

# Course Assessment Tool

## Test Plan

### CSE 4101

## Computer Science Projects

Chris Durel  
[timeaisis@gmail.com](mailto:timeaisis@gmail.com)

Matt Garabedian  
[mgarabed@fit.edu](mailto:mgarabed@fit.edu)

Zax Laubstein  
[zlaubste@fit.edu](mailto:zlaubste@fit.edu)

Faculty Advisor:  
Dr. Shoaf  
[wds@cs.fit.edu](mailto:wds@cs.fit.edu)

## Table of Contents

<u>Test Cases</u> .....	<u>page number</u>
Login() and Redirect().....	1
New_C_Outcome() and Del_C_Outcome().....	2
Modify_C_Outcome() and Rate_Student().....	3
Update_SA_S() and Update_C_Outcome().....	4
Take_CS_Survey() and Take_SA_Survey().....	5
Get_C_Achievment() and Course.....	6
Update_Database() and Graduate Student log in.....	7
Get_LO() and Set_LO().....	8
Get_CA() and Set_CA().....	9
Course Satisfaction Survey and Self Achievement Survey.....	10
Program Survey and Get_Survey().....	11
Update_Survey_DB() and Change_Question().....	12
Add_Question() and Remove_Question().....	13
Change_QA_Type() and Add_Course().....	14
Remove_Course() and Access_Course().....	15
Add_Course() and Remove_Course().....	16
Access_Course() and Take_Prog_Survey().....	17
Update_Prog_Survey() and Create_Course().....	18
Delete_Course() and Access_Course().....	19
Create_C_Report() and Update_C_Database().....	20
Display_C_Report() and Explanation of Test Plan.....	21

**Name:**

Login()

**Primary Actor:**

User

**Description:**

Main login into Moodle.

**Preconditions:**

Successful navigation to Moodle homepage.

**Expected Input Type:**

2 Strings (username, password)

**Expected Output:**

log-in successful message, or "Logging in..." message

**Error Handling:**

**Input:** invalid username/password combination

**Output:** reload webpage with warning about invalid username/password combination

**Name:**

Redirect()

**Primary Actor:**

User

**Description:**

Redirects to appropriate user-page (Student, professor, admin, etc).

**Preconditions:**

Successful login through Moodle (correct username/password combination)

**Expected Input Type:**

Click login button

**Expected Output:**

Display user homepage

**Error Handling:**

**Input:** invalid username/password combination

**Output:** reload webpage with warning about invalid username/password combination

**Name:**

New\_C\_Outcome()

**Primary Actor:**

Professor

**Description:**

Creates new course outcome

**Preconditions:**

Successful navigation to course outcome page of specific course

**Expected Input Type:**

String describing course outcome

**Expected Output:**

Course Outcome successfully added message

**Error Handling:**

**Input:** Invalid string (too long, invalid characters)

**Output:** Reload page with message saying "invalid course outcome, please re-enter"

**Name:**

Del\_C\_Outcome()

**Primary Actor:**

Professor

**Description:**

Delete specific course outcome.

**Preconditions:**

Successful navigation to course outcome page of specific course

**Expected Input Type:**

Choose from drop down box of course outcome to delete, then click "delete"

**Expected Output:**

Course outcome successfully added message

**Error Handling:**

**Input:** No course outcome selected and clicked "delete"

**Output:** Reload page with message "Please select course outcome to delete"

**Name:**

Modify\_C\_Outcome()

**Primary Actor:**

Professor

**Description:**

Modifies a specific course outcome

**Preconditions:**

Successful navigation to course outcome page of specific course

**Expected Input Type:**

Drop down box of which course outcome to modify, then input a string of fields to modify.

**Expected Output:**

Course outcome successfully modified

**Error Handling:**

**Input:** Course outcome not selected first

**Output:** Reload page with message "Please select a course outcome to modify"

**Name:**

Rate\_Student()

**Primary Actor:**

Professor

**Description:**

Rates a student's progress on course outcomes

**Preconditions:**

Successful navigation to course page

**Expected Input Type:**

Select student to rate, then select from drop down box for each outcome, "Good, Satisfactory, Fair, Poor", then click "rate student".

