**CMCS-5733 - SE Project I**

**System Requirements Document for**

**ECS Grad School Course Schedule Creation**

1. **Problem**

*This section is a prose description of the problem that is to be solved by the system including background and context for the problem and a clear statement of the main problem(s) to overcome.*

The Graduate School of Engineering and Computer Science offers Masters degrees in Engineering and Computer Science. There are five different tracks in Engineering and one track in Computer Science. The grad school offers about 50 different courses and has a faculty of 21 including full-time and adjunct faculty.

Prior to enrollment for each semester the Director of the Graduate School has to determine what courses to offer in the semester and how many section of each course to offer. Then the Director has to assign the courses to the two terms, nights and classrooms.

Currently this schedule is created manually which takes a long time, is prone to have errors and does not provide any real assurance that all the students can get the courses they need.

1. **Objectives**

*This section is a list of objects that the solution is required to achieve.*

* Generate a schedule (a list of sections) that maximizes the number of students in each section and minimizes the number of sections while ensuring that each student can enroll in four sections or the number remaining to graduate.
* Reduce the time it takes to generate the schedule and track all of the data related to the generation process.
* Reduce the number of times students can't get a set of classes they can take in a semester

1. **Existing system**

*This section describes how the existing system or process works today. This could include proses and/or swim-lane flowchart.*

Today, several weeks before enrollment, the Director and the Chair review the degree plans and update spreadsheets with their descriptions, review the faculty list with what the faculty can teach and update the spreadsheet with that data. They also estimate the number of new student in each program for the semester being planned.

They then build a schedule of course sections on the white board and assign sections with instructor to days of the week. They move these around until there are not conflicts with instructor, the course sections within the tracks don’t seem to conflict, and the total hours offered match the total hour needs of the students in the two departments. The student hour needs are determined by multiplying the number of students enrolled by 12 and dividing by the number of student-course-hours (25\*3) in one class section and seeing if that is close to the number of sections in the proposed schedule.

They get other members of the graduate school to review the proposed schedule and make adjustments before it is finalized.

The owner has identified these problems:

* Generation of the schedule is slow and error prone.
* There is no way to test the schedule to see if it meets all the criteria
* There is not a good way to determine what student need what course
* There is not a good way to easily track the degree plan data
* There is not a good way to track what faculty teach each course and how many courses the faculty can teach each semester

1. **Functional Requirements**

*This section is a list of functions the system is required to meet. Each of these functions will be support by a Use Case in a later section.*

*\* Requirements that are secondary and can be delayed to later release*

* 1. **Maintain Course information (add, update, delete)**
     1. Number
     2. Name
     3. Description
     4. Number of credit hours
     5. Capacity
     6. List of Prerequisite courses
     7. List of Semesters offered
  2. **Maintain Faculty Information (add, update, delete)**
     1. Name
     2. Title
     3. Maximum graduate teaching load by semester
     4. Courses that the faculty member is able to teach
     5. Days they can or prefer to teach\*
  3. **Maintain Degree information (add, update, delete)**
     1. Name
     2. Department (MSE, MSCS) and track (e.g., Mechanical Engineering)
     3. Degree Code (must match one of the degree codes used by Colleague)
     4. Hours Required
     5. Degree Course Requirements
        1. Required Courses
           1. Number of hours
           2. List of courses
        2. Elective Group one
           1. Number of hours
           2. List of Courses
        3. Elective Group two -
           1. Number of hours
           2. List of Courses
  4. **Maintain Rooms (add, update, delete)\***
     1. Building
     2. Room number
     3. Capacity
  5. **Update Forecast** 
     1. Specify he forecast for a semester: for each degree plan, the user can update the number of expected new students entering into the degree plan.
  6. **Import Students** 
     1. Import from file provided by Colleague
        1. Id
        2. Degree
        3. Track
        4. Graduation date
  7. **Import Student Courses** 
     1. Import from file provided by Colleague
        1. Id
        2. Course Number
        3. Grade
        4. Term
  8. **Generate Schedule**
     1. Create a list of sections that meet the following criteria
        1. Input class fill % goal and class overage % goal
        2. Sections have Semester, Course, Section number, Faculty, Term\*, Room\*, night\*, Start/End Time\*
        3. Sections are needed by students in their degree plan
        4. There are faculty assigned to teach each section and each course is assigned a faculty that can teach the course and no faculty is assigned more courses than they are allowed to teach.
        5. Want to maximize the number of students in the courses (subject to being below the capacity plus overage %)
        6. Want to minimize the number of courses
  9. **Test Schedule**
     1. Determine how many students can get the number of sections they need.
     2. Determine how many courses are within fill and overage goals
     3. Determine how many courses have more than 100% of capacity
  10. **Manually Adjust Schedule/Maintain Sections for Schedule (Add, Update, Delete)**
      1. Course
      2. Section Number
      3. Faculty
  11. **Generate Reports**
      1. Schedule Reports
         1. The schedule report shows a list of sections for the semester (for each section, the report must include the Course Number/Section Number and Faculty assigned to teach the section).
         2. The faculty load report shows which courses each professor is assigned to, and total teaching load (hours) for the semester for that professor.
      2. Student Reports
         1. The degree plan report shows how many students are in each degree plan (these are students that have not already graduated prior to the semester for which the schedule is being generated). This gives the Director a sense of the student load that the schedule has to handle per degree.
         2. The student report lists every student, along with a count of courses that student has completed and number of courses needed by that student to graduate. This gives the Director a sense of the student load that the schedule has to handle.
  12. **Control access to all functions (maintenance and** **reports**)
      1. Create/maintain user accounts in the system
      2. Identify specific user
      3. Ensure the user login in is who they say they are (single password is sufficient)
      4. Allow user to only access functions they are permitted to. There are two roles in the system: Admin and Director.
         1. Admin: Access is primarily for data entry only: maintain courses, degree plans, faculty assignments, rooms\*
         2. Director – Has access to all Data Entry operations, but can also generate, modify schedule
         3. All roles - Print reports

