UCMS - University Course Management System

A comprehensive and colorful terminal-based University Course Management System

Click to visit the repository!

Features • Installation • Usage • Project Structure

Table of Contents

Overview

UCMS (University Course Management System) is a modern, colorful, and user-friendly terminal application designed to streamline academic administration. Built with Java, it features a vibrant ANSI-colored interface that enhances user experience while managing students, courses, instructors, and enrollments.

Why UCMS?

- ✨ **Colorful Interface** - Beautiful ANSI-colored UI with icons and emojis
- 🌓 **Robust Validation** - Comprehensive input validation for all user entries
- 🎯 **Role-Based Access** - Different interfaces for Admin, Lecturer, and Student
- 📊 **Data Management** - Efficient handling of academic records
- 🚀 **Easy to Use** - Intuitive menus and clear navigation

⬆ Back to Table of Contents

Features

Admin Features

- **User Management**
  - Add, view, update, and delete students
  - Manage lecturer accounts
  - View all system users
- **Course Management**
  - Create and manage courses
  - Assign lecturers to courses
  - Manage course modules
  - Set course capacity and credits
- **Reporting**
  - Generate comprehensive system reports
  - View enrollment statistics
  - Monitor system activity

Lecturer Features

- **Course Management**
  - View assigned courses
  - Access student rosters
  - Manage course modules
- **Grade Management**
  - Input and update student grades
  - View grade distributions
  - Track student performance
- **Student Interaction**
  - View enrolled students
  - Access student information

Student Features

- **Course Enrollment**
  - Browse available courses
  - Enroll in courses
  - Drop courses
  - View course modules
- **Academic Records**
  - View enrolled courses
  - Check grades
  - Monitor academic progress
- **Profile Management**
  - Update personal information
  - View academic history

↑ Back to Table of Contents

System Requirements

- **Java Development Kit (JDK)**: Version 8 or higher
- **Operating System**: Windows, macOS, or Linux
- **Terminal**: Any terminal with ANSI color support
  - Windows 10+ (Command Prompt, PowerShell, Windows Terminal)
  - macOS Terminal
  - Linux Terminal
- **IDE** (Optional): IntelliJ IDEA, Eclipse, or VS Code with Java extensions

⬆ Back to Table of Contents

Installation

Step 1: Clone the Repository

```
git clone https://github.com/YOUR_USERNAME/UCMS.git
cd UCMS
```

Step 2: Compile the Project

```
javac -d out src/**/*.java src/*.java
```

Step 3: Run the Application

```
java -cp out Main
```

Alternative: Using IntelliJ IDEA

1. Open IntelliJ IDEA
2. Select **File** → **Open** and navigate to the project folder
3. Ensure JDK 8+ is configured in **File** → **Project Structure**
4. Right-click on `Main.java` and select **Run 'Main.main()'**

⬆ Back to Table of Contents

Usage

Starting the Application

When you run UCMS, you'll be greeted with a colorful ASCII art banner:

```
============================================================
                    PICK YOUR ROLE
```

```
============================================================
    1. Admin
    2. Lecturer
    3. Student
    4. Exit
************************************************************
Enter your choice:
```

Admin Workflow

1. Select **Admin** role
2. Enter login credentials (firstname, lastname, email, phone, password)
3. Access Admin menu:
    - Manage Courses
    - Manage Students
    - Manage Lecturers
    - Generate Reports

Sample Report Output

```
============================================================

                  UCMS SYSTEM REPORT

============================================================
Total Number of Courses           : 10
Total Number of Students Enrolled : 500
Total Number of Lecturers         : 50
------------------------------------------------------------
Report generated by: Admin Name
Date: 08-Nov-2025
============================================================
```

⬆ Back to Table of Contents

User Roles

- **Admin**: Full access to all system features, including user and course management, and reporting.
- **Lecturer**: Access to courses they teach, ability to manage grades, and communicate with students.
- **Student**: Access to enrolled courses, ability to check grades, and manage personal information.

