IUT SAINT-DIÉ-DES-VOSGES

INFORMATIQUE

PORTFOLIO UE1

COMPÉTENCE 1 : Réaliser

Étudiante : LIGEIRO, Eunice Eugenio

Saint-Dié-Des-Vosges, Juin 2025

Contents

Introduction 4

Analyse des Ressources et SAÉ's 5

R1.01 : Initiation au développement 5

Qu'ai-je appris ? 5

Comment ai-je appris ? 5

Comment cela m'a permis de développer la compétence "Réaliser" ? 5

R1.02 Développement d'interfaces web 5

Qu'ai-je appris ? 5

Comment ai-je appris ? 6

Comment cela m'a permis de développer la compétence "Réaliser" ? 6

R1.10 : Anglais 6

Qu'ai-je appris ? 6

Comment ai-je appris ? 7

Comment cela m'a permis de développer la compétence "Réaliser" ? 7

S1.01 : Implémentation d'un besoin client 7

Problématique professionnelle 7

Contexte de travail 8

Mise en œuvre de la compétence Réaliser 8

En résumé 10

Ressource R2.13 – Communication en milieu professionnel 10

Qu'ai-je appris ? 10

Comment ai-je appris ? 11

Comment cela m'a permis de développer la compétence « Réaliser » ? 11

R2.01 : Développement orienté objets 11

Qu'ai-je appris ? 11

Comment ai-je appris ? 11

Comment cela m'a permis de développer la compétence Réaliser ? 12

R2.01b Conception objet 12
Qu'ai-je appris ? 12
Comment ai-je appris ? 12
Comment cela m'a permis de développer la compétence Réaliser ? 13
R2.03 : Qualité de développement 13
Qu'ai-je appris ? 13
Comment ai-je appris ? 13
Comment cela m'a permis de développer la compétence Réaliser ? 13
R2.02 : Développement d'applications avec IHM 14
Qu'ai-je appris ? 14
Comment ai-je appris ? 14
Comment cela m'a permis de développer la compétence Réaliser ? 14
S2.01 : Développement d'une application 15
Problématique professionnelle 15
Contexte de travail 15
Mise en œuvre de la compétence Réaliser 15
Prolongements et regard critique 17
En résumé 17
Conclusion 18

Introduction

Ce portfolio présente les compétences que j'ai développées dans l'axe "Réaliser" au cours de mes deux premiers semestres en BUT Informatique. J'y décris les ressources et les projets (SAÉ) qui m'ont le plus appris, en m'appuyant sur des exemples concrets. L'objectif n'est pas de tout citer, mais de montrer comment j'ai évolué, ce que j'ai compris, et ce que j'ai appris de mes erreurs ou réussites.

Pour développer cette compétence, les ressources et SAÉ suivantes ont été étudiées et réalisées :

Semestre 1 :

- S1.01 : Implémentation d'un besoin client
- R1.01 : Initiation au développement
- R1.02 Développement d'interfaces web
- R1.10 : Anglais

Semestre 2 :

- S2.01 : Développement d'une application
- R2.01 : Développement orienté objets
- R2.01b Conception objet
- R2.02 : Développement d'applications avec IHM
- R2.03 : Qualité de développement
- R2.13: Communication en milieu professionnel

Analyse des Ressources et SAÉ’s

R1.01 : Initiation au développement

Qu’ai-je appris ?

Grâce à cette ressource, j’ai appris les bases de la programmation en langage C, notamment l'utilisation des conditions, des boucles, des fonctions, des tableaux, des structures et des piles. J’ai découvert comment structurer un programme, résoudre des problèmes étape par étape et tester mes solutions. J’ai aussi compris l’importance de la rigueur dans l’écriture du code et de la lisibilité pour faciliter la maintenance et la compréhension du programme.

Comment ai-je appris ?

Au début, j’ai eu du mal à comprendre les consignes et à traduire ma logique en code. Je passais beaucoup de temps sur les exercices, parfois sans réussir à les terminer. Mais en pratiquant régulièrement, en relisant les cours, en testant différents cas et en m’aidant des corrections, j’ai commencé à progresser. J’ai aussi échangé avec mes camarades, ce qui m’a permis de voir d’autres façons de penser et de coder. Les TP m'ont beaucoup aidée à appliquer la théorie et à gagner en autonomie.