1. **Non-Functional Requirements**

*This section is a list of non-functional requirements the system is required to meet. This would include: usability, reliability, performance, supportability, interface, training, operations, and legal.*

* Usability
  + The system must be easy to use so that administrator can quickly update data. The user must be able to find and execute all functions after one 30-minute training session.
  + The system must make it easy to import data from Colleague. This should only require a few keystrokes and not require any understanding of the contents of the data file.
* Reliability
  + The system must be highly reliable, since failure to operate prevents the director from making the semester schedule.
* Performance
  + All online screen updates should be performed in under 1 second.
  + All imports should be performed in under 30 seconds.
  + Generation of schedule should be performed in under 1 min.
  + Testing of schedule should be performed in under 1 min.
* Supportability
  + The system should be developed in a common technology that the students and faculty in the grad school should be able to use to make upgrades to the system
  + The system should be documented and coded in a way that a developer that was not originally on the development team could determine how to make updates.
* Interface
  + The system must support a file exchange interface with Colleague.
* Training
  + The system should provide on-screen help.
  + The system should come with training documentation.
* Operations
  + Grad School staff will operate this system.
* Legal
  + The system should meet legal and OC security requirements for student data.
  + No one should have access to specific student grades

1. **Target Environment**

*This section is describes the required system-operating environment if there is one.*

The system must normally function on a stand alone Windows 10 based PC with 4G of memory and 200 GB hard drive.

1. **Use Cases**

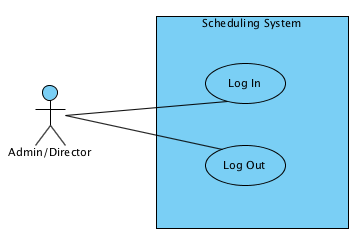
*This section describes the Use Cases that support the functional requirements. Each Use Case includes: name, initiating actor, participating actors, diagram, list and details of each scenario.*

This is a list of use cases identified for this system. The definition of each use case follows.

* 1. Log in
  2. Log out
  3. Add Course
  4. Update Course
  5. Delete Course
  6. Add Faculty
  7. Update Faculty
  8. Delete Faculty
  9. Add Degree Plan
  10. Update Degree Plan
  11. Delete Degree Plan
  12. Add Rooms \*
  13. Update Rooms\*
  14. Delete Rooms\*
  15. Update Forecast of New Students
  16. Import Student Data
  17. Import Student Course Data
  18. Generate Schedule
  19. Test Schedule
  20. Add Section
  21. Update Section
  22. Delete Section
  23. Generate Reports
  24. Assign Schedule Details (Assign Term, Nights and Rooms to schedule)\*
  25. Maintain User Accounts

