NCT

# NCT University Management System

## New Cairo Technology University

Comprehensive Academic Management Solution

Version 1.0.0 | December 2024

## 📋 Table of Contents

# 1. Overview

The **NCT University Management System** is a fully integrated desktop application developed in C# using WPF (Windows Presentation Foundation) with a modern, professional GUI. The system centralizes all university operations including student management, department administration, course scheduling, fee tracking, and academic grading.

> **Key Highlights:**
>
> - 🎓 Complete academic workflow management
> - 💰 Automated fee calculation based on department
> - 📊 Interactive grade distribution charts
> - 🔒 Data validation and integrity checks
> - 📈 Performance reports and analytics

# 2. Features

## Core Modules

| Module | Description |
|---|---|
| **Dashboard** | Real-time statistics, grade distribution pie chart, quick overview |
| **Students** | Full CRUD operations, department assignment, auto-generated IDs |
| **Departments** | Manage departments with unique annual fees, course/student counts |
| **Courses** | Course management with credits, year levels, department assignment |
| **Sections** | Section management with 40-student capacity, scheduling |
| **Fees** | Automatic fee calculation, payment tracking, status management |
| **Grades** | Score entry, automatic grade calculation, leniency algorithm |
| **Reports** | Performance analytics, course statistics, section utilization |

## Data Validation

- ✅ Email format validation

- ✅ Egyptian phone number format
- ✅ Student ID format (NCT + 5 digits)
- ✅ Score range validation (0-100)
- ✅ Duplicate record prevention
- ✅ Required field enforcement

## 3. Technology Stack

| Component | Technology |
|---|---|
| Framework | .NET 8.0 |
| UI Framework | WPF (Windows Presentation Foundation) |
| Architecture | MVVM (Model-View-ViewModel) |
| Database | SQLite with Entity Framework Core 8.0 |
| Charts | LiveCharts2 (SkiaSharp) |
| Toolkit | CommunityToolkit.Mvvm |

## 4. System Architecture

```
┌─────────────────────────────────────────────────────────────┐
│                     PRESENTATION LAYER                      │
│  ┌───────────────────────────────────────────────────────┐  │
│  │                     Views (XAML)                      │  │
│  │  Dashboard │ Students │ Departments │ Courses │ ...   │  │
│  └───────────────────────────────────────────────────────┘  │
│                            │                                │
│                            ▼                                │
│  ┌───────────────────────────────────────────────────────┐  │
│  │                   ViewModels (C#)                     │  │
│  │  Data Binding │ Commands │ Business Logic             │  │
│  └───────────────────────────────────────────────────────┘  │
└─────────────────────────────────────────────────────────────┘
                             │
                             ▼
┌─────────────────────────────────────────────────────────────┐
│                       BUSINESS LAYER                        │
│  ┌───────────────────────────────────────────────────────┐  │
│  │                       Services                        │  │
│  │  GradingService │ ReportService │ ValidationService   │  │
│  └───────────────────────────────────────────────────────┘  │
└─────────────────────────────────────────────────────────────┘
                             │
                             ▼
┌─────────────────────────────────────────────────────────────┐
│                         DATA LAYER                          │
│  ┌───────────────────────────────────────────────────────┐  │
│  │                 Entity Framework Core                 │  │
│  │             UniversityDbContext │ DbSeeder            │  │
│  └───────────────────────────────────────────────────────┘  │
│                            │                                │
│                            ▼                                │
│  ┌───────────────────────────────────────────────────────┐  │
│  │                 SQLite Database                    │    │
│  └───────────────────────────────────────────────────┘    │
└──────────────────────────────────────────────────────────────┘
```

## Project Structure

```
UniversityManagementSystem/
├── Models/                    # Entity classes
│   ├── Student.cs
│   ├── Department.cs
│   ├── Course.cs
│   ├── Section.cs
│   ├── Grade.cs
│   └── StudentFee.cs
├── ViewModels/               # MVVM ViewModels
├── Views/                    # XAML Views
├── Data/                     # Database layer
├── Services/                 # Business services
├── Converters/               # WPF value converters
├── Resources/                # Styles and themes
└── DOCUMENTATION/            # Documentation files
```

