Année 2013 – 2014

Rapport de projet – EI4 AGI

Développement d'un site de quizz en PHP

Projet réalisé par :

Guillaume Fache

Projet encadré par :

Mehdi Lhommeau

ISTIA – Université d'Angers

Remerciements

Je tiens tout d'abord à remercier M. Mehdi Lhommeau pour son encadrement, son support et ses suggestions tout au long de ce projet, qui m'ont permis d'avancer en ligne droite quand je ne savais pas quelle direction prendre. Je remercie aussi Valentin Gourdon, Erwan Peneau et les autres étudiants de l'IUT géothermique de Nantes pour leur apport et leur gestion de ce projet. Enfin je tenais à remercier pour son travail Armelle d'Anjour, qui s'est occupé de réaliser une maquette du site, et globalement toute la partie design du projet.

Table des matières

Introduction

L'apprentissage peut se faire de mille façons. Une des plus efficace est par le jeu, car l'assimilation de la connaissance se fait presque à son insu, en offrant une sensation d'accomplissement et de réussite bénéfique pour la personne. C'est ce principe que l'IUT géothermique de Nantes (GTE) a décidé d'exploiter en lançant le projet GTE Challenge, une plateforme Web où les étudiants de la formation peuvent se loguer, participer à différentes activités ludiques, consulter son historique de score, son évolution, ainsi que ceux de ses amis. Le projet allait jusqu'à inclure des regroupements de joueurs par groupes (promotion, année d'étude …) et réaliser des tournois entre ces groupes.

Ce projet a été réalisé sous l'initiative de l'IUT géothermique de Nantes, en collaboration avec Armelle d'Anjou, chargée du design du site, et de l'ensemble de la partie graphique. On va voir dans ce rapport comment s'est articulée la réalisation de ce projet, ainsi que l'utilité d'une communication efficace dans un projet de cette envergure, notamment à quel point quelques approximations de langage peuvent mener à d'assez radicales conséquences, quand on travaille à plusieurs et qu'on ne communique jamais de vive voix. Ce a aussi été l'occasion pour moi de mettre en parallèle deux techniques de programmation Web, avec deux langages très différents, Node.js et PHP5, et voir les différents types d'approches du problème proposé en fonction du langage utilisé. Ces approches, je les détaillerai dans trois parties, les deux premières consacrées à la découverte de JavaScript et de Node.js, ainsi que leur application à ce projet, et la troisième au site final, développé en PHP 5.

Présentation du projet :

I. Les prémices du projet

Le projet était à l'origine prévu pour être réalisé à l'aide de Node.js, un framework en JavaScript, aux spécificités particulièrement adaptées à ce type de projet. J'ai d'abord dû me familiariser avec JavaScript, langage que je n'avais vu que de très loin, et jamais en cours. Je me suis basé sur les tutoriaux [1] pour le JavaScript et [2] pourNode.js. J'ai aussi eu fréquemment recours à la documentation JavaScript [3]. A l'issu de ce tutoriel, j'avais appris à maitriser les différentes commandes console nécessaires pour faire fonctionner les scripts Javascript. Les premiers scripts, simplissimes, permettait la communication entre un serveur basique au possible et un client. Au fur et à mesure, j'apprenais à récupérer les informations dans la barre d'adresse, à écouter les évènements et à créer des fonctions de callback (fonctions de rappel qui s'exécutent quand un évènement se réalise). Ces fonctions sont à la base même de Node.js. En effet, cet outil permet d'adopter un modèle non bloquant, c'est-à-dire que le programme n'est pas obligé d'attendre le retour d'un processus particulièrement long. Il peut continuer à s'exécuter et revenir déclencher un évènement une fois le processus long terminé.

Par exemple :

- Télécharger une mise à jour
    - Installer la mise à jour
- Continuer l'exécution du programme sans attendre
- Tâche suivante

Suite au cours de Management Projet, on décide de découper mon travail en deux phases :

- phase 1 : assurer le bon fonctionnement des tâches principales : assurer une session de jeu pour joueur unique, et l'interface administrateur qui permet la gestion des questionnaires.

- phase 2 : ajouter tous les éléments secondaires : une interface graphique plus poussée, la gestion de sessions multiples, les historiques des joueurs plus ou moins poussés, le chat, la gestion de communautés de joueurs.