**System Authorization Use Cases**

1. Log In
2. Log Out

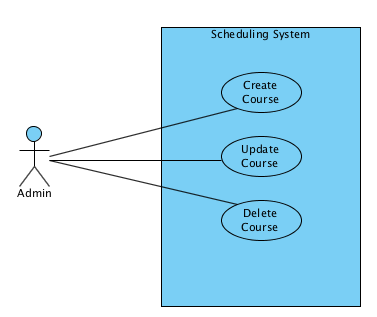
****

|  |  |
| --- | --- |
| **Name** | **Log in** |
| **Description** | Grant a user access as an authorized user |
| **Actor(s)** | User |
| **Flow of Events** | 1. A User wants to use the system.  2. User selects to Log in  3. User enters credentials  4. System tests the credentials and grants access or shows an error. |
| **Special Requirements** | Require credentials (username and password) |
| **Pre- Conditions** | User has previously been set up as a user in the system |
| **Post- Conditions** | User is authorized to the system if requirements are met and no exceptions occur |
| **Exceptions** | Credentials do not match what is in the system, user is given appropriate error message and not allowed access to the system |

|  |  |
| --- | --- |
| **Name** | **Log out** |
| **Description** | Remove a user access as an authorized user |
| **Actor(s)** | User |
| **Flow of Events** | 1. A User wants to end using the system.  2. User selects to log out.  3. System logs out the users |
| **Special Requirements** |  |
| **Pre- Conditions** | User is currently authorized to the system |
| **Post- Conditions** | No user is currently authorized if requirements are met. |
| **Exceptions** | There must be a user authorized before a use can be logged out. |

**Course Maintenance Use Cases**

1. Add Course
2. Update Course
3. Delete Course



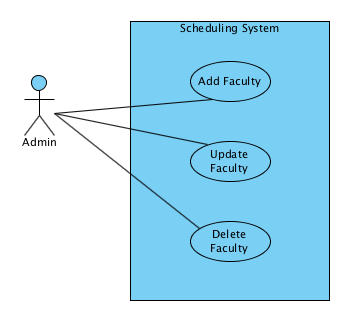
|  |  |
| --- | --- |
| **Name** | **Add Course** |
| **Description** | Add the information for a new Course to the system. This happens when a new Course is approved and added to the graduate catalog. |
| **Actor(s)** | Admin |
| **Flow of Events** | 1. A new Course is approved.  2. Admin selects to add a new Course.  3. Admin enters information about the Course.  4. Admin selects to save the Course. |
| **Special Requirements** | Require Course number, name, credit hours, description, capacity, list of prerequisite courses, and list of semesters offered. |
| **Pre- Conditions** | Admin has be authorized to the system |
| **Post- Conditions** | Course is added to system if requirements are met and no exceptions occur |
| **Exceptions** | Do not add if Course number exists in system |

|  |  |
| --- | --- |
| **Name** | **Update Course** |
| **Description** | Update the information for an existing Course in the system. This happens when a correction is required. |
| **Actor(s)** | Admin |
| **Flow of Events** | 1. A change in Course information is detected.  2. Admin selects to update a Course.  3. Admin selects Course to update.  3. Admin enters information about the Course.  4. Admin selects to save the Course. |
| **Special Requirements** | Require, name, credit hours, description, capacity, list of prerequisite courses, and list of semesters offered. |
| **Pre- Conditions** | Admin has be authorized to the system |
| **Post- Conditions** | Course is updated in system if requirements and no exceptions occur. |
| **Exceptions** | Do not allow change to a Course number that exists in system |

|  |  |
| --- | --- |
| **Name** | **Delete Course** |
| **Description** | Delete the information for an existing Course in the system. This happens when an item is entered in error. |
| **Actor(s)** | Admin |
| **Flow of Events** | 1. A Course is identified that needs deletion.  2. Admin selects to delete a Course.  3. Admin selects Course to delete.  4. Admin deletes a Course. |
| **Special Requirements** |  |
| **Pre- Conditions** | Admin has be authorized to the system |
| **Post- Conditions** | Course is deleted from system if requirements and no exceptions occur.  Course is removed from the list of Courses taught by faculty teaching the course. |
| **Exceptions** | Do not remove Course if it is in a degree plan or if a student has taken the course |

