

Rapport de Projet – EI4 AGI

Webquizz : Un site web éducatif de quizz

Projet réalisé par :

- Florian BAPPEL
- Antoine DE JESUS
- Pierre MARTIN

Projet encadré par :

- Mehdi LHOMMEAU
- Marc LENOIR
- Dominique EVROT



ISTIA – Université d'Angers

Remerciements

Nous tenons tout d'abord à remercier notre tuteur, M. Lhommeau pour son encadrement, son support et la confiance qu'il nous a accordés pour réaliser ce projet.

Nous remercions également les professeurs du IUT GTE de Nantes pour nous avoir confié la réalisation de cet outil pédagogique.

Nous tenons à remercier également l'ISTIA et son équipe pédagogique, et plus particulièrement M. Guimbail pour son aide sur la gestion de projet.

Table des matières

Remerciements.....	1
Introduction	3
I - Présentation du projet :.....	4
1 – Contexte.....	4
2 - Fonctionnalités désirées.....	5
3 - Organisation, phases du projet.....	6
3.1 - Le cahier des charges.....	6
3.2 - Gantt.....	10
3.3 - SCRUM	11
3.4 - Git.....	12
II - Technologies utilisées.....	13
2 - Couche base de données : MongoDB.....	16
3 - Couche serveur : NodeJS / Express / Mongoose	17
4 - Couche client/présentation : HTML 5 / CSS 3 / AngularJS / Bootstrap	20
III - Résultat	22
1 - Travail réalisé.....	22
1.1 - Page d'accueil	22
1.2 - Page de listing des quizz.	24
1.3 - Créer / Éditer un quizz	25
1.4 - La page Jouer.....	26
2 - Nos difficultés.....	29
3 - Amélioration/continuation	29
Conclusion :	30
Webographie :.....	31
Bibliographie :.....	31

Introduction

Nous avons tous eu recours à des méthodes d'apprentissage classique (CM, TD, TP) mais de nouvelles approches apparaissent. C'est dans cette optique que des professeurs de l'IUT GTE de Nantes ont voulu développer un nouvel outil d'apprentissage. Ils ont donc fait appel à l'ISTIA afin de réaliser un site web de quizz permettant de tester les connaissances des étudiants et d'apprendre différemment.

Après avoir découvert tous les projets mis à notre disposition, nous avons choisi de réaliser celui-ci. Ce projet nous a tout de suite intéressés de par les technologies qui pouvaient être mises en place. Celles-ci sont très récentes et de plus en plus appréciées par les professionnels.

Tout d'abord, nous présenterons le projet qui nous a été confié en détaillant le cahier des charges et les différentes phases nous avons mis en place pour le mener à bien.

Dans un deuxième temps, nous présenterons l'architecture globale de l'application ainsi que les différentes technologies que nous avons utilisées.

Pour finir, nous présenterons l'état final de notre projet avec les problèmes rencontrés et les possibles améliorations.

I - Présentation du projet :

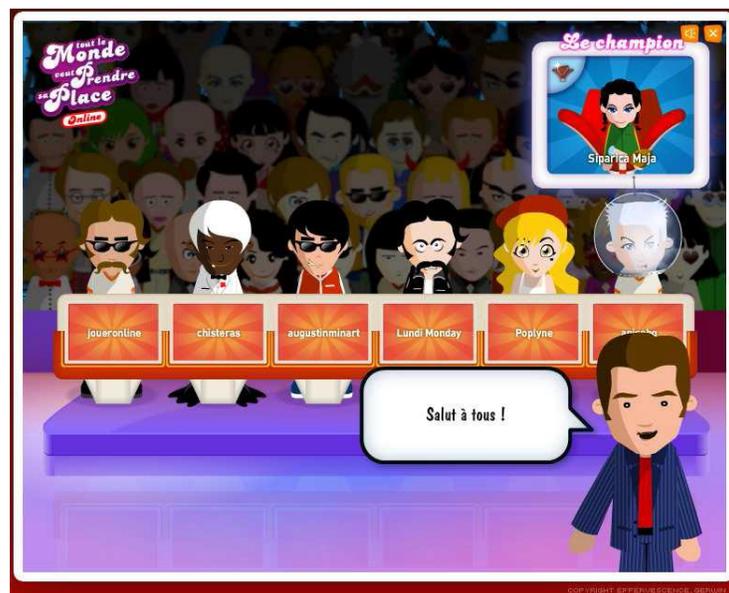
Dans cette partie, nous allons tout d'abord vous présenter dans quel contexte nous avons réalisé ce projet. Ensuite nous allons détailler quelles étaient les attentes pour ce site web. Enfin nous allons montrer comment nous avons mis à profit les cours de gestion de projet afin d'encadrer le nôtre.

1 - Contexte

Nous avons réalisé ce site web dans le cadre des projets tutorés d'EI4. Ces projets ont pour but d'appliquer les connaissances que nous avons apprises durant notre formation. Ils ont également pour but de nous exercer au travail d'équipe et à la gestion de projet.

Dans notre cas ce projet nous a également permis de découvrir de nouvelles technologies dynamiques de la programmation web.

Ce projet nous a été proposé par des professeurs de l'IUT GTE de Nantes. Ils souhaitaient que l'on réalise un site web de jeu éducatif. Ce site devait leur permettre de tester leurs connaissances sous forme de quizz en ligne. Ces quizz devaient être interactifs et s'inspirer du jeu "Tout Le Monde Veut Prendre Sa Place".



2 - Fonctionnalités désirées

En plus de sa fonctionnalité de base, qui était de proposer des quizz éducatifs. Le site web devait également intégrer les fonctionnalités suivantes :

- Gestion d'une communauté de joueurs :

Le site web devait permettre de gérer une communauté de joueurs. Les étudiants devaient se connecter au site afin d'avoir accès à leur page de profil, ainsi qu'à des quizz adaptés à leur formation.

- Gestion de classement, historique et scores :

Le site devait également garder un historique des résultats obtenus sur les quizz par les étudiants. Cet historique doit permettre d'obtenir un classement sur chaque quizz. Il permet également aux enseignants de pouvoir suivre la progression des étudiants.

- Interface administrateur pour la gestion des questionnaires :

Afin de pouvoir créer et modifier les quizz, le site devait posséder une interface utilisateur facile d'utilisation. Les professeurs devaient pouvoir créer de façon intuitive les quizz qu'ils proposaient aux étudiants.

- Application smartphone/tablette :

Le site web devait également pouvoir fonctionner sur smartphone et tablette. Pour cela nous pouvons réaliser une application ou bien rendre le site compatible et adapté à l'affichage d'un écran de smartphone ou tablette.

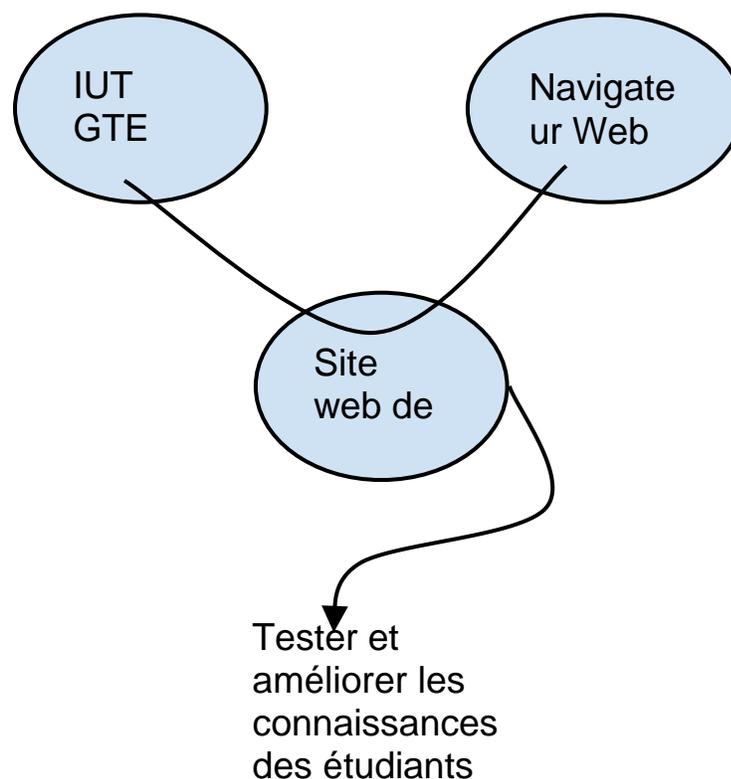
3 - Organisation, phases du projet

Afin d'encadrer notre projet, nous avons mis à profit nos cours de gestion de projet. Nous avons d'abord commencé par réaliser un cahier des charges détaillé afin de fixer les contours du travail à réaliser.