On a élaboré un premier diagramme de Gantt qui rend compte des objectifs premiers qui étaient fixé. Rétrospectivement, tous n'ont pas été réalisés, et certains étaient juste preuves d'une ambition bien trop grande, notamment le portage sous Android. Ce diagramme est présenté en annexe 1. Un deuxième diagramme est présenté en annexe 2, présentant le Gantt des tâches qui ont été effectivement réalisées durant ce projet, rétrospectivement. L'accompagne en annexe 3 la répartition temporelle circulaire de chacune des tâches sur le temps de projet total. Elle permet d'isoler les éléments qui ont concentré le plus de temps, et donc ceux sur lesquels il aurait convenu d'améliorer la méthode de travail pour mener le projet à bien plus rapidement. On peut noter que c'est souvent la partie apprentissage qui a nécessité le plus de temps, plus que la conception des scripts eux même.

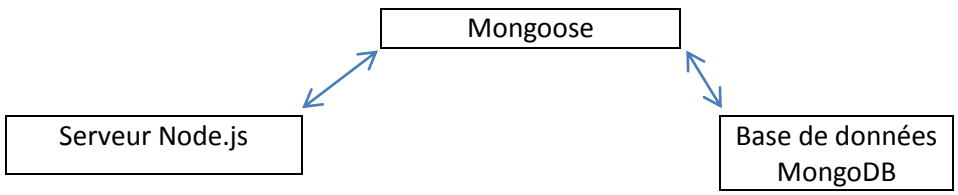
En parallèle, émerge le besoin de rencontrer les étudiants du GTE pour préciser certaines spécificités du cahier des charges transmis, et s'assurer de la faisabilité du projet tant dans son ensemble que dans les détails. Pour cela on s'est rendu à Carquefou l'espace d'un après-midi pour échanger avec les étudiants et leur tuteur sur leur vision du projet. On a abouti à un cahier des charges plus précis, et les impératifs à fournir sont décidés. Conformément à ce qui avait été anticipé, la première phase du projet consisterait à assurer le minimum fonctionnel, à savoir la possibilité de créer un compte, de s'identifier et jouer à un quizz. L'habillage, les affrontements entre joueurs, la sauvegarde de l'évolution du joueur seraient des améliorations à considérer dans un second temps uniquement.

II. L'association de Node.js avec MongoDB et jQuery

Très vite il est apparu qu'il y aurait besoin d'une communication avec une base de données, sur laquelle serait stockée les données des utilisateurs (nom, score, mot de passe …) ou encore les questions du quizz. Il nous a fallu commencer par rechercher quel SGBD (Système de Gestion de Base de Données) utiliser avec Node.js, et MongoDB s'est imposé comme un choix pertinent car il permettait une utilisation relativement aisée à l'aide d'un module spécifique prévu à cet effet : Mongoose. On s'est notamment servi du tutoriel [4]. En quelques mots, il fallait d'une part installer MongoDB sur la machine par un simple copier-coller dans le répertoire de notre choix, puis lancer dans l'invite de commande l'installation de Mongoose, qui permet la communication entre Node.js et le SGBD. On utilise pour ceci la commande :

```
> npm install mongoose
```

On pourrait schématiser l'utilité de Mongoose ainsi :



Après s'être placé dans le répertoire où l'on a copié mongoDB, on le lance à l'aide de la commande :

```
> mongod.exe --dbpath c:\mongodb\test
```

Dans une première étape, on aboutit à un script qui permet d'ajouter une association d'un pseudo et d'un mot de passe à une base de données MongoDB. On cherche dans un second temps à ce que le mot de passe ne soit pas fixé en dur dans le script mais reçu dynamiquement par socket. Cette manipulation impose de se familiariser avec le module SocketIO, notamment à l'aide du tutoriel [5].Cependant, pour ne pas passer trop de temps sur la partie consacrée à la conception du site lui-même, on laisse de côté la partie connexion via socket pour se focaliser sur la conception du quizz. On cherche sur internet des exemples de projets similaires pour s'en inspirer, et on s'aperçoit bien vite que la maitrise de jQuery est nécessaire pour avancer. Là encore l'apprentissage se fait depuis les fondamentaux, car on ne connait rien à cette bibliothèque JavaScript pourtant particulièrement répandue. On s'intéresse notamment au tutoriel [6] et à la documentation officielle [7]. La découverte de la plateforme JSfiddle grâce à M.Lhommeau permet de consulter un large choix de code de quizz proposant chacun des spécificités dont on peut s'inspirer librement. On apprend sur le tas, à partir du code que l'on trouve, car à ce stade du projet on n'a pas le temps de s'attarder sur chaque nouvelle notion.

