BTS CIEL – SESSION 2025  LESEIGNOUX Sacha

Dossier Technique du projet « Accès campus » - Partie Poste de Gestion Des Salles (PGS)



Épreuve E6 – Rapport Personnel de Projet

Spécialité : Informatique et Réseaux (IR)

Lycée Saint Aubin La Salle – Verrière en Anjou

Table des matières

[PRÉSENTATION DU PROJET : 3](#_Toc196916206)

[1. Situation dans le projet 4](#_Toc196916207)

[1.1. Synoptique de la réalisation 4](#_Toc196916208)

[1.2. Description de la partie personnelle 7](#_Toc196916209)

[2. Analyse UML/SysML 8](#_Toc196916210)

[3. Installation de l’environment de Travail 8](#_Toc196916211)

[4. Développement de la partie gestion des badges 8](#_Toc196916212)

[5. Développement de la partie gestion des salles 9](#_Toc196916213)

[5.1. Tests et Validation 9](#_Toc196916214)

[6. Réalisation de la fonction et/ou cas d'utilisation 9](#_Toc196916215)

[7. Conception détaillée 12](#_Toc196916216)

[8. Tests unitaires 14](#_Toc196916217)

[8.1. 2.3. Tests unitaires du module matériel/logiciel 14](#_Toc196916218)

[9. Problèmes rencontrés 16](#_Toc196916219)

[10. Bilan de la réalisation personnelle 18](#_Toc196916220)

PRÉSENTATION DU PROJET :

Dans le cadre de ce projet, il nous a été demandé de développer une solution complète pour gérer l’accès aux salles du Campus Saint Aubin La Salle.  
Le but était de contrôler qui pouvait entrer dans les salles de cours et de pouvoir suivre la présence des étudiants et du personnel. L’accès devait principalement se faire grâce à des badges RFID, utilisés comme des clés électroniques.

Le choix de ce type de système répondait à plusieurs besoins :  
D’abord, améliorer la sécurité des locaux et des utilisateurs, mais aussi simplifier la gestion des accès en réduisant l’usage des clés traditionnelles.  
Cela permettait également de moderniser l’image du campus et de proposer un service plus pratique aux étudiants et aux enseignants, notamment après leur installation dans de nouveaux locaux.

Le projet se déroulait autour de plusieurs éléments techniques :

* La PEA (Poignée Électronique Autonome) :  
  Chaque porte équipée possédait un dispositif autonome comprenant un lecteur de badge RFID, un digicode et une caméra pour la reconnaissance faciale. Chaque PEA était identifiée individuellement.
* Le PGS (Poste de Gestion des Salles) :  
  C’était depuis ce poste que l’exploitant pouvait créer et gérer les badges ainsi que planifier l’accès aux salles via des réservations.
* La BAE (Borne d’Appel Étudiant) :  
  Installée à l’accueil, cette borne avec écran tactile permettait aux utilisateurs de réaliser certaines opérations comme demander un accès ou obtenir des informations rapidement.
* Le PSW (Poste Serveur Web) :  
  Ce serveur était chargé d’héberger une application web destinée aux clients extérieurs souhaitant faire des réservations de salles.

1. Situation dans le projet
   1. Synoptique de la réalisation

Une image contenant texte, diagramme, capture d’écran, Plan

Le contenu généré par l’IA peut être incorrect.

Le projet de contrôle d’accès du campus était organisé en cinq grandes parties, chacune correspondant à un sous-projet soit électronique, soit informatique, comme un serveur ou une interface homme-machine.  
Dans cette partie, je vais présenter les différents éléments réalisés.

La Poignée Électronique Autonome (PEA) gérait le verrouillage des portes grâce à trois méthodes d’identification : le badge RFID, le digicode et l’empreinte digitale. Une clé physique était également prévue en secours en cas de dysfonctionnement.  
Chaque PEA devait être réplicable sur le réseau du campus et possédait un identifiant unique pour assurer son bon fonctionnement.

Le Poste Serveur Web (PSW), de son côté, hébergeait l’application web permettant la gestion des accès via les identifiants SALS.  
Cette application offrait également la visualisation des salles disponibles, le suivi des présences, ainsi que l'historique des absences et des retards, en lien direct avec la base de données.

Le Poste de Gestion des Salles (PGS), réservé aux administrateurs, permettait de réserver les salles et de gérer l’ensemble du parc, soit plus de 300 salles et environ 2000 badges d’accès.