Nous avons ensuite réalisé un planning détaillé de notre projet. Ce planning nous a permis d'identifier les différentes tâches à réaliser. Il nous a également permis de nous fixer une contrainte de temps sur chacune de ces tâches et de les répartir entre les membres du groupe.

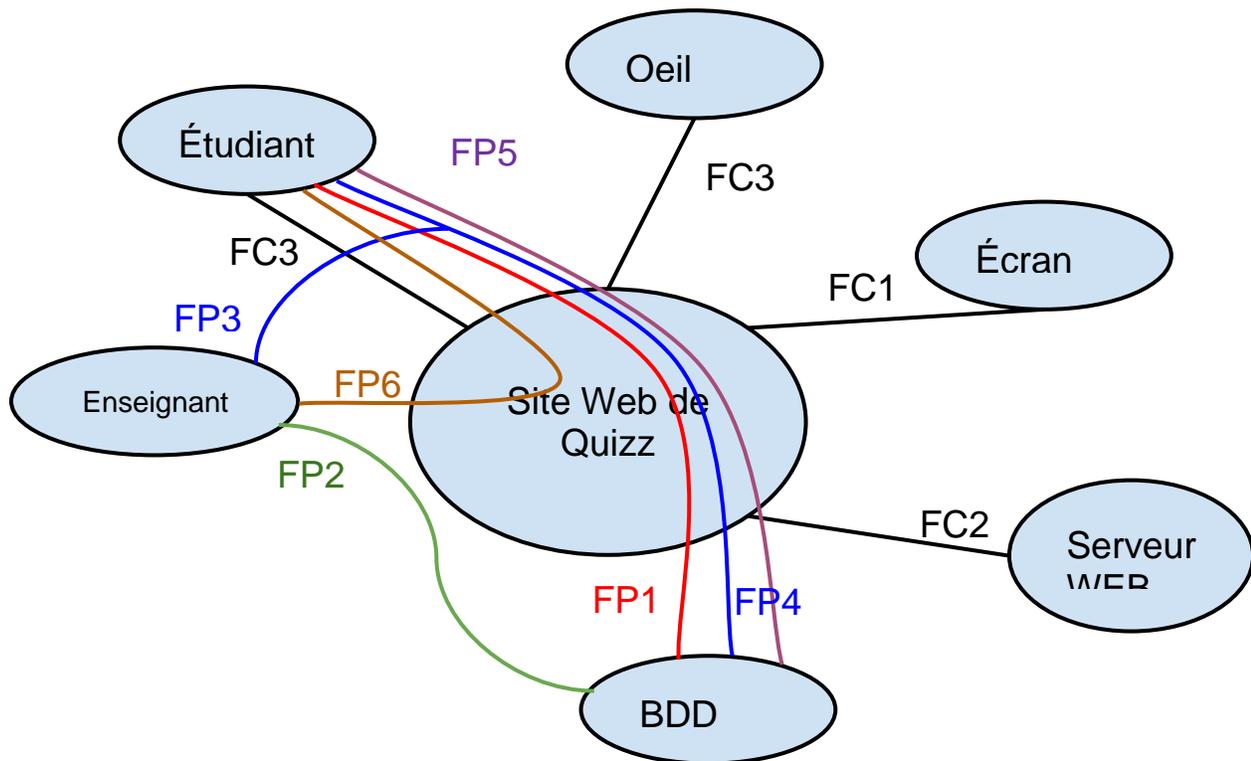
Ensuite nous avons choisi d'utiliser la méthode Agile SCRUM pour encadrer le déroulement de notre projet. Enfin, afin d'encadrer le développement de notre site web nous avons choisi de travailler avec Git et Github.

3.1 - Le cahier des charges



Nous avons tout d'abord réalisé une bête à corne afin de définir le but de notre site web.

Ensuite nous avons réalisé un diagramme pieuvre afin de définir les fonctionnalités et les contraintes de notre site web.



FP1 : Permettre aux étudiants de répondre au Quizz stocké dans la BDD
 FP2 : Permettre aux enseignants de créer, supprimer et modifier les Quizz dans la BDD
 FP3 : Permettre aux étudiants/enseignants de se connecter à partir des identifiants stockés dans la BDD
 FP4 : Stocker les résultats des étudiants sur la BDD
 FP5 : Afficher des historiques/classements des étudiants à partir de la BDD
 FP6 : Fournir un espace de discussion pour les étudiants/enseignants

FC1 : Savoir s'adapter à l'écran
 FC2 : Être hébergé sur un serveur Web
 FC3 : Être esthétique
 FC4 : Proposer des quizz adaptés aux étudiants
 FC5 : Pouvoir ajouter une contrainte de temps pour les quizz

Ensuite nous avons caractérisé nos milieux extérieurs. Cela nous a permis de définir des critères sur nos fonctions, ainsi que leurs niveaux. Ces critères ont pour but de vérifier si notre résultat final respecte notre cahier des charges.

Milieu environnant SV1	Caractéristiques
Étudiants	<ul style="list-style-type: none"> - Année d'étude - spécialité
Enseignant	<ul style="list-style-type: none"> - Age - Matière enseignée
BDD	<ul style="list-style-type: none"> - nb de Quizz disponible - difficulté des Quizz - type de Quizz - Nb utilisateur - Sécurité - Capacité
Serveur WEB hébergeur	<ul style="list-style-type: none"> - Temps de réponse - Technologie gérée - Coût - Sauvegarde des données (back-up)
Écran Utilisateur	<ul style="list-style-type: none"> - Taille du Support (smartphone, tablette, écran)
Œil	<ul style="list-style-type: none"> - Problème de vue (daltonisme) - esthétique

