CS673 Software Engineering

Lab 2 - Requirement Analysis

## Description:

The purpose of this lab is to make sure every student gets a pivotal tracker set up for their project and knows how to use user stories to represent requirements.

## Instructions:

1. Set up a pivotal tracker account if you don’t have already.
2. Go to your project pivotal tracker website.
3. Click the “+ Add Story” button on the top right corner to add a user story. You should include a title, a description using the template **“As ...,I want to … So that …”**. Please be aware that your user stories in this lab should not be tasks about how to implement the project (such as setting up the database server), but functional requirements from the user’s point of view. NO credit will be given if your user stories are implementation tasks. As in the “Add a Project” user story shown in Figure 1, your user story ***type*** should be “**Feature**”. Make sure that your user story follows the “INVEST” principle. Later in your project, you may also use Pivotal Tracker to track general tasks across the user stories such as setting up the database server. The type of these stories should be set as “*Chore*” and in general no points are assigned, as these are mostly created for developers, not users. You can also track bugs and releases on Pivotal Tracker. **However, in this lab, your user stories must be “Feature” stories**.
4. In the “tasks” section, you should analyze the user story and add a breakdown of tasks, as well as the estimation of each task (in terms of person hours). Make sure to include all possible tasks involved in this user story from the start of the story to the acceptance of the user story, as shown in Figure 2.
5. You should also assign a story point to indicate the story complexity based on the estimation of your breakdown tasks. You can also add labels to group your user stories into epics as shown in Figure 1.
6. In the “activity” section, **add at least 2 acceptance tests** using the template “Given … When… Then… ”, as shown in Figure 3. Make sure that your acceptance tests are specific enough. They are NOT a repetition of the description. The objective is that the client will know how to test this user story exactly by following these acceptance tests.

