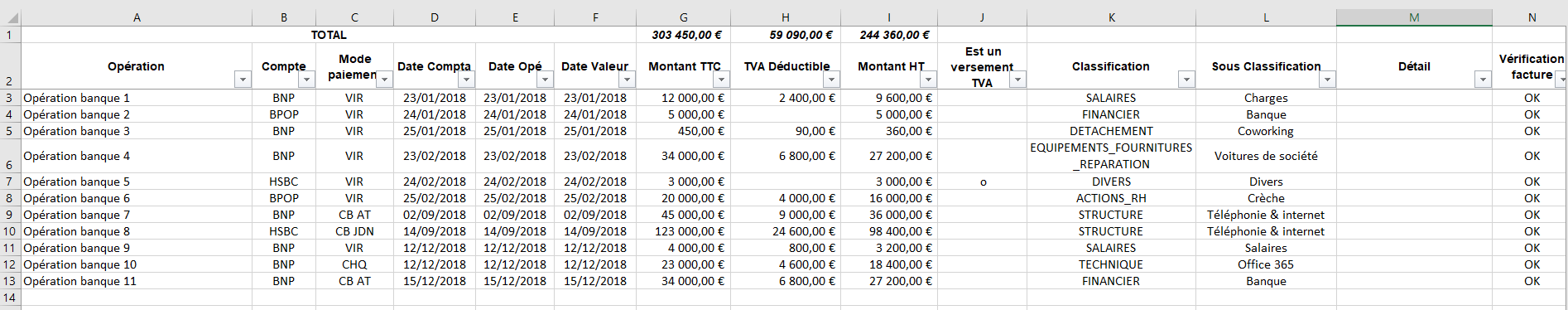
Suivie d’avancement du projet annuel

24-03-2019

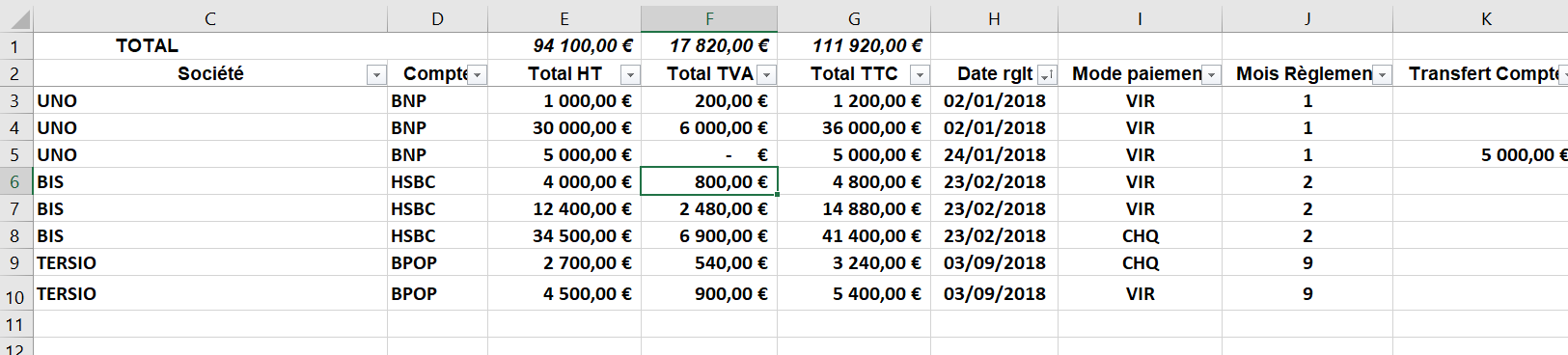
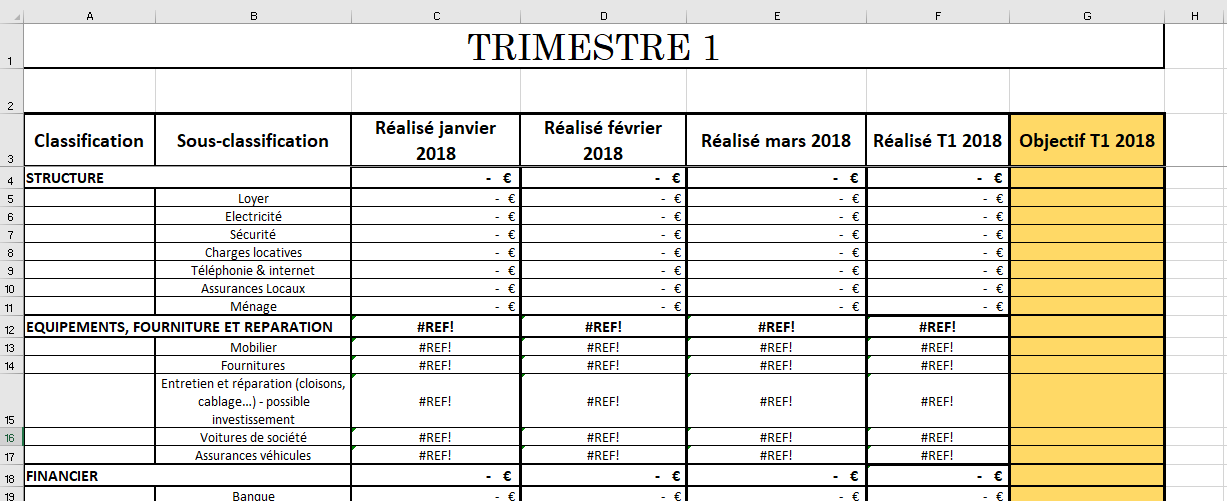
Romain Bignon

