**Bugs**

**Bug 1 (Critical)**

This Bug located in JavaSource/org/unitime/timetable/model/Class\_.java, Line 1866, instructors can’t be zero because we cant divide by zero ,so this case must handled

A screenshot of a computer

Description automatically generated with medium confidence

To fix this bug we can add to the method if (instructors == 0) ,it will return an error message to make sure that no division by zero will happen.

**if (instructors == 0) {System.out.println("Unexpected state: None of the teaching requests have any instructors assigned. Please check the input data.")}**

**Bug 2 (Major)**

This Bug located in JavaSource/org/unitime/timetable/model/CourseEvent.java, Line 88, the loop will exit after executing the first iteration, because the return statement is inside the loop body. This means that the method will return the session of the first course in the getRelatedCourses() list, and ignore any subsequent courses.

A screenshot of a computer

Description automatically generated with medium confidence

To fix this bug we can add a new session variable outside the loop, and replace the return statement inside the loop with an assignment to this variable, which is returned at the end of the method, break statement will be added to exit the loop when a session is found.

**for (RelatedCourseInfo rc: getRelatedCourses()) {**

**Session s = rc.getCourse().getInstructionalOffering().getSession();**

**if (s != null) {**

**session = s;**

**break;} }**

**return session;}**

**Bug 3 (Major)**

This Bug located in JavaSource/org/unitime/timetable/model/Department.java, Line 204, A null pointer exception could be thrown if c1 or c2 is null

A screenshot of a computer

Description automatically generated with medium confidence

To fix the bug ,we can check if either c1 or c2 is null before using them ,and if any of them is null we can throwm an IllegalArgumentException with the message "Invalid color string".

**if (c1 == null || c2 == null) {throw new IllegalArgumentException("Invalid color string"); }**

**Bug 4(Major)**

This Bug located in JavaSource/org/unitime/timetable/model/Event.java, Line 142,It is better that the comparison is done using "equals()" not “==”A screenshot of a computer

Description automatically generated with medium confidence

To fix this bug, we can use the equals() method to compare instead of == ,since equals() is preferred over == for comparing strings and boxed types, Bec In Java, the == operator checks whether two objects refer to the same memory location, rather than whether their contents are equal, while the equals() method checks whether two objects have the same contents, rather than whether they refer to the same memory location.

**if (!getEventName().equals(e.getEventName())) {…..}**

**Change 1: -**

**CourseCreditUnitConfig class**

The CourseCreditUnitConfig class is an abstract base class which define and manage different types of course credit configurations within the UniTime scheduling system,such as fixed credit and variable credit, and arranged credit. It defines common fields such as the credit format, credit type, and credit unit type. It provides several methods for creating credit configurations of different types, based on the credit format and other parameters. It also provides methods like getMinCredit() and getMaxCredit(), which return the minimum and maximum credit values for a credit configuration

**Feature to be added**

we can add a static method calculateTotalCredits to CourseCreditUnitConfig class , which takes a list of CourseCreditUnitConfig objects and returns the total number of credits for that set of configurations. The method could iterate over the list and call the getMaxCredit method for each configuration to get the maximum number of credits for that configuration, and then sum up all the maximum credits to get the total. This could be useful if we need to calculate the total credits for a student's course load.

**Change 2: -**

**CourseCreditType class**

This class is a part of the Unitime scheduling system, It extends BaseCourseCredit Type which is an abstract class that its methods can be inherited by CourseCreditType and it provides a base implementation of the CourseCreditType like "uniqueld" which is created within the object when the constructor is called which calls the constructor of the parent class "super(uniqueld)", This class represents a specific type of credit that a course provides. Each course can have multiple credit types (for ex: lecture credit-lab credit), and each credit type can be associated with one or more courses ,For example, a course may offer both lecture and lab credit types, and each credit type may be associated with different credit hours or units and this is managed by the administrators.

The CourseCreditType class provides methods for retrieving and managing credit types in the system. the getCourseCreditTypeList() method returns a list of all credit types defined in the system, the getCourseCreditTvpeForReference(String referenceString) method returns a specific credit type based on its reference string(argument), getCourseCreditTypeForUniqueld(Long uniqueld) method returns a specific credit type based on its unique id, and getAbbv) returns the abbreviation of the course credit type object, It calls getAbbreviation) method from the parent class.

**Feature to be added**

This feature will be to get credit types of a specific department Description of the change: Add a new method **getCreditTypesOfDepartment()** to class [CourseCreditType.java](http://CourseCreditType.java), in model package.It can be used to know which credit types are often chosen that can be used to preform calculations.