Documentation technique – rendev

Projet de gestion d’évènements, ajout, modification et suppression via application c# avec une interface web

MARINHO Johnny, JAUCH Walter, JEANRENAUD Nelson, GOUVEIA Andre, MACHADO Jorge

09.03.2020

Version 1.0

# Introduction

Ce rapport est une documentation technique d’une application faite, en groupe, dans le cadre du module 306 supervisé par Mme Terrier. Il documente le code et l’ensemble des fonctions, méthodes et algorithmes du projet Rendev, développé par Jauch Walter, Gouveia André et Jeanrenaud Nelson pour la partie C# ainsi que Marinho Johnny et Machado Jorge.

Le projet est un gestionnaire d’évènements qui sont affichés sur la carte à l’aide d’une interface web. Les évènements sont donc ajoutés, modifiées et supprimées via l’application de bureau et ensuite, elles sont visibles, de manière synchronisée sur le site web.

# Cahier des charges

## Matériel et logiciels à disposition

* Visual Studio 2017
* Mysql
* Visual Studio Code
* OpenStreetMap (api WEB)
* OpenLayers (librairie JS)
* GMap (nuGet C#)

## Descriptif complet du projet

### Explication :

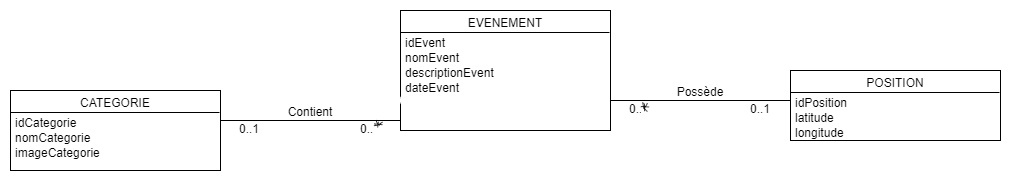
Un site web avec une carte sur laquelle on voit des évènements. Ces évènements peuvent être crée sur une application C#.

### Côté Web :

Un site avec une seule page qui permet de visualiser les multiples évènements sur une carte dynamique utilisant la technologie api OpenStreetMap. Le visiteur du site peut rechercher les évènements selon différent critères tel que le nom, la date ou encore la catégorie.

### Côté C# :

Une page d’accueil avec une carte qui permet à l’utilisateur de sélectionner une location l’utilisateur peut ensuite renseigner des champs, comme description, nom, date… sur un formulaire classique. Pour finir, il peut ensuite envoyer les points qu’il a créé sur le serveur pour qu’il puisse être affiché sur le site web.



## Livrables

* Planning
* Rapport de projet
* Manuel utilisateur (si applicable)
* Journal de travail

## Analyse de l’existant

### Analyse des programmes similaires

* Le site McDonald’s permet de trouver un emploi dans un périmètre donné et utilise un système d’objets sur une carte similaire à celle utilisée dans ce projet
* <https://www.ge-soif.ch/> aussi permet de trouver des fontaines placées sur une carte ou l’on peut ajouter, modifier et supprimer des objets

## Technologies à utiliser

### OpenStreetMap (api WEB)

Selon Wikipédia : « OpenStreetMap est un projet Service collaboratif de cartographie en ligne qui vise à constituer une base de données géographiques libre du monde, en utilisant le système GPS et d'autres données libres. ».

C’est donc une api parfaite pour le client WEB. Elle est largement utilisée et permet une implémentation facile et logique dans notre projet.

### OpenLayers (librairie JavaScript)

Selon Wikipédia : « OpenLayers est un logiciel libre, publié sous licence BSD. Il constitue une bibliothèque de fonctions JavaScript assurant un noyau de fonctionnalités orienté vers la mise en place d'applications clientes Web cartographiques fluides. »

Cette dernière va permettre aux événements d’apparaître sur la carte aux endroits où ils ont été créés via l’application C#. Elle permettra aussi d’afficher les informations de l’évènement en question quand l’utilisateur interagira avec.

### GMap

GMap (ou Great Map) est un contrôle .NET open source, puissante et, surtout, gratuit. On peut utiliser des cartes de Google, Bing, OpenStreetMap, Yahoo!, Pergo, WikiMapia et bien d’autres.