Comment cela m’a permis de développer la compétence “Réaliser” ?

Petit à petit, j’ai appris à passer d’un problème à une solution fonctionnelle. J’ai gagné en logique, en méthode et en confiance. Aujourd’hui, je suis capable d’analyser un énoncé, de planifier les étapes de la réalisation et de corriger mes erreurs. J’arrive à écrire du code plus clair, plus structuré, et à penser aux cas limites. Je sais aussi utiliser les outils de développement et la documentation pour m’aider. Je ne maîtrise pas encore tout, mais je me sens désormais capable de réaliser un programme simple de bout en bout, ce qui était impensable pour moi au début du semestre.

R1.02 Développement d'interfaces web

Qu’ai-je appris ?

Grâce à cette ressource, j’ai appris à créer une page web structurée en HTML et stylisée avec

CSS. J'ai découvert l'usage des balises HTML sémantiques (<header>, <section>, <article>, etc.), l'intégration de contenus variés (images, vidéos, tableaux) et les principes de base du design avec CSS : mise en forme, typographie, couleurs, espacements, etc. J'ai aussi appris à organiser mes fichiers correctement et à valider mon code avec les outils du W3C.

Comment ai-je appris ?

L'apprentissage s'est fait progressivement à travers un TD d'introduction suivi de trois TP évalués, chacun plus complexe que le précédent. Le premier TP m'a permis de m'approprier les bases du HTML, le deuxième m'a initiée à la mise en forme avec CSS, et le troisième m'a demandé de produire une page complète, structurée et stylisée. Par ailleurs, j'ai pu mettre en pratique ces connaissances lors du projet de la SAÉ 1.06, en binôme, pour créer une page web sur l'entreprise Scrub Daddy. En parallèle, les cours de design d'interface avec Figma m'ont aidée à mieux comprendre l'aspect visuel et ergonomique du web.

Comment cela m'a permis de développer la compétence "Réaliser" ?

Cette ressource m'a permis de développer la compétence "Réaliser" en me donnant les moyens de produire une page web fonctionnelle, lisible et conforme aux standards. J'ai appris à suivre un cahier des charges simple, à organiser mon travail, à respecter une maquette de design, et à coder de manière rigoureuse. Grâce à l'expérience de la SAÉ, j'ai également compris l'importance de la cohérence entre le contenu, la structure et l'apparence, ce qui est essentiel pour réaliser une application web efficace. Même si je n'ai pas encore exploré tous les aspects comme l'accessibilité (ARIA, navigation clavier), je suis désormais capable de créer une base solide sur laquelle je pourrai m'améliorer.

R1.10 : Anglais

Qu'ai-je appris ?

Cette ressource m'a permis d'enrichir mon vocabulaire spécifique à l'informatique en anglais, notamment grâce à deux documents : l'un sous forme de livre avec des exercices sur le hardware, le software et l'Internet, et l'autre qui rassemblait la terminologie informatique la plus courante. J'ai appris à mieux comprendre et utiliser des mots techniques liés aux composants,

aux logiciels, aux réseaux, ou encore à la cybersécurité. J'ai aussi appris à me présenter en anglais de façon claire, et à m'exprimer à l'oral devant un groupe sur un sujet professionnel.

Comment ai-je appris ?

L'apprentissage a été progressif. Au début, j'ai étudié les deux documents pour acquérir les bases du vocabulaire informatique. Ensuite, on a fait une activité en binôme où je devais poser des questions personnelles à un camarade pour ensuite le présenter à la classe, ce qui m'a permis de pratiquer l'anglais oral dans un contexte réel. Enfin, en duo, nous avons préparé et présenté un exposé en anglais sur un thème de l'informatique et nous avons choisi de parler des hackers. Cela m'a donné l'occasion d'effectuer des recherches, de structurer mes idées en anglais, et de les présenter de façon compréhensible et professionnelle.