Enfin, la Borne d’Appel Étudiant (BAE) était chargée d’identifier les étudiants grâce à leur badge et d’afficher leurs informations (nom, prénom, classe, effectif prévu).  
Elle vérifiait leur présence et signalait toute erreur d’affectation.  
En cas d’oubli de badge, un professeur pouvait valider la présence d’un étudiant via une application disponible sur PC ou sur mobile.

Une image contenant diagramme, croquis, ligne, conception

Le contenu généré par l’IA peut être incorrect.

Le diagramme de déploiement présente l'architecture complète du système de gestion des accès aux salles du Campus Saint Aubin La Salle. Il montre comment les différents équipements matériels interagissent entre eux pour assurer le contrôle d'accès et la supervision des présences sur le campus. Ce diagramme permet ainsi de visualiser la répartition des rôles et des responsabilités de chaque dispositif technique intégré dans le projet.

Dans ce projet, notre objectif principal était de développer une solution efficace et sécurisée pour gérer les entrées dans les salles de cours et pour suivre en temps réel la présence des étudiants et du personnel. L'accès aux salles devait s'effectuer principalement grâce à des badges RFID, mais aussi via d'autres méthodes d'identification telles que le digicode et l'empreinte digitale.

La Poignée Électronique Autonome (PEA) jouait un rôle clé dans le dispositif. Installée sur chaque porte, elle gérait l'ouverture grâce à trois moyens d'identification : badge RFID, digicode et empreinte digitale. En cas de panne, une clé physique permettait également d’ouvrir la porte. Chaque PEA possédait un identifiant unique et devait pouvoir être répliquée sur l’ensemble du réseau du campus afin d’assurer son bon fonctionnement.

Le Poste Serveur Web (PSW) hébergeait quant à lui l'application web dédiée à la gestion des accès via les identifiants SALS. Cette application offrait plusieurs fonctionnalités : la gestion des accès, la visualisation en temps réel des salles disponibles, le suivi précis des présences, ainsi que l'accès aux historiques d'absences et de retards. Elle était directement reliée à la base de données du campus.

Le Poste de Gestion des Salles (PGS) représentait un point important pour les administrateurs. Il permettait de créer et de gérer les badges d'accès, de planifier l'ouverture des salles et de gérer l'ensemble du parc comprenant plus de 300 salles et environ 2000 utilisateurs.

La Borne d'Appel Étudiant (BAE) avait pour mission d'identifier les étudiants par leur badge à l'entrée du campus. Elle affichait leur nom, prénom, classe et validait leur présence, tout en signalant toute erreur d’affectation éventuelle. En cas d'oubli de badge, un professeur pouvait confirmer manuellement la présence de l'étudiant à l’aide d'une application accessible sur PC ou sur mobile.

Enfin, tous ces équipements étaient interconnectés via un Switch, qui centralisait les communications réseau entre les différents dispositifs. Le Switch était lui-même relié à un Routeur, assurant la liaison avec l'extérieur si nécessaire.

Dans le diagramme de déploiement, le PGS est mis en évidence en rouge pour souligner ma partie mais aussi son importance stratégique dans l'architecture du système. Il symbolise le cœur de la de la gestion des accès et des réservations de salles, directement connecté au Switch pour une communication rapide et fiable avec les autres équipements du réseau.

* 1. Description de la partie personnelle

Dans le cadre de ce projet, ma mission principale a été de développer le Poste de Gestion des Salles (PGS) pour le Campus Saint Aubin La Salle. Ce système, destiné aux administrateurs du campus, permet de gérer les accès aux salles de cours et d’organiser les réservations tout en étant intégré aux dispositifs de contrôle comme les badges RFID et les digicodes. L’objectif était de centraliser la gestion des accès et des réservations dans une interface sécurisée et facile à utiliser.

Après avoir analysé le cahier des charges, j’ai structuré le développement autour de deux fonctionnalités essentielles : la gestion des badges RFID et la planification des réservations des salles. Pour la gestion des badges, j’ai conçu une interface permettant aux administrateurs de créer, modifier, désactiver ou supprimer des badges d’accès. Ce système doit pouvoir gérer efficacement près de 2000 badges, chacun étant associé à un utilisateur avec des droits d’accès spécifiques. Cela garantit un contrôle précis et fiable des accès aux différentes zones du campus.

Parallèlement, j’ai développé le module de réservation des salles, permettant aux administrateurs de planifier les créneaux horaires pour plus de 300 salles. Les utilisateurs peuvent consulter l’état d’occupation des salles en temps réel et ajuster leurs réservations en cas de besoin. Des outils de recherche, des filtres et des alertes ont été intégrés pour faciliter la gestion et assurer une navigation rapide et efficace dans l’application. Les administrateurs peuvent également modifier ou annuler des réservations directement depuis l’interface.