C’est ce que nous avons d’utiliser, sous conseils de notre enseignante (Mme. Anne Terrier), pour l’application C#. C’est grâce à GMap que nous afficherons les points déjà ajoutés et nous pourrons également récupérer l’adresse d’un point.

## Analyse fonctionnelle

## Méthodologie

Pour planifier ce projet nous nous sommes basés sur la méthode en 6 étapes que voici :

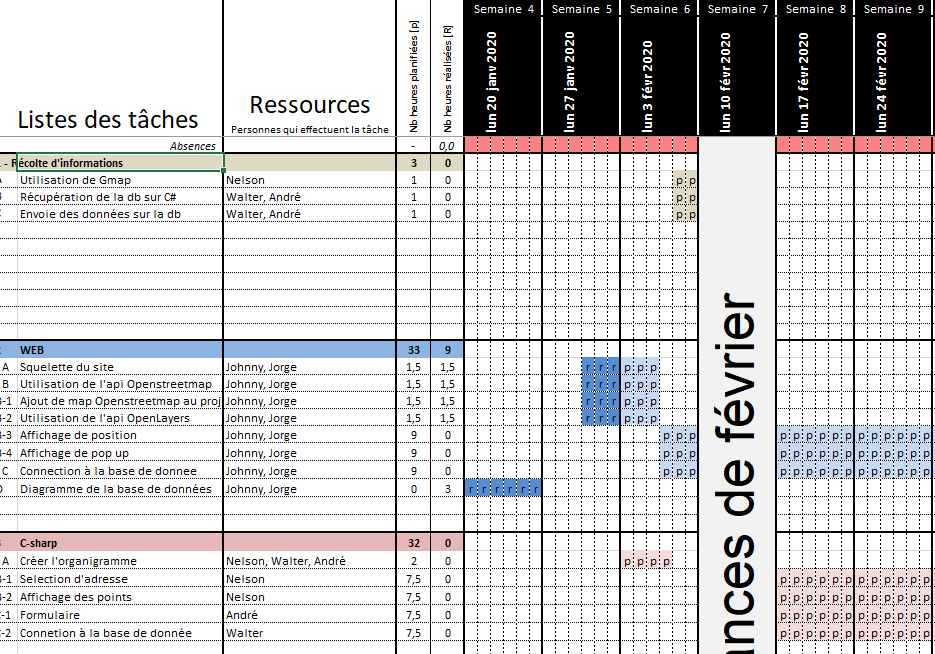
### 1. S’informer

La première étape du projet a été de s’informer sur l’énoncé de notre projet, sur les technologies à utiliser et les programmes similaires.

Nous avons donc défini le projet, compris le travail à faire. Ensuite, nous nous sommes documentés et informés sur les délais liés au projet.

## 2. Planifier

Dans cette deuxième étape nous avons plannifié notre projet à l’aide de taches attribuées aux membres de l’équipe. Nous en avons fait un tableau Excel pour suivre l’avancement du projet et des tâches à finir pour chaque personne dans le temps dont voici un extrait :



Nous nous sommes ensuite mis d’accord sur les logiciels et les outils à utiliser et sommes passé à la prochaine étape du projet.

### 3. Décider

Dans cette étape, nous avons fait valider le cahier des charges avec notre mandant (l’enseignante) et, il fut validé après quelques modifications.

### 4. Réaliser

Nous nous sommes alors mis au développement, scindés en deux groupes : messieurs Marinho et Machado pour la partie web et messieurs Gouveia, Jauch, Jeanrenaud pour la partie C#. Toute l’équipe a également participé à la rédaction des différentes documentations.