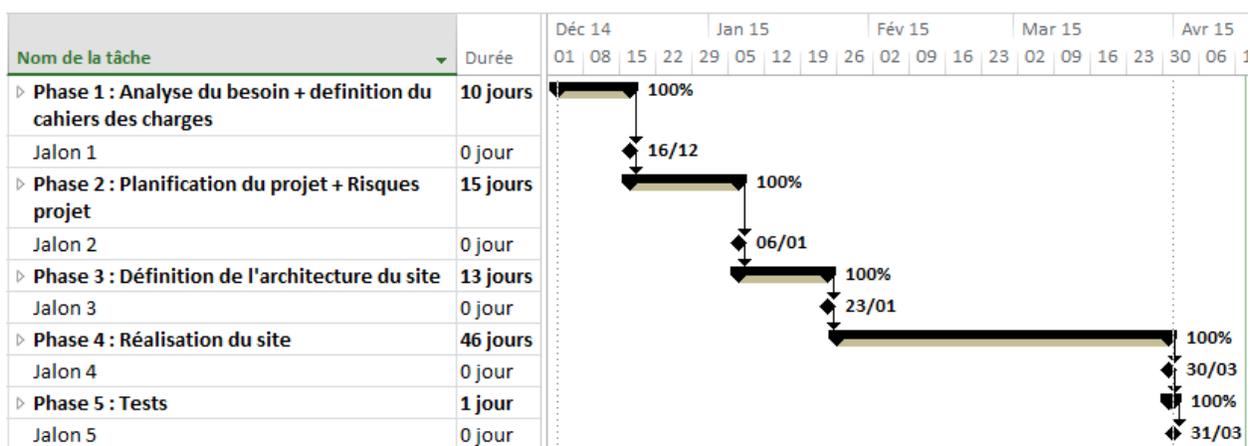
Fonctions	Critères	Niveaux
FP1 : Permettre aux étudiants de répondre aux Quizz stockés dans la BDD	<ul style="list-style-type: none"> - Ergonomie - Temps de réponse/traitement - Visibilité de l'avancement - Fiabilité du traitement des réponses - Visibilité des résultats 	<ul style="list-style-type: none"> • Utilisation sans notice • < 1s • Avancement visible en permanence • Aucune erreur • Affichage des résultats
FP2 : Permettre aux enseignants de créer, supprimer et modifier les Quizz dans la BDD	<ul style="list-style-type: none"> - Ergonomie - Temps de réponse - Visualisation des modifications 	<ul style="list-style-type: none"> • Utilisation sans notice • < 1s • En direct
FP3 : Permettre aux étudiants/enseignants de se connecter à partir des identifiants stockés dans la BDD	<ul style="list-style-type: none"> - Sécurité - Modification des infos perso - Ergonomie - Fiabilité 	<ul style="list-style-type: none"> • Haute • Possible • Utilisation sans notice • Pas de pertes
FP4 : Stocker les résultats des étudiants sur la BDD	<ul style="list-style-type: none"> - Sécurité (confidentialité) - Rapidité - Fiabilité 	<ul style="list-style-type: none"> • Haute • < 1s • Pas de pertes
FP5 : Afficher des historiques/classements des étudiants à partir de la BDD	<ul style="list-style-type: none"> - Ergonomie - Tri - Recherche 	<ul style="list-style-type: none"> • Utilisation sans notice, affichage clair • Possible par colonne • Possible
FP6 : Fournir un espace de discussion pour les étudiants/enseignants	<ul style="list-style-type: none"> - Ergonomie - Rapidité - Sécurité 	<ul style="list-style-type: none"> • Utilisation sans notice, lisibilité • Quasiment instantané • Haute
FC1 : Savoir s'adapter à l'écran	<ul style="list-style-type: none"> - Adaptabilité - Rapidité 	<ul style="list-style-type: none"> • Sur tous types d'écrans • Immédiat
FC2 : Être hébergé sur un serveur Web	<ul style="list-style-type: none"> - Compatibilité 	<ul style="list-style-type: none"> • Totale
FC3 : Être esthétique	<ul style="list-style-type: none"> - Attractivité - Adaptation au daltonisme 	<ul style="list-style-type: none"> • Haute • Correcte
FC4 : Proposer des quizz adaptés aux étudiants	<ul style="list-style-type: none"> - Difficulté - Cohérence 	<ul style="list-style-type: none"> • Fourchette adaptée à l'étudiant • En accord avec la/les spécialité(s) de l'étudiant
FC5 : Pouvoir ajouter une contrainte de temps pour les quizz	<ul style="list-style-type: none"> - Facilité - Ergonomie 	<ul style="list-style-type: none"> • Haute • Utilisation sans notice

3.2 - Gantt

Pour réaliser notre Gantt, nous avons d'abord commencé par identifier les différentes tâches à faire pour réaliser ce projet. Ensuite nous avons défini quelle personne devait réaliser chacune des tâches.

C'est ainsi que nous nous sommes réparti le travail à réaliser en fonction des compétences de chacun. Après nous avons estimé le temps nécessaire à la réalisation des tâches.

Enfin nous avons défini les dépendances entre celles-ci. C'est ainsi que nous avons réalisé notre Gantt, qui nous a permis de planifier notre projet.

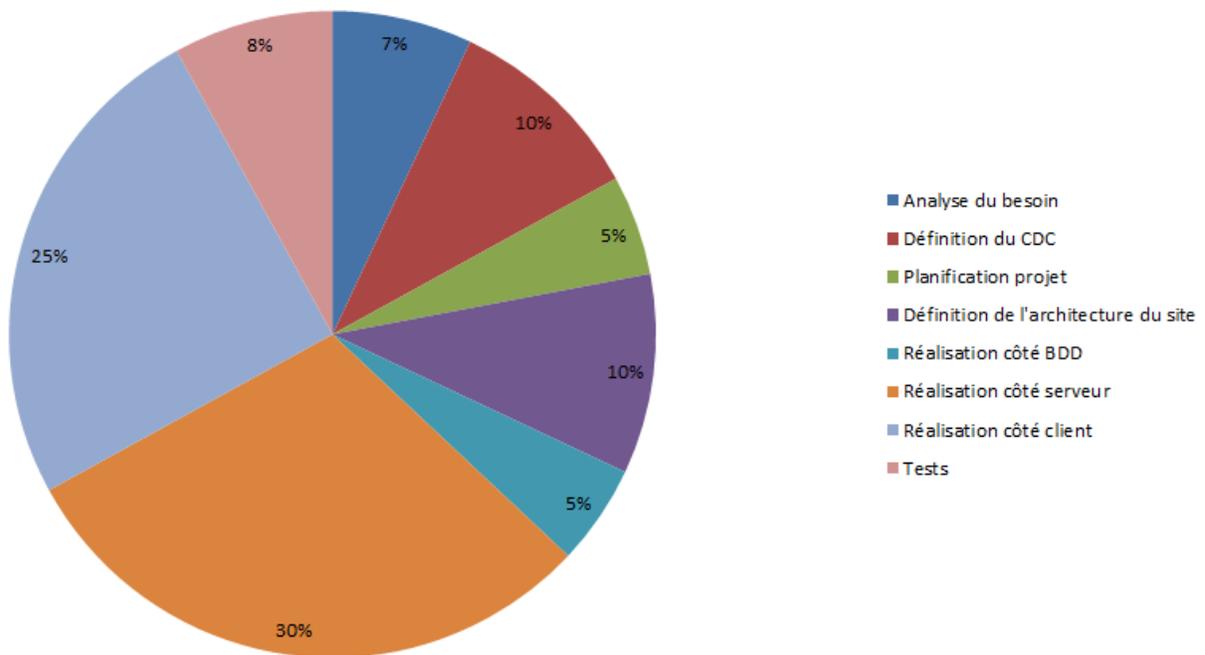


Ici les grandes phases apparaissent. Par souci de place, le Gantt complet est accessible à l'adresse suivante :

<https://drive.google.com/file/d/0B8lWt9u11FA4ZTBHbUcxaE5KcUU/view?usp=sharing>

La répartition du temps de travail a au final été la suivante :