**Faculty Maintenance Use Cases**

1. Add Faculty
2. Update Faculty
3. Delete Faculty



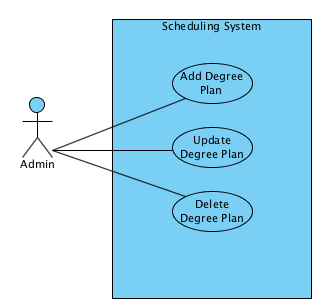
|  |  |
| --- | --- |
| **Name** | **Add Faculty** |
| **Description** | Add the information for a new Faculty to the system. This happens when new Faculty is hired. For a new Faculty, the name, title, courses taught and grad courses per semester should be entered. |
| **Actor(s)** | Admin |
| **Flow of Events** | 1. A new Faculty is identified.  2. Admin selects to add a Faculty.  3. Admin enters information about the Faculty.  4. Admin selects to save the Faculty. |
| **Special Requirements** | Require name and title. |
| **Pre- Conditions** | Admin has be authorized to the system |
| **Post- Conditions** | Faculty is added to system if requirements are met and no exceptions occur. |
| **Exceptions** | None |

|  |  |
| --- | --- |
| **Name** | **Update Faculty** |
| **Description** | Update the information for an existing Faculty in the system. This happens when a correction is required. |
| **Actor(s)** | Admin |
| **Flow of Events** | 1. A change in Faculty information is detected.  2. Admin selects to update a Faculty.  3. Admin selects Faculty to update  3. Admin enters information about the Faculty.  4. Admin selects to save the Faculty. |
| **Special Requirements** | Require Faculty Name |
| **Pre- Conditions** | Admin has be authorized to the system |
| **Post- Conditions** | Faculty is updated in system if requirements and exceptions are meet. |
| **Exceptions** | None |

|  |  |
| --- | --- |
| **Name** | **Delete Faculty** |
| **Description** | Remove a faculty member from the system. |
| **Actor(s)** | Admin |
| **Flow of Events** | 1. A Faculty is identified that needs deleting.  2. Admin selects to delete faculty.  3. Admin selects faculty to delete.  4. Admin deletes the faculty. |
| **Special Requirements** |  |
| **Pre- Conditions** | Admin has be authorized to the system |
| **Post- Conditions** | Faculty is deleted from system if requirements are met and no exceptions occur.  Faculty is removed from any listing of courses taught by Faculty. |
| **Exceptions** |  |

**Degree Plan Use Cases**

1. Add Degree Plan
2. Update Degree Plan
3. Delete Degree Plan

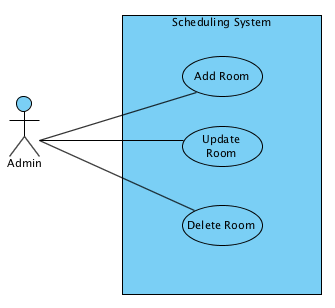


|  |  |
| --- | --- |
| **Name** | **Add Degree Plan** |
| **Description** | Add the information for a new Degree Plan to the system. This happens when new Degree Plan is created. |
| **Actor(s)** | Admin |
| **Flow of Events** | 1. A Degree Plan is identified.  2. Admin selects to add a Degree Plan.  3. Admin enters information about the Degree Plan.  4. Admin selects to save the Degree Plan. |
| **Special Requirements** | The following data is required for a new degree plan: name, dept/track, degree code, hours required, the list of required courses, the lists of set of elective courses and hours for each. |
| **Pre- Conditions** | Admin has be authorized to the system |
| **Post- Conditions** | Degree Plan is added to system if requirements are met and no exceptions occur. |
| **Exceptions** | Do not add if the code is the same as an existing Degree Plan code.  Do not update if the total number of hours doesn’t match the required courses and set of elective hours. |

|  |  |
| --- | --- |
| **Name** | **Update Degree Plan** |
| **Description** | Update the information for an existing Degree Plan in the system. This happens when a correction is required. |
| **Actor(s)** | Admin |
| **Flow of Events** | 1. A change in Degree Plan information is detected.  2. Admin selects to update a Degree Plan.  3. Admin selects Degree Plan to update  3. Admin enters information about the Degree Plan.  4. Admin selects to save the Degree Plan. |
| **Special Requirements** | See “Add Degree Plan” use case. |
| **Pre- Conditions** | Admin has be authorized to the system |
| **Post- Conditions** | Degree Plan is updated in system if requirements and exceptions are meet. |
| **Exceptions** | Do not update if the code is the same as an existing Degree Plan code.  Do not update if the number of hours don't match the required courses and set of elective hours. |

|  |  |
| --- | --- |
| **Name** | **Delete Degree Plan** |
| **Description** | Delete the information for an existing Degree Plan in the system. This happens when a Degree Plan is entered in error or is no longer used. |
| **Actor(s)** | Admin |
| **Flow of Events** | 1. A Degree Plan is identified that needs deleting.  2. Admin selects to delete a Degree Plan.  3. Admin selects Degree Plan to delete.  4. Admin deletes the Degree Plan. |
| **Special Requirements** |  |
| **Pre- Conditions** | Admin has be authorized to the system |
| **Post- Conditions** | Degree Plan is deleted from system if requirements and exceptions are meet. |
| **Exceptions** | Do not delete Degree Plan if the Students are assigned to Degree Plan. |