Comment cela m'a permis de développer la compétence "Réaliser" ?

Cette ressource m'a permis de réaliser des tâches concrètes en anglais dans un contexte professionnel. J'ai appris à adapter mon langage en fonction du public, à respecter un format d'exercice ou de présentation, et à travailler efficacement en binôme. En produisant une présentation structurée et pertinente sur un sujet technique, j'ai aussi développé ma capacité à transmettre des informations complexes de manière simple et claire. Ce sont des compétences essentielles dans le domaine informatique, où la communication en anglais est souvent incontournable.

S1.01 : Implémentation d'un besoin client

Problématique professionnelle

L'objectif de cette SAÉ était de concevoir et développer, en langage C, un jeu d'aventure textuelle très basique, où le joueur interagit uniquement via la console. Le principal défi consistait à structurer un monde virtuel en lieux interconnectés, permettant l'exploration via des commandes textuelles.

Contexte de travail

J'ai réalisé ce projet en binôme, ce qui représentait une première véritable expérience de collaboration en programmation. Nous avons utilisé Git pour gérer les versions et synchroniser notre code. Le projet s'est conclu par une soutenance orale où nous avons présenté notre jeu et répondu aux questions des enseignants.

Mise en œuvre de la compétence Réaliser

AC11.01 | Implémenter des conceptions simples

Nous avons conçu des structures Location, Mobile et Game, en utilisant les pointeurs et l'allocation dynamique.

Extrait - Déclaration de la structure Location :

```c
typedef struct Location {
    char *name;                 // Location name
    char *desc;                 // Description of the location
    struct Location *exits[6];  // Connections in each direction
} Location;
```

Extrait - Fonction LocationNew :

```c
// Function prototypes
Location *LocationNew(char *name, char *desc);
void LocationDelete(Location *l);
void LocationPrint(Location *l);
void LocationSetExit(Location *from, Direction dir, Location *to);
```

La logique de commandes (help, go, look...) a également été implémentée dans un fichier cmd.c, avec une structure conditionnelle claire.

Cette étape m'a permis de renforcer ma compréhension des pointeurs et de la structuration des données.

AC11.02 | Élaborer des conceptions simples

Avec mon binôme, nous avons d'abord réfléchi à l'architecture du jeu : comment représenter les lieux, les déplacements, et les commandes.

Extrait – Structure Game :

```c
typedef struct
{
    Mobile *player;
    Stack *locations;   // Stack to store all locations
} Game;
```

Ce travail préparatoire a été essentiel pour une mise en œuvre structurée et claire, même dans un projet simple.

AC11.03 | Faire des essais et évaluer leurs résultats

Chaque fonction a été testée au fur et à mesure. Les tests consistaient principalement à entrer des commandes dans la console pour vérifier les réactions du jeu.

Gestion d'erreurs – Commande inconnue :

```c
void parseAndExecute(char *line, Game *game)
{
    if (line && game) /* check pointers validity */
    {
        char *name = strtok(line, sep); /* extract first word = command */
        if (name)                       // null pointer if line just contains a return (empty line)
        {
            char *args = strtok(NULL, sep); /* extract the second word = argument. The rest of the line is disregarded */
            /* call the appropriate function, depending on the command */
            if (!strcasecmp(name, "help"))
                cmdHelp(); /* cmd=="help" -> call cmdHelp() */
            else if (!strcasecmp(name, "quit"))
                cmdQuit(game); /* cmd=="quit" -> call cmdQuit(). Also properly shuts the game down */
            else if (!strcasecmp(name, "look"))
                cmdLook(game, args); /* cmd=="look" -> call cmdLook(). args specify where/what to look. game contains the necessary info about st
            else if (!strcasecmp(name, "go"))
                cmdGo(game, args); /* cmd="go" -> call cmdGo(). args specify where to go. game contains info about locations and will update the
            else
                printf("Command not found. Try 'help'\n");
        }
        else
            printf("Command not found. Try 'help'\n");
    }
}
```

Nous avons également testé les déplacements dans toutes les directions et les cas limites (commandes erronées, directions bloquées…).

