TER L2 Informatique

PunyDuck

Objectifs : Créer une plateforme de distribution des projets des étudiants de la Faculté des sciences de l'Université de Montpellier.

Différentes étapes

- Mise en place d'un serveur qui servira d'intermédiaire entre les utilisateurs et la base de données.
- Création d'un framework pour faciliter la réalisation de l'application.
- Conception de l'application graphique à l'aide du framework.
- Connexion entre l'application et le serveur.
- Création d'une base de données pour stocker les comptes des utilisateurs et les projets.
- Mise en service d'un site internet permettant le téléchargement de l'application.

Cahier des charges

I - Le serveur

- Fonctionnement asynchrone
- Héberger de manière sécurisée les données des utilisateurs.
- Les utilisateurs pourront télécharger les projets hébergés.
- Doit pouvoir gérer la plupart des erreurs de réseau, comme la coupure de la connexion lors d'un téléchargement.

II - Le framework

- Ensemble de classes et fonctions qui serviront de base à la structure d'une nouvelle application.
- Son utilisation doit être facile avec assez de fonctionnalités déjà disponibles pour pouvoir réduire les appels de fonctions bas niveau.
- Permettre la plasticité des interfaces

III - L’application

- Interface graphique permettant de naviguer facilement entre différents onglets
- Possibilité d'accéder aux informations de son compte et de les modifier : pseudonyme, adresse mail, âge, niveau, badges.
- Personnalisation limitée : mode clair / sombre, position de certains éléments de la fenêtre, couleur de l’arrière plan.

V - La base de données

- Stocke les données relatives aux comptes des utilisateurs ainsi que les projets.
- Les données seront cryptées.

VI - Le site

- Une page pour télécharger l’application.
- Design responsive et dynamique

Description des moyens utilisés *(TODO : description)*

- C++ avec les normes modernes de 2011, 2014 et 2017 ainsi que l’API graphique OpenGL pour créer le framework.
- Python 3.8 avec la bibliothèque asyncio pour le serveur et sa communication avec l’application.
- Le système de gestion de base de données PostgreSQL pour gérer les comptes utilisateurs et le stockage des projets.
- Discord pour la communication entre les membres du groupe.
- GitHub pour le partage et l’hébergement des fichiers sources.
- L’éditeur de texte Visual Studio Code qui permet de programmer à plusieurs sur un même projet grâce à son extension Live Share.
- Le langage et système de composition de documents LaTex pour la rédaction du rapport.

Répartition des tâches indépendantes

- Framework (Esteban | Valentin F.)
- serveur (Julien | Paul)
- Base de données (Julien | Paul)
- Site Internet (Valentin F. | Valentin P.)

Répartition des tâches dépendantes

- Interface graphique de l'application (Esteban | Valentin F. | Valentin P.)
- Communication entre l'application et le serveur
- (facultative) Messagerie

Diagramme de Gantt

TODO : forme visuelle

17/01 - 09/02 : apprentissage des bibliothèques OpenGL et asyncio (tests + ébauche (papier + programmation) de la structure du framework et du serveur pour préparer l'architecture finale et réduire les erreurs de conception)

20/01 - 27/01 : définition formelle du cahier des charges (notre vision d'une plateforme de distribution)

23/01 - 26/01 : Diagramme de Gantt (sans expliciter la partie programmation du framework, application et serveur)

27/01 - 02/02 : Listing exhaustif des fonctionnalités ~~de chaque composantes du projet~~ du framework et de l'application (non exhaustif pour le serveur car impossible car manque d'expérience donc oublie / sous-estimation de certaines parties)

27/01 - 02/02 : Maquettes graphiques de l'application et des différents éléments visuels qui la composent

02/02 - 02/02 : répartition des tâches dépendantes et indépendantes (pour éviter les blocages) + *minis Gantt* explicitant le déroulement de la programmation de l'application

03/02 - 09/02 : diagramme UML du framework, de l'application (framework influencé par les maquettes graphiques) et du serveur

(20/01 - 09/02 : conception de la structure globale du projet et modélisation)

10/02 - 08/03 : Programmation du framework

10/02 - ??/?? : Création et mise en place du serveur

24/02 - 31/03 : Programmation de l'application

24/02 - ??/?? : création de la base de données et intégration au serveur

01/04 - 01/04 : première version stable de l’application, 100% fonctionnelle

01/04 - 20/04 : réalisation du site Internet (conception du design puis création)

01/04 - 30/04 : tests unitaires et correction d'éventuels bugs résiduels

01/04 - 30/04 : ajouts des fonctionnalités facultatives (si possible) et minis projets

01/02 - 30/04 : rédaction du rapport