Alors qu'on approche à grand pas d'une première version jouable, on est contacté par l'équipe du GTE de Nantes qui demande quel type d'hébergement serait le plus approprié pour ce projet. On leur fournit la taille supposée de l'ensemble du site finis, sans leur préciser à nouveau le langage de programmation qui devait être supporté, ceci ayant déjà été l'objet de précédents échanges. Cette erreur de communication aura des conséquences importantes sur la suite du déroulement du projet. Les options étaient d'héberger le site sur un serveur propre à l'IUT, sur lequel est déjà stocké le site de la formation, ainsi que celui du BDE, ou bien d'obtenir ces services d'un fournisseur externe. Pour des raisons techniques qui ne m'ont pas été communiquées, c'est OVH qui a été retenu pour héberger le site.

III. Un nouveau départ en PHP

A ce stade du projet, il a fallu repenser l'intégralité des outils utilisés pour réaliser le site. Node.js n'étant pas supporté par OVH, la plateforme d'hébergement choisie par les étudiants du GTE, on passe à des scripts en PHP et une base de données MySQL, à laquelle on accèdera via phpMyAdmin et des requêtes SQL intégrées aux scripts. Cette matière ayant été enseignée au premier semestre, on compte sur cette facilité pour avancer rapidement dans la programmation. On s'aidera régulièrement tout le long de cette partie de l'ouvrage [1] de la bibliographie. Dès le début, on essaiera de profiter pleinement des nouvelles fonctionnalités de PHP 5, qui n'avaient pas été explorées en cours mais qui s'avèrent être fortement demandées sur le marché des stages. On commence par réaliser un script de connexion qui couple PHP et MySQL, et qui permet l'inscription dans une base de données stockée sur OVH d'un ensemble de données. On choisit dans un premier

temps de développer en local à l'aide de Wamp, mais on se rend vite compte qu'il est aussi rapide de téléverser le code à chaque modification directement sur le serveur, tout en gardant à chaque séance une copie de sauvegarde du site qui fonctionne, pour qu'on ne régresse pas dans le projet suite à une manipulation malencontreuse. On apprend pour cela à se servir de l'outil Filezilla, qui permet le transfert de fichiers sur le serveur d'OVH, et ce par une connexion simple et un simple drag and drop. Voyons maintenant un peu plus en détail quelques parties clés des différents scripts réalisés. Chacun de nos scripts nécessitant une interaction avec la base de données commence par la partie :

```
    try
    {
        // On se connecte à MySQL
        $bdd = new PDO('mysql:host=mysql51-116.perso;dbname=gtechalltest', 'gtechalltest', 'ludotheque05');
    } catch (Exception $e)
    {
        // En cas d'erreur, on affiche un message et on arrête tout
        die ('Erreur : '.$e->getMessage());
    }
```

Une fois qu'on a géré la lecture dans la base de données, on peut développer un script qui permet de vérifier l'existence d'un compte dans la table 'login' de la base de données. On s'est généreusement servi du tutoriel [8] pour développer ce script. Ces scripts (inscription d'un nouveau compte et vérification de l'existence d'un compte) sont appelés à partir de notre page d'accueil, sous les noms de signin.php et login.php, à l'aide de formulaires. Le cœur du login repose sur le bout de code suivant :

```
$bdd = new PDO('mysql:host=mysql51-116.perso;dbname=gtechalltest', 'gtechalltest', 'ludotheque05');

if ($_POST['login'] != ""  && $_POST['pwd'] != "")
   {
// la preparation de la requête SQL
$req = $bdd->prepare('INSERT INTO login(nom, mdp) VALUES(?,?)');

//l'exécution de la requête
$req->execute(array(
   $_POST['login'],
   $_POST['pwd']
        ));
    }
```

Il n'y a pas de contraintes concernant la création d'un compte. Dans un second temps, il conviendrait de faire en sorte que seules des adresses mail valides finissant par @etud.univ-nantes.fr

puissent autoriser la création d'un compte. Une fois correctement logué, l'utilisateur peut ensuite être redirigé vers le quizz lui-même. C'est là que l'élaboration du jeu à proprement parlé a commencé.

