***SOFTWARE REQUIREMENTS SPECIFICATION***

Coursely Course Management Software

Christen Ford

Version 1.0

Table of Contents

[**1.0** **Introduction** 4](#_Toc526948493)

[**1.1 Goals and objectives** 4](#_Toc526948494)

[**1.2 Statement of scope** 4](#_Toc526948495)

[**1.3 Software context** 4](#_Toc526948496)

[**1.4 Major constraints** 5](#_Toc526948497)

[**2.0 Usage Scenario** 6](#_Toc526948498)

[**2.1 User profiles** 6](#_Toc526948499)

[**2.2 Use-cases** 7](#_Toc526948500)

[**3.0** **Data Model and Description** 9](#_Toc526948501)

[**3.1 Data description** 9](#_Toc526948502)

[**3.2 Data objects** 10](#_Toc526948503)

[**3.2.1 Course** 10](#_Toc526948504)

[**3.2.2 Section** 10](#_Toc526948505)

[**3.2.3 Student** 10](#_Toc526948506)

[**3.2.4 Instructor** 11](#_Toc526948507)

[**3.2.5 Login Information** 11](#_Toc526948508)

[**3.2.6 Building** 11](#_Toc526948509)

[**3.2.7 Department** 11](#_Toc526948510)

[**3.2.8 Majors** 12](#_Toc526948511)

[**3.2.9 Schedule** 12](#_Toc526948512)

[**3.3 Relationships** 12](#_Toc526948513)

[**3.4 Complete data model** 12](#_Toc526948514)

[**3.5 Data dictionary** 15](#_Toc526948515)

[**4.0 Functional Model and Description** 19](#_Toc526948516)

[**4.1 Description of major functions** 19](#_Toc526948517)

[**4.1.1 Major functionality** 19](#_Toc526948518)

[**4.1.2 Functional requirements** 19](#_Toc526948519)

[**4.1.3 Non-functional requirements** 22](#_Toc526948520)

[**4.2 Software interface description** 24](#_Toc526948521)

[**4.2.1 External system interfaces** 24](#_Toc526948522)

[**4.2.2 Human interfaces** 24](#_Toc526948523)

[**5.0** **Restrictions, Limitations, and Constraints** 24](#_Toc526948524)

1. **Introduction**

**1.1 Goals and objectives**

The major goal of Coursely is to provide university students the means to electronically manage their course schedule. Objectives for Coursely are as follows:

**1.2 Statement of scope**

Coursely provides facilities for course management. First, users may view courses in the course catalog. Students may add and remove courses to and from their schedule as well as view their schedule. Advisors may provide course prerequisite overrides for advisees. Finally, administrators may administer user accounts and manage the course catalog. This includes features for adding and removing courses from the catalog as well an archiving and unarchiving them.

Major user inputs of Coursely include:

* User information: First and last name; university identification number, university email address.
* Course information: Name, description, credit hours, terms offered.

Processing functionality of Coursely include:

**1.3 Software context**

Course management software is usually part of a larger educational enterprise planning systems (ERP). An excellent example of ERP in action is the highly customized version of Peoplesoft utilized by Lorain County Community College (LCCC). In addition to offering course management, LCCC’s Peoplesoft variant interfaces with email software, manages tuition, provides transcripts, offers degree planning, etcetera… One may ask then, with many educational ERP systems providing course management software, as well as a plethora of other useful features, why Coursely?

Despite the availability of ERP systems, many universities still choose to implement their own systems, likely as a cost saving measure. There is nothing wrong with this option if the software is implemented correctly. However, when the software is implemented incorrectly, students may not be able to successfully use the software to do what they need to do. The software may not function properly, and it may, in the worst case, cause monetary damages if the software prevents students from accomplishing time-sensitive tasks in a timely manner.

Coursely provides a compromise to those universities that do not have the financial means to employ a full-fledged ERP system and do not have the resources to develop their own course management software. By positioning Coursely in the middle of the market between home grown solutions and ERP systems, Coursely provides value to those not served by the current course management software market. In addition, Coursely is purely dedicated to course management. No additional features are included with the software. This allows Coursely to embrace a minimalist footprint, while providing all major course management functionality provided by educational ERP systems.

**1.4 Major constraints**

Restrictions on student records is a large constraint placed on Coursely. The software necessarily requires access to student records, so it may check prerequisite course fulfillment prior to letting students enroll in courses. This presents Coursely with several options to handling this constraint:

1. Coursely may interface with a student records system to request the information it needs. This is likely the best approach. It is modular; therefore, the student records system may be swapped in and out. It also places all security concerns over preserving student’s privacy to the student records system.
2. Coursely may store student records in the same data store used to keep track of the course catalog. While this is a feasible solution, it is not optimal. This option would provide a performance benefit as Coursely would only have to interface with a single data store, but this would also introduce the possibility of administrators having access to student records. A major security concern, as the only people who should have access to student records are those with permission from the university.

