user story j’ai décidé d’utiliser un Forms et d’utiliser le NuGet « ScottPlot » pour afficher le graphique. Une autre alternative aurait été d’utiliser les fonctions intégrées de base mais cela permettant moins de personnalisation j’ai décidé de m’orienter vers ScottPlot.

Les premières difficultés ont été l’utilisation de Forms, n’ayant pas effectué de projet avec cette technologie j’ai dû apprendre les syntaxes et comment cela fonctionnait.

Ensuite j’ai rencontré des difficultés avec le package ScottPlot car je souhaitais effectuer certaines choses que ScottPlot ne me permettait pas, par exemple l’affichage détaillé de chaque point.

### Effectuer un changement monétaire vers la même monnaie

*En tant qu'utilisateur*

*Je veux afficher les données dans la même devise*

*Afin de pouvoir les comparer*

Afin de réaliser cette user story j’ai décidé d’utiliser une API pour convertir les devises. J’ai premièrement rencontré des difficultés à trouver une API gratuite.

J’ai ensuite rencontré des difficultés à récupérer quelles devises étaient utilisé, mes données locales ne contenaient pas la devise mais le code du pays.

Le dernier problème rencontré venait d’une limite quotidienne de requête à l’API, cela m’a pris du temps avant de m’en rendre compte.

## Mise en place de l’environnement de travail

Le code source est stocké sur GitHub à l’adresse suivante : <https://github.com/joaberch/plot-that-lines.git>

Il peut être cloné avec cette commande :



4 - Commande pour cloner le dépôt

**Liste des fichiers :**

* README.md : Description du projet GitHub
* .gitignore :Décrit les fichiers à ne pas prendre en compte lors des commits.
* .env : Contient la clef de l’API.
  + APIKEY=93nd9, valeur d’exemple
* API\_MS.MIL.XPND.CN\_DS2\_fr\_csv\_v2\_3446916.csv : Donnée de test.

**Version des systèmes/outils :**

* IDE : Visual Studio Community 2022, Version 17.9.4
* Framework : .Net, version 8.0.202
* Bibliothèques externes : « ScottPlot.WinForms par Scott Harden », version 5.0.38

## Description des tests effectués

*Effectuer un changement monétaire vers la même monnaie*

|  |  |  |
| --- | --- | --- |
| *Choisir la monnaie de conversion* | *Dans la fenêtre des graphiques Quand je sélectionne une monnaie dans laquelle afficher mon graphique Alors mon graphique s'actualise avec les données converti, la légende est aussi mise à jour* | *???* |
| *Choisir la même monnaie de conversion que celle actuel* | *Dans la fenêtre des graphiques Quand je sélectionne la monnaie actuellement utilisé pour la convertir Alors rien ne se passe* | *???* |

*Afficher le graphique*

|  |  |  |
| --- | --- | --- |
| *Modifier le pays sélectionné* | *Dans le menu d'accueil de l'application (voir maquette) Quand je clique sur le nom d'un pays Alors le graphique du pays concerné apparaît dans la devise sélectionné* | *???* |
| *Lancer l'application* | *Au démarrage de l'application La page d'accueil apparaît (voir maquette)* | *???* |
| *Afficher l'année et l'argent* | *Dans un graphique Quand je passe ma souris sur un point du graphique Alors les données de ce point sont affichés (année et argent)* | *???* |
| *Filtrer la date de début* | *Dans la page des graphiques Quand je sélectionne une date de départ Alors le graphique s'actualise et commence à cette date* | *???* |
| *Filtrer la date de fin* | *Dans la page des graphiques Quand je sélectionne une date de fin Alors le graphique s'actualise et fini à cette date* | *???* |
| *Filtrer la date de début et de fin* | *Dans la page des graphiques Quand je sélectionne une date de départ et une date de fin Alors le graphique s'actualise et commence à la date de départ et fini à la date de fin* | *???* |
| *Exception de filtrage* | *Dans la page des graphiques Quand je défini une date de départ et/ou une date de fin invalide (chiffre, date illogique, commencer après la fin) Alors le graphique ne se met pas à jour et un message apparaît pour aider l'utilisateur* | *???* |

## Erreurs restantes

**Option de filtre :**

Description : Les options de filtre ne fonctionnent pas, elles avaient été anciennement implémentées dans un ancien commit mais la restructuration de l’application ne l’a pas restauré.

Conséquences : L’utilisateur ne peut pas appliquer d’options de filtre. Toutes les années sont affichés.

Actions envisagées : Récupérer les méthodes anciennement créées et les adapter au code actuel

**Changement de devise :**

Description : Lorsqu’un graphique est affiché et que la devise a modifié le graphique ne se met pas automatiquement à jour.

Conséquences : L’utilisateur ne peut pas automatiquement changer la devise, il devrait désélectionner et resélectionner le pays.

Action envisagée : Réafficher les graphiques lorsque la devise est modifiée.

# Conclusions

**Objectifs atteints :**

* L’application affiche un graphique contenant les données récupérées.
* La sélection des pays à afficher est fonctionnelle
* La conversion des devises en utilisant une API fonctionne.
* Dans le cas d’une erreur avec l’API les données sont toujours affichées selon les données locales

**Objectifs non-atteints :**

* Mise à jour automatique du graphique lors du changement de devise.
* Si une donnée n’est pas affichée converti avec l’API puis que d’autres données sont converties elles sont affichés les deux en même temps, il y a eu néanmoins un message d’erreur afficher à l’utilisateur pour le prévenir que la première donnée a été affiché dans la devise local

**Difficultés particulières :**

* Utilisation de l’API, intégrer l’API à été un problème pour gérer les erreurs de conversion et les données qui ne correspondaient pas à ce que l’API souhaitait.

**Suite possible :**

* Affinage des options de filtre, rajouter de nouvelles options ou améliorer celles existantes.
* Mise à jour automatique du graphique selon la devise.
* Optimisation des requêtes à l’API, sauvegarder les données pour ne pas avoir à réeffectuer la même requête.
* Améliorer l’interface utilisateur pour la rendre plus simple à utiliser.
* Certaines données pourraient être stocké dans le fichier « .env » afin de permettre à l’application d’être plus flexible.
* La mise à jour automatique du graphique ne fonctionne pas toujours si la conversion a beaucoup changé la valeur de base.
* Les tests unitaires peuvent être affiné.

# Utilisation de l’IA

Lors de ce projet il m’est arrivé d’utiliser l’intelligence artificielle « ChatGPT ». J’ai utilisé ChatGPT afin de modifier mes données CSV plus rapidement.

Je l’ai aussi utilisé pour lister toutes les fonctionnalités à tester dans mes tests unitaires.

J’ai aussi utilisé ChatGPT pour me donner des idées de comment refactoriser mon code pour qu’il soit plus proche du paradigme de programmation fonctionnel.

# Annexes

## Journal de travail

Un journal de travail plus détaillé est accessible dans le même dossier que ce rapport