**Expected Output:**

Student successfully rated message, reload page with current ratings.

**Error Handling:**

**Input:** "Rate Student" button clicked without ratings selected.

**Output:** Reload page with message "Please select rating for each course outcome for this student"

**Name:**

Update\_SA\_S()

**Primary Actor:**

Professor

**Description:**

Updates Self Achievement survey

**Preconditions:**

Successful navigation to professor survey page for specific course

**Expected Input Type:**

Input strings to fields specific to survey

**Expected Output:**

Survey updated message

**Error Handling:**

**Input:** Too many characters in a field, invalid characters

**Output:** "Field invalid" message

**Name:**

Update\_C\_Outcome()

**Primary Actor:**

Professor

**Description:**

Navigates to the Course Outcomes page

**Preconditions:**

Successful navigation to the course page

**Expected Input Type:**

Click button

**Expected Output:**

Load the course outcomes page

**Error Handling:**

**Input:** Click, page not loaded

**Output:** Reload page with page load error message

**Name:**

Take\_CS\_Survey()

**Primary Actor:**

Student

**Description:**

Allows student to take course survey

**Preconditions:**

Successful navigation to specific course page

**Expected Input Type:**

Click, then radio buttons for each question, then finally “Finish Survey” button.

**Expected Output:**

Navigation to the course survey to take

**Error Handling:**

**Input:** Click, but could not find survey

**Output:** Reload page with database error. Survey could not be grabbed from the database.

**Input:** Click “finish survey” button without answering every question

**Output:** Reload page with “please answer every question” message

**Name:**

Take\_SA\_Survey()

**Primary Actor:**

Student

**Description:**

Allows student to take self achievement survey

**Preconditions:**

Successful navigation to specific course page

**Expected Input Type:**

Click “take professor survey” button, then click radio buttons for each question, then finally “Finish Survey” button.

**Expected Output:**

Navigation to course homepage with message “Thank you for taking this survey!”

**Error Handling:**

**Input:** Click, but could not find survey

**Output:** Reload page with database error. Survey could not be grabbed from the database.

**Input:** Click “finish survey” button without answering every question

**Output:** Reload page with “please answer every question” message

**Name:**

Get\_C\_Achievment()

**Primary Actor:**

Student

**Description:**

Allows student to look at his course achievements

**Preconditions:**

Successful navigation to specific course page

**Expected Input Type:**

Click "My Course Achievements"

**Expected Output:**

Load student-specific course achievement page with his/her specific course achievements.

**Error Handling:**

**Input:** Click, but could not course achievements/page

**Output:** Reload page with database error. Course achievements could not be grabbed from the database.

**Name:**

Course

**Primary Actor:**

Admin

**Description:**

Allows admin to grab the course object, navigate to the specific page, and modify the course

**Preconditions:**

Successful navigation to the course list page.

**Expected Input Type:**

Click the specific course to load.

**Expected Output:**

Load specific course page.

**Error Handling:**

**Input:** Click, but could not find course.

**Output:** Reload page with database error. Course could not be found from the database. Possibly display "Course must first be created".

**Name:**

Update\_Database()

**Primary Actor:**

All users (this will be an automatically called method whenever a course object is modified)

**Description:**

Updates database with information in a recently modified course object

**Preconditions:**

Update-specific message saying the modification was made successfully (e.g. "Survey completed successfully!", or "Course Objective added!").

**Expected Input Type:**

Any user that will modify a course object

**Expected Output:**

Load specific success page, and return successful message

**Error Handling:**

**Input:** Could not connect with database

**Output:** Reload page with database error.

**Name:**

Graduate Student log in

**Primary Actor:**

Students

**Description:**

Ensure that proper homepage is displayed and same functionality as Student

**Preconditions:**

Successful log in to system as a graduate student.