Damien Chaudois

Client : Expaceo UK

But du projet : automatiser et réduire les interventions humaines sur l’établissement du suivi de la trésorerie d’Expaceo et d’Expaceo UK, et offrir un outil de reporting et de vue d’ensemble des chiffres clef des finances des entreprises.  
Point de départ de l’application : deux fichiers Excel aujourd’hui utilisés pour la comptabilité de l’entreprise. L’administration tient à jour ces fichiers en allant sur les sites des banques, copie les informations de chaque virement d’argent, et toutes les données les concernant, comme suit: 

Il existe un onglet pour les décaissements, et un pour les encaissements. Chacun renseigne des informations différentes :

Le bilan se trouve dans un onglet dédié et compare les dépense effectué avec les dépense prévues.

Objectifs annoncé par le client :

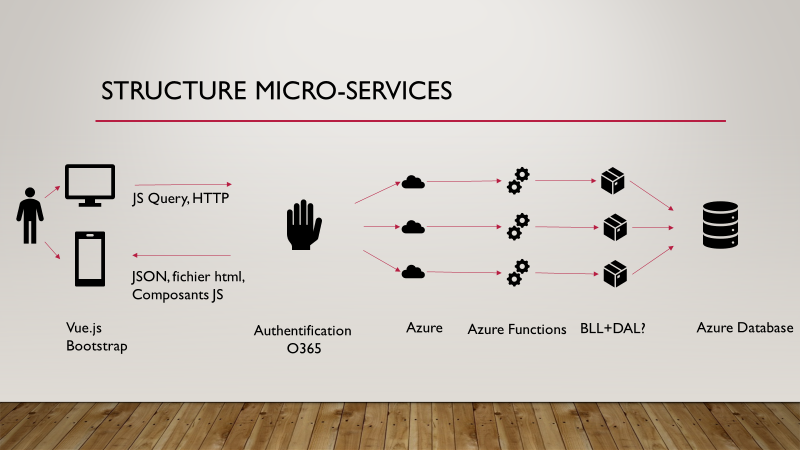
* Permettre à l’application d’aller automatiquement récupérer les mouvements bancaire d’encaissement et de décaissement si la/les banque propose une API pour accéder à ces informations, sinon, automatiser autant que possible le remplissage
* Permettre de mettre en relation les entrées et les sorties d’argent, et de facilement tracer leurs origines
* Offrir la possibilité d’exporter les rapports en format Excel ou PDF que l’entreprise pourra envoyer à des clients, comptable ou banques
* Améliorer la lisibilité de la trésorerie en général, et de son évolution futur et passé.
* Automatiser le calcul de la TVA
* Permettre de lier les ligne de mouvement avec un fichier (facture ou autre )