Cette phase m’a appris à tester rapidement, de manière empirique, tout en comparant les résultats aux spécifications.

Prolongements et regard critique

Ce projet m’a permis de consolider les bases de la programmation en C, tout en développant des compétences transversales. Mais plusieurs axes d’amélioration se dégagent :

- **Renforcer la gestion des erreurs** : Vérifier systématiquement les retours de malloc, strdup, et libérer proprement la mémoire en cas d’échec aurait permis un code plus robuste.
- **Mieux structurer les modules** : Une séparation plus nette entre la logique du jeu, la gestion des commandes et l’univers aurait favorisé la maintenabilité.
- **Approfondir Git** : L’usage de branches pour développer des fonctionnalités en parallèle aurait été bénéfique.

En résumé

Cette SAÉ a marqué une étape importante dans mon apprentissage : elle m’a permis de passer de la théorie à la pratique, de collaborer efficacement, et de mieux comprendre la logique de conception d’un programme en C. Grâce à elle, j’ai acquis les bases solides nécessaires pour évoluer vers des projets plus complexes et structurés.

Ressource R2.13 – Communication en milieu professionnel

Qu’ai-je appris ?

J’ai appris à vulgariser des informations techniques pour les rendre accessibles à un public non spécialiste. Cela implique de simplifier un langage complexe, structurer les idées et adapter son discours à différents interlocuteurs. Cette compétence est essentielle en informatique, car on doit

souvent expliquer nos choix ou notre travail à des clients, utilisateurs ou collègues qui n'ont pas le même niveau technique.

Comment ai-je appris ?

À travers l'analyse d'articles (comme « Météorite russe ») et de vidéos (« Ma thèse en 180 secondes »), j'ai compris comment repérer le public cible et adapter le niveau de langage. Ensuite, j'ai pu m'entraîner à simplifier des textes ou à reformuler des concepts vus en TP, comme la récursivité ou la manipulation de pointeurs. Ces exercices m'ont obligé à réfléchir à l'essentiel à transmettre, à l'ordre logique des idées, et à utiliser des exemples concrets.

Comment cela m'a permis de développer la compétence « Réaliser » ?

Savoir vulgariser m'aide à mieux structurer mon raisonnement quand je développe une solution informatique. Ça me force à clarifier mes idées avant même de coder ou de présenter mon travail. Dans mes projets, je suis plus à l'aise pour expliquer mes choix techniques, que ce soit à l'écrit ou à l'oral. J'ai aussi pris conscience que « réaliser », ce n'est pas seulement écrire du code, c'est aussi rendre ce qu'on fait compréhensible et utile pour les autres.

R2.01 : Développement orienté objets

Qu'ai-je appris ?

Dans la ressource R1.01, j'ai appris les bases de la programmation en langage C. J'ai découvert la syntaxe du langage, la logique algorithmique, les structures de contrôle (conditions, boucles), la manipulation des tableaux, la gestion de la mémoire, et la décomposition fonctionnelle d'un programme. J'ai aussi compris l'importance de la rigueur dans l'écriture du code et la manière de documenter son travail avec des commentaires clairs.

Comment ai-je appris ?

J'ai appris principalement à travers les séances de TP où l'on devait implémenter des programmes en suivant des consignes précises. J'ai pris l'habitude de commencer par une phase d'analyse du problème avant de coder, souvent en écrivant des algorithmes en pseudo-code. J'ai aussi tiré parti de mes erreurs : chaque bug rencontré m'a obligée à relire attentivement mon

code, à utiliser printf() pour déboguer, et à rechercher des solutions de manière autonome ou en demandant de l'aide à mes camarades. Grâce à des retours réguliers de l'enseignant, j'ai pu corriger mes mauvaises habitudes (comme l'oubli de libérer la mémoire) et améliorer progressivement mon style de programmation.

Comment cela m'a permis de développer la compétence Réaliser ?