**Maintain Rooms\***

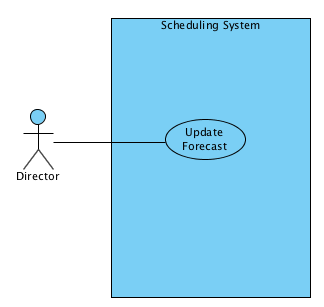


|  |  |
| --- | --- |
| **Name** | **Add Room\*** |
| **Description** | Add the information for a room. This happens when a new room should be made available for the system to use when generating a schedule. |
| **Actor(s)** | Admin |
| **Flow of Events** | 1. A new room is identified.  2. Admin selects to add a room.  3. Admin enters information about the room.  4. Admin selects to save the Degree Plan. |
| **Special Requirements** | Require building, room number, and capacity. |
| **Pre- Conditions** | Admin has be authorized to the system |
| **Post- Conditions** | room is added to system if requirements are met and no exceptions occur. |
| **Exceptions** | Do not add if the building/room number is a duplicate of an existing room in the system. |

|  |  |
| --- | --- |
| **Name** | **Update Room\*** |
| **Description** | Update the information for a room in the system. |
| **Actor(s)** | Admin |
| **Flow of Events** | * + - 1. Admin selects to modify a room.       2. Admin updates information about the room.       3. Admin selects to save the updated information. |
| **Special Requirements** | Require building, room number, and capacity. |
| **Pre- Conditions** | Admin has be authorized to the system |
| **Post- Conditions** | room is updated if requirements are met and no exceptions occur. |
| **Exceptions** | Do not update if the building/room number is a duplicate of an existing room in the system. |

|  |  |
| --- | --- |
| **Name** | **Delete Room\*** |
| **Description** | Delete a room from the system. |
| **Actor(s)** | Admin |
| **Flow of Events** | * + - 1. Admin selects to delete a room.       2. Admin deletes room. |
| **Special Requirements** | None |
| **Pre- Conditions** | Admin has be authorized to the system |
| **Post- Conditions** | room is deleted if requirements are met and no exceptions occur. |
| **Exceptions** | None |

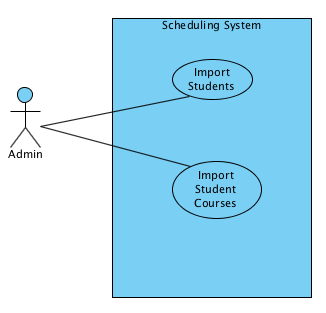
**Update Forecast**

****

|  |  |
| --- | --- |
| **Name** | **Update Forecast** |
| **Description** | Update the forecast of new students coming into the program. |
| **Actor(s)** | Director |
| **Flow of Events** | 1. Director receives estimates of new students for the semester in which a schedule is needed.  2. Director selects to update the forecast.  3. For each Degree Plan, the director inputs the expected number of new students.  4. Director selects to save the changes. |
| **Special Requirements** | None |
| **Pre- Conditions** | Director is authorized to the system |
| **Post- Conditions** | Forecast is updated in system if requirements and exceptions are meet. |
| **Exceptions** | Do not change the forecast if one or more input values are invalid. |