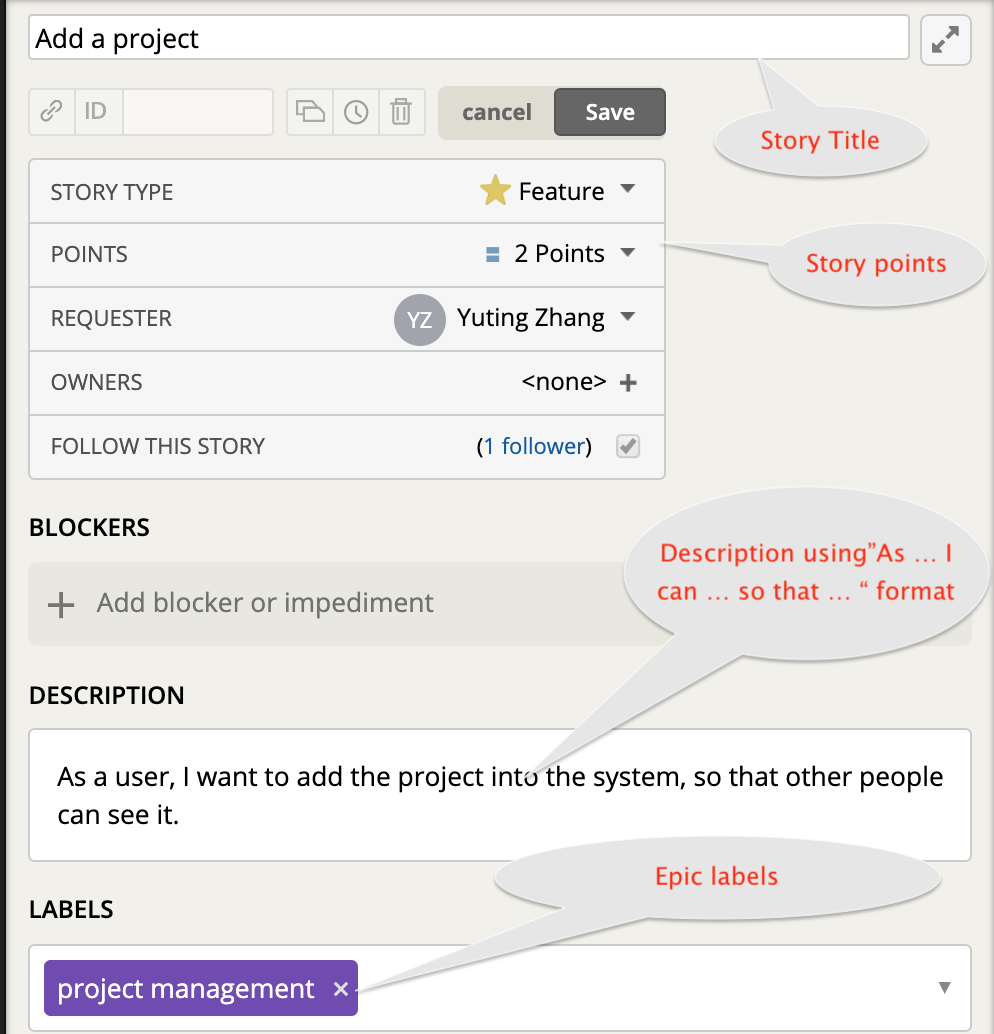


Figure 1 Add a Project