Plutôt que de partir de zéro, on a choisi de partir de l'ossature d'un projet déjà existant, en l'arrangeant aux besoins de notre projet. Le quizz original puisait ses questions dans un fichier PHP distinct, sobrement nommé questions.php, mais cette solution ne permettait pas assez de flexibilité, dans la mesure où chaque question devait être élaborée sur le même modèle, chaque quizz devait avoir le même nombre de questions. Dans un premier temps on a considéré l'option de faire en sorte que l'administrateur du site pourrait modifier le contenu de ce fichier à son bon vouloir, quitte à lui développer une petite interface pour lui faciliter le travail, mais après mûre réflexion, et selon l'avis de M.Lhommeau, il s'est avéré que cette solution n'était pas la meilleure, et qu'il valait mieux privilégier une inscription des questions dans une base de données.

Le script quizz.php a été l'occasion d'expérimenter un peu avec la Programmation Orientée Objet permise par PHP 5, et on a simplifié notre code en créant une classe 'questions' avec des attributs pertinents :

```
class Question
{
		// Attributs
		public $question;
		public $reponse_1;
		public $reponse_2;
		public $reponse_3;
		public $reponse_4;
		public $bonne_rep;
		public $comm;
		public $img;
}
```

Puis, tant qu'on a des données à lire dans la base de données, on enregistre les questions dans des objets de type Question qu'on stock ensuite dans un tableau :

```
$i=0;
		// Nous traitons les résultats en boucle
		while( $enregistrement = $select->fetch() )
		{

		$question = new Question();

		// on met chacun des attributs de questions  dans une variable de type Question
```

```
                $question->question=$enregistrement->question;
                $question->reponse_1=$enregistrement->reponse_1;
                $question->reponse_2=$enregistrement->reponse_2;
                $question->reponse_3=$enregistrement->reponse_3;
                $question->reponse_4=$enregistrement->reponse_4;
                $question->bonne_rep=$enregistrement->bonne_rep;
                $question->comm=$enregistrement->comm;
                $question->img=$enregistrement->img;

                $arrayQuestion=array($question->question,
                                     $question->reponse_1,
                                     $question->reponse_2,
                                     $question->reponse_3,
                                     $question->reponse_4,
                                     $question->bonne_rep,
                                     $question->comm,
                                     $question->img);

                $quizzfinal[$i]=$arrayQuestion;

                unset($question); // On supprime l'objet question pour pouvoir en recréer un au
//prochain appel de la boucle
                $i++;
                }

$quizz= array($quizzfinal[0],
 $quizzfinal[1],
 $quizzfinal[2],
 $quizzfinal[3]
);
```

On pourrait sans doute simplifier le code en créant une méthode Set_Tableau() à la classe Question qui intégrerait directement chacune des questions à un tableau directement manipulable par l'utilisateur. Cette démarche pourra être complétée dans une reprise du projet ultérieure.

On créé donc une nouvelle table qu'on appelle questions. Les champs à renseigner sont l'intitulé de la question, les quatre réponses possibles, l'indice de la bonne réponse, et un éventuel commentaire. La première étape a donc été de communiquer entre le script quizz.php et la base de données pour afficher chacune des données au bon endroit. Dans un second temps, on s'est rendu compte que certaines questions avaient besoin d'être illustrées par une image pour être comprises. Il a donc fallu intégrer ces images à la base de données. On a d'abord choisi de les stocker directement, à l'aide de variables BLOP, qui représentent des objets binaires de grande taille. Il est cependant vite apparu qu'en dehors du fait que ces variables étaient très peu pratiques à manipuler, il était préférable de stocker l'adresse des images comme texte, ce qui permettait de ne pas alourdir inutilement la base de données, et permettait donc un meilleur accès aux données.

Après avoir assuré un quizz fonctionnel, il convenait de réaliser une page d'administration qui aurait pour fonction de permettre la visualisation, l'ajout et la suppression de questions de la banque de données. Cette page est accessible en se connectant avec un compte administrateur spécifique. Cette page reprenait des principes déjà utilisés dans les parties précédentes, à savoir la lecture et l'écriture dans une base de données. La seule réelle nouveauté résidait dans l'utilisation de la requête SQL UPDATE. Plusieurs problèmes sont cependant apparus. L'ordre des questions affichées dans la base de données dépendait de l'ordre dans lequel elles avaient été entrées, et n'avait pas forcément de sens. L'idéal aurait été de pouvoir créer une interface modifiable d'un simple drag and drop par l'administrateur, voire même créer des sous catégories de questions (par exemple « Mécanique des fluides », « Electrotechniques »,…) . Cette partie reste encore à réaliser.