**Import Data Use Cases**

1. Import Students
2. Import Student Courses

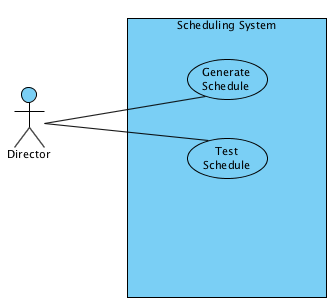


|  |  |
| --- | --- |
| **Name** | **Import Students** |
| **Description** | Add the graduate student information generated by Colleague to the system |
| **Actor(s)** | Admin |
| **Flow of Events** | 1. Admin makes request for Colleague to generate Student data  2. Colleague generates data.  3. Admin selects to import Colleague data.  4. Admin specifies file to import.  5. System deletes all students currently in system  5. System imports data.  6. System displays results (counts of import) |
| **Special Requirements** | Data file meets specified format (CSV) |
| **Pre- Conditions** | Admin has be authorized to the system |
| **Post- Conditions** | Students are added to system if requirements are met and no exceptions occur. |
| **Exceptions** | None |

|  |  |
| --- | --- |
| **Name** | **Import Student Courses** |
| **Description** | Add the student course information generated by Colleague to the system. |
| **Actor(s)** | Admin |
| **Flow of Events** | 1. Admin make request for Colleague to generate Student Course data  2. Colleague generates data.  3. Admin selects to import Colleague data.  4. Admin specifies file to import.  5. System deletes all student courses currently in system  5. System imports data.  6. System displays results (counts of import) |
| **Special Requirements** | Data file meets specified format (CSV) |
| **Pre- Conditions** | Admin has be authorized to the system |
| **Post- Conditions** | Student courses are added to system if requirements are met and no exceptions occur. |
| **Exceptions** | No course is added if the student ID number for the record does not match a student ID in the system. |

**Make a Schedule**

1. Generate Schedule
2. Test a Schedule

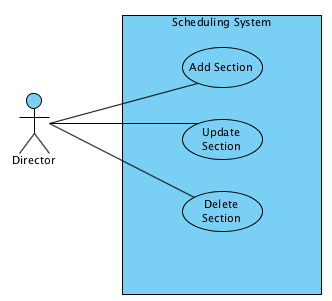


|  |  |
| --- | --- |
| **Name** | **Generate a Schedule** |
| **Description** | Create a list of Sections to offer in the semester |
| **Actor(s)** | Director |
| **Flow of Events** | 1. Director Selects to Generate Schedule.  2. Director enters Semester number, the Section Fill % and the Section Overage % desired  2. System deletes schedule for semester if it exists  3. System generates list of sections for schedule. Each section should have Semester, Course, Section number, Faculty, Term\*, Room\*, night\*, Start/End Time\*.  4. System displays the statistics for the schedule |
| **Special Requirements** | Use course needs for existing students that have not graduated prior to semester and new students forecast for the semester  Assumptions:   * All students will take 4 courses in the fall and spring unless they need less to meet their degree plan requirements * All students will take 2-4 courses in the summer * All new students will take three required courses and 1 elective in Fall and Spring. |
| **Pre- Conditions** | Students and Student Courses have been imported for prior semester.  Forecast for Semester has been created and updated. |
| **Post- Conditions** | Schedule is generated and Sections added. |
| **Exceptions** | None |

|  |  |
| --- | --- |
| **Name** | **Test Schedule** |
| **Description** | Determine the goodness of the schedule. A Schedule is good if :  1) all existing students that have not yet graduated and all new students forecast can enroll in 4 (or fewer if that is all they need) courses for the semester  2) all courses have between the course Section Overage % and the Section Fill % for the number of students assigned to course |
| **Actor(s)** | Director |
| **Flow of Events** | 1. Director Selects to Test Schedule.  2. Director enters Semester number  3. System displays the statistics for the schedule  Number of students  Number and percent of students with required courses  Number and percent of students without required courses  Number of sections  Number of sections within tolerance  Number and percent of sections above cap  Number and percent of sections below percent |
| **Special Requirements** | None |
| **Pre- Conditions** | Schedule exists for semester |
| **Post- Conditions** | None |
| **Exceptions** | None |

**Schedule Maintenance**

1. Add Section
2. Update Section
3. Delete Section

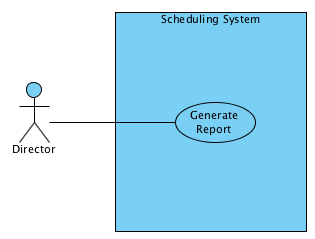


|  |  |
| --- | --- |
| **Name** | **Add Section** |
| **Description** | Add the information for a new Section to the Schedule. This happens when new Section is needed to improve the schedule that was produced by the system. For the section, the course number and section number should be added. |
| **Actor(s)** | Director |
| **Flow of Events** | 1. A new Section is needed.  2. Director selects to add a new Section.  3. Director enters information about the Section.  4. Director selects to save the Section. |
| **Special Requirements** | Require Section number, course, faculty and semester |
| **Pre- Conditions** | Director has be authorized to the system |
| **Post- Conditions** | Section is added to Semester if requirements are met and no exceptions occur. |
| **Exceptions** | Do not add if course and section number exists in Semester.  Do not allow assignment of Faculty to more Sections than they are allowed. |