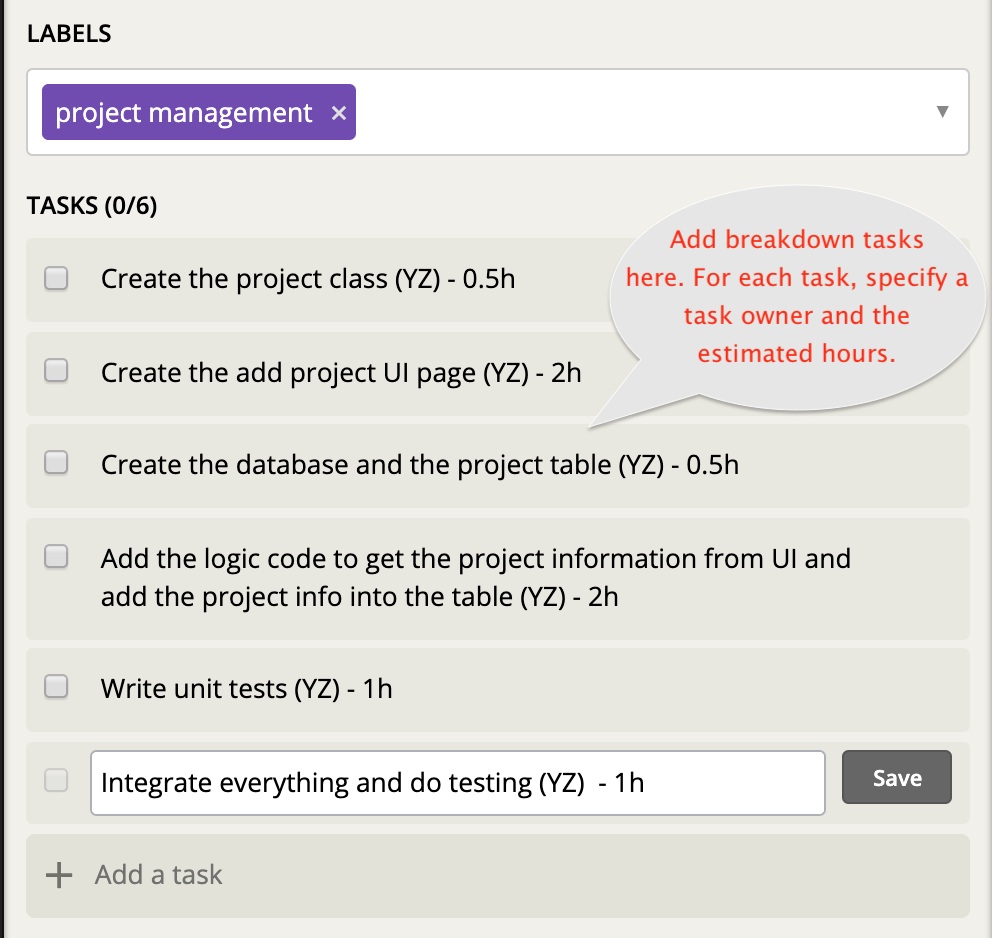


Figure 2 Add Tasks

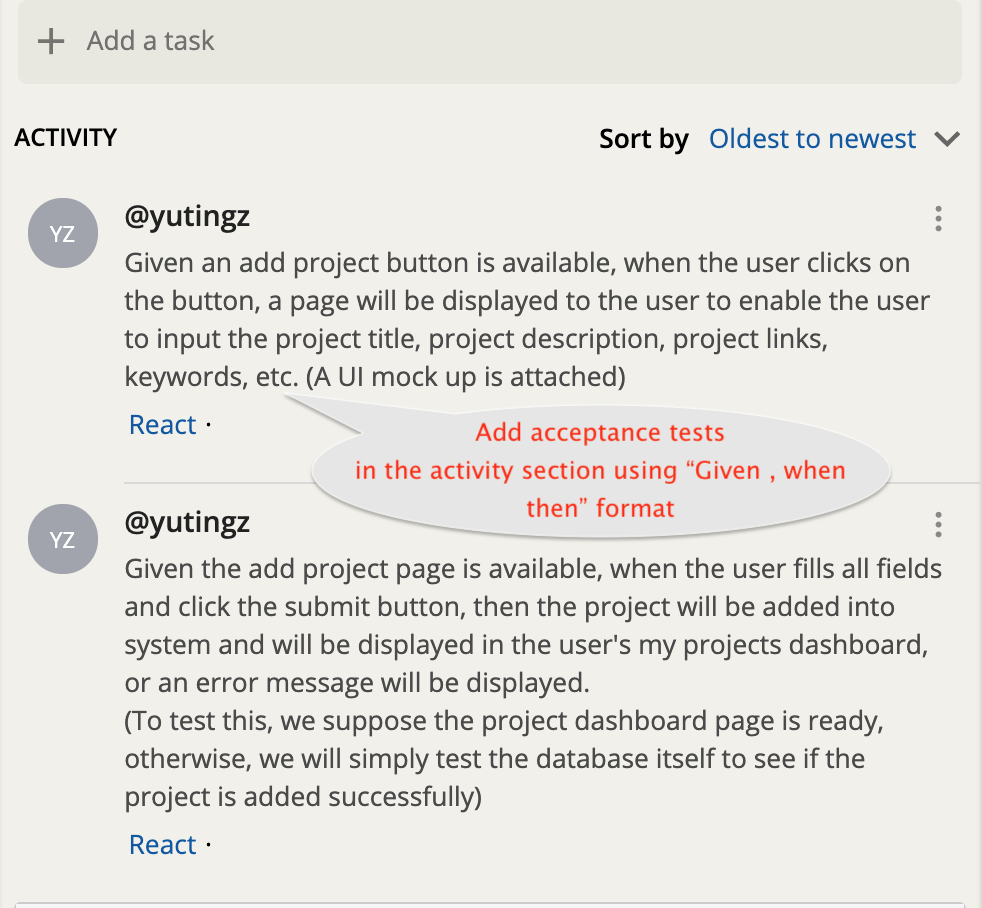