# 5. Entity Relationship Diagram (ERD)

```
                              ┌─────────────────────┐
                              │     DEPARTMENT      │
                              ├─────────────────────┤
                              │ PK  Id              │
                              │ UK  Code            │
                              │     Name            │
                              │     HeadOfDepartment│
                              │     Description     │
                              │     AnnualFees      │
                              └─────────┬───────────┘
                                        │
                       ┌────────────────┼────────────────┐
                       │ 1              │ 1              │
                       ▼ N              ▼ N              │
          ┌────────────────────┐  ┌────────────────────┐ │
          │      STUDENT       │  │       COURSE       │ │
          ├────────────────────┤  ├────────────────────┤ │
          │ PK  Id             │  │ PK  Id             │ │
          │ UK  StudentId      │  │ UK  Code           │ │
          │     Name           │  │     Name           │ │
          │ UK  Email          │  │     Credits        │ │
          │     Phone          │  │     YearLevel      │ │
          │     YearLevel      │  │ FK  DepartmentId ──┘ │
          │ FK  DepartmentId ──┘  │     Description    │
          │ FK  SectionId ──────┐ └─────────┬──────────┘
          │     EnrollmentDate  │           │ 1
          │     IsActive        │           ▼ N
          └──────────┬──────────┘ ┌────────────────────┐
                     │            │      SECTION       │
        ┌────────────┤            ├────────────────────┤
        │ 1          │            │ PK  Id             │
        ▼ N          │            │     Name           │
┌───────────────────┐│            │ FK  CourseId ──────┘
│   STUDENT_FEE     ││            │     Capacity (40)  │
├───────────────────┤│            │     Schedule       │
│ PK  Id            ││            │     Room           │
│ FK  StudentId ────┘│            └────────────────────┘
│     AcademicYear   │
│     Amount         │     1      ┌────────────────────┐
│     AmountPaid     │◄───────────│       GRADE        │
│     Status         │            ├────────────────────┤
│     AssignedDate   │            │ PK  Id             │
│     LastPaymentDate│            │ FK  StudentId      │
└───────────────────┘             │ FK  CourseId       │
                                  │     Assignment1    │
                                  │     Assignment2    │
                                  │     FinalExam      │
                                  │     TotalScore     │
                                  │     SymbolicGrade  │
                                  │     LeniencyApplied│
                                  │     GradedDate     │
                                  └────────────────────┘
```

## Relationships Summary

| Relationship | Type | Description |
|---|---|---|
| Department → Student | 1:N | One department has many students |
| Department → Course | 1:N | One department offers many courses |
| Course → Section | 1:N | One course has multiple sections |
| Section → Student | 1:N | One section contains many students (max 40) |
| Student → StudentFee | 1:N | One student has fees for each year |
| Student → Grade | 1:N | One student has grades for each course |
| Course → Grade | 1:N | One course has grades from many students |

# 6. Database Schema

## Department Table

```
CREATE TABLE Departments (
    Id INTEGER PRIMARY KEY AUTOINCREMENT,
    Code TEXT NOT NULL,
    Name TEXT NOT NULL,
    HeadOfDepartment TEXT,
    Description TEXT,
    AnnualFees DECIMAL(18,2) DEFAULT 0
);
```

## Student Table

```
CREATE TABLE Students (
    Id INTEGER PRIMARY KEY AUTOINCREMENT,
    StudentId TEXT,              -- Format: NCT00001
    Name TEXT NOT NULL,
    Email TEXT NOT NULL UNIQUE,
    Phone TEXT,
    YearLevel INTEGER CHECK(YearLevel BETWEEN 1 AND 4),
    DepartmentId INTEGER REFERENCES Departments(Id),
    SectionId INTEGER REFERENCES Sections(Id),
    EnrollmentDate DATETIME DEFAULT CURRENT_TIMESTAMP,
    IsActive BOOLEAN DEFAULT 1
);
```

## Course Table

```
CREATE TABLE Courses (
    Id INTEGER PRIMARY KEY AUTOINCREMENT,
    Code TEXT NOT NULL,          -- Format: CS101
    Name TEXT NOT NULL,
    Credits INTEGER CHECK(Credits BETWEEN 1 AND 6),
    YearLevel INTEGER CHECK(YearLevel BETWEEN 1 AND 4),
    DepartmentId INTEGER REFERENCES Departments(Id),
    Description TEXT
);
```

## Section Table

```
CREATE TABLE Sections (
    Id INTEGER PRIMARY KEY AUTOINCREMENT,
    Name TEXT NOT NULL,
    CourseId INTEGER NOT NULL REFERENCES Courses(Id),
    Capacity INTEGER DEFAULT 40,
    Schedule TEXT,
    Room TEXT
);
```

## StudentFee Table

```
CREATE TABLE StudentFees (
    Id INTEGER PRIMARY KEY AUTOINCREMENT,
    StudentId INTEGER NOT NULL REFERENCES Students(Id) ON DELETE CASCADE,
    AcademicYear INTEGER CHECK(AcademicYear BETWEEN 1 AND 4),
    Amount DECIMAL(18,2) NOT NULL,
    AmountPaid DECIMAL(18,2) DEFAULT 0,
    Status TEXT DEFAULT 'Pending',  -- Pending, Partial, Paid, Overdue
    AssignedDate DATETIME DEFAULT CURRENT_TIMESTAMP,
    LastPaymentDate DATETIME
);
```