Le développement a été réalisé avec Visual Studio en C# et le .NET Framework, permettant une gestion fluide des données et des interactions avec la base de données. Un point important a été portée à l’ergonomie de l’application, afin d’offrir une expérience aux utilisateur optimale.

Ma contribution a permis de créer un outil centralisé et sécurisé, offrant aux administrateurs un moyen efficace de gérer les accès et les réservations des salles.

1. Analyse UML/SysML

Suite à l'analyse des besoins, j’ai donc pris appuis sur le diagramme de cas d’utilisation, ou je pouvais voir toutes les étapes que je devais réaliser pour faire ma partie PGS et comprendre les échanges que je devais mettre en œuvre avec la base de données pour pouvoir faire fonctionner mon codage.

1. Installation de l’environment de Travail

Avant de passer à la partie codage, j’ai du faire quelque étapes qui sont :

Récupérer toutes les informations dont j’avais besoins.

Installer des paquets json très important pour la suite de mon codage et donc communiquer avec la base de données.

J’ai aussi pris en compte toutes les routes dont j’avais besoins pour relier mon code a la base de données.

La configuration des outils de versioning (Git) pour ne pas perdre de trace de ce que je faisais ainsi que de mettre en commun avec le reste du groupe.

1. Développement de la partie gestion des badges

Donc cette partie est la phase principale de mon travail dans mon projet, j’ai développé la fonctionnalité de création de badges. Lorsqu'un nouveau badge est créé, un identifiant unique est attribué à chaque utilisateur, et ce badge est associé à des droits d’accès spécifiques, comme l’accès à certaines salles. Ces informations sont stockées de manière sécurisée dans la base de données pour garantir une gestion fiable et cohérente.

J'ai également mis en place une fonctionnalité permettant de modifier les informations des badges existants. Cette fonctionnalité est essentielle pour effectuer des mises à jour lorsque les utilisateurs changent de rôle ou doivent accéder à de nouvelles salles. Grâce à cette option, les administrateurs peuvent rapidement ajuster les droits d’accès d’un utilisateur ou même changer les informations associées à un badge sans difficulté.

Pour la gestion de la désactivation des badges, une fonctionnalité a été ajoutée pour permettre aux administrateurs de désactiver un badge rapidement, par exemple, si un badge est perdu ou si un utilisateur quitte le campus. Cette option permet de garantir que l'accès aux salles est coupé immédiatement, assurant ainsi la sécurité du système.

Enfin, la possibilité de supprimer définitivement un badge a été implémentée. Cette fonctionnalité permet aux administrateurs de nettoyer la base de données en supprimant des badges qui ne sont plus nécessaires. La suppression est une action contrôlée et sécurisée, accessible uniquement aux utilisateurs autorisés, pour éviter toute manipulation accidentelle ou non souhaitée.

1. Développement de la partie gestion des salles

Pour la gestion des salles, ma première étape a été de concevoir une interface simple permettant aux administrateurs de gérer les réservations de plus de 300 salles. J’ai créé une fonctionnalité qui permet de réserver une salle en choisissant un créneau horaire, avec une mise à jour en temps réel de l’occupation des salles. Cela permet aux administrateurs de consulter facilement les disponibilités et d’éviter les erreurs de réservation.

J’ai également ajouté des options pour modifier ou annuler une réservation existante, offrant ainsi une flexibilité dans la gestion des salles. Des outils de recherche et des filtres permettent de trouver rapidement des salles disponibles en fonction de critère comme la capacité. Toutes ces actions sont sécurisées par des validations en temps réel pour garantir la fiabilité des données.

Enfin, toutes les informations sont stockées dans la base de données, assurant une gestion centralisée et cohérente des réservations. Ce système a permis d’optimiser l’utilisation des salles tout en offrant aux administrateurs un contrôle efficace et simple sur les réservations.

* 1. Tests et Validation

Une fois ma partie gestion des badges ainsi que la partie gestion des salles développées, j'ai donc effectué des tests unitaires et d'intégration pour m'assurer que chaque partie réaliser fonctionne correctement. Ce qui inclut :

Tester les interfaces graphiques pour voir si j’ai bien les résultats attendus.

Vérifier s’il y a des problèmes via la communication avec la base de données, vérifier si les informations effectuer se sont bien enregistré sur la base de données.