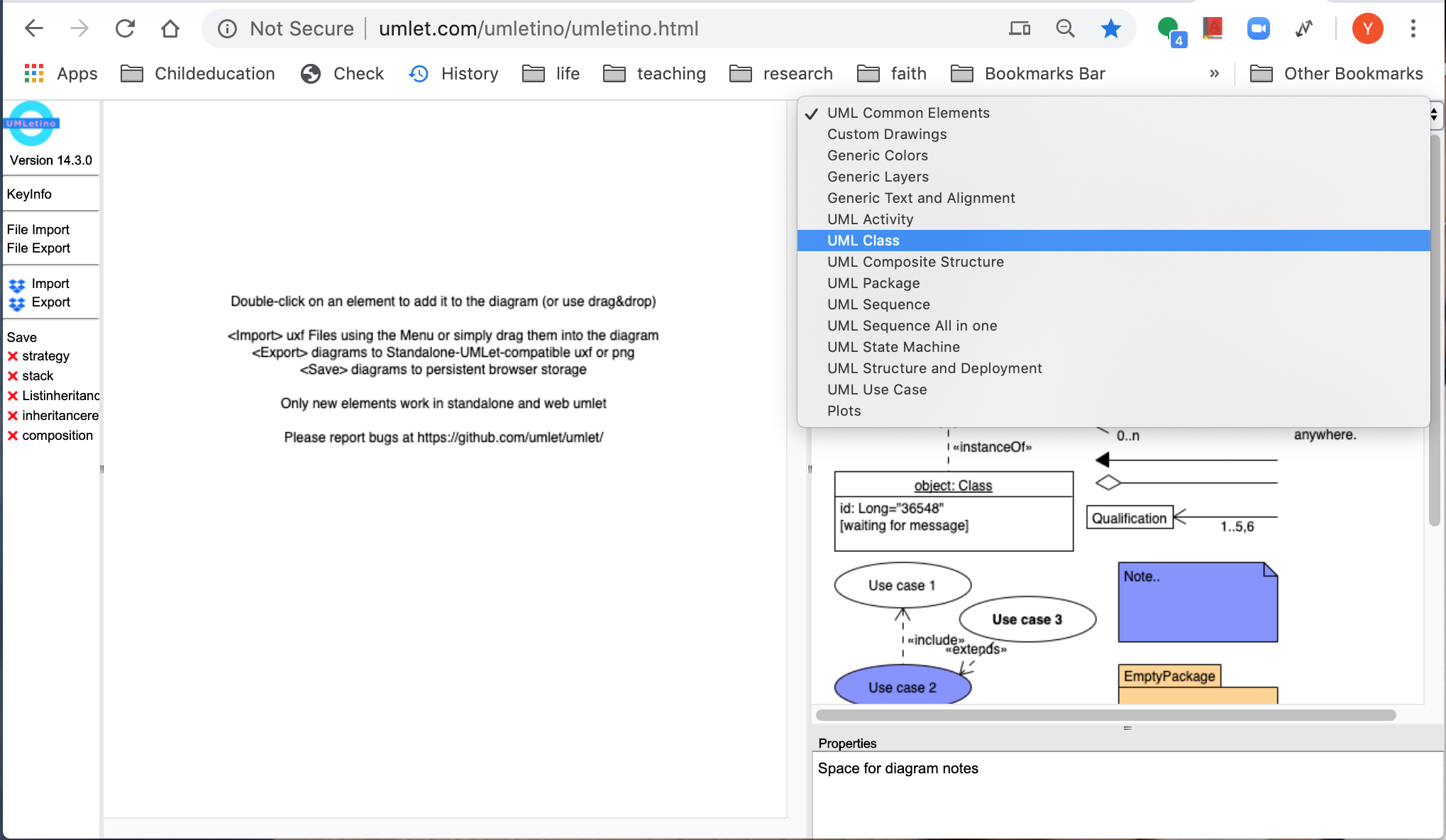
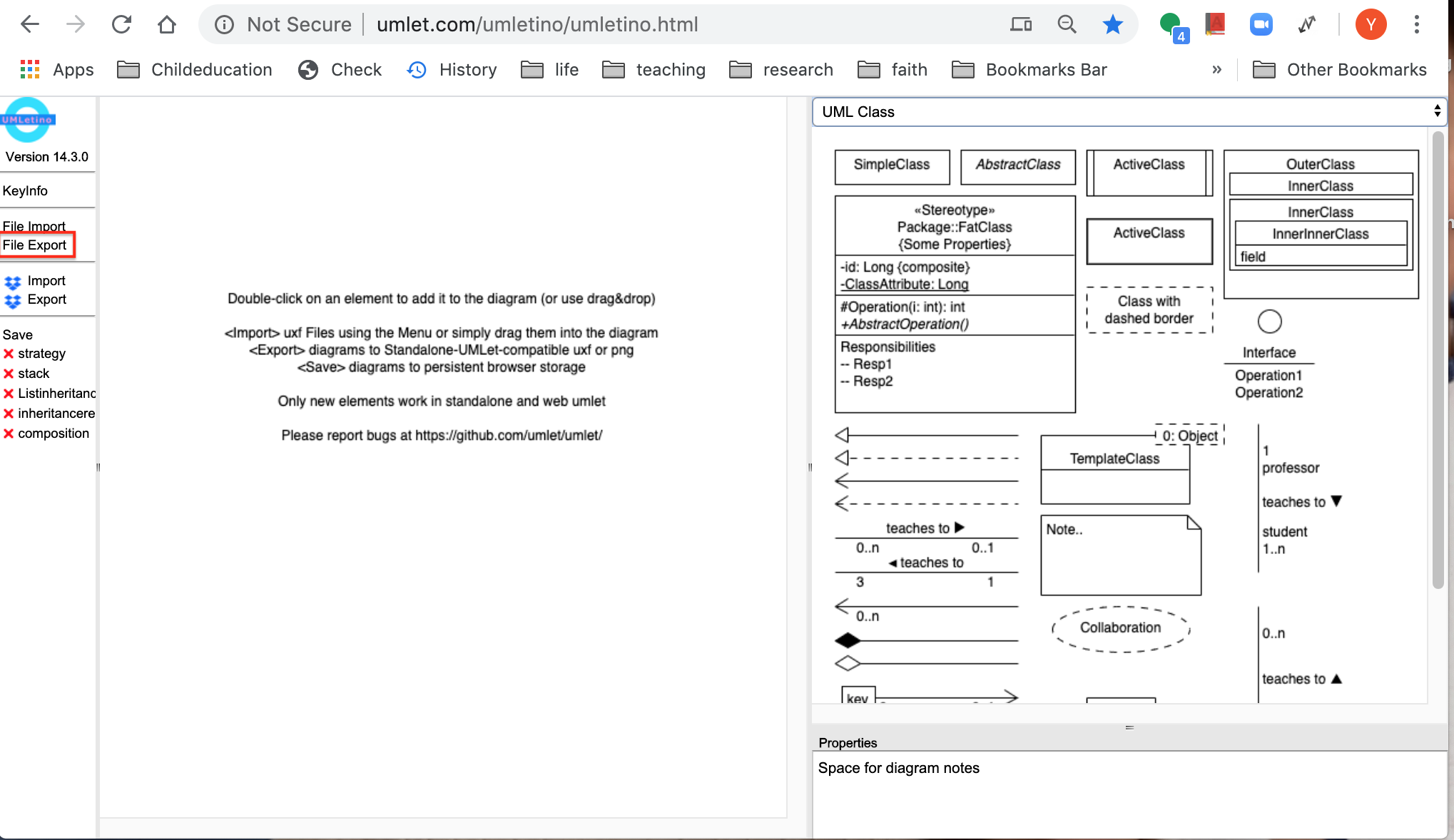