⬆ Back to Table of Contents

Project Structure

```
SoftDev_UCMS/
│
├── .idea/                          # IntelliJ IDEA project settings
├── assets/
│   └── Exercise_2.pdf              # Project specification document
│
├── out/                            # Compiled class files
│
├── src/
│   ├── CourseManagement/
│   │   ├── Course.java             # Course entity and management
│   │   ├── CourseManagementModule.java
│   │   ├── Module.java             # Course module entity
│   │   └── ModuleManagement.java   # Module operations
│   │
│   ├── Interfaces/
│   │   ├── Assignable.java         # Interface for assignable entities
│   │   ├── Authenticatable.java    # Interface for authentication
│   │   ├── Displayable.java        # Interface for displayable entities
│   │   └── Searcheable.java        # Interface for searchable entities
│   │
│   ├── UserManagement/
│   │   ├── Admin.java              # Admin user class
│   │   ├── Lecturer.java           # Lecturer user class
│   │   ├── LecturerManagementModule.java
│   │   ├── Student.java            # Student user class
│   │   ├── StudentManagementModule.java
│   │   └── User.java               # Base user class
│   │
│   ├── Utilities/
│   │   └── Utility.java            # Utility functions and helpers
│   │
│   ├── HelperFunctions.java        # General helper functions
│   ├── Main.java                   # Application entry point
│   ├── UCMS.java                   # Main system controller
│   ├── UIHelper.java               # UI utility functions
│   └── UserRoles.java              # User role definitions
│
├── .gitignore                      # Git ignore file
├── README.md                       # This file
├── README.pdf                      # PDF version of README
└── SoftDev_UCMS.iml                # IntelliJ IDEA module file
```

⬆ Back to Table of Contents

UI Components

UIHelper Class

The `UIHelper.java` file provides utility functions for creating a colorful terminal interface:

- **ANSI Color Codes**: Utilizes ANSI escape codes for vibrant colors and text formatting
- **Icons and Emojis**: Enhances visual appeal and user guidance
- **Formatted Menus**: Creates consistent and attractive menu displays
- **Headers and Dividers**: Provides visual separation and structure

Key UI Features

- Color-coded messages (success in green, errors in red, warnings in yellow)
- Formatted tables for displaying data
- Progress indicators and status messages
- Clear and intuitive navigation prompts

⬆ Back to Table of Contents

Validation Features

Input Validation (Utility.java)

- **Email Validation**: Ensures valid email format
- **Phone Number Validation**: Checks for proper phone number format
- **Password Strength**: Validates password complexity requirements
- **Numeric Input**: Validates integer and numeric inputs
- **Date Validation**: Ensures valid date formats
- **Range Checks**: Validates values within acceptable ranges

Error Handling

- Graceful handling of invalid inputs
- User-friendly error messages
- Input retry mechanisms
- Exception handling throughout the application
- Confirmation prompts for critical actions (e.g., deletions)

⬆ Back to Table of Contents

Contributing

We welcome contributions! Please follow these guidelines:

1. **Fork the repository**
2. **Create a feature branch**: `git checkout -b feature/amazing-feature`
3. **Commit changes**: `git commit -m 'Add amazing feature'`
4. **Push to branch**: `git push origin feature/amazing-feature`
5. **Open a Pull Request`

Development Guidelines

- Follow Java coding conventions and best practices

- Use meaningful variable and method names
- Write clear comments for complex logic
- Test all features thoroughly before submitting
- Update documentation for any changes
- Ensure code compiles without errors or warnings

Code Style

- Use proper indentation (4 spaces)
- Follow Object-Oriented Programming principles
- Implement proper encapsulation
- Use interfaces where appropriate
- Write clean, maintainable code

⬆ Back to Table of Contents

License

This project is licensed under the MIT License - see the LICENSE file for details.

⬆ Back to Table of Contents

Contact

For questions or feedback regarding this implementation:

- Review the code comments for detailed explanations
- Check the exercise specification document
- Refer to course materials for OOP concepts

⬆ Back to Table of Contents

Made with ♥ for academic excellence