Oral Informatique et Sciences du Numérique

1ère slide :

Bonjour, je vais vous présenter le site web que j’ai développé avec mon camarade de classe Arthur Goldberg.

Nous avons choisi de réaliser ce site après avoir réalisé une présentation sur l’Affaire Snowden et l’outil PRISM. Nous avons compris l’importance des questions juridiques et éthiques liés à cette affaire (vie privée, confidentialité, sécurité, etc). Ainsi nous voulions faire en sorte qu’un système de cryptographie et donc toutes ces questions soient au cœur de notre projet. De là nous avons décidé après une longue discussion de réaliser cette messagerie instantanée cryptée.

Sur cette image nous avons une vue actuelle du site : il s’agit du profil d’un membre, avec une image de profil modifiable, une photo de couverture, ainsi qu’un bouton pour éditer ses infos. La page affichée est en Anglais grâce au système de multilinguisme.

2ème slide :

Le but de ce site web est assez simple, nous voulons fournir à nos utilisateurs le moyen de communiquer avec leurs amis. Pour cela nous avons choisis un système d’amitié et de fils de discussion le plus intuitif possible.

Le contrôle passe par la souris et le clavier sur ordinateur et sur tablette et mobile le système s’adapte à la largeur d’écran plu étroite et aux commandes tactiles.

3ème slide :

Dès le début de notre projet nous avons établi un cahier des charges très précis afin de ne pas avoir à revenir en arrière par la suite dans notre projet et/ou de se retrouver dans impasse.

La fonction primaire est la fonction communication, c’est le cœur de notre projet.

L’id ici correspond à un pseudonyme qui servira pour la recherche mais aussi lors de l’envoie des messages afin de savoir à qui l’on parle. Il ne peut excéder 10 caractères. Pour ce qui est de la sécurité nous utilisons une fonction symétrique afin de crypter et de décrypter le message avec une clé définie pour chaque membre. Ainsi les messages sont illisibles dans notre BDD et uniquement lisibles par les deux utilisateurs concernés.

Pour ce qui est du front-end, nous avons développé une interface entièrement responsive.

Il y a des pages de connexion / inscription. Nous avons aussi développé le système de photo de profil modifiable, ainsi que le système de traduction. Nous avons aussi donné la possibilité d’accompagner les messages d’émojis.

Pour finir dans le back-end : nous avons choisis d’utiliser comme langage le PHP. Pour ce qui est de l’hébergement et du stockage nous avons utilisés un serveur du rectorat ainsi qu’une base de donnée MySQL appartenant au rectorat. Notre code était sauvegardé et géré en utilisant Git sur la plate-forme GitHub.

4ème et 5ème slides :

Nous avions dès le départ un planning de travail après avoir établis le cahier des charges que vous venez d’observer.

Nous avons voulu répartir le travail de la manière la plus égale possible. Cependant il n’a pas été simple car nous ne partions pas avec les même connaissances.

Arthur a ainsi commencé par se former pendant que je m’occupait d’architecturer le code durant les premières semaines.

Pour autant nous avions tous les deux déjà réalisé un projet informatique nous ayant fait utiliser des outils utiles pour travailler à plusieurs et que nous maîtrisions déjà.

Malgré quelques retards ponctuels de quelques jours, nous avons tenu ce planning prévisionnel.

6ème slide :

Nous avons réalisé ce schéma afin d’expliquer les différentes possibilités que vous avez en tant qu’utilisateur du site.

Tout commence lorsque l’utilisateur arrive sur l’accueil, de là il peut soit se connecter, soit s’inscrire.

Par la suite il peut ajouter ses amis, grâce à leurs pseudonymes. Si la demande est acceptée, il peuvent commencer à discuter ensemble.

Il est aussi possible de passer tout le système en anglais. Ou n’importe quelle autre langue, le système gérant de manière très simple l’ajout de langues, mais nous n’étions familiers qu’avec l’anglais.

Pour finir vous pouvez modifier vos informations personnelles, y compris celles choisies lors de l’inscription.

7ème slide :

Nous avons utilisé une architecture Modèle-Vue-Contrôleur pour l’ensemble du système. Il s’agit d’un design pattern très utilisé dans la réalisation des sites web.

Il permet de séparer l'affichage des informations, la gestion des actions de utilisateur et l'accès aux données et leur traitement.

Concrètement c’est à dire que lorsque l’utilisateur demande d’accéder à une page, il n’accède pas vraiment à la page en question il accède en réalité à un fichier maître qui s’occupe charger un contrôleur appelant les modèles et vues nécessaires suivant une série de conditions.

8ème slide :

Dans ce projet, comme précisé dans le planning prévisionnel, je me suis occupé de différentes fonctionnalités. Mais là je vais me concentrer sur une partie précisément ; le système de gestion du membre. C’est-à-dire l’inscription, la connexion, la déconnexion et tout ce que cela implique de lien avec la base de données et les entrées du membre.