In a realistic scenario, option one would be the obvious choice, for the reasons outlined. Option two also introduces data duplication. Regardless of whether student records are maintained by the Coursely software, or by a separate system in use at the university; if option two is taken, then someone with proper permissions would have to maintain the student records in the coarsely data store, while simultaneously having the university keep their own separate records. This could introduce data inconsistencies, synchronization issues, and authenticity conflicts. If new student records data is entered into Coursely that has not been entered into the student records system yet, how does Coursely know whether the data can be trusted? How is Coursely certain the data is authentic? Clearly, the best option available to Coursely is to utilize option one.

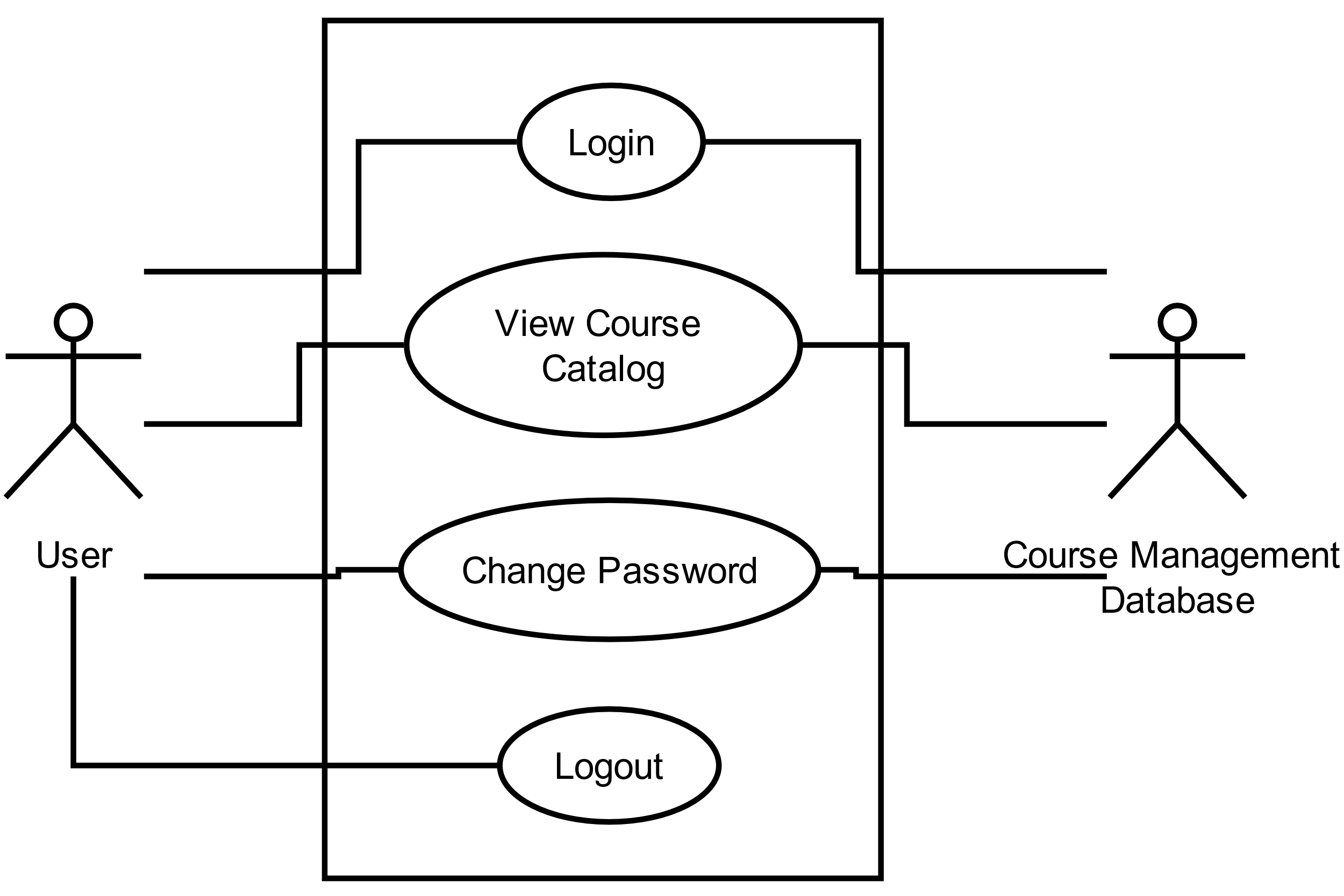
**2.0 Usage Scenario**

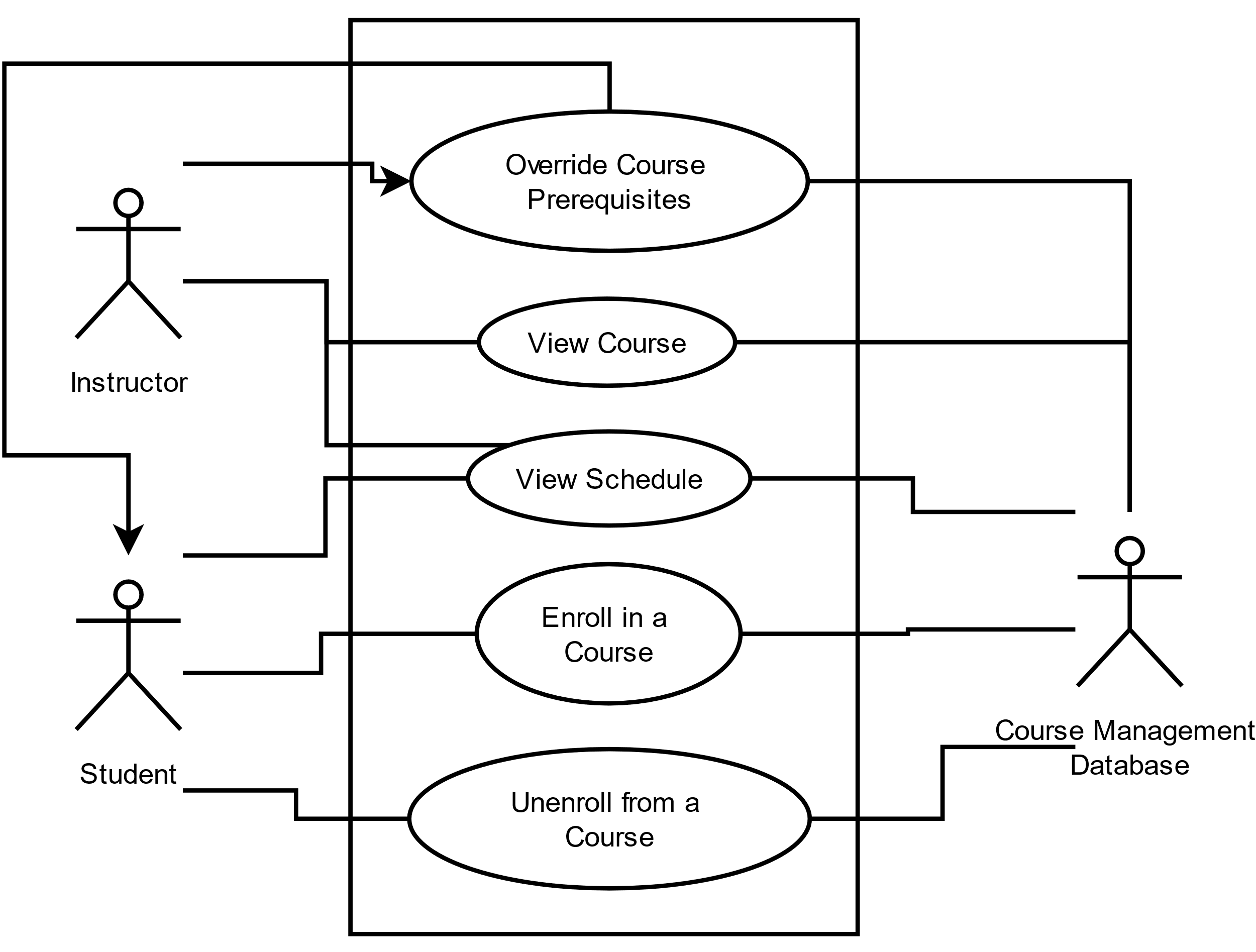
**2.1 User profiles**

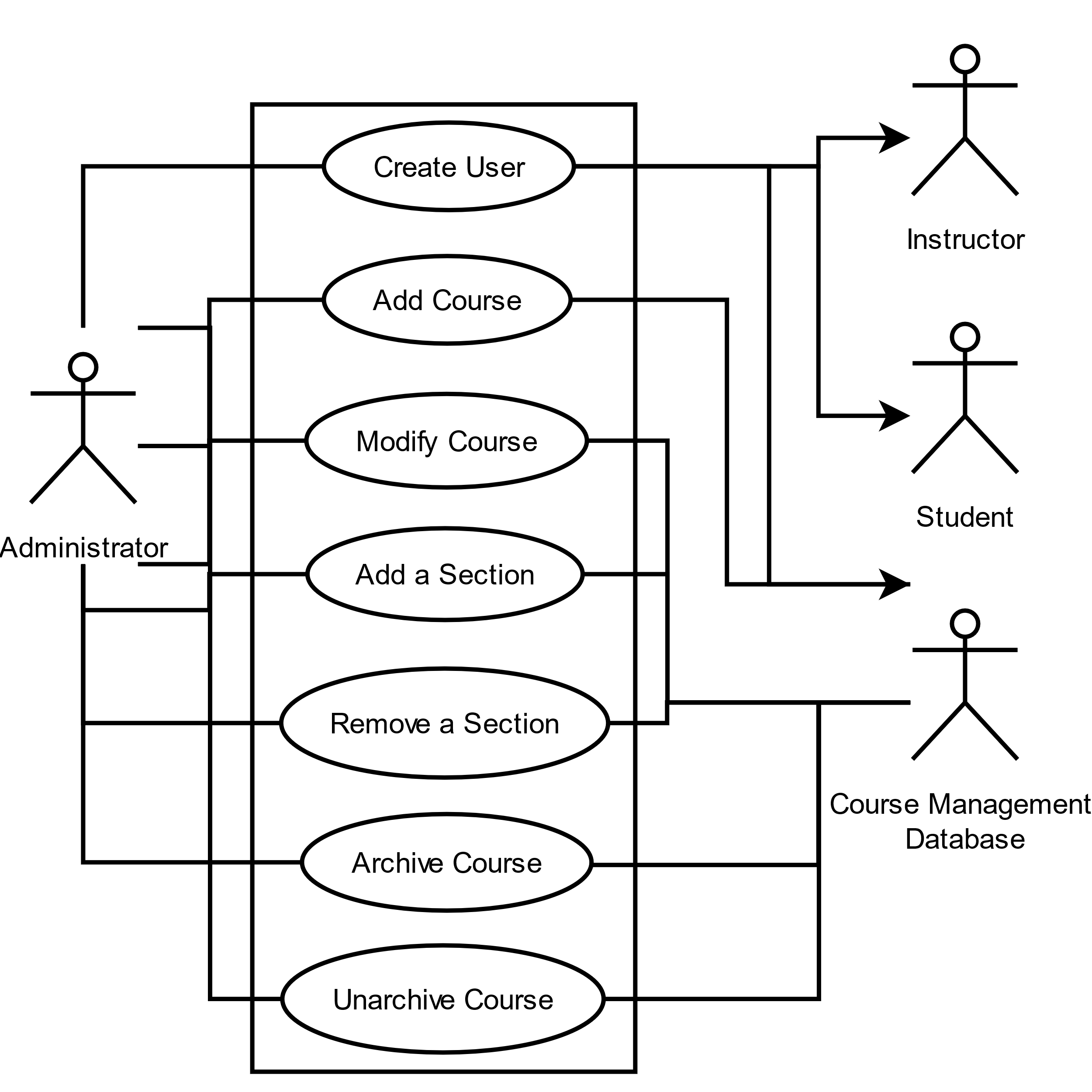
Coursely has three (3) user profiles:

* Student: A student at the university where Coursely is employed. Students are the primary user of Coursely and may consist of undergraduate students up to doctoral students.
