Ferris DeHart
Seth Richard
Nicklaus Settoon
Professor Mukhopadhyay
3380-Object_Oriented_Design
due 2020-Nov-1st
Github_link: ***https://github.com/nicksettoon/3380-xCourse***

Test Cases with User Stories for xCourse

1. Homepage

a. User logs in to the app

i. Greeted with xCourse homepage

ii. TESTCASE:

1. Inputs:
    a. Username: filled
    b. Password: filled
    c. Successful SSO login
2. Expected Output(EO):
    a. Display xCourse homepage
3. Actual Output(AO): ?
4. Pass/Fail Criterion: AO == EO

b. If the user selects “Plan”

i. Enter Plan Mode

ii. TESTCASE:

1. Inputs:
    a. planMode: Off
    b. planSelected: True
2. Expected Output(EO):
    a. planMode: On
    b. Enter Plan Mode
3. Actual Output(AO): ?
4. Pass/Fail Criterion: AO == EO

c. If user selects “Compare”

i. Enter Comparisons View

ii. TESTCASE:

1. Inputs:
    a. compareMode: Off
2. Expected Output(EO):
    a. compareMode: On
    b. Enter Comparisons View
3. Actual Output(AO): ?
4. Pass/Fail Criterion: AO == EO

2. Comparisons View

a. Description: User can create comparison pairs and save them for later use
   i. A comparison pair is a set of two degree flowcharts saved to the user account

b. User enters Comparisons View for the first time

   i. Enter create new comparison pair
   ii. TESTCASE:
      1. Inputs:
         a. compareView: On
         b. compareViewVisited: False
      2. Expected Output(EO):
         a. compareViewVisited: True
         b. compareView: Off
         c. compareMode: On
         d. enter new Compare Mode
      3. Actual Output(AO): ?
      4. Pass/Fail Criterion: AO == EO

c. User clicks button to create new comparison Pair

   i. User's current degree is loaded and displayed in the left panel
   ii. TESTCASE:
      1. Inputs:
         a. compareView: On
         b. newComparisonsButton: clicked
         c. compareMode: Off
      2. Expected Output(EO):
         a. compareView: Off
         b. compareMode: On
         c. enter new Comparison Mode
         d. leftPanel: user.currentDegree
         e. rightPanel: None
      3. Actual Output(AO): ?
      4. Pass/Fail Criterion: AO == EO

d. User clicks "add degree" on rightPanel

   i. Provide list of degrees
   ii. TESTCASE:
      1. Inputs:
         a. compareMode: On
         b. rightPanel: None
         c. addDegree: clicked
      2. Expected Output(EO):
         a. compareMode: Off
         b. showDegreeList: True
      3. Actual Output(AO): ?

4. Pass/Fail Criterion: AO == EO

iii. user selects degree from list

1. degree added to Compare Mode
2. ***TESTCASE:***
    a. Inputs:
        i. showDegreeList: True
        ii. degreeSelected: True
        iii. desiredDegree: FlowchartObject
        iv. compareMode: Off
    b. Expected Output(EO):
        i. showDegreeList: Off
        ii. compareMode: On
        iii. leftPanel: user.currentDegree
        iv. rightPanel: desiredDegree
        v. Go back to Compare Mode
    c. Actual Output(AO): ?
    d. Pass/Fail Criterion: AO == EO

e. User clicks comparison Pair

i. Open that comparison Pair for viewing and editing

ii. TESTCASE:

1. Inputs:
    a. compareView: On
    b. compareMode: Off
2. Expected Output(EO):
    a. compareView: Off
    b. compareMode: On
    c. enter Compare Mode for that pair
3. Actual Output(AO): ?
4. Pass/Fail Criterion: AO == EO

3. Compare Mode

a. Description:
    i. Two Plan Mode modules are displayed side by side
    ii. New layer is generated and added to both plan modes
        1. Displays classes in both degree paths in yellow
    iii. Compare mode works the same as Plan Mode but with two charts, side by side.

b. User selects panel and clicks view degree button

i. enter Plan Mode for that degree

ii. TESTCASE:

1. Inputs:
    a. compareMode: On
    b. panelSelected: True

c. selectedPanel: [leftPanel | rightPanel]
d. viewDegreeButton: clicked

2. Expected Output(EO):
   a. compareView: Off
   b. planMode: On
   c. enter Plan Mode for [leftPanel.degree | rightPanel.degree]
3. Actual Output(AO): ?
4. Pass/Fail Criterion: AO == EO

c. User selects panel and clicks change degree button

i. Pull up degree list and allow them to change the degree

ii. TESTCASE:

1. Inputs:
   a. compareMode: On
   b. panelSelected: True
   c. selectedPanel: [leftPanel | rightPanel]
   d. changeDegreeButton: clicked
   e. showDegreeList: False
2. Expected Output(EO):
   a. compareMode: Off
   b. showDegreeList: True
3. Actual Output(AO): ?
4. Pass/Fail Criterion: AO == EO

iii. user selects degree from list

d. User clicks back button

i. Enter Comparisons View

ii. TESTCASE:

1. Inputs:
   a. compareMode: On
   b. previousMode: CompareView
   c. backButton: clicked
   d. compareView: Off
2. Expected Output(EO):
   a. compareMode: Off
   b. previousMode: Whatever it was before entering compareMode
   c. compareView: On
   d. Enter Comparisons View
3. Actual Output(AO): ?
4. Pass/Fail Criterion: AO == EO

4. Plan Mode

a. Description: Flowchart for user's current degree plan is automatically loaded

b. User clicks on course bubble on flowchart

i. Course description is displayed

ii. TESTCASE:

