ESPRIM

ClassCheck Absence Manager

Academic Attendance Management System

Development Team:

Mokni Hamdi
Oussama Fekih Hassen

https://github.com/Mk-1000/ClassCheck
hamdimokni@gmail.com
hssanoussama@gmail.com

December 31, 2024

Executive Summary

ClassCheck Absence Manager represents a state-of-the-art .NET Core application designed to revolutionize attendance management in educational institutions. This comprehensive system leverages cutting-edge technologies to deliver:

- Attendance tracking and monitoring
- Real-time analytics and reporting
- Role-based access control
- Seamless integration capabilities
- Mobile-responsive interface

Built on .NET Core 6.0 and utilizing modern software architecture principles, the system ensures scalability, maintainability, and security while providing an intuitive user experience for all stakeholders.

Contents

List of Figures

List of Tables

Introduction

1.1 Project Overview

ClassCheck Absence Manager is a state-of-the-art attendance management system designed to modernize and streamline educational administrative processes. This sophisticated solution leverages modern .NET technologies to provide:

- Real-time attendance tracking and monitoring
- Comprehensive reporting and analytics
- Role-based access control
- Mobile-responsive user interface

1.2 Problem Statement

Traditional attendance management systems face numerous challenges that impact educational institutions' efficiency and effectiveness. Key issues include:

- ⚠ Time-consuming paper-based processes
- ⚠ Limited accessibility to historical attendance data
- ⚠ Inefficient reporting mechanisms
- ⚠ Lack of real-time insights and analytics
- ⚠ Difficulty in maintaining attendance records across multiple courses

1.3 Project Objectives

The ClassCheck system aims to address these challenges through the following objectives:

1. **Accessibility**: Provide anywhere, anytime access to attendance data
2. **Accuracy**: Ensure data integrity and reduce human error
3. **Analytics**: Deliver comprehensive reporting and analytical capabilities
4. **Security**: Implement robust role-based access control

System Analysis and Design

2.1 Requirements Analysis

2.1.1 Functional Requirements

- **User Management**
  - User registration and authentication
  - Role-based access control
  - Profile management
- **Attendance Management**
  - Record and track student attendance
  - Manage course schedules
  - Handle multiple class sections
- **Reporting System**
  - Generate attendance reports
  - Export data in multiple formats
  - Create statistical analyses

2.1.2 Non-Functional Requirements

- **Performance**: Response time ¡ 2 seconds
- **Scalability**: Support for 10,000+ concurrent users
- **Reliability**: 99.9
- **Security**: Data encryption and secure authentication
- **Usability**: Intuitive interface with minimal training required

Technical Architecture

3.1 Technology Stack

primaryblue		
Backend Framework	.NET Core	6.0
Database	MySQL	8.0
ORM	Entity Framework Core	6.0
Frontend	Razor Pages	6.0
CSS Framework	Bootstrap	5.0
JavaScript	jQuery	3.6
Authentication	ASP.NET Identity	6.0