On s'est vite rendu compte qu'il était possible d'accéder à la page d'administration en évitant l'identification en rajoutant juste '/page_admin.php' à la fin de la barre d'adresse. Pour éviter combler cette faille de sécurité importante, on insère la protection suivante au début du script de la page d'administration :

```
<? if ($_POST['login']!="admin"||$_POST['pwd']!="ludotheque05")
{
        echo "erreur d'identification";
        header('Location: index.html'); // redirige vers la page d'accueil
}
else
{ // le reste du code de la page d'administration …
```

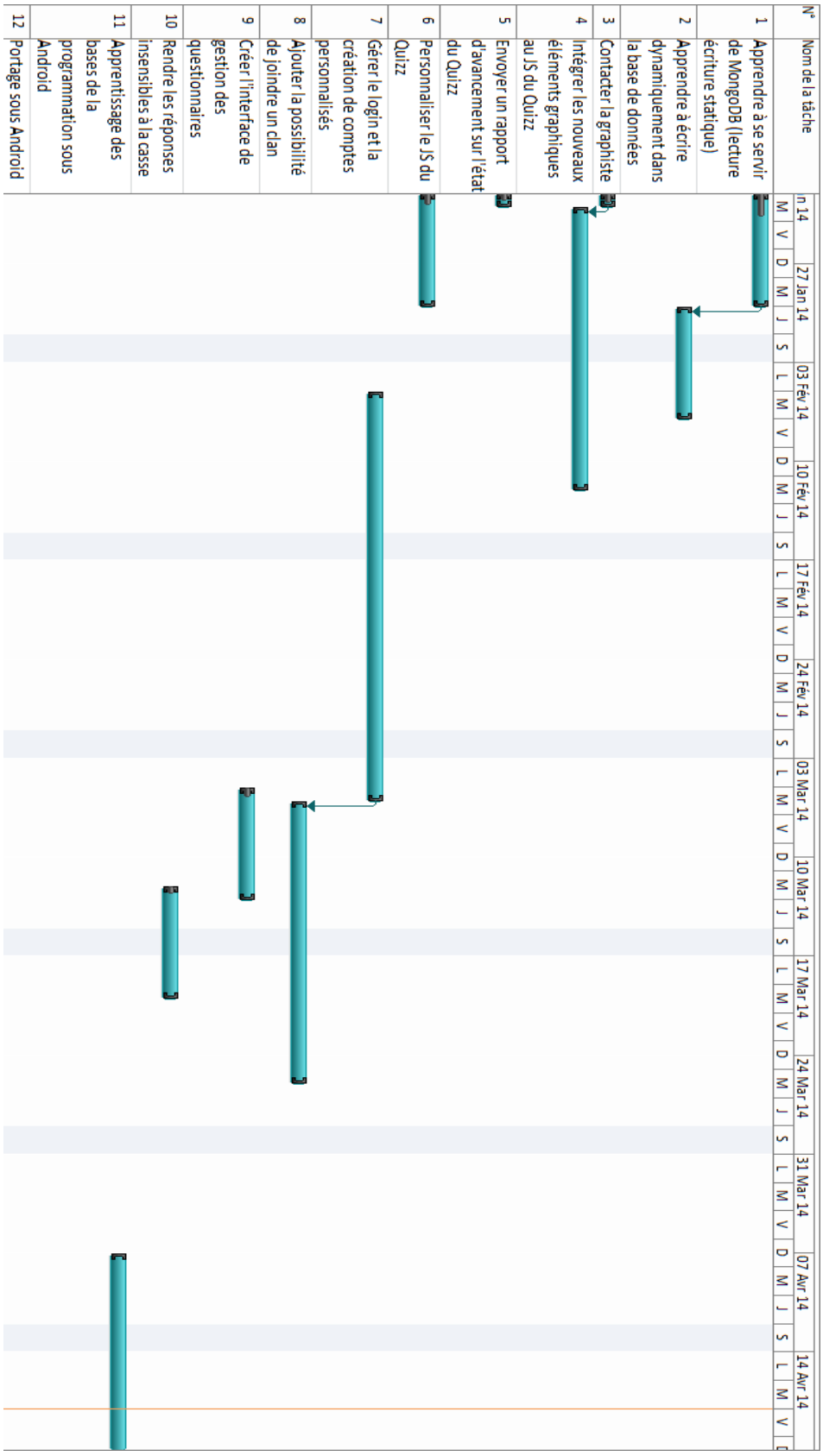
Conclusion

Le projet Quizz était ambitieux. En s'inspirant du site de Tout le monde veut prendre sa place, il était dès le départ certain que ce serait un projet sur le long terme, car d'une grande complexité, et avec des possibilités d'améliorations nombreuses. La structure logique du site a pu être réalisée, ainsi qu'une première esquisse d'un des jeux souhaités, le quizz de révision en quelques minutes. Cette ossature pourra être reprise à l'avenir, grâce à la facilité d'utilisation de PHP, ainsi que les nombreux commentaires présents dans le code, et améliorée par une intégration d'une interface graphique plus développée, ainsi que d'autres jeux, et pourquoi pas l'affrontement de joueurs en