|  |  |
| --- | --- |
| **Name** | **Update Section** |
| **Description** | Update the information for an existing Section in the system. This happens when a correction is required. |
| **Actor(s)** | Director |
| **Flow of Events** | 1. A change in Section information is detected.  2. Director selects to update a Course.  3. Director selects Course to update.  3. Director enters information about the Course.  4. Director selects to save the Course. |
| **Special Requirements** | Require Section number, course, faculty and semester |
| **Pre- Conditions** | Director has be authorized to the system |
| **Post- Conditions** | Course is updated in system if requirements and exceptions are met. |
| **Exceptions** | Do not allow change to a Section number that exists in Semester  Do not allow assignment of faculty member to more sections than they are allowed. |

|  |  |
| --- | --- |
| **Name** | **Delete Section** |
| **Description** | Delete the information for an existing Section in the system. This happens when an item is entered in error. |
| **Actor(s)** | Director |
| **Flow of Events** | 1. A Section is identified that needs deletion.  2. Director selects to delete a Section.  3. Director selects Section to delete.  4. Director deletes a Section. |
| **Special Requirements** |  |
| **Pre- Conditions** | Director has be authorized to the system |
| **Post- Conditions** | Section is deleted from Schedule if requirements and exceptions are met. |
| **Exceptions** | None |

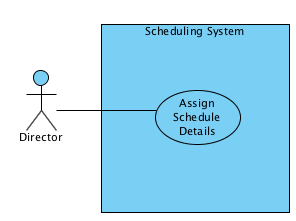
**Generate Report**



1. Generate Report
   * 1. Schedule Reports
        1. The schedule report shows a list of sections for the semester (for each section, the report must include the Course Number/Section Number and Faculty assigned to teach the section).
        2. The faculty load report shows which courses each professor is assigned to, and total teaching load (hours) for the semester for that professor.
     2. Student Reports
        1. The degree plan report shows how many students are in each degree plan (these are students that have not already graduated prior to the semester for which the schedule is being generated).
        2. The student report lists every student, along with a count of courses that student has completed and number of courses needed by that student to graduate.

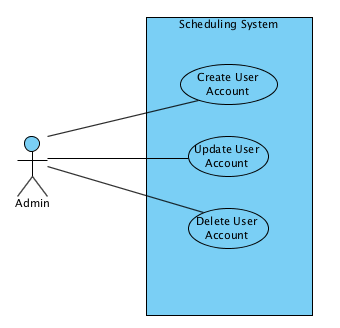
|  |  |
| --- | --- |
| **Name** | **Generate Report** |
| **Description** | Generate a report for the Director based on information in the system and report selected |
| **Actor(s)** | Admin, Director |
| **Flow of Events** | 1. A User determines a need for a report  2. User selects report  3. System generates report  4. User views/prints report |
| **Special Requirements** | None |
| **Pre- Conditions** | User has be authorized to the system |
| **Post- Conditions** | Generated report for viewing |
| **Exceptions** | None |

**Assign Schedule Details\***

****

|  |  |
| --- | --- |
| **Name** | **Assign Schedule Details\*** |
| **Description** | After generating a proposed schedule, the director can (optionally) ask the system to modify the generated schedule to include the term, room, night, and start/end time for each section on the schedule. |
| **Actor(s)** | Director |
| **Flow of Events** | 1. Director asks for the system to generate a schedule that includes the additional information for each section.  2. System generates modified schedule. |
| **Special Requirements** | None |
| **Pre- Conditions** | User has be authorized to the system |
| **Post- Conditions** | Generated schedule is updated to show term, night, and room |
| **Exceptions** | None |