### 5. Contrôler

[A FAIRE]

### 6. Evaluer

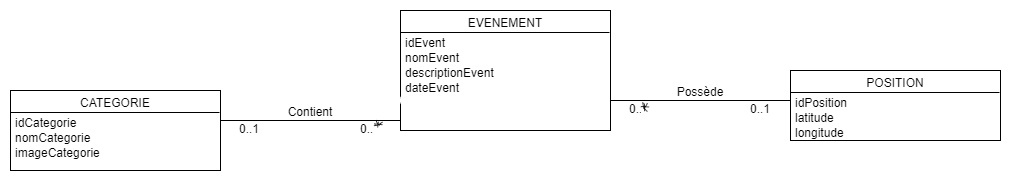
Nous avons, dans le but d’avoir un suivi, tenté d’analyser un maximum notre travail dans le journal de bord. Cela nous permet de garder un regard critique afin d’améliorer notre travail.

# Implémentation

## Base de données

Que ce soit l’application WEB ou l’application de bureau (C#), une base de données est indispensable pour que nous puissions stocker les données nécessaires au bon fonctionnement du système.

Nous nous sommes donc mis d’accord sur un schéma de base de données qui convenait aux utilisations des différentes application. La base devait être capable de stocker tous les événements ajoutés. Cela implique la position à laquelle l’évènement est enregistré ainsi que la catégorie d’événements (Loisir, Travail, etc.). Voici le modèle que nous avons décidé d’appliquer :



C’est une base de données composée de trois tables. Les tables événement et position se remplissent simultanément étant donné que les positions stockées correspondent aux événements ajoutés sur l’application de bureau. La table concernant les catégories est également maintenue via l’application C#. L’utilisation d’images permet de mieux distinguer les événements en regardant la carte et l’application WEB devient donc plus intuitive.

## Structure du projet

### Partie WEB

La structure du projet web est la suivante : trois dossiers, un pour le style, les fichier css, un autre pour le modèle de la base ainsi que pour le fichier de création de la base et enfin un dernier pour les images. Les fichiers php sont à la racine du projet.

[A FAIRE] 🡪 parler de la structure des dossiers et tout, les classes si yen a

### Partie C#

Après on peut parler de git aussi

## Analyse du code

### Partie WEB

### Les fonctions

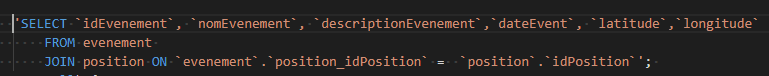
Côté PHP, nous avons 3 fonctions. La première, **getEvents()**, nous permet de recevoir tous les évènements, ce qui permet, lors du lancement de la page, de récupérer leur latitude, longitude ainsi que leur id et de les placer sur la carte. Cette fonction utilise le design pattern singleton. On va donc créer une variable statique pour y stocker la requête. Ensuite, si cette variable n’a jamais été instanciée, nous allons le faire et l’exécuter dans la base et retourner le résultat à la fin. Si elle a déjà été instanciée, nous allons juste retourner son résultat puisque ça veut dire qu’elle a déjà été exécutée auparavant.

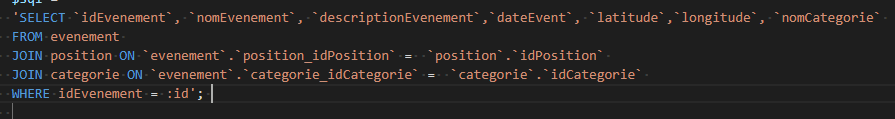
La deuxième, **getEventByID()**, va entrer en jeu lorsque l’utilisateur va sélectionner un évènement sur la carte. Cela va déclencher une action Javascript, que l’on décrira plus loin, qui enverra l’id de l’évènement sélectionné à cette fonction, qui retournera ensuite un tableau avec les informations relatives à cet évènement. Cette dernière ne permet pas d’implémenter le design pattern singleton car elle prend en compte un paramètre qui peut changer dans la requête.

La troisième fonction, **displayEvent()**, permet d’afficher les informations précédemment reçues.

Coté Javascript, nous commençons par créer trois variables qui reprennent, avec la fonction json\_encode, tous les évènements ainsi que la latitude et la longitude de celui sélectionné.

### Les requêtes à la base

Voici les requêtes que nous avons utilisées dans les fonctions décrites ci-dessus :

Cette requête est utilisée dans la fonction **getEvents().**

Celle-ci est utilisée dans la fonction **displayEvent()**.

## Rapport de tests :