Ces apprentissages m'ont permis de développer la compétence **Réaliser** car j'ai appris à produire du code fonctionnel, lisible et structuré, en suivant une démarche rigoureuse. Je suis désormais capable de traduire un besoin simple en programme exécutable, de tester mes fonctions, de corriger des erreurs de compilation et d'exécution, et de structurer mon travail à travers plusieurs fichiers. Le fait de coder moi-même et de comprendre mes erreurs m'a donné plus de confiance en mes capacités. Je sais aussi mieux estimer le temps nécessaire à chaque étape du développement.

R2.01b Conception objet

Qu'ai-je appris ?

Grâce à la ressource conception objet, j'ai appris les fondements de la Programmation Orientée Objet (POO) ainsi que la modélisation UML, deux compétences essentielles pour concevoir des logiciels robustes et bien structurés. J'ai compris les principes clés de la POO : encapsulation, héritage, abstraction et modularité, ainsi que leur traduction concrète en Java. J'ai également appris à utiliser des diagrammes UML (cas d'utilisation, classes, séquence) pour représenter visuellement le comportement et la structure d'un système avant sa mise en œuvre.

Comment ai-je appris ?

L'apprentissage s'est fait progressivement, en alternant entre cours théoriques et exercices pratiques dans les TPs. Les cours et TP's m'ont aidée à faire le lien entre la théorie (modélisation) et la pratique (implémentation). Par exemple, dans un TP, j'ai modélisé une médiathèque avec ses classes, puis je l'ai implémentée en Java, en utilisant des notions comme les classes abstraites ou les relations d'héritage. J'ai ainsi appris à traduire une modélisation UML en un programme fonctionnel, en respectant les bonnes pratiques de la POO.

Comment cela m’a permis de développer la compétence Réaliser ?

Cette ressource m’a permis de développer la compétence Réaliser en me donnant une méthodologie claire pour concevoir puis coder un programme orienté objet. J’ai appris à analyser un besoin, à le modéliser de façon rigoureuse, puis à implémenter la solution en Java de manière structurée. Le travail sur les TPs m’a également permis de mieux comprendre l’intérêt de la modularité et de la réutilisabilité du code. Aujourd’hui, je suis plus autonome pour créer un programme orienté objet complet, en partant d’un diagramme UML jusqu’à un code fonctionnel.

R2.03 : Qualité de développement

Qu’ai-je appris ?

Dans cette ressource, j’ai appris que la qualité de développement repose sur plusieurs aspects essentiels. Premièrement, l’importance de structurer correctement le code, de bien documenter ses fonctions et variables, et de suivre des bonnes pratiques pour rendre le code lisible et maintenable. Ensuite, j’ai découvert le traitement des exceptions, ce qui permet de gérer les erreurs potentielles sans interrompre brutalement l’exécution du programme. Pour cela, nous avons réalisé un mini-projet de tournoi de volley-ball où il fallait gérer toutes les exceptions possibles selon les spécifications données. Enfin, j’ai appris à créer des tests unitaires en Python pour vérifier que chaque fonction fonctionne comme attendu, ce que nous avons mis en pratique à travers deux travaux pratiques.

Comment ai-je appris ?

J’ai appris à travers un mélange de cours théoriques, d’exemples pratiques et de projets. La partie sur la structuration du code et la documentation m’a été expliquée avec des exemples concrets. Pour le traitement des exceptions, nous avons codé un mini-projet qui m’a obligé à anticiper et gérer tous les cas d’erreur possibles. Enfin, les TP sur les tests unitaires m’ont permis d’écrire des tests automatiques en Python, renforçant ma compréhension et mes compétences par la pratique.

Comment cela m’a permis de développer la compétence Réaliser ?

Ces apprentissages m’ont permis de produire un code plus propre, plus sûr et plus fiable. En

structurant mieux mon code et en le documentant, je facilite sa lecture et sa maintenance, ce qui est essentiel en contexte professionnel. La gestion des exceptions garantit que le programme reste stable malgré les erreurs, ce qui augmente sa robustesse. Les tests unitaires assurent la qualité du logiciel en détectant rapidement les bugs. Tout cela contribue directement à la compétence Réaliser, car je suis capable de développer des applications informatiques de qualité, fonctionnelles et prêtes à l'usage.

R2.02 : Développement d'applications avec IHM

Qu'ai-je appris ?

