

Table des matières

[1. Analyse préliminaire 2](#_Toc182814530)

[1.1 Introduction 2](#_Toc182814531)

[1.2 Objectifs 2](#_Toc182814532)

[1.3 Gestion de projet 2](#_Toc182814533)

[1.4 Planification initiale 2](#_Toc182814534)

# Analyse préliminaire

## Introduction

Le projet « P\_DevOps\_Test » est réalisé dans le cadre du module 450, qui a comme thème les processus DevOps et en spcilité la mise en place d’un environnement de test automatisé. Le but est de créer un environnement de tests ainsi que des tests unitaires pour une application de To Do List, tout en utilisant les outils et méthodes vue en classe. Le projet utilise GitHub Project pour la gestion des taches et un journal de travail afin de suivre le déroulement du projet. Le chef de projet est Aurélie Curchod notre enseignante.

## Objectifs

Les objectifs du projet « P\_DevOps\_Test » sont les suivants :

**Mettre en place un environnement de Test automatisé** :

1. Mise en place de test unitaires.
2. Mise en place de test End to End.

**Mettre en place une politique et une stratégie de test incluant :**

1. Les types de tests effectués.
2. Les outils utilisés.
3. L’environnement de test utilisé.
4. Les critères de succès des tests.

**Mettre en place un plan de test pour les fonctionnalités suivantes :**

1. Création du profil.
2. Authentification.
3. Gestion du profil.
4. Ajout d’un TODO
5. Gestion des TODO

**Mise en place des rapports de test associés :**

## Gestion de projet

Ce projet est effectué en groupe de deux. Pour cela, nous utilisons l'outil GitHub Project afin de lister les tâches et attribuer les tâches à réaliser. Nous utilisons également GitHub Desktop ainsi que des commits pour que chacun puisse voir une version du projet à jour.

Un journal personnel est aussi utilisé sous forme de fichier CSV. Ce journal permet de suivre l'avancement du projet et de recenser les problèmes rencontrés. Ainsi, si une des deux personnes est absente, il est toujours possible de suivre l'avancement du projet. Le but est qu'aucun événement ne ralentisse l'avancement et le bon déroulement du projet.

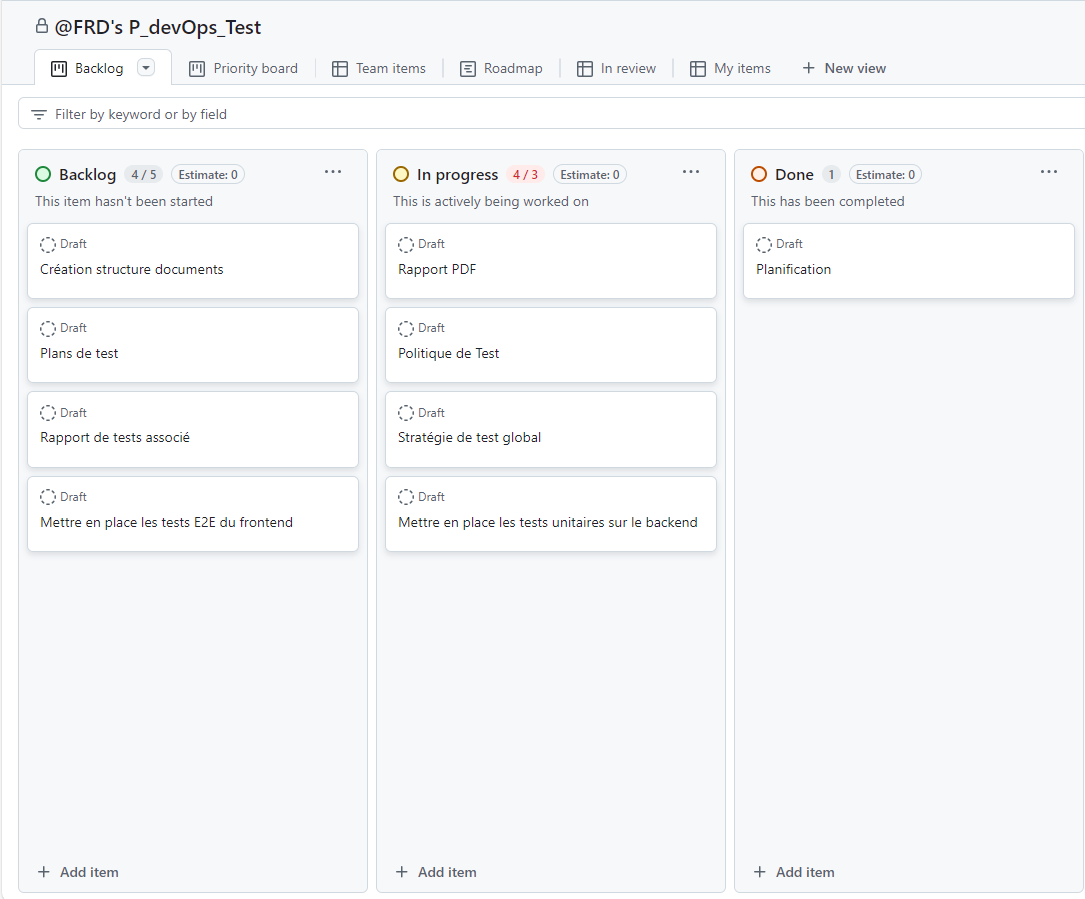
Le nommage des commits est également un point important, car il permet au professeur d'évaluer le projet à 80 % et d'obtenir un retour.

## Planification initiale

Dans ce projet, nous partons avec une toute petite expérience de test dû à la création de test unitaire en C# Cependant c’est bien la première fois que nous allons effectuer ce type de test en Web de plus il est important de noter que mise à part un peu de théorie nous n’avons jamais utilisé des outils tel que «Jest » « supertest » et « cypress »  . Nous nous sommes donc concentrés sur ce qui nous était spécifiquement demandé et avons cherché à comprendre le CDC afin de pouvoir au mieux planifier les tâches principales.

Nous avons réfléchi à ce qui était le plus important à faire au début, comme la mise en place de l’environnement, ainsi qu’aux différentes étapes nécessaires pour avancer efficacement, petit à petit, dans le projet.

Nous n’allons pas directement commencer par la mise en place des tests dans le code, mais d'abord par l’écriture de la politique, la stratégie et le plan de test :



Planification

# Test

## Politique de Test

***Introduction***

Cette politique de test définit les approches, les outils et les méthodologies pour assurer la qualité de l'application Todo, qui comprend un frontend Vue.js, un backend Node.js/Express, et une base de données MongoDB.

***Objectifs***

* Valider le fonctionnement des fonctionnalité prioritaire (création de profil, authentification, gestion de profil, gestion des tâches).
* Identifier les potentiels problèmes des fonctionnalités déjà codé
* Vérifier le bon fonctionnement complet du code via des test unitaire pour le backend.

## Stratégie de test

***Type de tests***

* Tests unitaires pour le BackEnd.
* Test End to End pour le FrontEnd.

***Tests unitaires***

Outils :

* Jest.
* Supertest

Environnements :

* Local
* Intégration

***Test End to End***

Outils :

* Cypress

Environnements :

* Production

***Environnement***

* Environnement de développement local.
* Environnement intégration (CI/CD, GithubAction).
* Environnement production.

***Critères de succès***

* 100% des test unitaire réussis

## Plan de test

***Création de compte***