**Expected Input Type:**

2 strings (Username/password)

1 student object

**Expected Output:**

Student homepage with full functionality

**Error Handling:**

**Input:** invalid username/password combination

**Output:** reload webpage with warning about invalid username/password combination

**Input:** null student object

**Output:** go to log-in page and alert user to contact administration with detailed error display

**Name:**  
Get\_LO()

**Primary Actor:**  
Professor and Students

**Description:**  
Method to return a course outcome object.

**Preconditions:**  
Successful navigation to a specific course and access to database.

**Expected Input Type:**  
1 int

**Expected Output:**  
Display of course outcome object's variables.

**Error Handling:**  
**Input:** invalid int  
**Output:** reload webpage with warning about proper integer input

**Name:**  
Set\_LO()

**Primary Actor:**  
Professor and Students

**Description:**  
Method to change a course outcome object. Setting a course outcome's learning objective to null should automatically delete the course outcome, although it should warn user before deleting course outcome.

**Preconditions:**  
Successful navigation to a specific course and access to database.

**Expected Input Type:**  
1 String

**Expected Output:**  
Display of course outcome learning objective variable.

**Error Handling:**  
**Input:** null string  
**Output:** warn user that they are about to delete this course outcome

**Name:**

Get\_CA()

**Primary Actor:**

Professor and Students

**Description:**

Method to return a course outcome achievement level for individual students.

**Preconditions:**

Successful navigation to a specific course and access to database.

**Expected Input Type:**

1 Student object

**Expected Output:**

Display of student's achievement for all course related course outcomes.

**Error Handling:**

**Input:** null student

**Output:** return to course webpage and warn user about choosing a student that is not enrolled in their class.

**Name:**

Set\_CA()

**Primary Actor:**

Professor

**Description:**

Method to change a course outcome achievement level for individual students.

**Preconditions:**

Successful navigation to a specific course and access to database.

**Expected Input Type:**

1 Student object

**Expected Output:**

Display of student's achievement for all course related course outcomes updated properly.

**Error Handling:**

**Input:** null student

**Output:** return to course webpage and warn user about choosing a student that is not enrolled in their class.

**Name:**

Course Satisfaction Survey

**Primary Actor:**

Students

**Description:**

Ensure that generic Course Satisfaction Survey is properly generated.

**Preconditions:**

Successful navigation to a specific course and access to database (immediately after creation of course).

**Expected Input Type:**

1 Course Object

**Expected Output:**

Display Course Satisfaction Survey properly.

**Error Handling:**

**Input:** null course

**Output:** return to course webpage and warn user about attempting to get a survey for a course that does not exist.

**Name:**

Self Achievement Survey

**Primary Actor:**

Students

**Description:**

Ensure that proper default Self Achievement Survey is properly generated.

Self Achievement Survey will have parts of the survey based on course outcomes.

**Preconditions:**

Successful navigation to a specific course, access to database, course outcomes are established for the course.

**Expected Input Type:**

1 Course Object

**Expected Output:**

Display Self Achievement Survey properly.

**Error Handling:**

**Input:** 1 empty Course Object

**Output:** return to course webpage and warn user that no course outcomes exist so a proper survey cannot be generated for this course at this moment

**Name:**

Program Survey

**Primary Actor:**

Students

**Description:**

Ensure that generic Degree Program Survey is properly generated.

**Preconditions:**

Successful navigation to student homepage, student is marked in system as graduated for their program.

**Expected Input Type:**

1 Student Object

**Expected Output:**

Display Degree Program Survey properly.

**Error Handling:**

**Input:** 1 Student Object (who is not a graduate)

**Output:** return to student homepage and warn user that they are not permitted to take the survey until they graduate.

**Name:**

Get\_Survey()

**Primary Actor:**

User

**Description:**

Ensure that correct survey object is returned.

**Preconditions:**

Successful navigation to specific course or student homepage.

**Expected Input Type:**

void

**Expected Output:**

display proper survey

**Error Handling:**

**Input:** void

**Preset Condition:** course or degree program association is not established with survey

**Output:** return to course/student homepage and alert user that the survey does not exist for the course/degree program yet

**Name:**

Update\_Survey\_DB()

**Primary Actor:**

User

**Description:**

Ensure that the database is updated properly and with the correct information.