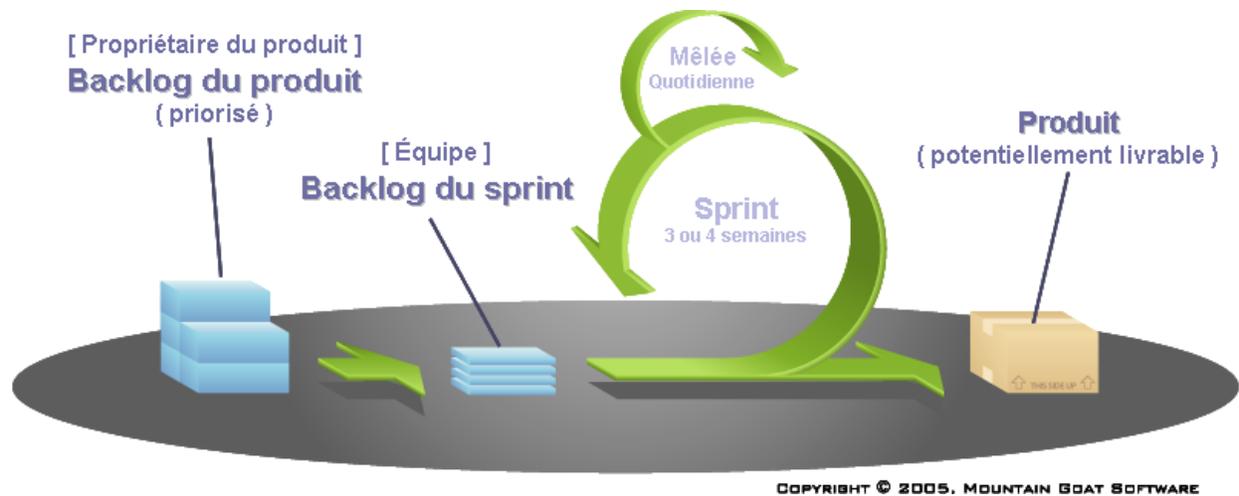
Répartition du temps de travail



3.3 - SCRUM

SCRUM est une méthode d'organisation et de développement de produit dite agile. Cette méthode s'appuie sur le découpage d'un projet en portions de temps appelées "sprint".

Chaque sprint commence par une planification et se termine par une démonstration de ce qui a été réalisé. Pendant les sprints on réalise des mêlées quotidiennes. C'est une réunion de planification qui permet de mettre au point la coordination sur les tâches en cours et les problèmes rencontrés.



Nous pensions mettre en place cette méthode au début, mais comme la communication avec l'IUT s'est révélée plus difficile que prévu, nous avons à la place envoyé et demandé des conseils à M. Lhommeau chaque semaine.

3.4 - Git

Pour encadrer le développement de notre projet et afin de pouvoir gérer nos codes nous avons choisi d'utiliser Git et Github.

Git est un logiciel de gestion de version (Versioning) créé par Linus Torvalds. Il permet à plusieurs développeurs de travailler de façon collaborative sur les mêmes sources.



Github est un site web fournissant des repositories (dépôts de projets) hébergés. Ce site permet de voir facilement les dernières tâches effectuées. Voici celui qui a hébergé le projet : <https://github.com/Kiruchi/WebQuizz>



II - Technologies utilisées

Pour ce projet, il était recommandé d'utiliser du NodeJS, mais les spécifications techniques n'étaient pas réellement définies. Nous avons ainsi pu proposer une architecture que nous pensons à même de répondre au besoin défini.

Notre choix s'est porté sur une architecture en couches de type MEAN. Ce chapitre expliquera les spécificités de celle-ci, et les technologies/librairies que nous avons utilisées en complément.

1 - Architecture globale

L'architecture utilisée est de type **MEAN**, ce qui veut dire **M**ongoDB **A**ngularJS **E**xpress **N**odeJS. On parle de "stack" (pile, architecture) MEAN par opposition au "stack" **LAMP** classique : **L**inux **A**pache **M**ySQL **P**HP.

L'architecture MEAN a pour spécificité d'utiliser du JavaScript tout au long de la chaîne. Ceci a plusieurs avantages :

- Si un développeur connaît le JavaScript, il peut comprendre du code aussi bien côté client que côté serveur, ce qui rend le code global plus compréhensible par n'importe quel membre d'une équipe.
- Les informations transmises par la base de données gardent le même format jusqu'à la couche présentation, ici du JSON, ce qui évite d'avoir à manipuler ces données.
- Pour des applications de petite à moyenne taille, MEAN est réputé pour être plus rapide et plus réactif que LAMP.



Dans notre cas, nous avons choisi une implémentation de MEAN baptisée MEAN.js.

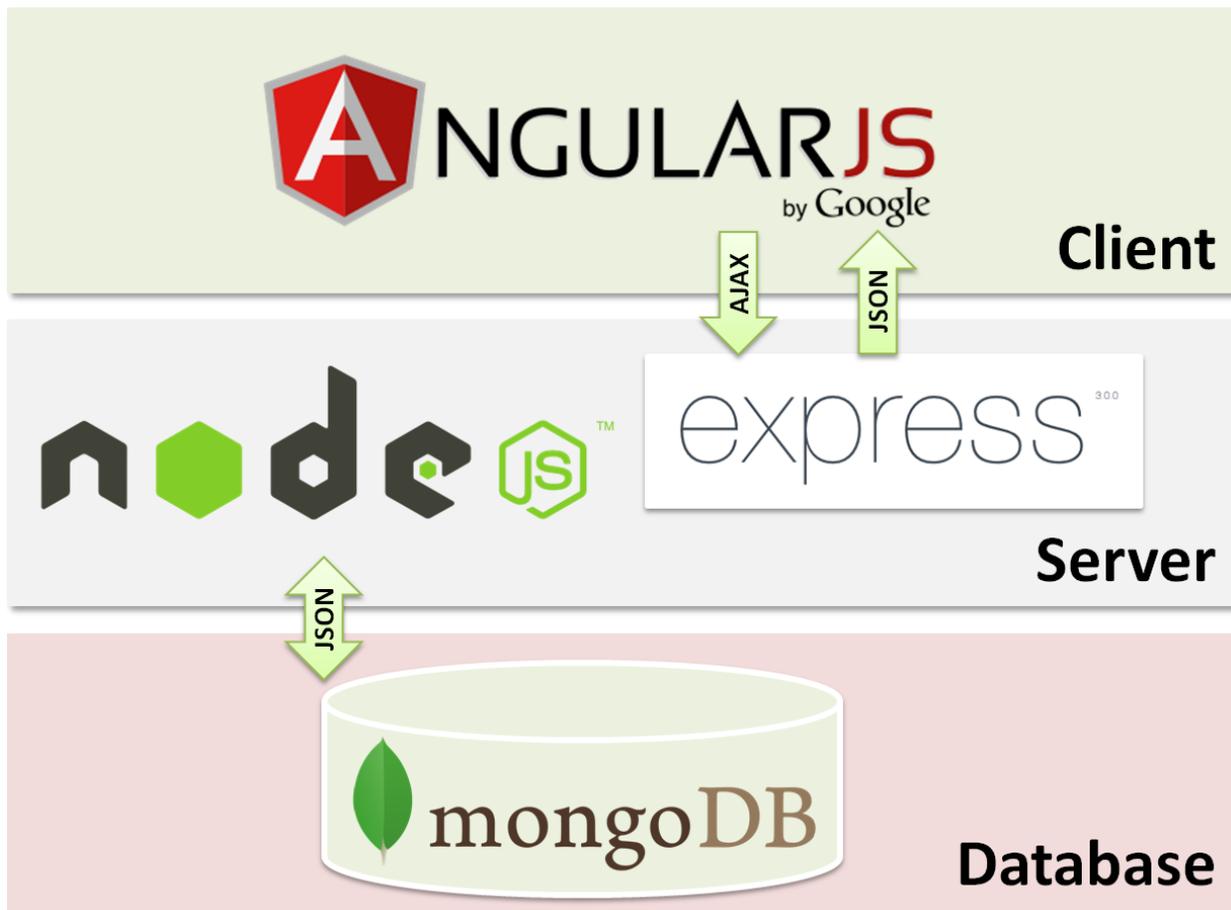
MEAN est une architecture en couches 3-tiers.

On distingue donc 3 couches :

- une couche base de données qui est MongoDB
- une couche serveur gérée par NodeJS/Express/Mongoose
- une couche client / présentation gérée par HTML 5 / CSS 3 / AngularJS / Bootstrap

Le chapitre II détaille ces technologies.