* Instructor: An instructor is the other primary user of Coursely. They may view the courses they are schedule to teach, as well as the number of students who have signed up for them. Instructors may also act as advisors for students.
* Administrator: An administrator is an invisible user of Coursely. They do not directly interact with students or advisors except to create their accounts and assign them their roles. Administrators are also responsible for managing the course catalog employed by Coursely.

**2.2 Use-cases**







1. **Data Model and Description**

**3.1 Data description**

The following presents the various data objects involved in the system. These include courses, students, and instructors. Courses and students are the main data objects utilized by the system. Instructors provide additional information regarding courses and are related to students via an advisor-advisee relationship.

**3.2 Data objects**

Following is a detailed description of the data objects presented in the previous section:

**3.2.1 Course**

A course is one of the main data objects used by Coursely. It contains the following data:

* Course name
* Course number
* Course department
* Course description
* Prerequisites
* Semester(s) offered
* Day(s) offered
* Section(s)
* Credits
* Archived

**3.2.2 Section**

This data object used by Coursely represents an individual section of a course. Courses may have different section with different attributes.

* Section number
* Day(s)
* Start date
* End date
* Start time
* End time
* Building
* Room
* Instructor(s)
* Student(s)
* Year
* Semester
* Max enrollment

**3.2.3 Student**

Another data object used by coarsely is the student. It contains the following data:

* First name
* Last name
* Identification number
* Email
* Phone number
* Courses completed
* Schedule
* Grades

**3.2.4 Instructor**

The next data object used in the Coursely system is the instructor. It contains the following data:

* First name
* Last name
* Identification number
* Email
* Phone number
* Courses taught
* Office hours
* Advisees

**3.2.5 Login Information**

This data object is used to represent user login information.

* Username
* Password

**3.2.6 Building**

This data object is used to represent university building information.