**Preconditions:**

Must be called from one of four method in Survey that will modify the survey's variables.

**Expected Input Type:**

1 survey object

**Expected Output:**

display proper survey

**Error Handling:**

**Input:** null survey object

**Output:** return to homepage and warn user that survey does not exist.

**Name:**

Change\_Question()

**Primary Actor:**

Professors, Admins

**Description:**

Ensure that the database is updated properly and with the correct information.

**Preconditions:**

Must be logged in as Professor or Administrator and chosen a specific course/program to do selected operation on.

**Expected Input Type:**

1 int

**Expected Output:**

updated question

**Error Handling:**

**Input:** 1 int (outside of array boundaries)

**Output:** reload webpage and warn user of the specific array boundaries for the survey they are working with

**Name:**

Add\_Question()

**Primary Actor:**

Professors, Admins

**Description:**

Ensure that the database is updated properly and with the correct information.

**Preconditions:**

Must be logged in as Professor or Administrator and chosen a specific course/program to do selected operation on.

**Expected Input Type:**

1 string

**Expected Output:**

display question added

**Error Handling:**

**Input:** null string

**Output:** reload webpage and warn user that they cannot create a blank question

**Input:** extremely long string

**Output:** reload webpage and warn user about the character length limit on questions.

**Name:**

Remove\_Question()

**Primary Actor:**

Professors, Admins

**Description:**

Ensure that the database is updated properly and with the correct information. Ensure that user is sure about deleting the index specified question.

**Preconditions:**

Must be logged in as Professor or Administrator and chosen a specific course/program to do selected operation on.

**Expected Input Type:**

1 int

**Expected Output:**

updated survey without question that was deleted

**Error Handling:**

**Input:** 1 int (outside of array boundaries)

**Output:** reload webpage and warn user of the specific array boundaries for the survey they are working with

**Name:**

Change\_QA\_Type()

**Primary Actor:**

Professors, Admins

**Description:**

Ensure that the database is updated properly and with the correct information.

**Preconditions:**

Must be logged in as Professor or Administrator and chosen a specific course/program to do selected operation on.

**Expected Input Type:**

1 int

**Expected Output:**

updated question with answer field displaying according to updated answer type

**Error Handling:**

**Input:** 1 int (outside of array boundaries)

**Output:** reload webpage and warn user of the specific array boundaries for the survey they working with

**Name:**

Add\_Course()

**Primary Actor:**

Professor

**Description:**

Make sure that a professor can add a course to the course list

**Preconditions:**

User is a registered professor

Course with given name does exist

**Expected Input Type:**

1 String: Name of course

**Expected Output:**

New course comes up under list of courses, and is fully functional

**Error Handling:**

**Input:** Invalid course name

**Output:** Bring user back to original screen

**Name:**

Remove\_Course()

**Primary Actor:**

Professor

**Description:**

Make sure that a professor can remove a course from the course list

**Preconditions:**

User is a registered professor

Course with given name does exist

**Expected Input Type:**

1 String: Name of course

**Expected Output:**

Course no longer comes up under list of courses

**Error Handling:**

**Input:** Invalid course name

**Output:** Bring user back to original screen

**Name:**

Access\_Course()

**Primary Actor:**

Professor

**Description:**

Make sure that a professor has the ability to access one of his/her courses

**Preconditions:**

User is a registered professor

Course with given name does exist

**Expected Input Type:**

1 String: Name of course

**Expected Output:**

Professor is able to access a given course

**Error Handling:**

**Input:** Invalid course name

**Output:** Bring user back to original screen

**Name:**

Add\_Course()

**Primary Actor:**

Student

**Description:**

Make sure that a student can add a course that he/she is taking

**Preconditions:**

User is a registered student

Course with given name does exist

**Expected Input Type:**

1 String: Name of course

**Expected Output:**

New course comes up under list of courses, and is fully functional

**Error Handling:**

**Input:** Invalid course name

**Output:** Bring user back to original screen

**Name:**

Remove\_Course()

