Mission VisiTTracking (VTT)

<u>présentation générale :</u>
Les visiteurs médicaux de GSB ont besoin d'un suivi pour leurs visites.

La mission actuelle va consister à enregistrer des informations en provenance des visiteurs afin d'obtenir plus tard des statistiques sur les médecins visités.

Dans ce cadre, le responsable d'équipe du service développement missionne votre équipe sur ce projet :
Le service chargé du suivi des visites souhaite récupérer des informations sur les visites des délégués médicaux
Les informations à récupérer par visiteur médical sont les suivantes :

- le médecin visité
- si la visite était fixée avec le médecin (rendez-vous ou non)
- l'heure d'arrivée
- le temps d'attente
- l'heure de départ
- le temps de visite.

Ces informations devront pouvoir être saisies au fur et à mesure de la journée par le délégué à partir de son smartphone ou sa tablette, ou sur son ordinateur.
Un service du site de Paris sera chargé de mettre à disposition les informations concernant les médecins et les visiteurs.

Dans ce cadre, votre mission sera de concevoir et livrer, une application client lourd nommée VTT qui propose :
- un CRUD pour les visites
- un CRUD pour les visiteurs
- un CRUD pour les médecins

Cette application doit communiquer avec une base de données grâce à des API.

L'application doit être livrée sur un GitHub Equipe et la BDD et les API sur la machine équipe du nuage pédagogique
Nommé un responsable de projet.
Document de suivi de projet à fournir **pour chaque sprint** :

	État - date	Nom étudiant 1		Nom étudiant 2				Nom étudiant 3			
SPRINT		Tps prev	Tps réel								
Us + tâche											

État : à faire / en cours (date/heure début) / fait (date/heure de fin) / validé

contraintes :
chaque étudiant doit faire au moins un sprint d'API.
Les autres sprints sont à partager.

Les technologies choisies par GSB :
- c# pour architecture applicative (front et back)
- PHP pour les API

Découpage du projet en sprint

n°		prédécesseurs	deadLine
00	Mise en place de l'environnement de développement et de test	-	5 septembre minuit
01	Sprint VTT_V0	-	12 septembre 15h30
02	Sprint VTT_V1	01	12 septembre minuit
03	Sprint VTT_V2	02 / 05 / 06 / 07	19 septembre 17h30
05	Sprint API_VISITEUR	08	18 septembre minuit
06	Sprint API_MEDECIN	08	18 septembre minuit
07	Sprint API_VISITE	08	18 septembre minuit
08	Sprint BDD avec jeu d'essai	-	5 septembre minuit
09	Sprint automatiser sauvegarde BDD	08	12 septembre minuit

Pour la deadLine :
- suivi de projet
- contenu du sprint
- gitHub et machine équipe à jour (faire une nouvelle branche pour chaque sprint)

Sprint VTT_V0 (aucun accès à la base de données)

tâche : concevoir classe abstraite Personne
tâche : concevoir classe Visiteur (mettre en place la double navigabilité entre Visiteur et Visite)
tâche : valider classe Visiteur
tâche : concevoir classe Médecin
tâche : valider classe Médecin
tâche : concevoir classe Visite
tâche : valider classe Visite
tâche : concevoir classe Passerelle (liste des Visiteurs / liste des Visites / Liste des Visites)
tâche : concevoir procédure de création de jeu d'essai pour implémenter Passerelle
tâches : création de méthodes classe Passerelle (A partager) + validation
List<Visites> getLesVisites(idVisiteur)
addVisite(uneVisite)
updateVisite(unVisite)
deleteVisite(unVisite)
List<Visiteur> getLesVisiteurs()
addVisiteur(unVisiteur)
updateVisiteur(unVisiteur)
deleteVisiteur(unVisiteur)
List<Medecin> getLesMedecins()
addMedecin(unMedecin)
updateMedecin(unMedecin)
deleteMedecin(unMedecin)
tâche : documentation technique (diagramme de classe + extraction de documentation de code sous forme d'un site web)
tâche : livraison sur GitHub (application + documentation) + BDD sur machine équipe du nuage pédagogique + script sauvegarde BDD automatisée.