* Name
* Abbreviation

**3.2.7 Department**

This data object is used to represent university department information.

* Name
* Abbreviation

**3.2.8 Majors**

* Name
* Description

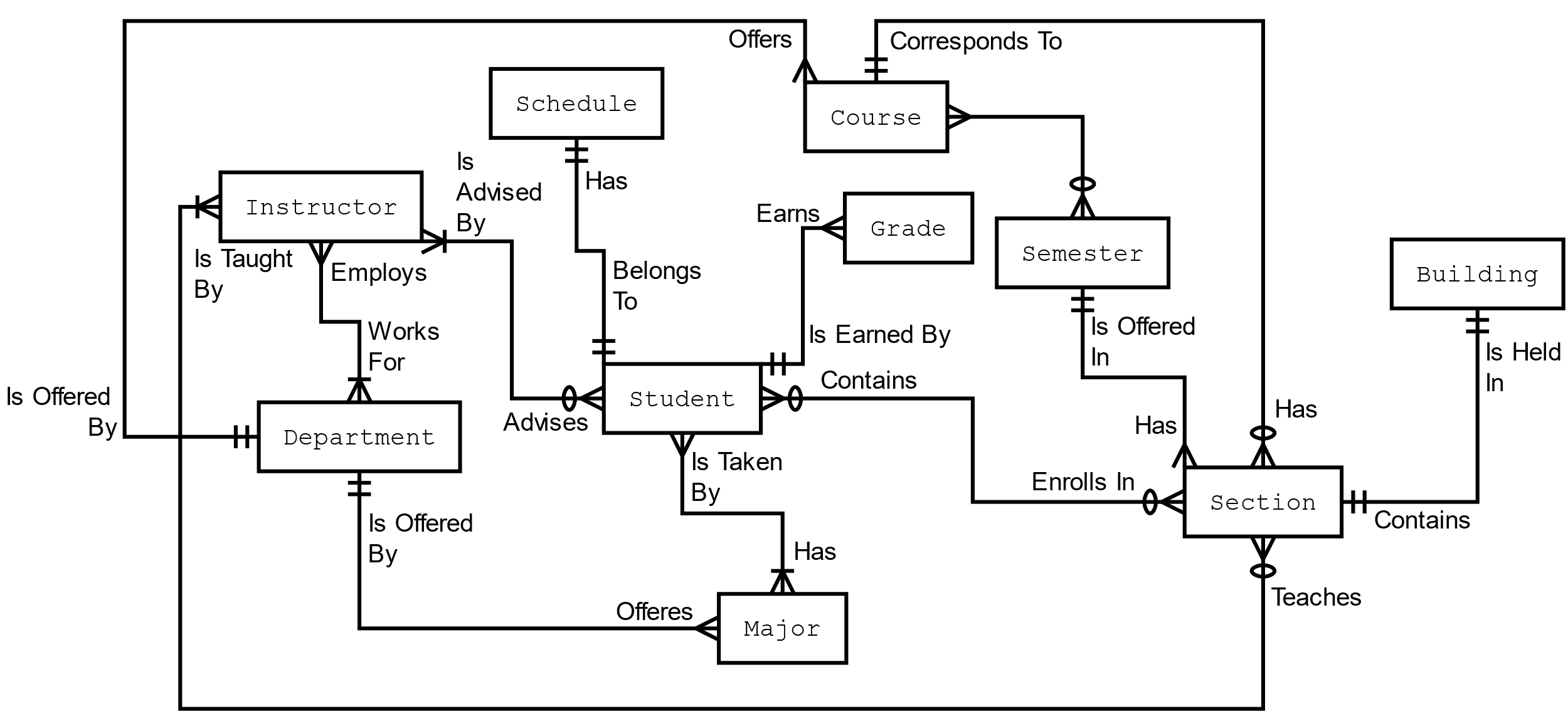
**3.2.9 Schedule**

This data object is used to represent student schedule information.