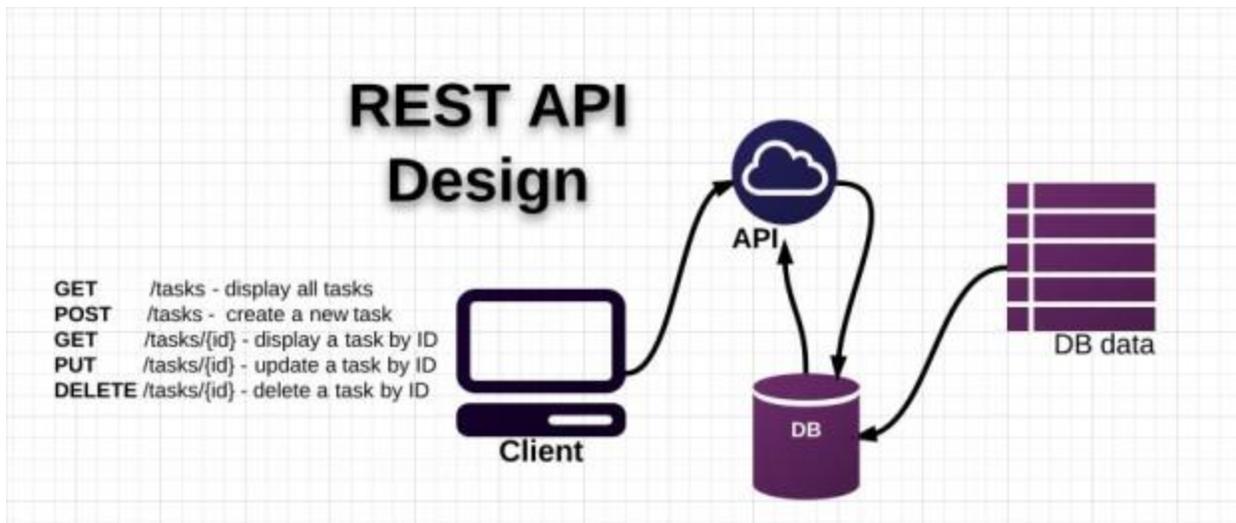
Ces couches sont organisées comme ceci :



Source : <http://joaopsilva.github.io/talks/End-to-End-JavaScript-with-the-MEAN-Stack/>

Flux de données provenant de la base de données :

MongoDB transfère ses informations en JSON à NodeJS, en passant par l'ORM (Object-Relational Mapping) Mongoose. Node.JS fournit alors des services web REST permettant à n'importe quel client de récupérer ces données.



Source : <https://github.com/nelsonic/learn-hapi>

Exemple : On désire créer une application Android pour jouer aux quizz. Vu que Node.JS fournit des services web, il n'est pas nécessaire pour l'application de tout refaire.

Par exemple, un appel GET à <http://www.monapplication.com/quizz/numeroDuQuizz> pourra renvoyer directement les informations d'un quizz en JSON.

L'approche service web permet de développer la couche serveur une seule fois, et de la réutiliser sur n'importe quel type de site web ou application.

Flux des pages web :

Node.JS utilise Express, qui est une implémentation de serveur web. Express s'occupe d'envoyer les pages demandées au client. Celles-ci sont alors affichées grâce à HTML5 et CSS3. On utilise également un Framework CSS3 : Bootstrap. Celui-ci permet de rendre le site responsive et accélère l'intégration CSS (voir partie II.4).

AngularJS sert pour le rendu dynamique et les appels aux services web. Il permet de faire des appels asynchrones, ce qui évite de devoir rafraîchir la page.

2 - Couche base de données : MongoDB

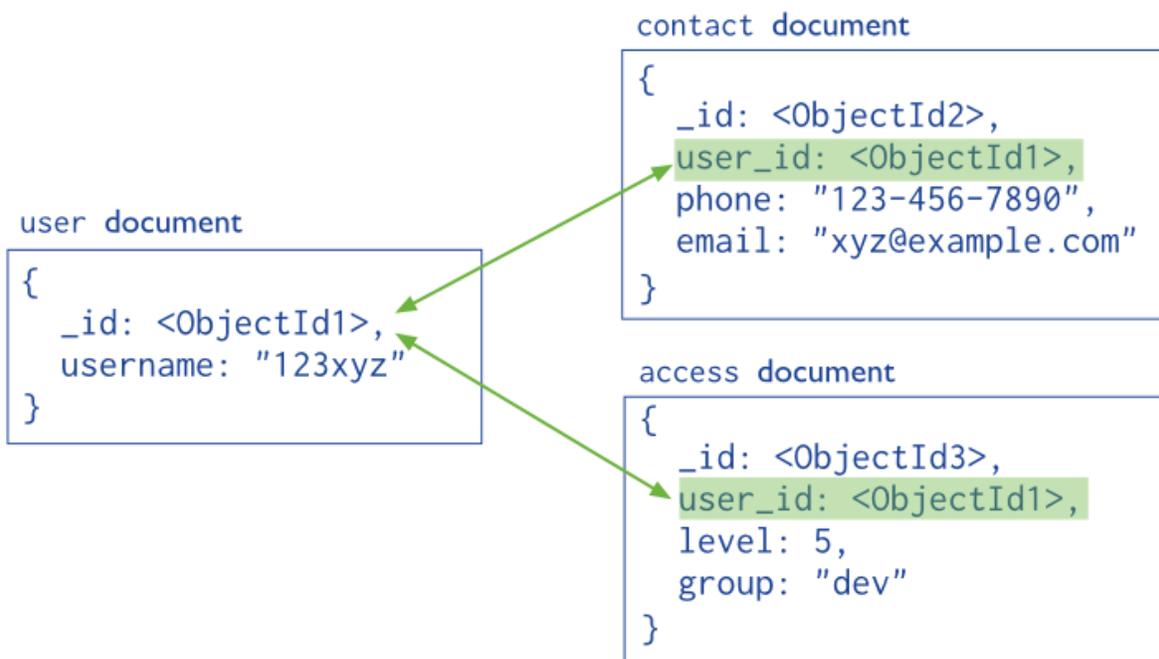


{ name: mongo, type: DB }

MongoDB est une base de données NoSQL, et qui n'est donc pas relationnelle. Elle est composée de collections et non pas de tables. Chaque entrée est appelée document et a une syntaxe JSON et un identifiant.

Voici un exemple avec trois documents. Les documents « contact » et « access » ont tous les deux une relation avec le document « user ».

MongoDB rend la formulation de requêtes très simple et très intuitif :



```
db.users.find(
  { age: { $gt: 18 } },
  { name: 1, address: 1 }
).limit(5)
```

← collection
← query criteria
← projection
← cursor modifier

Ici on recherche dans la collection users des users qui ont plus de 18 ans (\$gt veut dire greater than). On ne veut garder que leur nom et adresse, et on ne veut que les 5 premiers.