Table 3.1: Technology Stack Specification

3.2 System Architecture

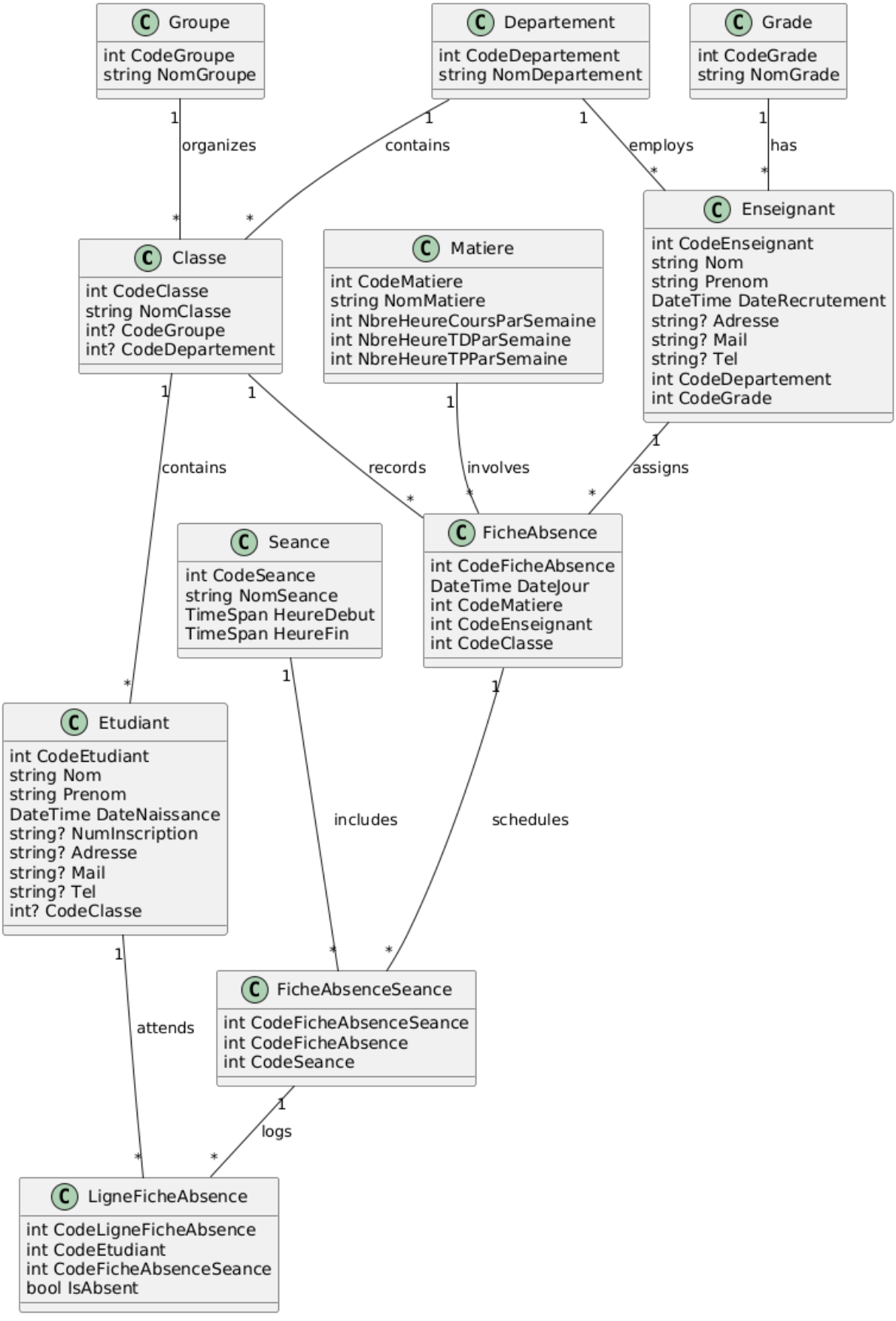

Figure 3.1: System Architecture Diagram

3.3 Core Entities

This section outlines the core entities of the system, detailing their attributes, relationships, and roles within the overall architecture. These entities enable robust data management and operational efficiency.

3.3.1 Entity Overview

The system's data models are grouped into the following categories:

- **Student Management:** Entities such as 'Etudiant' and 'Classe' manage student information and class assignments.
- **Attendance Tracking:** Entities like 'FicheAbsence' and 'LigneFicheAbsence' track and manage attendance records.

3.3.2 Detailed Entity Models

Student Management

Student Entity ('Etudiant'): The 'Etudiant' entity captures personal and academic information about students. It is linked to 'Classe' for class assignments and to 'LigneFicheAbsence' for attendance records.