1. Inputs:
    a. courseBubble: clicked
    b. courseSelected: CourseObject
2. Expected Output(EO):
    a. courseSelected.displayDescription()
3. Actual Output(AO): ?
4. Pass/Fail Criterion: AO == EO

c. User clicks elective bubble on flowchart

i. Displays elective options.

ii. TESTCASE:

1. Inputs:
    a. electiveBubble: clicked
    b. selectedBubble: ElectiveObject
2. Expected Output(EO):
    a. selectedBubble.displayOptions()
3. Actual Output(AO): ?
4. Pass/Fail Criterion: AO == EO

d. User checks/unchecks box for any layer

i. the colors for all courses in that layer are recalculated

ii. TESTCASE:

1. Inputs:
    a. layerHidden: [True | False]
    b. selectedLayer: LayerObject
    c. currentDegree: FlowchartObject
2. Expected Output(EO):
    a. currentDegree.updateColors(selectedLayer.getCourses())
3. Actual Output(AO): ?
4. Pass/Fail Criterion: AO == EO

e. User presses a button that says “Add a Minor.”

i. List of Minors is displayed

ii. User selects the desired minor.

iii. Adjusted flowchart is generated and displayed.

iv. TESTCASE:

1. Inputs:
    a. addMinorButton: clicked
    b. Minor: selected
    c. currentDegree: FlowchartObject
2. Expected Output(EO):
    a. currentDegree.recalculate()
3. Actual Output(AO): ?

4. Pass/Fail Criterion: AO == EO

f. User clicks button to export flowchart

i. save as pdf, png, jpg, json options given

ii. ***TESTCASE:***

1. Inputs:
    a. exportButton: clicked
    b. fileType: [pdf | png | jpg | json]
    c. currentDegree: FlowchartObject
2. Expected Output(EO):
    a. currentDegree.export(fileType)
3. Actual Output(AO): ?
4. Pass/Fail Criterion: AO == EO

g. User clicks button to make new plan

i. User selects a semester to assign the plan to (fall, spring, summer, intercessions)

ii. Enter Edit Mode

iii. ***TESTCASE:***

1. Inputs:
    a. newPlanButton: clicked
    b. planSemester: [fall | spring | summer | wintersession]
    c. planMode: On
    d. editMode: Off
2. Expected Output(EO):
    a. planMode: Off
    b. editMode: On
    c. Enter Edit Mode
3. Actual Output(AO): ?
4. Pass/Fail Criterion: AO == EO

5. Edit Mode

a. Description: Mode for editing a semester's courses (layer)

b. User clicks name of layer at the top to edit the name

i. Edit and save new name

ii. ***TESTCASE:***

1. Inputs:
    a. layerTitle: clicked
    b. currentName: string
    c. newName: string
2. Expected Output(EO):
    a. currentName: newName
3. Actual Output(AO): ?
4. Pass/Fail Criterion: AO == EO

c. User clicks the body of a course bubble

i. add to current plan layer

ii. update metadata, and time conflicts

iii. TESTCASE:

1. Inputs:
    a. courseBubble: clicked
    b. currentLayer: LayerObject
    c. selectedCourseBubble: CourseObject
    d. currentDegree: FlowchartObject
2. Expected Output(EO):
    a. LayerObject.add(selectedCourseBubble)
    b. currentDegree.update()
3. Actual Output(AO): ?
4. Pass/Fail Criterion: AO == EO

d. User clicks body of elective bubble

i. Display list of available courses
ii. user selects course from popup
iii. color of box is re-evaluated
iv. Add course to semester plan

v. TESTCASE:

1. Inputs:
    a. editMode: On
    b. electiveBubble: clicked
    c. selectedBubble: ElectiveObject
    d. currentLayer: LayerObject
    e. selectedBubble.displayOptions()
    f. selectedCourse: CourseObject from course list
    g. currentDegree: FlowchartObject
2. Expected Output(EO):
    a. currentLayer.add(selectedCourse)
    b. currentDegree.update()
3. Actual Output(AO): ?
4. Pass/Fail Criterion: AO == EO

e. User drags course bubble to a different plan layer

i. adds that course to that layer
ii. does not leave current layer's edit mode

iii. TESTCASE:

1. Inputs:
    a. courseBubble: dragged
    b. onLayer: True
    c. destinationLayer: LayerObject
    d. selectedCourseBubble: CourseObject
2. Expected Output(EO):
    a. destinationLayer.add(selectedCourseBubble)
3. Actual Output(AO): ?
4. Pass/Fail Criterion: AO == EO

f. User clicks on a different layer

i. switches to editing that layer

ii. TESTCASE:

1. Inputs:
    a. layerButton: clicked
    b. currentLayer: LayerObject
    c. selectedLayer: LayerObject
    d. editMode: On
2. Expected Output(EO):
    a. editMode: On
    b. currentLayer: selectedLayer
3. Actual Output(AO): ?
4. Pass/Fail Criterion: AO == EO

g. User clicks finish-plan button

i. prompt to select a color displays

ii. After finished,

1. lock the semester plan
2. return to Plan Mode, recalculating metadata

iii. TESTCASE:

1. Inputs:
    a. currentLayer: LayerObject
    b. editMode: On
    c. finishButton: clicked
    d. currentDegree: FlowchartObject
2. Expected Output(EO):
    a. editMode: Off
    b. currentLayer.save()
    c. currentLayer.lock()
    d. currentDegree.update()
    e. Enter Plan Mode
3. Actual Output(AO): ?
4. Pass/Fail Criterion: AO == EO