**Maintain User Accounts**

****

|  |  |
| --- | --- |
| **Name** | **Create User Account** |
| **Description** | The user creates a new user account. The user must provide the following data:   * user name * initial password * Role (Admin or Director) |
| **Actor(s)** | Admin/Director |
| **Flow of Events** | Admin selects to create a user account  Admin enters required data  Admin saves the user account. |
| **Special Requirements** | The system should always have at least one Admin account. |
| **Pre- Conditions** | Admin is currently authorized to the system |
| **Post- Conditions** | New user account is stored in the system if requirements are met and no exceptions occur. |
| **Exceptions** | An error should be generated if roles are changed such that no Admin users exist. |

|  |  |
| --- | --- |
| **Name** | **Update User Account** |
| **Description** | The admin user can update user account information. The user may update the following data:   * user name * password * Role (Admin or Director) |
| **Actor(s)** | Admin/Director |
| **Flow of Events** | Admin selects user account to update, and changes desired data  Admin saves the modified data. |
| **Special Requirements** | * There are no special requirements for the quality of the password. * At least one admin user must exist in the system at all times. |
| **Pre- Conditions** | Admin is currently authorized to the system |
| **Post- Conditions** | Modified user account is stored in the system if requirements are met and no exceptions occur. |
| **Exceptions** | No changes are made to the system if the user attempts to remove the only Admin account. |

|  |  |
| --- | --- |
| **Name** | **Delete User Account** |
| **Description** | A user is removed from the system. |
| **Actor(s)** | Admin/Director |
| **Flow of Events** | Admin selects the user and requests that the system delete the user.  The system deletes the user. |
| **Special Requirements** | At least one admin user must exist in the system at all times. |
| **Pre- Conditions** | Admin is currently authorized to the system |
| **Post- Conditions** | The specified user account is deleted from the system if requirements are met and no exceptions occur. |
| **Exceptions** | No changes are made to the system if the user attempts to remove the only Admin account. |

1. **Glossary**

*This section is a list of terms used in the document with their definition. This would include actors and data elements.*

|  |  |
| --- | --- |
| **Term** | **Definition** |
| GSECS | This is the acronym for the Graduate School of Engineering and Computer Science. This graduate school grants degrees to students offers |
| Chair | A Chair is an employee of the GSECS that is responsible for the academic program and operation of the GSECS. |
| Director | A Director is an employee of the GSECS that is responsible for the operation of the GSECS. |
| Admin | An Admin is an employee of the GSECS that works to support the operations of the GSECS. |
| Faculty | A Faculty (Faculty Member, Professor, Instructor) is an employee of the GSECS that is responsible for the teach classes that are offered by the GSECS. |
| Teaching Load | Teaching Load is the number of Courses (or hours) a Faculty can teach in a semester. |
| Student | A Student is a person that is enroll for a specific degree/track in the GSECS. |
| Course | A Course is a |
| Course Number | A Course Number is the identifier used across the university for the course it has the standard format AAAA-LNNH where the AAAA (alpha) designates the department and LNNH (numeric) the number of the course within that department. The L is a number 1-5 that indicted the level of the course and H is a number that indicates the number of credit hours. |
| Course Cap | A Course Cap (or Cap) is maximum number of students allowed to enroll in the courses. Sometimes is increased temporarily instead of adding a new section. |
| Section Fill % | A Section Fill % is the percentage of the Cap that needs to be maintained for a good schedule. |
| Section Overage % | A Section Overage % is the percentage of the Cap that is allowed to go over for a good schedule. |
| Section | A Section (or Course Section) is a specific instance of a course offered in a semester (and term) and taught by a specific faculty. |
| Section Number | A Section Number is a suffix appended to a Course Number to indicate the specific section of a course in a semester. It has the pattern AAAA-LNNH-XX were XX is a number. The sections in a semester start with 01 and are number sequential in numerical order. |
| Schedule | A Schedule is a list of Sections for a specific Semester. |
| Degree | A Degree is an official certification of a plan of completion of a plan of study granted by the GSECS and the university. GSECS grans a MSE and MSCS. |
| Track | A track is a plan of study within a degree. There is 1 current plan of study in the MSCS degree and 5 plans of study in the MSE degree. |
| Degree Plan | A Degree Plan is a description of the course requirements for a specific track within a degree. |
| Semester | A Semester is a period of time that a set of course is offered. For the GSECS it is 16 weeks. It is divided into two eight week Terms. Sections may be offer for the full semester or for a Term. |
| Semester Number | A Semester Number is an identifier for a specific semester. It is has the form AAYYYY. Where AA is F,S, SU and YYYY is the year. |
| Term | A Term is a period of time that a set of course is offered. For the GSECS it is 8 weeks. It is part of a Semester. There are only two in a Semester. |
| Term Number | A Term Number is a suffix appended to a Semester Number to indicate the specific Term. It has the pattern AAYYYY-XX were XX is a number. The Terms in a Semester start with 01 and are number sequential in numerical order. |