# Contexte

Suite à une étude de santé publique sur les étudiants en cursus informatique, le ministère de la santé nous a demandé de développer une solution permettant la réalisation d'un parcours virtuel via la pratique d’une activité sportive en réel. Elle doit permettre de créer des parcours virtuels via un site web et de se déplacer via une application mobile.

Composition de l’équipe :

* Alabic Nemanja
* Derousseaux-Lebert Nathanael
* Metzger Nicolas
* Sacher Marilyn
* Steiner Kélio

# Organisation du projet et technologies

Pour la mise en place de notre application, nous avons réalisé 3 parties distinctes :

* Une API qui gérera l'accès et les requêtes à la base de données
* Un site web qui permettra la gestion des challenges par l’administrateur
* Une application mobile qui sera utilisé par le joueur pour se déplacer sur un parcours

Le projet est divisé comme suit :

* Un répertoire backend dans lequel se trouve un répertoire server où se trouve l’ensemble des répertoires, librairies et fichiers nécessaires
* Un répertoire frontend avec deux sous répertoires :
  + Web avec l’ensemble des répertoires, librairies et fichiers nécessaires au site web
  + Mobile avec l’ensemble des répertoires, librairies et fichiers nécessaires à l’application mobile

En ce qui concerne les technologies employées :

* L’API a été développé en python avec le framework de développement web, Pyramid
* Le site web et l’application sont en javascript :
  + Le site web, il est développé avec la librairie reactJS et pour la partie graphique, Material ui
  + L’application mobile est développée avec le framework React Native

# Intégration de la stratégie de test dans le projet

Le développement du projet Acrobatt a suivi les principes de la méthode Scrum, méthode de développement agile la plus utilisée et documentée aujourd’hui.

Au vu de l’ajout et des modifications constantes des fonctionnalités au fil des itérations, l’intégration d’une stratégie de test dans un développement agile est cruciale. Elle doit être mise en œuvre lors de la première itération afin de mettre en place les outils, l’approche d’automatisation des tests, les ressources et les environnements de test dont l’équipe aura besoin.

La stratégie de test est un élément clé d’un développement agile, elle permet :

* L’automatisation des tests grâce aux outils de test
* De garantir la non régression des fonctionnalités déjà développées
* De soulager la charge de test des testeurs
* De maîtriser les coûts des tests

# Présentation de l’outil de test Postman

POSTMAN est un client pour API web permettant d’interroger des webservices et des API. Il propose de nombreuses fonctionnalités telles que la gestion de tests unitaires, la gestion de série de test, du monitoring, la production de documentation, et même un outil de travail collaboratif. Son environnement graphique est réputé complet, intuitif et propose une prise en main rapide.

Nous avons choisi d’utiliser cet outil dans le cadre de notre stratégie de test pour réaliser les tests sur l’API web car il comporte de nombreuses fonctionnalités que nous estimons pertinentes dans le cadre de notre stratégie de test, dont notamment :

* La construction et l'exécution de requêtes HTTP et de série de requêtes.
* La création de collections de test permettant de stocker et exécuter toutes les requêtes HTTP, elles permettent de regrouper les requêtes de façon fonctionnelle.
* La maintenance d’un historique de requêtes afin de pouvoir les rejouer.
* L’exécution de scripts javascript avant chaque requête.
* La génération d’une documentation de l’API sur la base des tests réalisés.
* Le partage des requêtes, collections et résultats en ligne entre les membres de l’équipe.
* Une interface simple et facile à prendre en main.

# Fonctionnalités à tester

Au cœur de l’élaboration de notre stratégie de test, nous avons dû définir l’ensemble des tests à réaliser, et aussi quantifier l’effort de test à fournir. Parmi ces tests se trouvent les tests fonctionnels sur les fonctionnalités métiers essentielles de l’application, les tests d’Interface Homme-Machine, les tests de performance et les tests de sécurité.

Si une stratégie de test optimale passe avant tout par la recherche d’un outil adapté, c’est l'automatisation des tests qui apporte le plus de valeur ajoutée, en permettant notamment de réduire le temps de test, de planifier leur exécution, et de garantir le bon fonctionnement des fonctionnalités essentielles de l’application.

Dans le cas du projet Acrobatt, seules des fonctionnalités métiers de l’API web étaient à même d’être automatisées. Nous avons donc établi une liste des fonctionnalités critiques et essentielles de cette API afin d’en automatiser les tests et des les exécuter à chaque ajout ou modification d’un composant.