Des demandes seront rajoutées au fur et à mesure que le projet avance. Celui-ci est évidemment réalisé en méthode agile. Un sprint dure 2 semaine, pour se synchroniser avec le rythme de l’école. Une présentation est donc prévue au moins le vendredi une semaine sur deux pour obtenir les retours du client.  
La gestion du projet se fait avec les outils Microsoft en utilisant des licences offert par l’entreprise pour leur permettre d’avoir accès au suivi de l’avancement du projet sous azure devOps. Le tableau des taches ( le trello ) est privé et il est impossible de donner l’accès à un externe, même en read-only, et il est impossible d’en faire un export.  
La gestion du versioning se fait sous Git, installé sur l’azure devOps. Il n’est pas possible à ma connaissance d’offrir des credential read-only à un utilisateur sans lui donner l’accès complet au devops. Il existe une branche master qui trigger le déploiement du site. La branche staging permet de faire les merge. Il existe deux branches supplémentaire pour chaque développeur.  
Les pipelines de déploiement sont également sous azure devops. Un pipeline se déclenche lors d’un push sur la branche master du git et ce dernier possède deux job pour le moment, auquel nous en rajouterons plusieurs par la suite, notamment concernant l’automatisation des test :

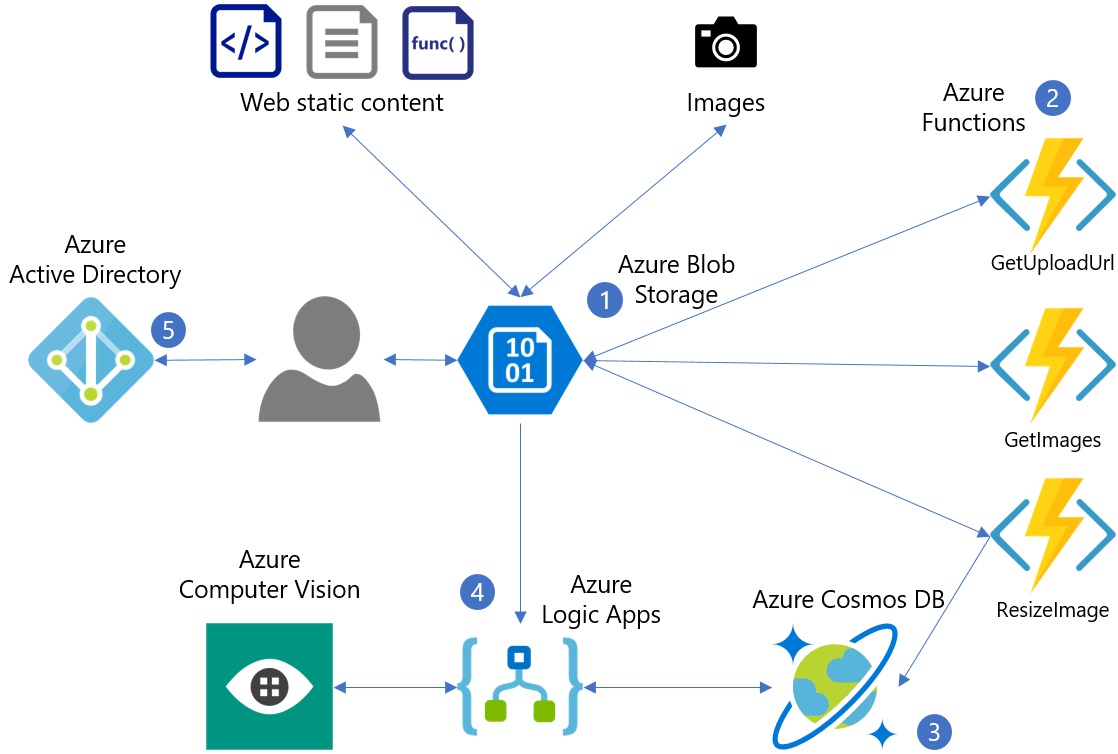
* Copie des fichiers source dans le compte de stockage ( voir la partie architecture )
* Build, packaging et déploiement des fonction azure

L’architecture

Le client a été très clair sur l’aspect critique des données circulant sur le site : la sécurité est donc la priorité. En partant du constat que Microsoft au travers d’azure aura toujours une meilleure sécurité que tout ce que nous pourrions proposer, il a été décider d’implémenter une architecture dite « serverless ».