1. Réalisation de la fonction et/ou cas d'utilisation

**Présentation du cas d’utilisation** : Gérer les badges

Dans le cadre de mon projet, j'ai développé une application permettant la gestion complète des badges NFC. Celle-ci repose sur une interface principale qui centralise plusieurs fonctionnalités essentielles. L'utilisateur peut créer, modifier et supprimer un badge. Il peut également authentifier un badge en scannant celui-ci pour vérifier sa validité, et consulter la liste des utilisateurs enregistrés dans le système. Pour compléter ces fonctionnalités, j'ai développé une seconde interface reliée à la première. Cette interface permet de gérer les associations entre les badges et les utilisateurs. L’utilisateur peut facilement associer un badge à une personne ou désassocier un badge existant. La sélection des badges et des utilisateurs se fait grâce à des menus déroulants, offrant une navigation simple et rapide. Ce cas d'utilisation couvre ainsi toute la gestion des badges : depuis leur création jusqu'à leur association aux utilisateurs, en passant par leur modification, leur suppression et leur authentification. L'ensemble a été conçu pour garantir une utilisation simple, rapide et sécurisée.

**Justification du choix technologique :**

Pour réaliser mon projet, j'ai choisi d'utiliser Visual Studio 2022 comme environnement de développement. Ce choix est principalement dû au fait que cet IDE est celui avec lequel je suis le plus à l’aise. Il propose de nombreuses fonctionnalités pratiques comme un excellent débogueur, une gestion efficace des projets et une intégration parfaite avec le Framework .NET. Cela m’a permis de travailler de manière fluide et de gagner du temps dans le développement de l'application. Concernant le langage, j'ai opté pour C#, et ce pour plusieurs raisons. Tout d’abord, C# offre un large choix de fonctionnalités qui m'ont permis de répondre facilement aux besoins du projet, qu'il s'agisse de la gestion des interfaces graphiques ou de l'interaction avec les périphériques comme les lecteurs de badges. Ce langage est aussi réputé pour sa sécurité renforcée, grâce à sa gestion stricte des types et son système de gestion automatique de la mémoire, ce qui évite de nombreuses erreurs courantes dans le code. Enfin, C# est conçu pour fonctionner de manière optimale avec le Framework .NET, qui met à disposition un grand nombre de bibliothèques et d’outils prêts à l’emploi. Cela m’a permis de me concentrer sur les fonctionnalités spécifiques du projet sans avoir à créer des composants de base à chaque fois. J’ai aussi utilisé des paquets Json pour pouvoir communiquer avec la base de données ainsi que des routes. Pour la gestion des badges, j'ai choisi d'utiliser un lecteur NFC, qui présente plusieurs avantages. Le premier est sa rapidité d’utilisation : il suffit de poser le badge à proximité du lecteur et la connexion se fait immédiatement, sans qu’il soit nécessaire de taper un code ou d’insérer physiquement un badge. Cela simplifie l'expérience utilisateur et permet un accès plus fluide. De plus, la sécurité a été un critère important dans ce choix. Le lecteur NFC offre une communication chiffrée entre le badge et le lecteur, ce qui garantit que les données échangées sont protégées contre les interceptions ou les falsifications. Ainsi, avec Visual Studio 2022, C#, .NET et le lecteur NFC, j'ai pu réaliser une application performante, sécurisée et facile à utiliser pour la gestion des badges, tout en répondant efficacement aux besoins de sécurité et d'ergonomie du projet.

**Présentation du cas d’utilisation** : Gérer les salles

Dans le cadre de mon projet, j'ai créé une application pour la gestion des salles. L'interface principale permet à l'utilisateur de réserver une salle, de donner l'accès à une salle et de visualiser l'état d'occupation des différentes salles. Pour réserver une salle, l'utilisateur peut simplement choisir la date et l'heure, ce qui facilite la planification. Il peut aussi attribuer l'accès à certaines personnes pour chaque salle, en fonction des besoins. L'application offre aussi une vue en temps réel de l’occupation des salles, permettant ainsi de vérifier facilement si une salle est libre ou déjà occupée. Cela permet de gérer efficacement les réservations et d'éviter tout conflit. En résumé, l'application couvre toutes les étapes de la gestion des salles : de la réservation à l'attribution des accès, en passant par le suivi de l'occupation. Le tout a été conçu pour être simple, rapide et pratique.

**Justification du choix technologique :**

Pour la gestion des salles, j'ai continué avec Visual Studio 2022 et C# pour garder une cohérence avec la partie gestion des badges. Cependant, pour cette fonctionnalité, l'intégration avec .NET m'a permis de travailler facilement avec des outils spécifiques à la gestion des calendriers, des réservations et des bases de données. J'ai utilisé une base de données pour gérer les réservations et les accès aux différentes salles, ce qui permet une gestion centralisée des informations. Cette solution assure un suivi en temps réel de l'occupation des salles et facilite les mises à jour immédiates des réservations. Pour l'interface, j'ai opté pour des éléments comme des menus déroulants et des calendriers, permettant à l'utilisateur de réserver une salle facilement et de voir instantanément l'état d'occupation.

