Rapport de Projet Informatique : « THOMAS » – a Train to HOMe And Stations

---

Introduction
Dans le cadre de notre deuxième année en CUPGE à l'UPSSITECH, nous avons réalisé un projet informatique intitulé « THOMAS ». Ce projet avait pour objectif de concevoir un programme en langage C permettant de gérer une compagnie de chemin de fer : création et gestion des trajets, réservation de places, et administration des utilisateurs (administrateurs, contrôleurs, voyageurs). Ce rapport présente l'organisation du programme, son mode d'emploi, un bilan qualitatif, ainsi que les difficultés rencontrées et les solutions apportées.

---

1. Organisation du Programme

Découpage en modules

- **Module auth** (`auth.c` / `auth.h`)
    - Gestion des utilisateurs (authentification, création de comptes, changement de mot de passe).
    - Hash SHA-256 pour sécuriser les mots de passe, stockés dans *users.txt*.
- **Module trajet** (`trajet.c` / `trajet.h`)
    - Gestion des trajets (ajout, modification, suppression, recherche, réservation).
    - Stockage dans *trajet.txt* : gares, temps de trajet, places disponibles.
- **Module voyageur** (`voyageur.c` / `voyageur.h`)

    - Gestion des voyageurs et de leurs réservations, historique personnel.

    - Synchronisation avec *trajet.txt* pour garantir la cohérence des données.

- **Module menu** (`menu.c` / `menu.h`)

    - Interface en ligne de commande, menus adaptés selon le rôle (administrateur, contrôleur, voyageur).

    - Chaque menu propose uniquement les fonctionnalités autorisées pour ce rôle.

- **Module json_export** (`json_export.c` / `json_export.h`)

    - Exportation et importation des données (trajets, réservations, utilisateurs) au format JSON.

    - Utilisation de la bibliothèque cJSON pour générer *exported_data.json*.

- **Module place** (`place.c` / `place.h`)

    - Gestion des places (affichage, réservation, libération).

Rôle des fichiers

- **trajet.txt** : liste des trajets (gares, temps, places).

- **users.txt** : identifiants et mots de passe hashés.

- **exported_data.json** : export JSON de l'ensemble des données.

- **admin.txt** et **controleur.txt** : comptes et mots de passe des administrateurs et contrôleurs.

---

2. Rôle des principales fonctions

- **afficher_trajets** : affiche les trajets disponibles (gares, horaires, places restantes).
- **ajouter_nouveau_trajet** : création d'un trajet (gares, durées).
- **supprimer_trajet_par_nom** : suppression d'un trajet et mise à jour des fichiers.
- **rechercher_trajet_txt** : recherche de trajets directs ou avec correspondances, proposition de réservation.
- **reserver_trajet_par_numero** : réservation d'un trajet sélectionné par numéro, gestion des places.
- **consulter_mes_reservations** : affichage des réservations de l'utilisateur connecté.
- **annuler_reservation** : annulation d'une réservation et libération de place.
- **exporter_trajets_txt** : export des trajets vers un fichier texte.
- **exporter_json** : export de l'ensemble des données au format JSON.

---

3. Mode d'Emploi

Compilation:

1. Installer les dépendances (ex. `libcjson-dev`).
2. Lancer la commande pour compiler : *make*

Exécution :

./bin/programme

Utilisation des menus

- **Administrateur**
    1. Gérer les trajets (ajouter, modifier, supprimer)
    2. Exporter les données au format JSON
    3. Changer le mot de passe (soi-même ou un autre utilisateur)
- **Contrôleur**
    1. Rechercher les trajets d'un voyageur (par identifiant)
    2. Changer son mot de passe
- **Voyageur**
    1. Rechercher des trajets (avec correspondances)
    2. Réserver ou modifier une réservation
    3. Consulter et annuler ses réservations

Les données sont sauvegardées automatiquement dans les fichiers texte et JSON à chaque modification.

---

4. Bilan Qualitatif

Points Positifs

- **Modularité** : code découpé en modules clairs, facilite le travail collaboratif et la maintenance.
- **Sécurité** : hash SHA-256 pour les mots de passe, protection des données sensibles.

- **Export JSON** : format structuré, facilite la portabilité et l'analyse externe.

- **Gestion des rôles** : accès limité aux fonctionnalités selon le profil de l'utilisateur.

Difficultés rencontrées et solutions :

- **Synchronisation des fichiers**

    - Problème : maintenir la cohérence entre la mémoire et les fichiers (*trajet.txt*, *users.txt*, *exported_data.json*).

    - Solution : centralisation des opérations de sauvegarde et export JSON forcé après chaque modification critique.

- **Gestion des correspondances**

    - Problème : recherche d'itinéraires avec correspondances, réservation multi-segments.

    - Solution : algorithme amélioré pour parcourir plusieurs trajets consécutifs en vérifiant la disponibilité des places.

- **Débogage et gestion des erreurs**

    - Problème : bugs de segmentation liés aux suppressions dans les tableaux, fuites mémoire.

    - Solution : ajout de vérifications systématiques, messages d'erreur explicites, et tests unitaires ciblés.

- **Sécurisation de l'authentification**

    - Problème : failles potentielles lors de la création et du changement de mot de passe.

    - Solution : utilisation rigoureuse du hash SHA-256 et enregistrement immédiat dans *users.txt*.

Améliorations futures

- Développement d’une interface graphique.
- Migration vers une base de données SQL pour la gestion en temps réel.
- Optimisation des algorithmes de recherche et de gestion des places pour de meilleurs temps de réponse.

Conclusion :

Ce projet nous a permis de consolider nos compétences en langage C, en architecture logicielle et en gestion de projet. Malgré les défis techniques, nous avons abouti à un système stable, sécurisé et conforme au cahier des charges. L’exportation JSON ouvre de nouvelles possibilités d’intégration et d’analyse.

Auteurs

- BOUDOUHI-MEZROUI Ayoub
- BOUJDAA Adam
- SAHLI Aziz