L’architecture suivante a été proposée au client (Voir fichier joint Projet annuel architecture.pptx) : 

Nous nous sommes basés sur ce descriptif provenant directement de la doc Microsoft pour établir notre architecture :

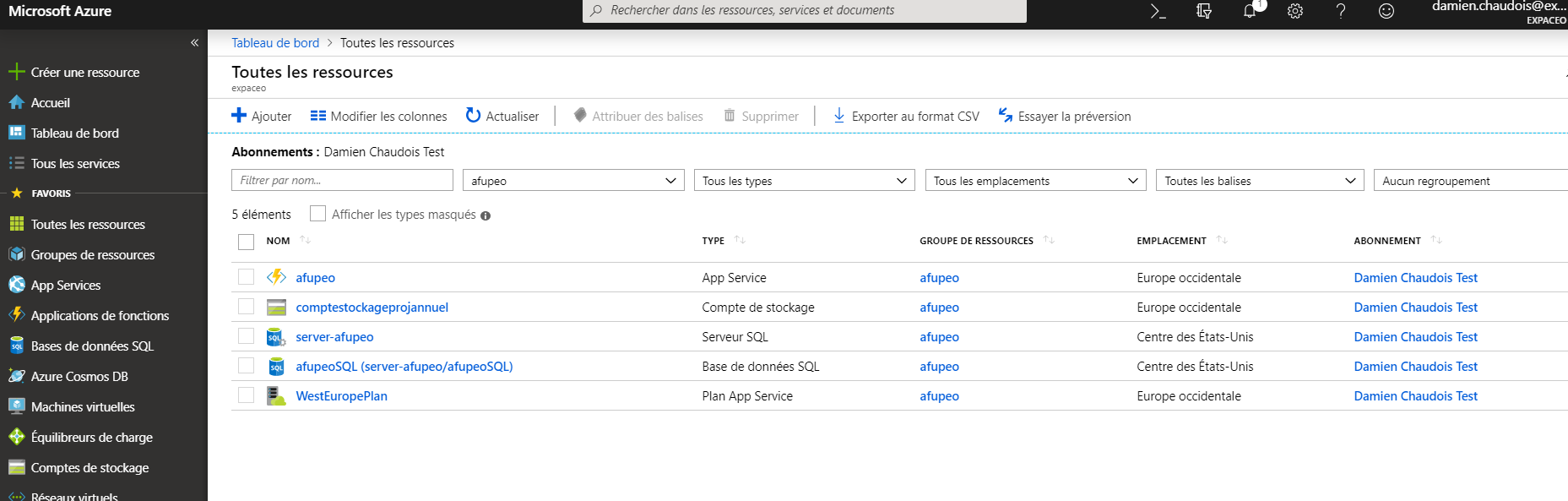
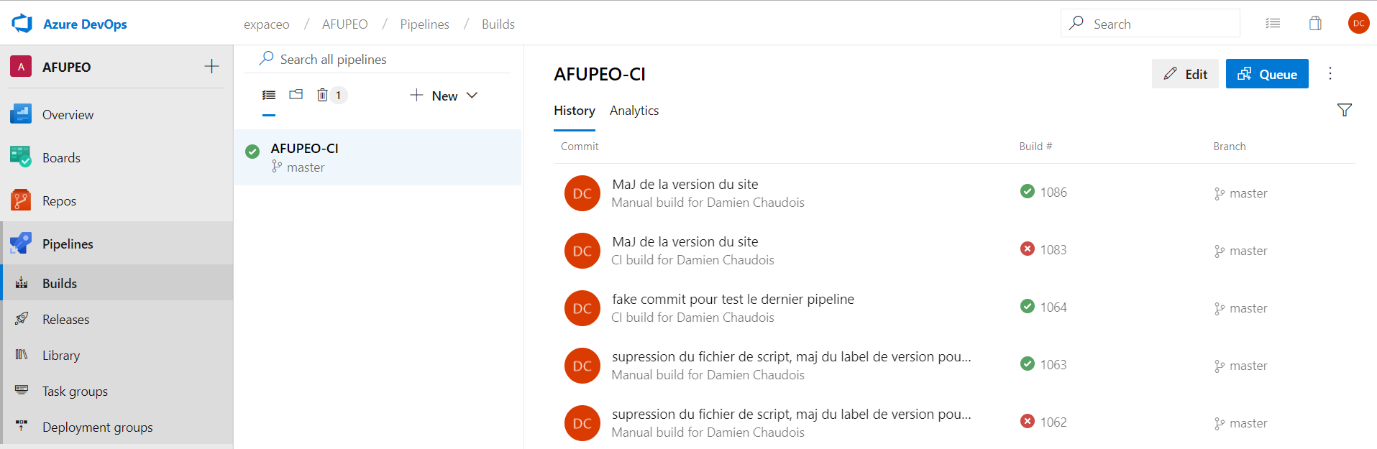
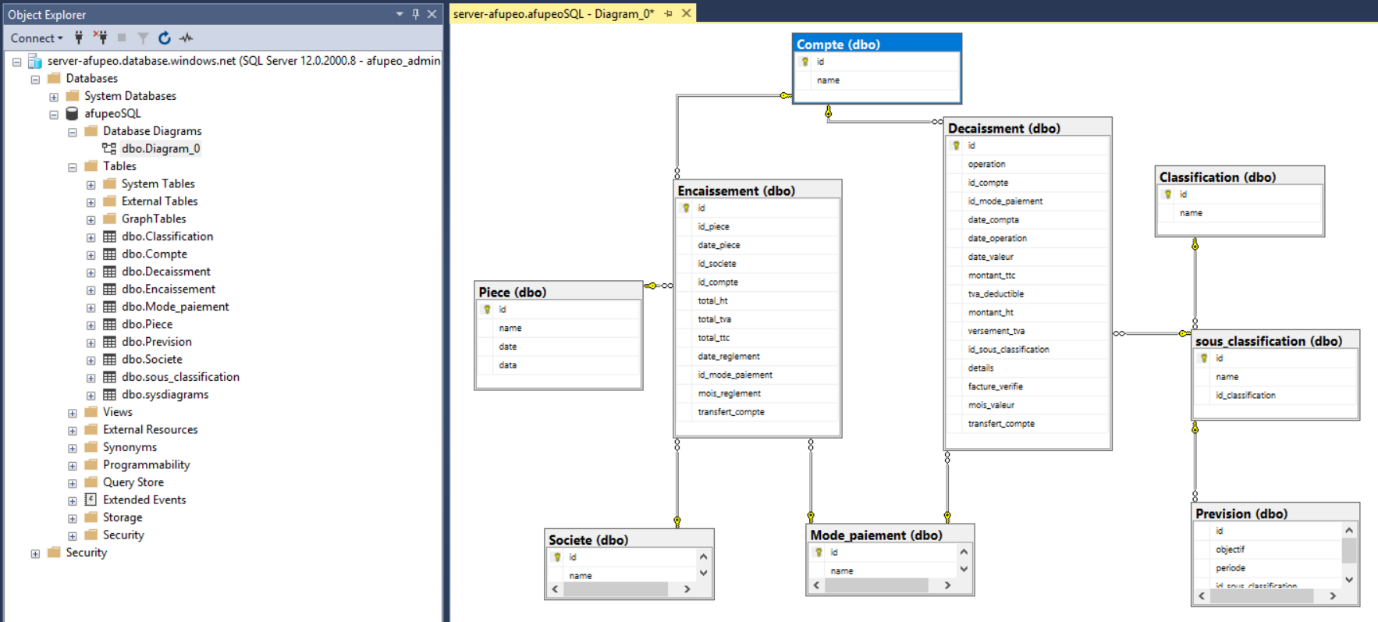
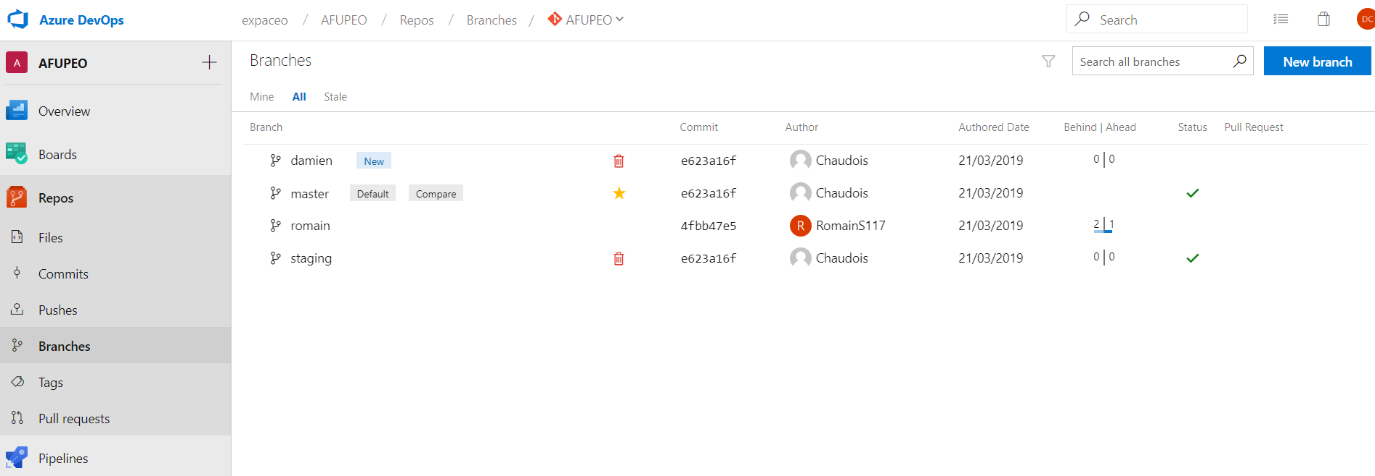
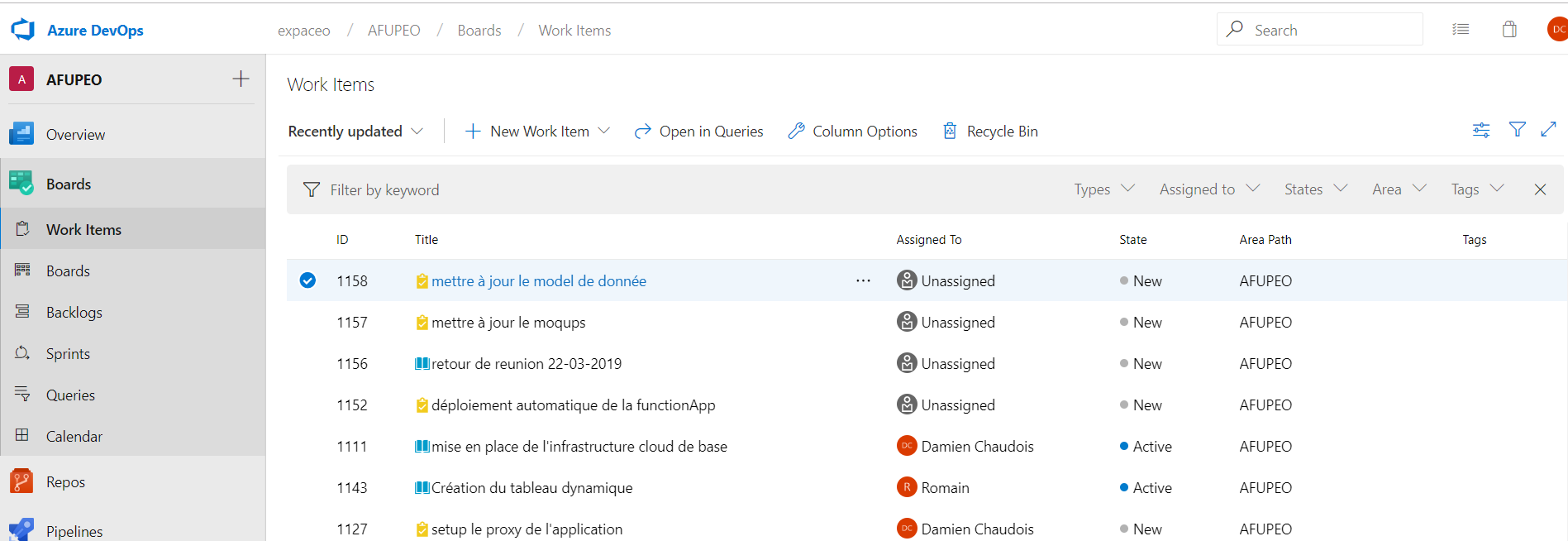


Nous n’utiliserons pas ici les Azure Logic apps ni Azure Computer Vision.

Les technologies choisies sont décrites dans le fichier joint Projet annuel architecture.pptx

Avancement :

A ce jour , nous possédons :

* Une maquette applicative soumise au client le 22-03 ( les retours n’ont pas encore été appliqué) : <https://app.moqups.com/4qwL8CV7d1/view/page/ab15fb6f3>
* L’architecture de base comme décrite dans le schéma ci-dessus : 
* Un pipeline de déploiement automatique fonctionnel 
* Un modèle de base de données et une base SQL server sur azure qui l’implémente :
* Un compte de stockage azure en ligne qui contient notre ébauche d’IHM récupéré sur un site de partage de Template HTML de base : <https://comptestockageprojannuel.z6.web.core.windows.net/>
* Un Git proprement mis en place avec une police de pull request et des branches dédié :
* Un Trello comportant tout les work items que nous avons déduit de nos réunions avec le client 
* Un début de function azure app qui renvoie un JSON simple : <https://afupeo.azurewebsites.net/api/Function1>