1. Une image contenant texte, diagramme, Parallèle, ligne

   Le contenu généré par l’IA peut être incorrect.Conception détaillée

Le module FrmGestionBadgesSalles constitue l’interface principale de gestion des badges. Il permet d’afficher la liste des badges via un DataGridView et de modifier leur état actif grâce à la case checkBoxIsActive. Il sert aussi de point d’entrée vers la visualisation des utilisateurs à travers le bouton btnAfficherUtilisateurs, qui ouvre le formulaire FormAfficherUtilisateurs.

FormAfficherUtilisateurs centralise l’affichage des données utilisateurs en utilisant un DataGridView alimenté par la méthode GetUtilisateursAsync() de ApiService. Ce formulaire permet une lecture claire et rapide des comptes enregistrés.

Le formulaire FormAssocierBadge gère l’association et la dissociation entre badges et utilisateurs. Il comprend deux ComboBox pour sélectionner l’utilisateur et le badge, ainsi que trois boutons d’action : Associer, Dissocier et Modifier l’état. Ces opérations déclenchent des appels vers ApiService, qui agit comme couche de communication avec l’API distante.

ApiService est le point central des échanges réseau. Il fournit des méthodes asynchrones pour récupérer, ajouter, supprimer ou modifier les entités Badge et Utilisateur, ainsi que pour gérer leurs associations. Il utilise les informations contenues dans AppSetting pour configurer les appels (URL de base et clé API).

La classe Badge représente un identifiant NFC, avec un nom, un identifiant numérique et un état d’activation. La méthode ToString() permet son affichage dans les interfaces. La classe Utilisateur, quant à elle, contient les informations personnelles et les droits d’un usager, liés à une Salle via l’attribut IdClasse.

La classe NFCReader permet la lecture de badges physiques à l’aide de la méthode asynchrone LireUIDNFCAsync(), utilisée notamment pour enregistrer ou identifier un badge existant.

Dans l’ensemble, cette architecture assure une gestion complète et fluide des utilisateurs et des badges NFC au sein d’un établissement, en s’appuyant sur une organisation claire entre interface, logique métier et communication réseau.

**Spécifications des données :**

Les échangent réaliser durant le projet se sont fait en JSON via API REST.

J’ai donc dû installer des packages NuGet sous format Json qui sont :

{

Install-Package System.Net.Http.Json

Install-Package System.Text.Json

Newtonsoft.Json

}

1. Tests unitaires
   1. 2.3. Tests unitaires du module matériel/logiciel

Une image contenant texte, capture d’écran, Parallèle, nombre

Le contenu généré par l’IA peut être incorrect.

Une image contenant texte, capture d’écran, Police, ligne

Le contenu généré par l’IA peut être incorrect.

Une image contenant texte, capture d’écran, Parallèle, nombre

Le contenu généré par l’IA peut être incorrect.

Une image contenant texte, capture d’écran, Police, algèbre

Le contenu généré par l’IA peut être incorrect.

Une image contenant texte, capture d’écran, nombre, Police

Le contenu généré par l’IA peut être incorrect.

Une image contenant texte, capture d’écran, nombre, Police

Le contenu généré par l’IA peut être incorrect.

1. Problèmes rencontrés

Au début de mon projet, j'ai rencontré des difficultés pour communiquer correctement avec l'API. J'ai rapidement compris que pour pouvoir traiter les données, il était nécessaire d'installer certains paquets comme Json.

Durant la phase de codage, j'ai fait face à plusieurs problèmes, car certains de mes codes ne donnaient pas les résultats attendus.

La communication avec l'API n'était pas fluide, et l'association des badges ne se faisait pas comme prévu. J'ai dû tester et ajuster le code à plusieurs reprises pour parvenir à une version fonctionnelle.

1. .Bilan de la réalisation personnelle

Le début de cette partie a été un peu compliqué, car je me suis retrouvé à réaliser certaines actions qui n'étaient finalement pas nécessaires.

Cependant, après m'être consacré pleinement au projet, j'ai rapidement compris les enjeux et les étapes à suivre. J'ai pu créer mes diagrammes et développer mon code en conséquence.

Bien sûr, des erreurs sont survenues, mais j'ai réussi à les résoudre et à faire en sorte que ma partie fonctionne correctement. Aujourd'hui, mes codes n'ont plus d'erreurs, la communication avec l'API se fait parfaitement, et les résultats sont désormais conformes aux attentes