Table 3.2: Attributes of the 'Etudiant' Entity

Attribute	Type	Description
CodeEtudiant	int	Unique identifier for the student.
Nom	string	Last name of the student.
Prenom	string	First name of the student.
...	...	...

```
1 public class Etudiant
2  {
3      [Key]
4      public int CodeEtudiant { get; set; }
5      public string Nom { get; set; }
6      public string Prenom { get; set; }
7      public DateTime DateNaissance { get; set; }
8      public string? NumInscription { get; set; }
9      public string? Adresse { get; set; }
10     public string? Mail { get; set; }
11     public string? Tel { get; set; }
12     [ForeignKey("Classe")]
13     public int? CodeClasse { get; set; }
14     // Navigation properties
15     public Classe? Classe { get; set; }
16     public ICollection<LigneFicheAbsence> LignesFicheAbsence { get;
     set; } = new List<LigneFicheAbsence>();
17
18 }
```

Listing 3.1: Student Entity Model

Attendance Tracking

Attendance Record (‘FicheAbsence‘): The ‘FicheAbsence‘ entity records attendance for specific classes, including references to ‘Matiere‘, ‘Enseignant‘, and ‘Classe‘.

Table 3.3: Attributes of the ‘FicheAbsence‘ Entity

Attribute	Type	Description
CodeFicheAbsence	int	Unique identifier for the attendance record.
DateJour	DateTime	Date of the attendance record.
...	...	...

```
public class FicheAbsence
 {
     [Key]
     public int CodeFicheAbsence { get; set; }

     [Required]
     public DateTime DateJour { get; set; }

     [Required]
     [ForeignKey("Matiere")]
     public int CodeMatiere { get; set; }

     [Required]
     [ForeignKey("Enseignant")]
     public int CodeEnseignant { get; set; }

     [Required]
     [ForeignKey("Classe")]
     public int CodeClasse { get; set; }

     // Navigation properties
     public Matiere? Matiere { get; set; }
     public Enseignant? Enseignant { get; set; }
     public Classe? Classe { get; set; }
     public ICollection<FicheAbsenceSeance> FichesAbsenceSeances { get;
    set; } = new List<FicheAbsenceSeance>();
 }
```

Listing 3.2: Attendance Record Model

3.3.3 Database Context Implementation

The ‘MyDbContext‘ class defines the database schema, including relationships and constraints. Below is the implementation:

```
public class MyDbContext : DbContext
{
    public MyDbContext(DbContextOptions<MyDbContext> options) : base(
    options) { }

    // DbSet properties
    public DbSet<Classe> Classes { get; set; }
    public DbSet<Seance> Seances { get; set; }
    ...

    // OnModelCreating configuration
    protected override void OnModelCreating(ModelBuilder modelBuilder)
    {
        // Primary Key configurations
        modelBuilder.Entity<Classe>().HasKey(c => c.CodeClasse);
        modelBuilder.Entity<Groupe>().HasKey(g => g.CodeGroupe);
        ...

        // Composite Key configurations
        modelBuilder.Entity<FicheAbsenceSeance>()
            .HasKey(fas => fas.CodeFicheAbsenceSeance);
        ...

        // Relationships and Foreign Key configurations
        modelBuilder.Entity<Classe>()
            .HasOne(c => c.Groupe)
            .WithMany(g => g.Classes)
            .HasForeignKey(c => c.CodeGroupe)
            .OnDelete(DeleteBehavior.SetNull);
        ...
    }
}
```

Listing 3.3: Database Context Implementation

3.3.4 Entity Relationship Model

Figure 3.2 illustrates the relationships among the core entities in the system.