## Grade Table

```
CREATE TABLE Grades (
    Id INTEGER PRIMARY KEY AUTOINCREMENT,
    StudentId INTEGER NOT NULL REFERENCES Students(Id) ON DELETE CASCADE,
    CourseId INTEGER NOT NULL REFERENCES Courses(Id),
    Assignment1 REAL,            -- 0-100
    Assignment2 REAL,            -- 0-100
    FinalExam REAL,              -- 0-100
    TotalScore REAL,             -- Calculated weighted score
    SymbolicGrade TEXT,          -- D, M, P, NA
    LeniencyApplied BOOLEAN DEFAULT 0,
    GradedDate DATETIME,
    UNIQUE(StudentId, CourseId)
);
```

# 7. Modules Description

## 1. Dashboard Module

**Purpose:** Provides an at-a-glance overview of the university system

- Total students, courses, sections, departments count
- Active vs inactive student ratio
- Interactive pie chart showing grade distribution
- Quick navigation to other modules

## 2. Students Module

**Purpose:** Complete student lifecycle management

- Add, edit, delete students
- Auto-generated Student IDs (NCT format)
- Department and section assignment
- Automatic fee generation on enrollment
- Search and filter by name, year, department

## 3. Departments Module

**Purpose:** Manage academic departments and their fee structures

- Create departments with unique codes
- Set annual tuition fees per department
- View student and course counts
- Automatic fee updates when fees change

## 4. Fees Module

**Purpose:** Financial management and payment tracking

- Automatic fee calculation based on department
- Payment recording (full or partial)
- Status tracking: Pending Partial Paid Overdue
- Summary statistics (total, collected, pending)

## 5. Grades Module

**Purpose:** Academic assessment and grade calculation

- Enter Assignment 1, Assignment 2, Final Exam scores
- Automatic grade calculation with weighted formula
- Leniency algorithm for borderline cases
- Real-time grade preview

# 8. Grading System

## Grade Calculation Formula

> **Total Score = (Assignment1 × 20%) + (Assignment2 × 20%) + (FinalExam × 60%)**

## Grade Thresholds

| Grade | Symbol | Score Range | Description |
|---|---|---|---|
| Distinction | D | 85 - 100 | Excellent performance |
| Merit | M | 70 - 84.99 | Very good performance |
| Pass | P | 50 - 69.99 | Satisfactory performance |
| Not Achieved | NA | 0 - 49.99 | Below passing standard |

## Leniency Algorithm

The system applies a **leniency algorithm** for students who:

1. Show improvement (Assignment2 > Assignment1)
2. Are within 2 points of the next grade boundary

> **Example:** A student with 83.5 total score showing improvement would be upgraded from Merit (M) to Distinction (D).

```
// Leniency conditions (C# pseudocode)
bool showsImprovement = Assignment2 > Assignment1;
bool nearBoundary = score >= (threshold - 2) && score < threshold;

if (showsImprovement && nearBoundary)
{
    ApplyLeniency(); // Upgrade to next grade
}
```

# 9. Installation & Setup

## Prerequisites

- Windows 10/11
- .NET 8.0 SDK or later
- Visual Studio 2022 or VS Code with C# extension

## Installation Steps

```
1. Navigate to project folder
   cd "C:\Users\YourUsername\Desktop\ZIAD task\UniversityManagementSystem"

2. Restore NuGet packages
   dotnet restore

3. Build the project
   dotnet build

4. Run the application
   dotnet run
```

## First Run

The database (`university.db`) is automatically created with sample data including:

- 4 Departments (CS, IT, BUS, ENG)
- 16 Courses (4 per department)
- 32 Sections
- 40 Students
- Fee records
- Sample grades

# 10. Color Palette & Branding

## NCT University Colors

| Color | Hex Code | Usage |
|---|---|---|

|  |  |  |
|---|---|---|
| □ Navy Blue (Primary) | #112250 | Headers, sidebar, primary buttons |
| □ Gold (Accent) | #cb810d | Highlights, hover states, badges |
| □ Gray (Secondary) | #75757d | Secondary text, borders |
| □ White | #FFFFFF | Backgrounds, cards |

## Status Colors

| Status | Color | Hex |
|---|---|---|
| Success/Paid | Green | #10B981 |
| Warning/Pending | Orange | #F59E0B |
| Error/Overdue | Red | #EF4444 |
| Info/Partial | Blue | #3B82F6 |

**New Cairo Technology University**

NCT University Management System v1.0.0

December 2024

Excellence in Education