* Course sections

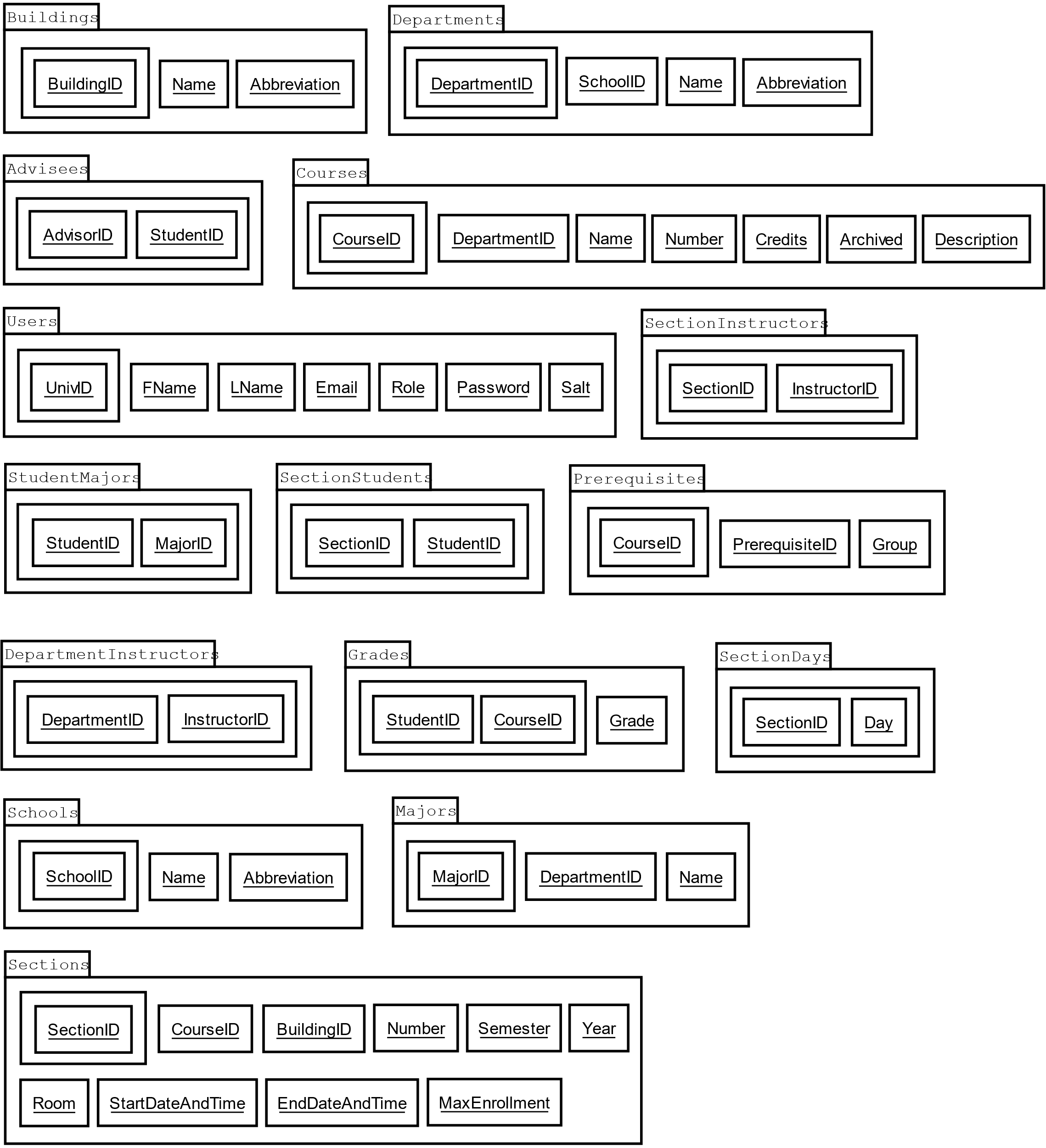
**3.3 Relationships**

The relationships of the various data objects in Coursely are as follows:



**3.4 Complete data model**

An entity relationship diagram details the entities that make up a system as well as their data containing attributes as well as any functional dependencies the entities may have on each other. Usually such entity relationship diagrams are used to develop the schema for a relational database. This section details the full schema for the Coursely database. Entities are represented as rectangles with plural tags while attributes are represented as singular nouns located within rectangles in the entities. An entities primary key is indicated by a second rectangle that groups one or more attributes within an entity.



|  |  |  |
| --- | --- | --- |
| Description of the Tables Present in the Coursely ERD | | |
| Table Name | Related To | Description |
| Advisees | Users | Holds the relationships between an advisee and advisor. |
| Buildings | N/A | This table holds records related to buildings on campus. It is used by the Sections table to indicate where the section is held. |
| Courses | Departments | This table holds records regarding a course, it is one of the primary tables in the database model. |
| DepartmentInstructors | Departments, Users | This table relates instructors to the departments they belong to. |
| Departments | Schools | This table holds records regarding a department within a school. |
| Grades | Courses, Users | This table holds a student’s academic record. |
| Majors | Departments | This table holds records regarding academic majors. |
| Prerequisites | Courses | This table holds records relating courses to their prerequisites. |
| Schools | N/A | This table holds records regarding academic schools. |
| SectionDays | Sections | This table holds records regarding the days on which a section is held. |
| SectionInstructors | Sections, Users | This table holds records regarding the instructors that teach a section. |
| Sections | Buildings, Courses | This table holds records regarding a course section. |
| SectionStudents | Sections, Users | This table holds records regarding the students enrolled in a section. |
| StudentMajors | Majors, Users | This table holds records relating students to their majors. |
| Users | N/A | This table holds records regarding users in the system. |