**Liste des fonctionnalités à tester**

* Requête utilisateurs
  + Inscription d’un utilisateur
  + Connexion d’un utilisateur
  + Inscription à un challenge
  + Affichage des challenges ou l’utilisateur est inscrit
  + Affichage des challenges où il peut s’inscrire
  + Création d’évènements sur un challenge
    - Départ d’un challenge
    - Arrivée à un challenge
    - Déplacement
    - Arrivée à un point de passage
    - Arrivée à un obstacle
    - Réponse à un obstacle
* Requête Challenge et parcours
  + Création des informations d’un challenge (nom, description, échelle, ...)
    - Téléchargement de la carte du parcours
    - Création d’un parcours :
  + Création des points de passages
  + Création des segments
  + Choix du départ et de l’arrivé du parcours
  + Création des obstacles
  + Validation d’un parcours
  + Affichage des informations pour un administrateur (Liste de ses challenges créés, en cours d’édition ou validés et jouable par les utilisateurs).
  + Validation des propositions d’obstacles sur un challenge par l’admin du challenge
  + Recherche d’un challenge par son identifiant

# 

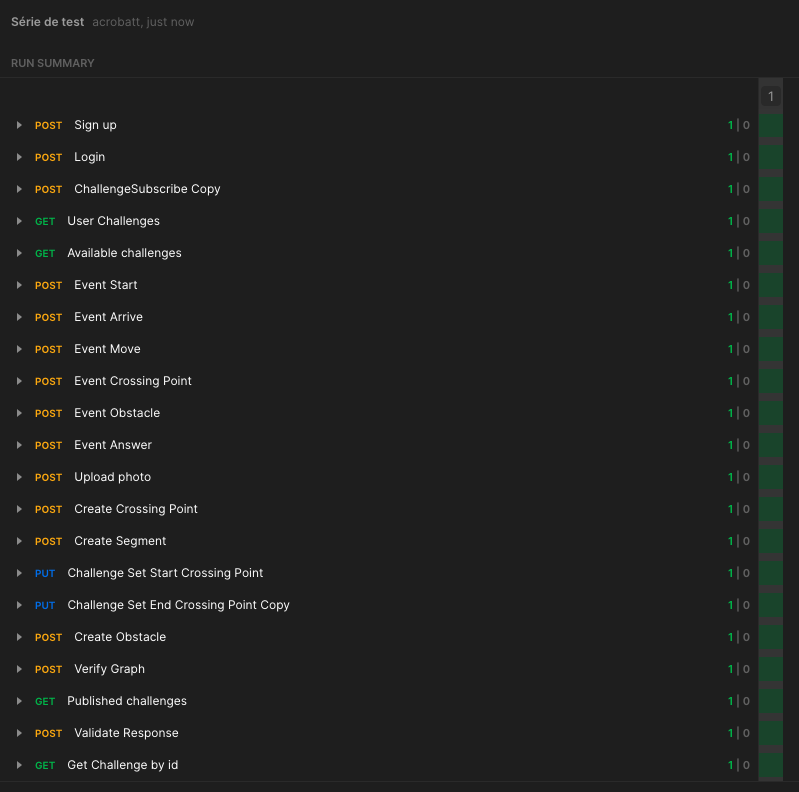
# 

# 

# Création d’une série de tests

La mise en place de séries de requêtes HTTP est similaire à celle proposée par de nombreux outils de tests. Elle consiste à spécifier le résultat attendu à chaque requête pour valider le test. Il est également possible d'exécuter une collection de requêtes sous la forme d’une série de tests, et d’en vérifier le résultat.

L’ensemble des paramètres des requêtes et de leurs réponses sont finalement disponibles pour rapidement identifier d’où proviennent les erreurs.



# Synthèse sur l’utilisation de postman

Il existe beaucoup de solutions logicielles pour tester une API web. Après avoir utilisé l’outil de test Postman pendant le première semestre dans le cadre de mini-projet Sharelock, il semblait évident pour notre équipe de se tourner vers cet outil que nous estimons fonctionnel et pertinent pour réaliser les tests fonctionnels de notre API. De plus, l’outil possède une documentation bien faite et très détaillée, ce qui est un plus dans son utilisation.

Dans le cadre de la mise en place de notre API, les tests Postman nous ont permis de nous affranchir de l’aspect graphique de l’application et donc de nous concentrer sur le retour de l’API uniquement. De plus, ils nous ont aussi permis de tester bien en amont les fonctionnalités de celle-ci, sans attendre d’avoir à disposition une IHM.

Le partage des collections de tests, en les publiant en ligne ou en les exportant en JSON, ainsi que la génération d’une documentation pour l’API, cet outil nous a permis d’améliorer l’intégration des tests dans notre développement et de faciliter un travail collaboratif.

L’automatisation des tests a aussi été un plus, bien programmés ils pouvaient être relancer à chaque évolution du code et intervenir dans la mise en place des tests de non-régression et surtout nous permettre de détecter d’éventuels effets de bord qui auraient pu passer inaperçus. Et dans certains cas, la mise en place des tests, nous a confortée dans le choix de nos logiques métiers.

En développant l'application nous avons atteint les limites de l’outil dans certains cas où le volume de données récupérées par les requêtes étaient trop importantes pour être traitées, notamment lorsqu'elles comprenaient de nombreuses images. Postman s'arrêtait alors brusquement. Donc nous avons essayé de passer sur la version en ligne de l’outil mais sans grande amélioration.

Un des bémols de l'outil est que pour disposer de entièreté de ces fonctionnalités, il faut payer dans le cadre d’une formation cela est un frein et même dans des petites structures, le bénéfice tiré de l’outil peut être réduit par ces contraintes. Il va aussi sans dire que le temps consacré à la rédaction des tests doit aussi être pris en compte pour en calculer les avantages. Néanmoins, utilisé à bon escient et avec suffisamment de temps pour mettre en place des tests complets, Postman est un aide non négligeable pour la robustesse d’une application.