différé, comme imaginé originalement. Les problématiques rencontrées, conception d'un jeu, communication d'un site avec une base de données, architecture d'un site complexe, auraient facilement pu être traitées par une équipe plus nombreuse, d'au moins deux personnes. L'indépendance de nombre de tâches aurait permis un gain de temps non négligeable, et cette organisation pourrait être améliorée à l'avenir. Le portage sur Android et iOS, comme cela avait été évoqué nécessiterait une troisième personne rien que pour cette tâche. Une meilleure coopération entre les différents partis pourrait être profitable, notamment sur la répartition des tâches, pour éviter les malentendus, en organisant par exemple au moins une rencontre avec l'intégralité des membres du projet.

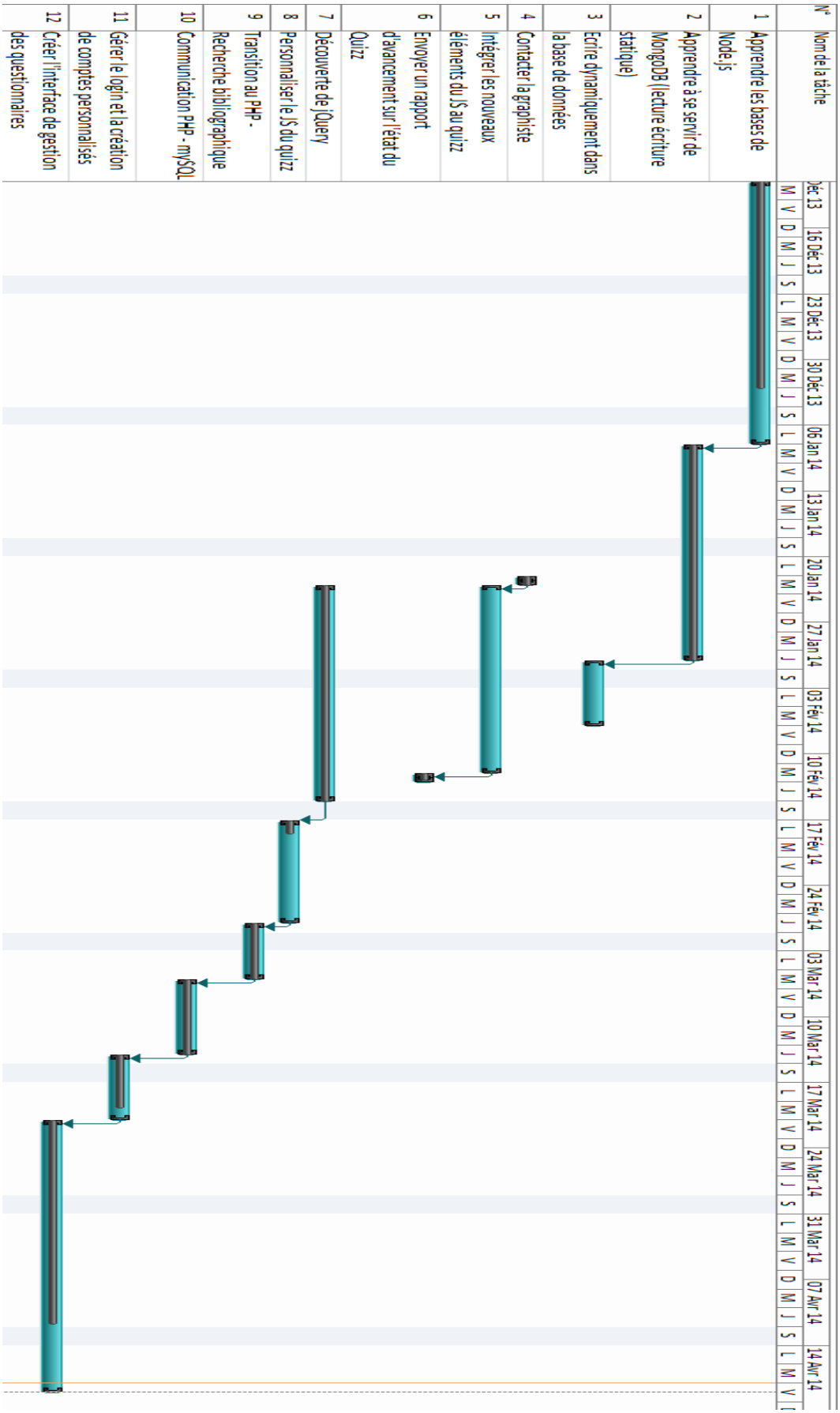
D'un point de vue personnel, ce projet a été l'occasion de traiter le même problème avec deux outils différents, et voir lequel était le plus approprié à chaque situation. Souvent PHP s'est avéré une solution de facilité, mais pas forcément la plus performante. Alors qu'il est demandé à chaque élève ingénieur de savoir être force de proposition dans son domaine, j'ai trouvé cet exercice enrichissant et me permettra à l'avenir d'aborder un projet du même type avec un panel de solution plus varié en tête.