**3.5 Data dictionary**

The data dictionary lists all the data utilized by the various objects in the Coursely system. The data dictionary acts as a reference for developers implementing the software. It also provides structure for the data store(s) that will house the data defined in the dictionary. The data dictionary is organized by field name, data type, and description. The description may additionally state any constraints that may be placed on the respective field. Coursely’s data dictionary is as follows:

|  |  |  |
| --- | --- | --- |
| **Data Dictionary for Coursely** | | |
| **Name** | **Data Type** | **Description** |
| Abbreviation | String | The abbreviation of a name. Constrained to 4 characters in length. |
| AdvisorID­ | Integer | Used to pair an instructor with a student. Foreign key of UnivID. |
| Archived | Boolean | Whether a course is archived or not. |
| BuildingID | Integer | Uniquely identifies a building in the system. |
| CourseID | Integer | Uniquely identifies a course in the system. |
| Credits | Integer | The amount of credit hours a course is worth. |
| Day | String | One of: {Monday, Tuesday, Wednesday, Thursday, Friday, Saturday} |
| DepartmentID | Integer | Uniquely identifies a department in the system. |
| Description | String | A description for a course. Constrained to 1,024 characters in length. |
| Email | String | A users email address, constrained to 128 characters in length. |
| EndDateAndTime | DateTime | An end date and end time. Used to indicate the last day of a section as well as the time the section ends each day. |
| FName | String | A users first name. Constrained to 32 characters in length. |
| Grade | String | A grade that a student has received for a course. |
| GroupID | Integer | Used to group prerequisites together. |
| InstructorID | Integer | Used to relate instructors to sections. |
| LName | String | A users last name. Constrained to 32 characters in length. |
| MajorID | Integer | Uniquely identifies a major in the system. |
| MaxEnrollment | Integer | The maximum amount of student that can enroll in a section. |
| Name | String | The name of aa building, course, department, major, or school in the database. Constrained to 64 characters in length. |
| Number | String | A course number. Constrained to 4 characters in length. Stored as a string for easier formatting. |
| Password | String | A hashed and salted password. Constrained to 64 characters in length. |
| PrerequisiteID | Integer | A prerequisite for a course. References a CourseID. |
| Role | String | A role of a user, one of: {Administrator, Instructor, Student} |
| Room | Integer | A room number. Used to identify where a course section is held. |
| Salt | String | The salt used to salt the password. Must be stored to validate the password prior to login. Constrained to 32 characters in length. |
| SchoolID | Integer | Uniquely identifies a school in the system. |
| SectionID | Integer | Uniquely identifies a section in the system. |
| Semester | String | A semester at the university. One of: {Fall, Spring, Summer} |
| StartDateAndTime | DateTime | A start date and start time. Used to indicate the first day of a section as well as the time the section begins each day. |
| StudentID | String | Used to link a student to a section. References a UnivID. |
| UnivID | String | A university identifier. Uniquely identifies a user in the system. |
| Year | Integer | A year. |

**4.0 Functional Model and Description**

**4.1 Description of major functions**

**4.1.1 Major functionality**

Coursely is course management software intended for use at the university level. The goal of the software is to make course management easy for students as well as enable advisors to better understand the needs of their advisees. The major functions for Coursely are as follows:

* All users may login, view the course catalog, and change their password.
* Users with the student role may view their schedule, enroll and unenroll from courses, and view their student record.
* Users with the instructor role may view the courses they are teaching, view the number of students in their courses, and view the academic record of an advisee.
* Users with the administrator role may create, remove, modify, archive, and unarchive courses in the course catalog, and may create new users and assign them roles.

**4.1.2 Functional requirements**

Functional requirements listed here have been prioritized in ascending order using the MoSCoW approach to requirements triage. All major functionality will be implemented, however there are some extraneous functional requirements that may or may not be implemented due to time constraints imposed on the project.

The prioritization of the requirements may be read from one (1) to (4). Requirements ranked with a one (1) correspond to must-have requirements while requirements ranked with a four (4) correspond to wont-have requirements. The two intermediary rankings of two (2) and three (3) correspond to should-have and could-have requirements respectively. Rankings may be found in square brackets next to the requirements definitions.

**4.1.2.1 [1]** The system shall have three distinct roles for users: student, instructor, and administrator.

**4.1.2.2 [1]** The system shall allow users to have only one role.

**4.1.2.3 [1]** The system shall allow students to enroll in unarchived courses that have at least one (1) not full section that they meet the prerequisites for.