Pour plus d'informations, voir <http://docs.mongodb.org/manual/>

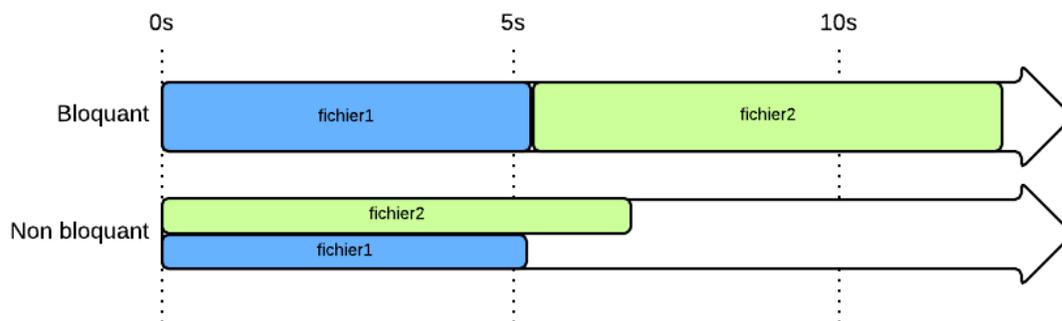
3 - Couche serveur : NodeJS / Express / Mongoose

La couche serveur est composée de trois éléments distincts :

- Node.JS
- Express
- Mongoose

Node.JS :

Node.JS est une plateforme serveur en JavaScript utilisant la machine virtuelle V8 de Google. Elle permet d'exécuter du code JavaScript en étant non-bloquant. Plusieurs tâches peuvent ainsi être effectuées en parallèles. Voici un schéma explicatif :



Node.JS est très vaste et beaucoup de modules existent étendant ses usages. Pour plus d'informations, voir <https://nodejs.org/documentation/>

Express :

Express est un Framework d'application web. C'est un module pour Node.JS qui permet de simplifier l'utilisation d'un serveur web sous Node.js.

Exemple pour un Hello World :

```
var express = require('express');
var app = express();

app.get('/', function (req, res) {
  res.send('Hello World!');
});

var server = app.listen(3000, function () {
  var host = server.address().address;
  var port = server.address().port;
  console.log('Example app listening at http://%s:%s', host, port);
});
```

On peut avec Express servir des pages spécifiques à la demande d'une URL. C'est le principe d'une route. Dans l'exemple précédent la route était /.

Pour plus d'informations, voir <http://expressjs.com/>

Mongoose :

Mongoose est un ORM (Object-Relational Mapping), mais pour une base NoSQL, on appelle donc cela un ODM (Object-Document Mapping).



Mongoose fonctionne par "Schemas". On donne à Mongoose la forme que doit avoir notre document, les conditions à respecter, les conditions par défaut, et Mongoose se chargera de faire les vérifications.

Exemple :

```
// Create schema for blog post
var blogSchema = new mongoose.Schema({
  title: String,
  author: String,
  body: String,
  comments: [{ body: String, date: Date }],
  date: { type: Date, default: Date.now },
  hidden: Boolean,
  meta: {
    votes: Number,
    favs: Number
  }
});
```

Ici on a créé pour un blog un titre, un auteur, un corps, des commentaires, une date qui est par défaut le moment où est appelé le schéma, un booléen pour savoir si le blog est caché, et des métadonnées. On remarque pour les commentaires qu'on peut avoir un tableau comme champ, ce qui est très pratique.

Mongoose permet aussi de créer des requêtes plus simplement qu'avec MongoDB par défaut :

```
// find each person with a last name matching 'Ghost', selecting the `name` and
`occupation` fields
Person.findOne({ 'name.last': 'Ghost' }, 'name occupation', function (err, person) {
  if (err) return handleError(err);
  console.log('%s %s is a %s.', person.name.first, person.name.last,
person.occupation) // Space Ghost is a talk show host.
})
```

Pour plus d'informations, voir : <http://mongoosejs.com/docs/>

4 - Couche client/présentation : HTML 5 / CSS 3 / AngularJS / Bootstrap

Cette couche est celle que verra l'utilisateur sur son navigateur web ou mobile.

L'affichage se fait donc classiquement en HTML 5/CSS 3. Afin de simplifier le développement CSS, on utilise SASS qui permet d'utiliser entre autres des variables dans un fichier que l'on compilera en CSS.

On utilise également un Framework CSS permettant d'accélérer l'intégration et de faciliter le Responsiveness. Ce Framework s'appelle Bootstrap et est développé par Twitter.



Mais qu'est-ce que le responsive ?

Le responsive web design est le fait que le contenu d'une page web s'adapte à la taille d'écran utilisée. Cela permet de créer un unique site web qui sera adapté aussi bien pour un écran de PC fixe que pour un smartphone ou une tablette.

Bootstrap :

Bootstrap propose tout d'abord un système de grille qui permet de positionner les éléments CSS par rapport à celle-ci. La grille s'adapte ensuite automatiquement à l'écran utilisé.



Bootstrap fournit aussi des composants prédéfinis qui sont ensuite modifiables, comme des onglets, des barres de menus, des listes, etc...

Pour plus d'informations, voir <http://getbootstrap.com/>

Nous utilisons également la librairie UI-Bootstrap qui permet de gérer des composants Bootstrap en AngularJS au lieu de JQuery par défaut. Voir <https://angular-ui.github.io/bootstrap/>

AngularJS :

AngularJS, développé par Google, est un Framework JavaScript côté client. Il permet de rendre dynamique l'affichage d'une page web, mais également de faire des appels asynchrones à des services web.

AngularJS ajoute des balises ou des attributs au HTML, et interprète ensuite ceux-ci. Il est doté d'un système de double data-binding. C'est à dire que lorsque le contrôleur AngularJS modifie une variable, la variable est modifiée en direct sur la page web. Et si par exemple la variable est un champ texte, si l'utilisateur le modifie, la variable est modifiée côté contrôleur instantanément.

De plus, AngularJS est un des Frameworks les plus en vogue de ces dernières années. Porté par Google, beaucoup d'entreprises s'y sont intéressé.

De très bons exemples sont accessibles sur le site officiel d'AngularJS : <https://angularjs.org/>



III - Résultat

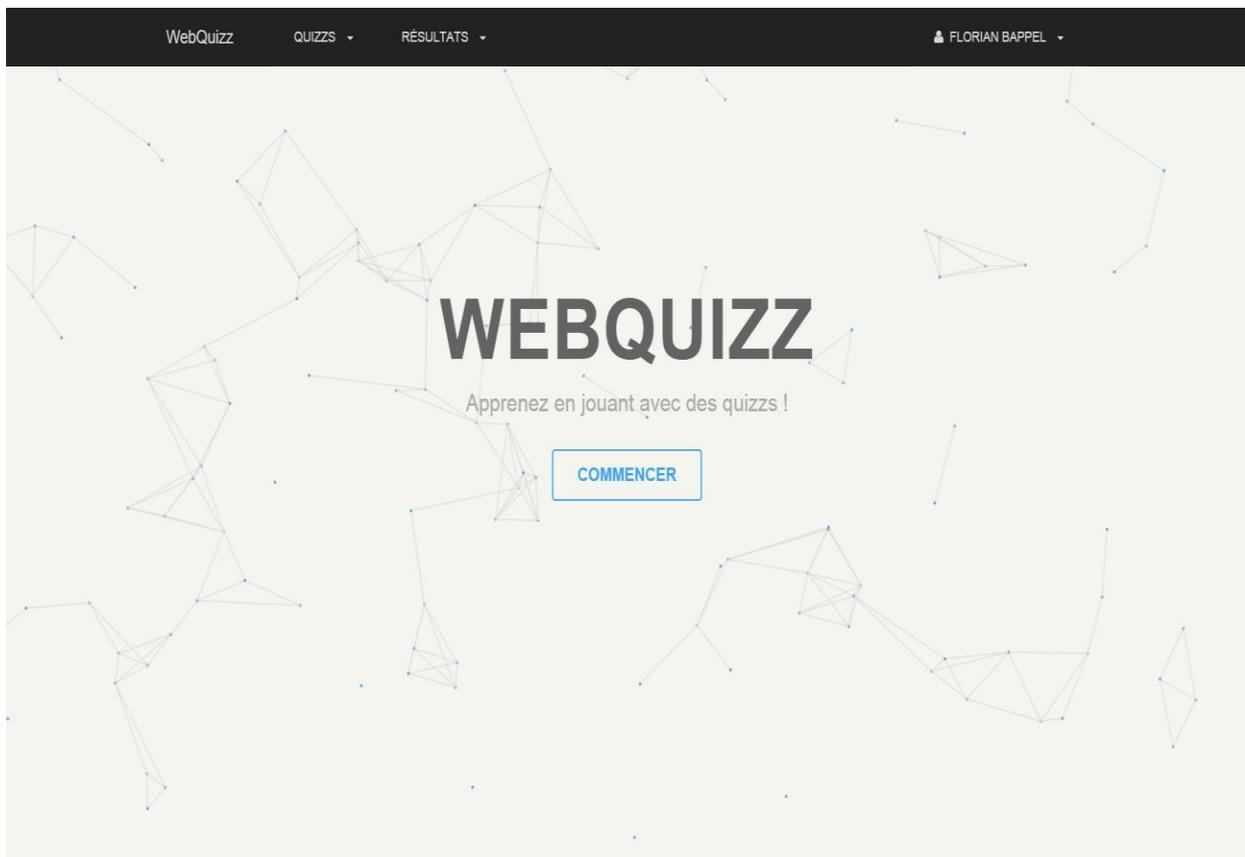
Dans cette partie, nous allons montrer les résultats que nous avons pu obtenir au cours de nos séances de projet tutoré (3 mois) et les problèmes que nous avons rencontrés. Nous terminerons par expliquer quelles améliorations sont possibles.