Annexes :

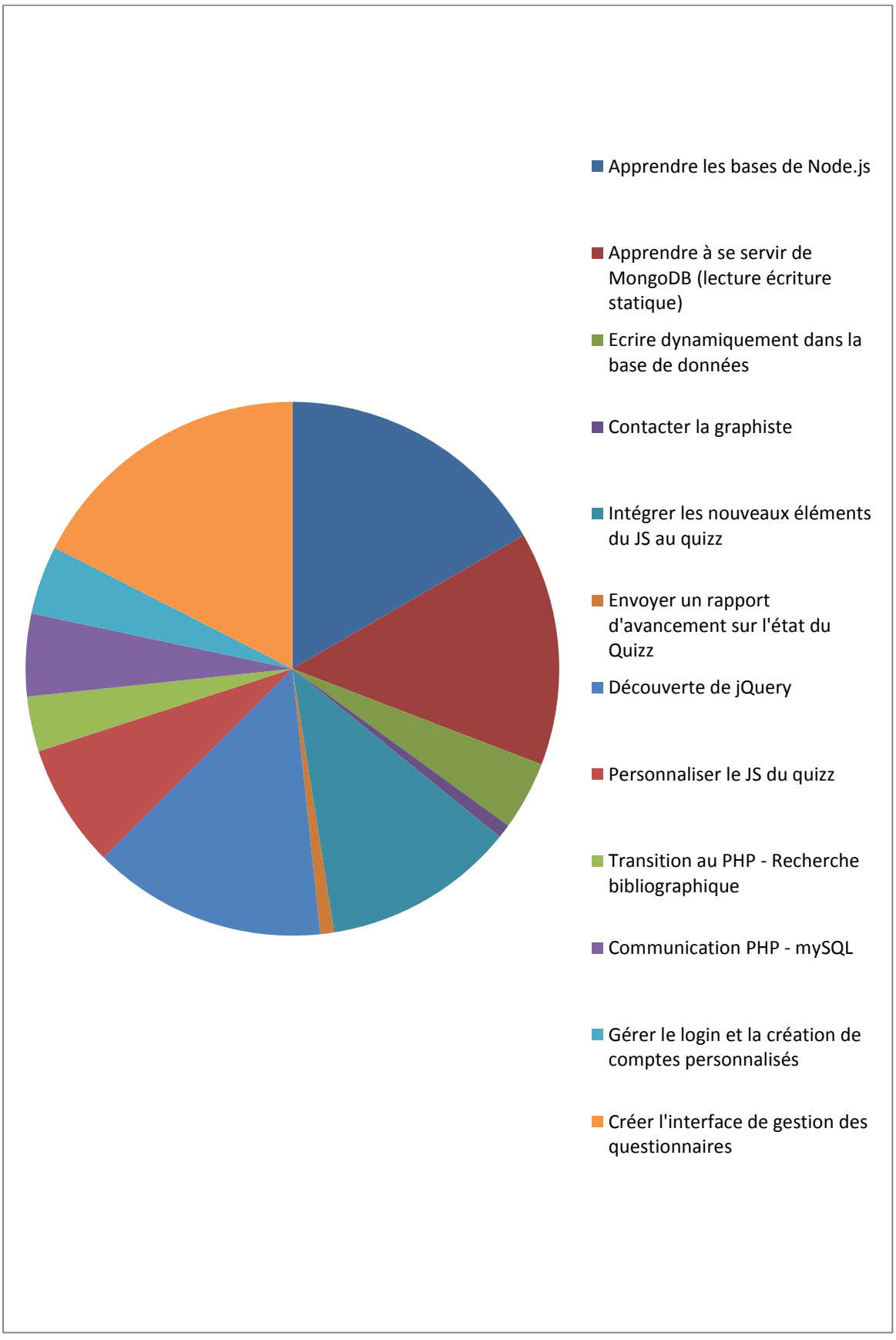
Annexe 1 : Diagramme de Gantt, première version