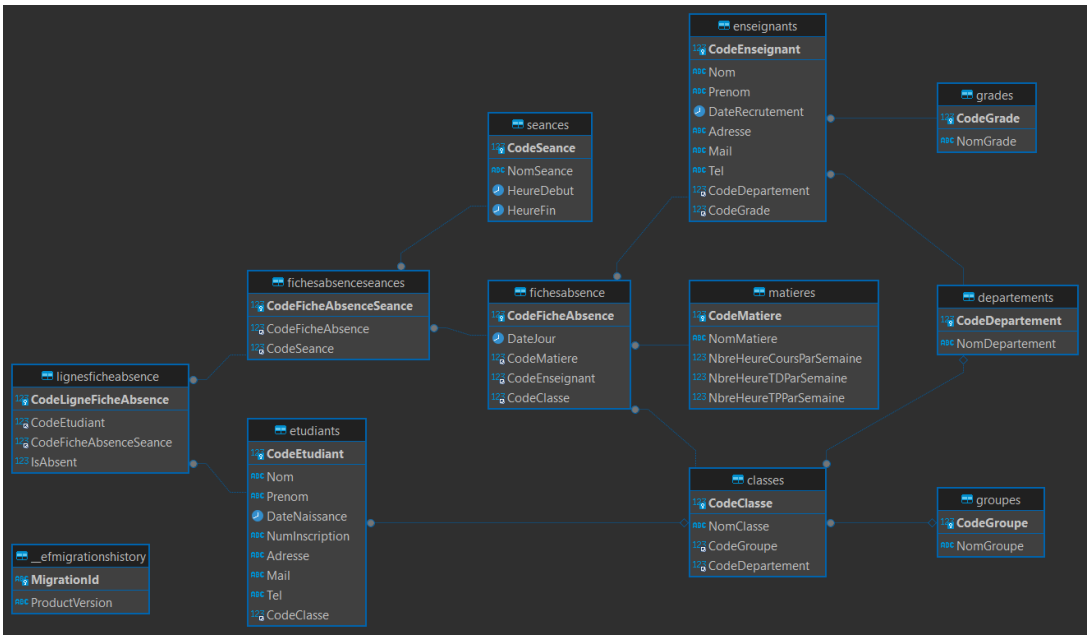

Figure 3.2: Entity Relationship Diagram

User Interface

4.1 Home Page

The home page serves as the central navigation hub for the system. It features a grid layout with interactive cards representing different management modules, such as **Etudiants**, **Classes**, **Groupes**, **Enseignants**, and more. Each card includes:

- **Title:** The name of the module (e.g., "Etudiant Management").
- **Icon:** A contextual icon for visual clarity.
- **Description:** A brief overview of the module's functionality.
- **Action Button:** A link to the respective module's management page.

Key Features:

- Clean and intuitive design for effortless navigation.
- Responsive layout ensuring usability on various devices.

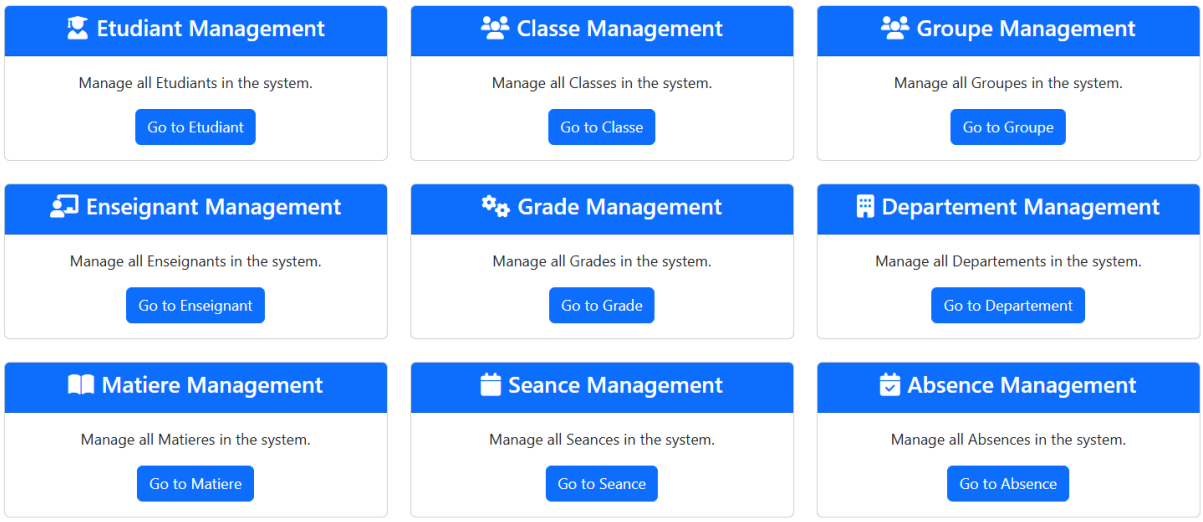

Figure 4.1: Home Page

4.2 Absence Management Page

The Absence Management page allows users to manage student absences effectively. It includes a table listing all attendance forms with the following columns:

- **Date:** The date of the absence.
- **Subject:** The associated subject.
- **Teacher:** The teacher responsible.
- **Class:** The class affected.
- **Sessions & Absences:** Details of sessions and absent students.
- **Actions:** Options for adding sessions, marking absences, editing, or deleting records.