**4.1.2.4 [1]** The system shall allow students to unenroll from course sections they are currently enrolled in.

**4.1.2.5 [1]** The system shall allow students to view their schedule.

**4.1.2.6** **[1]** The system shall allow students to view their academic record.

**4.1.2.7 [1]** The system shall allow instructors to view the courses they are scheduled to teach.

**4.1.2.8 [1]** The system shall allow instructors to view the number of students who are signed up for their courses.

**4.1.2.9 [1]** The system shall allow instructors to view a student’s record only if they have an advisor-advisee relationship.

**4.1.2.10 [1]** The system shall allow instructors to view a student’s schedule only if they have an advisor-advisee relationship.

**4.1.2.11** **[1]** The system shall allow administrators to create new courses in the course catalog.

**4.1.2.12 [2]** The system shall allow administrators to modify courses in the course catalog.

**4.1.2.13 [2]** The system shall allow administrators to archive courses in the course catalog.

**4.1.2.14** **[2]** The system shall allow administrators to unarchive courses in the course catalog.

**4.1.2.15 [1]** The system shall allow administrators to create new users.

**4.1.2.16 [1]** The system shall not allow anyone other than administrators to create new users.

**4.1.2.17 [1]** The system shall not allow users to change their role.

**4.1.2.18** **[1]** The system shall allow all users to view the course catalog.

**4.1.2.19 [1]** The system shall require all users to login with a university identifier and password.

**4.1.2.20 [3]** The system shall allow users to change their password.

**4.1.2.21** **[1]** The system shall allow the users to logout of the system.

**4.1.2.22 [1]** The system shall allow administrators to add a course section.

**4.1.2.23 [1]** The system shall allow administrators to modify a course section.

**4.1.3 Non-functional requirements**

**4.1.3.1** The system shall require all users to be associated with a first name, last name, university identification number, email address, and a role.

**4.1.3.2** The system shall require students to be associated with a major field of study.

**4.1.3.3** The system shall validate the user’s university identification number and password prior to allowing them to login to the system.

**4.1.3.4** The system shall allow students to only register for eighteen (18) credit hours maximum per term.

**4.1.3.5** The system shall register students for courses within three (3) seconds after the student submits their course registration information.

**4.1.3.6** The system shall require passwords to be at least eight (8) characters in length.

**4.1.3.7** The system shall not store the user’s password in plain text.

**4.1.3.8** The system shall hash and salt the user’s password before storing it in the user data store.

**4.1.3.9** The system shall hash and salt the user’s password before validating it with the password stored in the user data store.

**4.1.3.10** The system shall use the same hash and salt mechanism for storing the user’s password and validating the user’s password.

**4.1.3.11** The system shall require all courses to be associated with a department, name, description, and number of credit hours.

**4.1.3.12** The system shall require all course sections to be associated with dates and times offered as well as at least one (1) instructor.

**4.1.3.13** The system shall not allow course section start times to be later than course section end times.

**4.1.3.14** The system shall require course section start times and course section end times to differ by at least 50 minutes.

**4.1.3.15** The system shall display the section number in the format SO‑### where ### is the section number.

**4.1.3.16** The system shall display the course number in the format [Department Abbreviation]‑### where ### is the course number.

**4.1.3.17** The system shall not allow a class to be a prerequisite of itself.

**4.1.3.18** The system shall require all university identification numbers to be unique.

**4.1.3.19** The system shall not allow section numbers to be less one (1), or greater than 999.

**4.2 Software interface description**

**4.2.1 External system interfaces**

The system will interface with the following external systems:

* Bootstrap: Provides the theme for the web application and the ability to easily customize it.
* JQuery: Required for Bootstrap to function correctly.
* Microsoft SQL Server Express: Allows for direct integration into an ASP.NET web application. Not suitable for real-world deployment. Fortunately, SQL Server Express is very easy to migrate to SQL Server or Azure with minimal changes to the web server’s code base.

**4.2.2 Human interfaces**

The web application produced here will utilize a user role mechanic to change the presentation of information as seen by the user. The role mechanic also exists to separate users into distinct groups each with different functionality. As stated earlier the three roles are: Administrator, Instructor, and Student. The web application will consist of an initial homepage and set of dynamic web pages that display content and actions based on the user’s role in the system.

The home page will welcome the user to the system and prompt them to login by entering their username and password as well as selecting the type of role they wish to login as. Once successfully logged in, the system will redirect the user to their role specific homepage, where they will be shown possible actions based on their role.

If the user is an administrator, they will be presented with the options affiliated with the administrator role such as managing the course catalog and managing user accounts.

If the user is an instructor, they will be presented with the options affiliated with the instructor role such as managing their advisees and managing their courses.

If the user is a student, they will be presented with the options affiliated with the student role such as managing their schedule and viewing their courses history.

1. **Restrictions, Limitations, and Constraints**

User’s cannot currently reset their password if they forget it. If this system is deployed in a real‑world scenario, then this feature will be implemented.