Premièrement, le contrôleur, c’est lui qui va être appelé par le fichier maître en premier et qui va permettre de différencier les pages. Voici celui de la page de déconnexion.

On peut voir que le code est relativement simple et est constitué uniquement de conditions qui résultent en appels de fonctions ou d’autres branchements de conditions. C’est l’exemple typique d’un contrôleur.

On voit que pour vérifier si le membre est connecté on se contente de vérifier si son ID en session est « vrai ». Vérifier simplement si l’ID est différent de 0 peut sembler léger mais c’est très résilient et extrêmement rapide, de plus dans notre base de code 0 correspond au membre « guest », banni.

9ème, 10ème et 11ème slides :

Ensuite, le modèle, qui fait le lien entre la base de données et l’utilisateur et vice-versa. Voici, celui de l’inscription :

Même si le code des modèles est de loin le plus compliqué de tous, nous avons quand même essayé de le garder suffisamment maintenable. Ainsi ce qui saute aux yeux c’est qu’une majorité des actions de ces méthodes consiste à appeler d’autres méthodes,

Cela est l’un des atouts de la programmation orienté objet, nous avons de nombreuses méthodes (une centaine), correctement catégorisées et qui, chacune, n’ont qu’un but précis et sont donc relativement simples.

Ici, le plus intéressant est sans doute le mot de passe, qui n’est pas stocké dans la base de données, à la place on stocke un « salage » de celui-ci ; c’est-à-dire qu’on va le crypter mais avant de faire cela on rajoute une chaîne de caractères avant celui rendant bien plus compliqué les attaques par collision. Ici, le « sel » est le slug du membre. Une sorte de pseudonyme nettoyé de caractères non-ASCII.

12ème et 13ème slides :

Finalement, on passe aux vues, ce que voit l’utilisateur final. Voici celle de la page de connexion, ne s’affichant que si les conditions nécessaires sont validées par le contrôleur évidemment,

Il s’agit d’une vue typique, le code contient uniquement du HTML et le PHP est réduit au strict appel de « echo ». Dans certains cas, des boucles ou conditions simples peuvent êtres utilisées mais jamais plus, le contrôleur ou les modèles doivent se charger d’avoir préparé les données pour la vue. La seule partie réellement intéressante de cette vue est cette variable $clauses appelée en permanence. Il s’agit en réalité d’un objet « Languages » (une autre de nos classes dans les méthodes) créée très tôt dans la vie du fichier maître et qui a une méthode extrêmement pratique nommée « get » qui permet d’appeler une chaîne de caractère dans la langue désirée en fonction de son identifiant ; ainsi l’identifiant « register » appellera « S’inscrire » en français et « Register » en anglais.

14ème slide :

Dans ce tableau sont listés les différents systèmes présents dans le code mais que nous n’avons codé nous même :

- Nous avons utilisé du code externe pour le design du profil et celui du bloc de discussion. Même si le code ne vient pas de nous, il ne gérait que le design, il n’était pas fonctionnel.

- Ce qui est différent avec les bibliothèques JavaScript c’est que ce sont des fichiers qui nous permettent d’appeler des méthodes et propriétés prêtes à l’emploi facilitant le développement. Nous avons aussi utilisé list.js pour le bloc de discussion, nous permettant de gérer chaque membre comme un élément d’un tableau, pratique pour la recherche.

- Nous avons aussi utilisé un plugin jQuery se nommant timeago.js nous permettant de gérer le formatage du temps dans le bloc de discussion.

- Nous avons aussi utilisé Bootstrap, un framework permettant de créer les vues tout en gérant simplement le responsive design et nous permettant de ne pas avoir à nous embêter à recréer des tas de systèmes basiques dont nous avons eu avoir besoin.

- Pour finir avons utilisé Font Awesome, une bibliothèque qui nous a permis d’avoir de nombreuses icônes vectorielles prêtes-à-l’emploi à utiliser.

15ème slide :

Nous sommes arrivés à remplir le cahier des charges et aujourd’hui le projet est fonctionnel, toutefois des problèmes subsistent dans notre code :

- Il reste quelques problèmes aux niveaux du responsive design : même si l’interface est entièrement responsive, la navigation sur mobile pourrait parfois être polie et plus fluide.

- La modification de photo de profil n’est également pas fonctionnel. L’équipe informatique du rectorat de Paris nous ayant indiqué que le téléchargement vers le serveur de fichiers utilisateurs est interdit. Toutefois le code que nous avons implémenté fonctionne bien dans nos tests.

- Pour finir, au niveau de la sécurité : la cryptographie étant un domaine très compliqué il y a sans doute de nombreux détails que nous avons manqué et des connaissances qui nous dépassaient complètement. Par exemple nous pourrions sans doute renforcer la sécurité de la clé échangée entre membres.

De même, la connexion au serveur du rectorat est faite en clair, ainsi les messages peuvent techniquement être interceptés.

En guise de conclusion, je vous propose de tester le site et plus particulièrement de créer un compte et le modifier, mon collègue se chargeant de la suite…