**User stories**

“As a Student, I want to rank my choices for second year discipline so that I can be placed in my preferred program.”

“As a Student, I want to view the available courses to register in and be able to register so I can take new courses.”

“As a Student, I want to view my financial summary and pay my tuition so I don’t get kicked out of UBC.”

“As a Professor, I want to place students into engineering disciplines so that they can continue their degrees.”

“As a Professor, I want to be able to report a cheating incident so that the cheating student can be disciplined.”

“As a Professor, I want to submit a course grade for a student so they can receive their grade.”

“As the Dean, I want to be able to suspend students who have cheated so that they will not want to cheat again.”

**Use Cases**

|  |
| --- |
| Use case: **Login**   * Ask user for username/password * User enters their username and password * User presses “Login” button * System checks database for username * System compares the entered password with the one in the database, * User granted access |

|  |
| --- |
| Use case: **Rank disciplines**   * Include **Login** * If user is not a student who is eligible to rank disciplines <<Scenario 1>> * Student clicks “Rank disciplines” button * System shows student a list of all available engineering disciplines, with a description of each * Student uses drop down boxes to rank each discipline from first choice to last choice * Student clicks “submit rankings” button * System stores user’s program rankings in user’s profile   Scenario 1: System hides “View rank disciplines button” |

|  |
| --- |
| Use case: **View courses**   * Include **Login** * If user is not a student who is eligible to register for courses <<Scenario 1>> * Student clicks “View courses” button * System shows student a list of available courses at UBC with description of each   Scenario 1: System hides “View courses button” |

|  |
| --- |
| Use case: **Register course**   * Include **View courses** * Student clicks on course they wish to register in * System gets course’s info and displays it to student * Student clicks “Register course” * If student’s year level and academic standing do not make them eligible for the course, or if the course is full <<Scenario 1>> * System adds student to course and updates database * System displays success message   Scenario 1: System tells student why they are ineligible to register this course |

|  |
| --- |
| Use case: **View financial summary**   * Include **Login** * Student clicks “Financial summary” button * System gets student’s financial info and displays student’s outstanding dues |

|  |
| --- |
| Use case: **Pay tuition**   * Include **View financial summary** * User clicks “Pay now” button * System shows user payment page * User inputs payment information * System displays “payment successful” message and updates student’s outstanding dues in database to reflect payment |

|  |
| --- |
| Use Case: **Assign student to discipline**   * Include **Login** * Professor selects “assign disciplines” button * System displays a list of students ready to be assigned a discipline * Professor selects a student * System shows student’s info and asks Professor to select a discipline * Professor selects discipline to assign * System saves the Professor’s selection in student’s account |

|  |
| --- |
| Use Case: **Suspend Student**   * Include **Login** * Dean selects “suspend student” * System shows Dean a list of students * Dean selects a student to suspend * System gives Dean “are you sure you want to suspend this student” message * Dean confirms suspension * System updates student’s account to reflect suspension * System shows Dean confirmation of suspension |

|  |
| --- |
| Use Case: **Report Academic Misconduct**   * Include **Login** * Professor selects “report academic misconduct” button * System displays a list of all students * Professor selects a student * System shows student’s info and asks Professor to confirm suspension * Professor selects “confirm” * System saves the academic misconduct strike in Student’s account * System shows Professor confirmation of report |

|  |
| --- |
| Use Case: **Submit Grade**   * Include **Login** * Professor selects “submit grade” button * System displays a list of all students * Professor selects a student * System shows asks Professor to enter grade * Professor enters grade * System asks Professor to confirm grade * Professors clicks “confirm” * System stores grade in appropriate array in Student’s profile |

**Objects and Their Responsibilities**

|  |  |
| --- | --- |
| **Object** | **Responsibilities** |
| Login | * Prompt user for username and password * Send username and password to Database for authentication |
| Home | * Display options to user * Accept user input * Request information from various other objects |
| Course | * Store course information * Send course information to other objects who request it |
| CourseManager | * Keep track of all course objects * Send list of all courses to other objects who request it |
| Discipline | * Store discipline information * Send discipline information to other objects who request it |
| DisciplineManager | * Keep track of all discipline objects * Send list of all disciplines to other objects who request it |
| User | * Store information that is common to all types of accounts (Student, Prof, Dean etc.) |
| Student | * Store information for a student (courses, year level, discipline…) * Send student’s information to other objects who request it |
| Professor | * Store information for a Professor (name, account info…) |
| Dean | * Store information for a Dean (name, account info…) |
| Database | * Store all accounts and their information (Students, Professors, Deans…) * Authenticate username and password sent from Login and send back the appropriate user account |