1 - Travail réalisé

Le site est composé de 5 grandes parties qui sont les suivantes:

1.1 - Page d'accueil

Cette page permet à l'utilisateur de se connecter ou de s'inscrire. Elle permet également d'accéder à la liste des quizz.



- Page de connexion / inscription

Comme le nom l'indique, ces 2 pages sont destinées à l'inscription de l'utilisateur ou à sa connexion si celui-ci possède déjà un compte.

La page de connexion permet de rentrer ses identifiants afin de pouvoir accéder aux parties sécurisées de l'application comme la validation de quizz ou l'accès aux résultats de ceux-ci.

Si l'utilisateur n'a pas de compte, il pourra se diriger vers la page d'inscription où il renseignera les informations demandées afin de créer un compte.

Ces 2 pages utilisent le middleware Passport intégré au Framework ExpressJS et les mots de passe sont chiffrés via un chiffrement Md5.

The image displays two screenshots of the WebQuizz application's user interface. The top screenshot shows the registration page, titled 'Inscrivez vous :'. It features a form with five input fields: 'Prénom', 'Nom', 'Email', 'Nom d'utilisateur (Nom sous lequel vous apparaîtrez)', and 'Mot de passe'. Below the fields are two buttons: 'S'inscrire' and 'Se connecter'. The bottom screenshot shows the login page, titled 'Connectez vous :'. It features a form with two input fields: 'Nom d'utilisateur' and 'Mot de passe'. Below the fields are two buttons: 'Se connecter' and 'S'inscrire', along with a link for 'Mot de passe oublié ?'. Both screenshots have a dark header with 'WebQuizz' on the left and 'INSCRIPTION' and 'CONNEXION' on the right.

1.2 - Page de listing des quizz.

Nom	Matière	Difficulté	Nombre de questions	Date de création	Auteur
test gte	tst	★★★★☆	2	Apr 9, 2015 à 11:46 AM	marc Lenoir
Tryo	Musique	★★★★☆	3	Apr 9, 2015 à 11:08 AM	Etienne Hatte
Anime	Anime	★★★★★	4	Apr 2, 2015 à 4:07 PM	Ketro *
CMDR Doux	SPAAAAAAAAAAAAAAAAACCCCEEEEEEE IIIIIII	★★★★★	3	Apr 2, 2015 à 3:35 PM	Florian Bappel
fezgez	zehezhzeh	★★★★☆	3	Apr 1, 2015 à 2:34 PM	Kiruchi Thorann
fzaaga	zafzaf	★★★★☆	3	Apr 1, 2015 à 12:33 PM	Kiruchi Thorann
fazfazfz	dazdad	★★★★★	4	Apr 1, 2015 à 12:17 PM	Kiruchi Thorann
gerherè	gerger	★★★★★	2	Apr 1, 2015 à 12:04 PM	Kiruchi Thorann
test	test	★★★★☆	2	Apr 1, 2015 à 10:48 AM	Antoine DeJesus
Quizz Ant-49	Ant-49	★★★★☆	2	Apr 1, 2015 à 10:17 AM	Antoine DeJesus

Cette page liste tous les quizz qui ont été créés et enregistrés dans la base de données. Ils sont affichés sous la forme d'un tableau dynamique où les différentes colonnes peuvent être triées à notre convenance. Ces différentes colonnes sont les suivantes : Nom, matière, difficulté, nombre de questions, date de création, auteur.

On peut également réaliser une recherche par mot clé qui nous affichera tous les quizz où ce mot apparaît (dans n'importe quel champ).

Ces différentes fonctionnalités sont gérées du côté client à l'aide d'AngularJS. Nous avons choisi cette solution pour alléger la charge du serveur.

Sur cette même page, différentes actions sont possibles dès lors que l'utilisateur est connecté. Ces actions sont les suivantes :

- Jouer
- Créer un quizz

Si l'utilisateur a déjà créé des quizz, 2 autres options vont apparaître sur les quizz qui lui appartiennent :

- Éditer un quizz
- Supprimer un quizz

Dans les paragraphes qui suivent, nous allons vous présenter les fonctionnalités précédemment citées:

1.3 - Créer / Éditer un quizz

Ces 2 fonctionnalités étant basées sur le même fonctionnement, nous les traiterons dans ce paragraphe. Ces pages sont composés de 2 sous-modules: un module information et un module questions/réponses :

- Le 1er module permet d'éditer les informations générales concernant le quizz.

The screenshot displays the 'Edition du Quizz : CMDR Doux' interface. At the top, there is a navigation bar with 'WebQuizz', 'QUIZZS', and 'RÉSULTATS' menus, and a user profile 'FLORIAN BAPPEL'. The main content area is titled 'Edition du Quizz : CMDR Doux'. It features several input fields and controls: a 'Titre' field containing 'CMDR Doux', a 'Description' text area with '3... 2... 1... Engage', a 'Difficulté' section with five stars, a 'Type' dropdown menu set to 'Classique', and a 'Matière' field containing 'SPAAAAAAAAAAAAAAAAACCCCEEEEEEE IIIIIII'. Below these, there is a 'Date de début' section with a date picker for '02-Apr-2015', a time picker for '15:27', and an 'Ajouter une date de fin' button. At the bottom, there is a blue 'Etape suivante >' button.

- Le 2ème module permet l'édition des questions ainsi que des réponses qui y sont associées. On peut ajouter autant de questions et de réponses par question que l'on souhaite. On sélectionne les bonnes réponses via des check-box.

WebQuizz QUIZZS RÉSULTATS FLORIAN BAPPEL

Questions du quizz (2/4) ← Etape précédente

Premier Précédent 1 2 3 4 Suivant Dernier

Asking docking ?

Yes, port 21, please! Correcte ✖

Docking request denied Correcte ✖

Greating back ally, please dock port 34! Correcte ✖

Nouvelle proposition Correcte +

✖ Supprimer la question

Etape suivante ▶

1.4 - La page Jouer

L'action de jouer permet de lancer un quizz en nous envoyant vers une page dédiée. Afin de valider le quizz, il faut le faire dans son ensemble et cliquer sur le bouton terminer.

WebQuizz QUIZZS RÉSULTATS FLORIAN BAPPEL

Anime Créé le Apr 2, 2015 par Ketro *

★★★★★

1/4

Qui est Ryuk ?