**Primary Actor:**

Student

**Description:**

Make sure that a student can remove a course that he/she is no longer taking

**Preconditions:**

User is a registered student

Course with given name does exist

**Expected Input Type:**

1 String: Name of course

**Expected Output:**

Course no longer comes up under list of courses

**Error Handling:**

**Input:** Invalid course name

**Output:** Bring user back to original screen

**Name:**

Access\_Course()

**Primary Actor:**

Student

**Description:**

Make sure that a student has the ability to access one of his/her courses

**Preconditions:**

User is a registered student

Course with given name does exist

**Expected Input Type:**

1 String: Name of course

**Expected Output:**

Student is able to access a given course

**Error Handling:**

**Input:** Invalid course name

**Output:** Bring user back to original screen

**Name:**

Take\_Prog\_Survey()

**Primary Actor:**

Student

**Description:**

Make sure that a student has the ability to take a progress survey for one of his/her classes

**Preconditions:**

User is a registered student

Student has the ability to access the given course

Progress survey was successfully created

**Expected Input Type:**

1 Course object

**Expected Output:**

Student is able to take the survey

**Error Handling:**

**Input:** Invalid course survey

**Output:** Bring user back to original screen

**Name:**

Update\_Prog\_Survey()

**Primary Actor:**

Student

**Description:**

Make sure that a student has the ability to update a progress survey for one of his/her courses

**Preconditions:**

User is a registered student  
Student has the ability to access the given course  
Progress survey was successfully created

**Expected Input Type:**

1 Course object

**Expected Output:**

Student is able to update the survey with new information

**Error Handling:**

**Input:** Invalid course survey  
**Output:** Bring user back to original screen

**Name:**

Create\_Course()

**Primary Actor:**

Admin

**Description:**

Make sure that the admin is able to create a course

**Preconditions:**

User is a registered admin  
Course with this name does not already exist

**Expected Input Type:**

1 Course object

**Expected Output:**

Course is added

**Error Handling:**

**Input:** Invalid course name  
**Output:** Bring user back to original screen

**Name:**

Delete\_Course()

**Primary Actor:**

Admin

**Description:**

Make sure that the admin is able to delete a course

**Preconditions:**

User is a registered admin

Course does already exist

**Expected Input Type:**

1 Course object

**Expected Output:**

Course is removed

**Error Handling:**

**Input:** Invalid course name

**Output:** Bring user back to original screen

**Name:**

Access\_Course()

**Primary Actor:**

Admin

**Description:**

Make sure that the admin is able to access a course

**Preconditions:**

User is a registered admin

Course does already exist

**Expected Input Type:**

1 Course object

**Expected Output:**

Admin is able to access a course

**Error Handling:**

**Input:** Invalid course name

**Output:** Bring user back to original screen

**Name:**

Create\_C\_Report()

**Primary Actor:**

Admin

**Description:**

Make sure that the admin is able to create a course report

**Preconditions:**

User is a registered admin

Course exists

**Expected Input Type:**

1 Course object

**Expected Output:**

The course report will be added to the course page

**Error Handling:**

**Input:** Invalid course

**Output:** Bring user back to original screen

**Name:**

Update\_C\_Database()

**Primary Actor:**

Admin

**Description:**

Make sure that the admin is able to update a course database

**Preconditions:**

User is a registered admin

Course exists

Course database exists

**Expected Input Type:**

1 Course object

**Expected Output:**

The course database will be updated with new information

**Error Handling:**

**Input:** Invalid course database

**Output:** Bring user back to original screen

**Name:**

Display\_C\_Report()

**Primary Actor:**

Admin

**Description:**

Make sure that the admin is able to display a course report

**Preconditions:**

User is a registered admin

Course exists

Course report exists

**Expected Input Type:**

1 Course object

**Expected Output:**

The course report will be brought up

**Error Handling:**

**Input:** Invalid course report

**Output:** Bring user back to original screen

## **Explanation of Test Plan**

Since the program we are developing depends heavily on user interaction and the proper methods for interacting with the database, we are taking a black box testing approach. In this sense it is important to ensure that each function works properly especially if the program may be implemented.