J'ai appris à concevoir et développer des interfaces homme-machine (IHM) en Java, en utilisant deux technologies principales : Swing puis JavaFX. J'ai compris les concepts fondamentaux de l'IHM, comme la gestion des événements, la séparation entre la partie graphique (vue) et la logique (contrôleur), ainsi que l'importance d'une interface claire et intuitive pour l'utilisateur.

Comment ai-je appris ?

Les apprentissages se sont faits d'abord par des cours théoriques expliquant les bases des IHM et leurs enjeux. Ensuite, nous avons mis en pratique ces notions via des exercices Swing, où nous développions des interfaces graphiques simples à présenter au professeur. Par la suite, nous avons découvert JavaFX, où nous utilisions SceneBuilder pour créer graphiquement nos interfaces, puis nous écrivions la logique dans les classes contrôleurs associées. Cette alternance théorie/pratique m'a permis d'intégrer progressivement les concepts et de les appliquer concrètement.

Comment cela m'a permis de développer la compétence Réaliser ?

Ce module m'a donné les outils et méthodes nécessaires pour réaliser des applications avec interfaces graphiques efficaces et fonctionnelles. Savoir manipuler Swing et JavaFX, maîtriser SceneBuilder et la programmation orientée contrôleur me permet aujourd'hui de concevoir des logiciels avec une meilleure ergonomie et organisation. Cela développe ma capacité à créer des solutions complètes, répondant aux besoins des utilisateurs, ce qui est un élément clé de la compétence Réaliser.

S2.01 : Développement d'une application

Problématique professionnelle

L'objectif de cette SAÉ était de concevoir, en trinôme, une application dotée d'une interface graphique destinée à des utilisateurs non-informaticiens. Le besoin exprimé portait sur la gestion d'un calendrier universitaire, jusqu'ici basée sur l'édition manuelle de scripts. Notre mission était donc de proposer une interface intuitive permettant la création, la modification et la visualisation de ce calendrier à travers des interactions simples.

Contexte de travail

Ce projet a été mené en trinôme et a représenté une nouvelle étape dans la gestion collaborative d'un code plus complexe. Contrairement à la SAÉ 1.01, l'ajout d'une interface graphique (AWT/Swing en Java) nous a confrontés à des problématiques nouvelles : gestion des événements utilisateurs, mise à jour dynamique des vues, et structuration du code orienté objet. Le projet s'est terminé par une soutenance avec démonstration de l'application.

Mise en œuvre de la compétence Réaliser

AC11.01 | Implémenter des conceptions simples

Nous avons implémenté différentes classes et méthodes pour gérer la logique métier et l'affichage. L'une des méthodes clés est createNewCalendar(), qui guide l'utilisateur à travers une série de boîtes de dialogue pour créer un nouveau calendrier universitaire, de manière interactive et sécurisée :

```java
private void createNewCalendar() {  1 usage  Wendy-Patricia
        // Étape 1 : Demande de l'année
        String yearInput = JOptionPane.showInputDialog(
                parentComponent: this,
                message: "Entrez l'année pour le nouveau calendrier:",
                title: "Nouveau Calendrier",
                JOptionPane.QUESTION_MESSAGE
        );

        if (yearInput == null || yearInput.trim().isEmpty()) return;

        try {
            int year = Integer.parseInt(yearInput);

            // Valider que l'année est dans une plage raisonnable (ex: 1900-2100)
            if (year < 1900 || year > 2100) {
                JOptionPane.showMessageDialog(
                        parentComponent: this,
                        message: "Veuillez entrer une année entre 1900 et 2100.",
                        title: "Année invalide",
                        JOptionPane.ERROR_MESSAGE
                );
                return;
```