- Un chat
- Un avion de chasse
- Un dieu de la mort
- Un virus
- Une tueuse a gage

Question suivante

Cette action permet d'enregistrer les résultats dans la base de données et de lancer la page de résultat avec les réponses de l'utilisateur et les réponses attendues.

WebQuizz QUIZZS RÉSULTATS FLORIAN BAPPEL

75.00%

Questions Réponses données en bleu, bonnes réponses sur fond vert

Qui est Ryuk ?
Un chat
Un avion de chasse
<i>Un dieu de la mort</i>
Un virus
Une tueuse a gage

Quelle l'identité du terroriste Zero ?
<i>Kururugi Suzaku</i>
Lelouch Lamperouge
Charles VI Britania
Kazuki Karen

Combien d'étoile vaut Ussop pour avoir assomer Sugar ?
1
2

1.5 - La page de listing des résultats

Cette page est similaire à la page de listing des quizz à ceci près que celle-ci recense tous les résultats de quizz enregistrés sur la BDD. Elle possède en outre les mêmes filtres et outils de recherche que cette dernière.

WebQuizz QUIZZS RÉSULTATS FLORIAN BAPPEL

Résultats

+ Jouer à un quizz

Rechercher...

Nom du quizz	Difficulté	Résultat	Joué le	Joueur
Quizz Ant-49	★★★★☆☆	50.00%	4/1/15 à 3:57 PM	Florian Bappel
fezgzeg	★★★★☆☆	0.00%	4/1/15 à 5:21 PM	Antoine DeJesus
test	★★★★☆☆	50.00%	4/1/15 à 5:21 PM	Kiruchi Thorann
test	★★★★☆☆	100.00%	4/1/15 à 5:23 PM	Antoine DeJesus
fezgzeg	★★★★☆☆	33.33%	4/1/15 à 9:17 PM	Mehdi Lhommeau
fezgzeg	★★★★☆☆	0.00%	4/2/15 à 12:59 PM	Kiruchi Thorann
fazfazfaz	★★★★★★	50.00%	4/2/15 à 1:01 PM	Kiruchi Thorann
fezgzeg	★★★★☆☆	33.33%	4/2/15 à 3:24 PM	Florian Bappel
CMDR Doux	★★★★★★	33.33%	4/2/15 à 3:37 PM	Antoine DeJesus
Anime	★★★★★★	25.00%	4/3/15 à 11:05 AM	laurent laurent

1 2 3

Avec tri par date et recherche du mot "try"

WebQuizz QUIZZS RÉSULTATS FLORIAN BAPPEL

Résultats

+ Jouer à un quizz

try

Nom du quizz	Difficulté	Résultat	Joué le	Joueur
Tryo	★★★★☆☆	100.00%	4/9/15 à 11:16 AM	Etienne Hatte
Tryo	★★★★☆☆	33.33%	4/9/15 à 1:18 PM	Kiruchi Thorann
Tryo	★★★★☆☆	100.00%	4/15/15 à 1:47 PM	Florian Bappel

2 - Nos difficultés

Après avoir passé les difficultés inhérentes à la découverte de nouveaux outils et langages/framework, nous avons rencontré des problèmes liés à la modification de quizz. Nous étions tout d'abord partis pour réaliser une mise à jour des données de la BDD mais cette méthode nécessitait trop de vérifications et de redondances. Nous avons donc opté pour une solution plus simple mais plus efficace : supprimer le quizz en question et le recréer avec les données modifiées.

3 - Amélioration/continuation

Dans une démarche d'amélioration continue, nous avons déjà intégré au code source de l'application des modifications permettant une intégration facile de plusieurs ajouts de fonctionnalités qui sont les suivantes :

- un champ "date de fin" optionnel a été prévu afin de limiter dans le temps les quiz.
- un champ type de quizz qui permettrait de définir différents affichages des quizz dans le mode Jouer (mot fléché, mode "Tout le monde veut prendre sa place")
- des quizz à essai unique ou des quizz avec temps limité.

D'autres améliorations possibles pourraient être les suivantes :

- Ajout d'un tchat
- Section réservée au professeur

Conclusion :

Au cours de ce projet, nous avons pu mettre en place des outils et techniques vus en cours comme les méthodes de gestion de projet, la définition d'un cahier des charges et de ses jalons. Nous avons pu faire preuve d'autonomie et d'esprit d'équipe afin de gérer notre temps sur les différentes séances que nous avons.

Nous avons également dû faire de la recherche de documentation pour mieux appréhender les nouvelles technologies sur lesquelles nous nous sommes formés. Cette veille technologique nous sera sans aucun doute d'un réel intérêt dans le monde professionnel, afin de rester à jour et en phase avec ce qui se fait de mieux.

Ces différents apprentissages à l'ISTIA nous ont aidés à aboutir à une première version de l'application, jugée "très prometteuse".

L'architecture que nous avons choisie s'est révélée payante, même si plus compliquée à prendre en main pour un niveau E14 qu'une architecture classique. Nous pensons qu'elle permettra à terme d'ajouter les fonctionnalités que nous n'avons pas intégrées par manque de temps.

Nous avons tous trois aimé travailler sur ce projet, et attendons à présent les cours d'E15 pour renforcer nos connaissances dans le domaine du web.

Webographie :

<https://www.mongodb.org/> : Site et documentations officielles pour MongoDB

<http://expressjs.com/> : Site et documentations officielles pour Express

<https://angularjs.org/> : Site et documentations officielles pour AngularJS

<https://nodejs.org/> : Site et documentations officielles pour Node.JS

<https://egghead.io/> : Tutoriels vidéo pour AngularJS

<http://meanjs.org/> : Site de MEAN.js, implémentation de l'architecture MEAN

<http://stackoverflow.com/> : Communauté de développeurs répondant à des questions techniques

Bibliographie :

MEAN Machine - Chris Sevilleja - Editions Leanpub

MEAN Web Development - Amos Q. Haviv - Editions Packt

Full stack Javascript development with MEAN - Adam Bretz & Colin J. Ihrig - Editions Sitepoint

Webquizz, un site web éducatif de quizz

Projet réalisé par : Florian BAPPEL, Antoine DE JESUS, Pierre MARTIN
Projet encadré par : Mehdi LHOMMEAU, Marc LENOIR, Dominique EVROT

Résumé

Ce projet tutoré a eu pour but de développer une application Web éducative visant à tester les connaissances des étudiants via des quizz interactifs. Les technologies choisies pour réaliser ce projet sont toutes des technologies récentes et innovantes.

L'application est basée sur une architecture en couche 3-tiers appelé MEAN. C'est une architecture entièrement en JavaScript réputée pour sa rapidité pour des projets de petite et moyenne envergures.

Le travail s'est fait en équipe, à l'aide de Github : un système de gestion de versions. Nous avons ainsi pu historiser notre code et travailler en collaboration de façon efficace.

Le résultat final est une première version de l'application. Cette version gère la gestion de quizz basiques, la gestion d'utilisateurs, et la recherche de résultats et de quizz. Des améliorations sont encore à faire, et rendues possibles par le choix d'architecture effectué.

Mots-clés

MEAN - site web - quizz - Bootstrap - html 5 - gestion de projet

Summary

This project aimed to develop an educational web application. Its goal was to test students' knowledge using interactive quizz. The technologies we chose for this project are all dynamic and modern.

The app was based on a 3-tier architecture layer called MEAN. This architecture is entirely in Javascript. It is well known to be fast for small to medium applications.

We worked in team with Github, a versioning system. It made possible to have an history of our code and to work efficiently together.

The final result is a 1.0 version of the application. It handles basic quizz management, users' management, et research through quizz and results. There is still room for improvement, but they are made possible by the chosen architecture.

Keywords

MEAN - website - quizz - bootstrap - html5 - project management