Sprint VTT_V1 (aucun accès à la base de données)

Interface graphique + code pour affichage jeu d'essai « en dur »
je veux pouvoir voir tous les visiteurs
je veux pouvoir créer un visiteur
je veux pouvoir modifier un visiteur
je veux pouvoir supprimer un visiteur
je veux pouvoir voir toutes les visites
je veux pouvoir créer une visite
je veux pouvoir modifier une visite
je veux pouvoir supprimer une visite
je veux pouvoir voir tous les médecins
je veux pouvoir créer un médecin
je veux pouvoir modifier un médecin
je veux pouvoir supprimer un médecin
tâche : documentation utilisateur
tâche : livraison

Sprint VTT_V2 (avec appel API)
intégration appel des api

Interface graphique + modification classe Passerelle pour appel API
je veux pouvoir voir tous les visiteurs
je veux pouvoir créer un visiteur
je veux pouvoir modifier un visiteur
je veux pouvoir supprimer un visiteur
je veux pouvoir voir toutes les visites
je veux pouvoir créer une visite
je veux pouvoir modifier une visite
je veux pouvoir supprimer une visite
je veux pouvoir voir tous les médecins
je veux pouvoir créer un médecin
je veux pouvoir modifier un médecin
je veux pouvoir supprimer un médecin
tâche : documentation utilisateur
tâche : livraison

Sprint API_VISITEUR

tâche : création classe en PHP
tâche : test de la classe
tâche : création classe DaoVisiteur

tâche : création getLesVisiteurs()
tâche : test
tâche : getUnVisiteur(id)
tâche : test
tâche : création addVisiteur(unVisiteur)
tâche test
tâche : updateVisiteur(unVisiteur)
tâche : test
tâche : deleteVisiteur(unVisiteur)
tâche test

tâche : concevoir API visiteur (GET) + script de test.
tâche : modifier API visiteur (GET , un visiteur en fonction id) + script de test.
tâche : modifier API visiteur (POST) + script de test.
tâche : modifier API visiteur (PATCH) + script de test.
tâche : modifier API visiteur (DELETE) + script de test.
tâche : documentation technique de l'API
Tâche : livraison

Sprint API_VISITE

tâche : création classe en PHP
tâche : test de la classe
tâche : création classe DaoVisite

tâche : création getLesVisites()
tâche : test
tâche : getUneVisite(id)
tâche : test
tâche : création addVisiteur(uneVisite)
tâche test
tâche : updateVisiteur(uneVisite)
tâche : test
tâche : deleteVisiteur(uneVisite)
tâche test

tâche : concevoir API visite (GET) + script de test.
tâche : modifier API visite (GET , une visite en fonction id) + script de test.
tâche : modifier API visite (POST) + script de test.
tâche : modifier API visite (PATCH) + script de test.
tâche : modifier API visite (DELETE) + script de test.
tâche : documentation technique de l'API
Tâche : livraison

Sprint API_MEDECIN

tâche : création classe en PHP
tâche : test de la classe
tâche : création classe DaoMedecin

tâche : création getLesMedecins()
tâche : test
tâche : getUnMedecin(id)
tâche : test
tâche : création addMedecin(unMedecin)
tâche test
tâche : updateMedecin(unMedecin)
tâche : test
tâche : deleteMedecin(unMedecin)
tâche test

tâche : concevoir API medecin (GET) + script de test.
tâche : modifier API medecin (GET , un visiteur en fonction id) + script de test.
tâche : modifier API medecin (POST) + script de test.
tâche : modifier API medecin (PATCH) + script de test.
tâche : modifier API medecin (DELETE) + script de test.
tâche : documentation technique de l'API
Tâche : livraison