Annexe 2 : Gantt du travail réellement réalisé durant ce projet :



Annexe3 : Diagramme circulaire de la répartition des tâches en fonction du temps de projet total



Bibliographie

[1] *PHP 5 avancé* de Eric Daspet et Cyril Pierre De Geyer, aux éditions Eyrolles

Webographie

[1] http://fr.openclassrooms.com/informatique/cours/dynamisez-vos-sites-web-avec-javascript/introduction-au-javascript

[2] http://fr.openclassrooms.com/informatique/cours/des-applications-ultra-rapides-avec-node-js

[3] https://developer.mozilla.org/fr/docs/JavaScript/R%C3%A9f%C3%A9rence_JavaScript

[4] http://atinux.developpez.com/tutoriels/javascript/mongodb-nodejs-mongoose/

[5] http://fr.openclassrooms.com/informatique/cours/des-applications-ultra-rapides-avec-node-js/socket-io-passez-au-temps-reel

[6] http://fr.openclassrooms.com/informatique/cours/simplifiez-vos-developpements-javascript-avec-jquery/attendre-la-disponibilite-du-dom

[7] http://learn.jquery.com/using-jquery-core/working-with-selections/

[8] http://www.gilbert-pernot.fr/apprendre_php.html#systeme_securise

Projet réalisé par : Guillaume Fache

Projet encadré par : Mehdi Lhommeau

Résumé

Le GTE de Nantes souhaitait concevoir un site de révision ludique pour leurs étudiants, avec par exemple de petits quizz. Ce projet est important dans la mesure où il permet de faciliter le succès des étudiants de la formation. Il était prévu d'être conçu en associant Node.js à une base de données MongoDB. Suite à une mauvaise communication, le projet a dû être repris depuis le départ en PHP et avec une base de données mySQL, et a abouti à un site permettant la création de compte joueur, l'accès à un quizz de quatre questions qui donne les corrections, ainsi qu'une page administrateur permettant la modification de la banque de questions. Ce travail pourra être repris plus tard pour pousser plus loin la complexité du jeu et l'aspect visuel de l'ensemble.

Mots-clés : site web, développement, gestion de projet, PHP, Node.js, mySQL, base de données

Summary :

The GTE of Nantes decided they needed a website for their students to learn and study, using tools they understand and like. This project is rather important because it will make students success easier. The website had to be created with Node.js, a relatively new language that has many advantages for the kind of use that was intended. After a few weeks, the problem of where the site would be hosted was solved by deciding it would be on an external server provider. The choosing of the host was poorly managed, and resulted in an incompatibility between the work completed and the storage possibilities. As a result, the project had to be done over from the start using a new language, usable with the site requirements: PHP 5. The website is still at an early state yet, and many features are yet to be created, but a bone structure has been created, and could be useful for others to end the project later on. A login system is already working, leading to a self-correcting quizz that can tell you what your mistakes are, and what the correct answer is. This work will possibly be continued next year, to improve the game's complexity and the graphic aspect of the website.

Keywords : Website, development, project management, PHP, database, Node.js