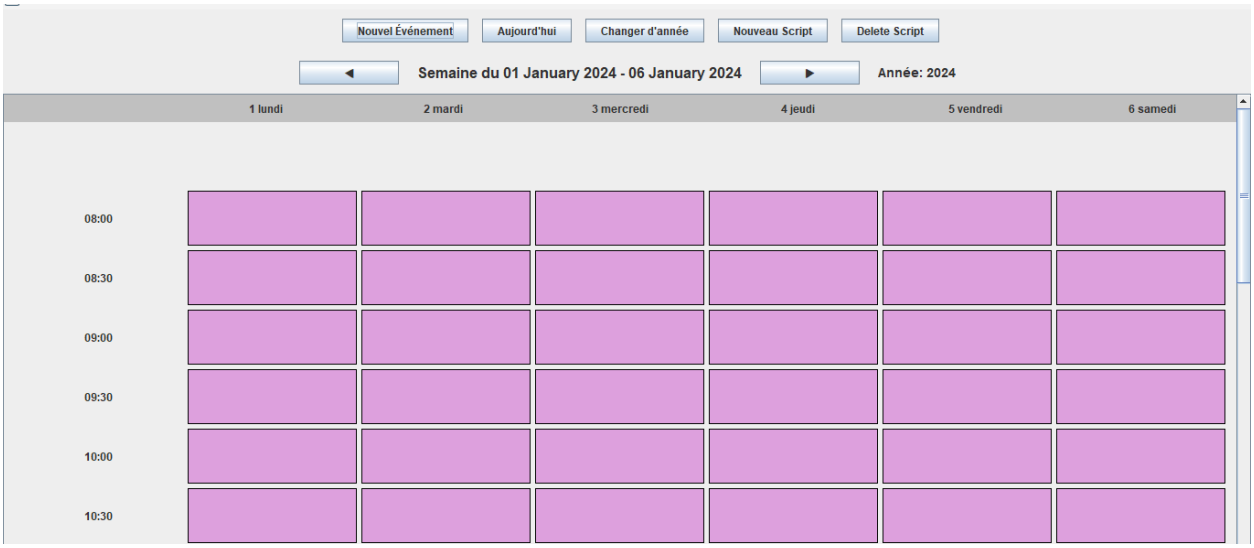
Cette implémentation met en œuvre des principes d'interaction utilisateur, de validation des entrées, et de persistance des données.

AC11.03 | Faire des essais et évaluer leurs résultats

Nous avons procédé à de nombreux tests empiriques, en simulant des cas d’usage réels :

- Création d’un calendrier avec des valeurs valides et invalides
- Tentatives de suppression de fichiers inexistants
- Ajout de contraintes ou d’événements aux mauvais endroits

Résultat final :



Prolongements et regard critique

Ce projet m'a permis d'aborder concrètement la programmation événementielle et la gestion d'interfaces. Plusieurs pistes d'amélioration ont été identifiées :

- **Modulariser davantage le code** : Certaines classes sont encore trop chargées (ex. : une classe fait interface et logique).
- **Ajouter une couche modèle/contrôleur claire (MVC)** pour une meilleure séparation.
- **Améliorer l'expérience utilisateur** : confirmation visuelle plus fluide, prévisualisation du calendrier avant validation.
- **Approfondir l'utilisation de Git** : Comme pour la SAÉ 1.01, l'usage plus rigoureux de branches aurait amélioré notre processus de développement.

En résumé

La SAÉ 2.01 m'a permis de passer de la programmation en ligne de commande à la conception d'une application graphique, en intégrant les notions d'événement, d'ergonomie, et de persistance. Elle m'a appris à penser « utilisateur final », à structurer un projet orienté objet, et à collaborer efficacement autour d'un projet plus réaliste.

Conclusion

Cette compétence a pris tout son sens au fil des ressources, en particulier grâce aux projets pratiques qui m'ont poussée à aller au-delà de la simple écriture de code. J’ai appris à structurer un programme, à anticiper les erreurs, à tester rigoureusement et à toujours garder en tête la qualité et la lisibilité de ce que je développe.

Comprendre que « réaliser » ne se limite pas à faire fonctionner un code, mais implique aussi de le rendre fiable, maintenable et compréhensible, a changé ma façon de travailler.

Aujourd’hui, je me sens plus confiante dans ma capacité à concevoir et développer des applications complètes, tout en sachant que chaque nouveau projet est une opportunité d’aller plus loin. Ce que j’ai acquis me donne une base solide et professionnelle pour continuer à progresser.