Key Functionalities:

1. **Add New Attendance Record:** A modal form for creating a new absence record, capturing details like date, subject, teacher, and class.
2. **Mark Absences:** A modal for selecting absent students from a dropdown.
3. **Responsive Table:** A searchable and sortable table for quick data access.

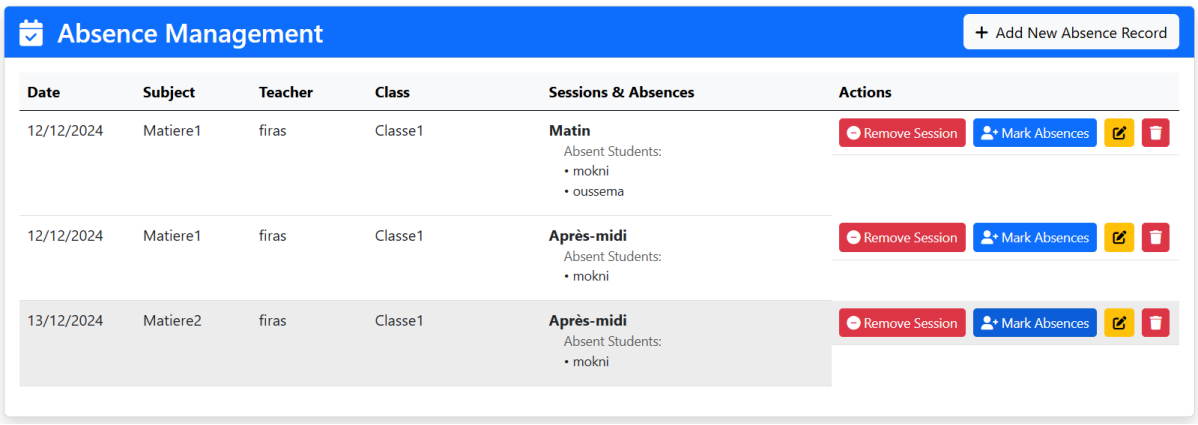

Figure 4.2: Absence Management Page

Future Enhancements

5.1 Planned Features

Future development plans include:

- Mobile application development
- Machine learning for attendance pattern analysis
- Integration with learning management systems
- Advanced reporting and analytics
- Automated notification system
- Biometric authentication integration

5.2 Technology Roadmap

primaryblue		
Phase 1	Mobile App Development	Q1 2025
Phase 2	AI/ML Integration	Q2 2025
Phase 3	Advanced Analytics	Q3 2025
Phase 4	System Integration	Q4 2025

Table 5.1: Development Roadmap

Conclusion

6.1 Project Achievements

The ClassCheck Absence Manager has successfully achieved its primary objectives:

- Improved data accuracy
- Enhanced reporting capabilities
- Streamlined administrative processes
- Increased user satisfaction

6.2 Lessons Learned

Key insights gained during development:

- Importance of thorough requirement analysis
- Benefits of modular architecture
- Need for scalable design from the start

References

Bibliography

[1] Microsoft (2024). .NET Documentation. https://docs.microsoft.com/en-us/dotnet/

[2] Microsoft (2024). Entity Framework Core Documentation. https://docs.microsoft.com/en-us/ef/core/

[3] Bootstrap Team (2024). Bootstrap Documentation. https://getbootstrap.com/docs/

[4] Microsoft (2024). Azure DevOps Documentation. https://docs.microsoft.com/en-us/azure/devops/

[5] OWASP (2024). Web Security Guidelines. https://owasp.org/www-project-web-security-testing-guide/