Figure 3 Add Acceptance Tests

1. Create **at least 2 functional user storie**s for your group project and store them in the icebox. Please do not change or delete other people’s stories. Try to create a different user story from others. All these stories should be stored in the icebox initially. They can be either the essential features, desirable features, or optional features in your group project proposal. The requirement leader can move some into the current iteration.
2. Analyze your 2 user stories, and identify three types of application domain classes that are related to your three user stories: entity, boundary, and control. These domain classes are for **both** of your user stories. You do NOT need to analyze them separately.
3. For each **entity** class, specify the **essential attributes and operations**,
4. Draw a **class diagram** to show the **entity class** relationships. Your class diagram should not be an EBC diagram, but the class diagram for all entity classes you identified in your two user stories. . You can use Umlet (<http://www.umlet.com/umletino/umletino.html>) to draw the class diagram. Choose “UML Class” in the dropdown box. You can also use other tools to draw the class diagram.



1. You can save your diagram as a uxf file using “File Export”, so that you can import it to Umlet using “File Import” in order to edit it later. You can also save it as an image file to add into your lab report.



1. Please submit a document on Blackboard to include the following sections in either pdf or doc format on blackboard.
   1. User stories on Pivotal Tracker
      1. Include your pivotaltracker project URL and your username in pivotaltracker.
      2. For each user story, include your **user story URLs** and some screenshots of your user story titles, their descriptions, breakdown tasks and acceptance tests. (Please make sure your screenshots are large enough to see).
   2. Entity, boundary and Control objects (You only need to provide one set for both of your stories). For each entity class, specify the essential attributes and operations.
      1. Entities
      2. Boundaries
      3. Controls
   3. A class Diagram to show the entity class relationships.

Aidan Duffy

METCS 673

Lab 2

1. User Stories on Pivotal Tracker:
   1. <https://www.pivotaltracker.com/n/projects/2531653>
   2. Username: anduffy
   3. Story #1 (Search for recipes by name):
      1. URL: <https://www.pivotaltracker.com/story/show/179651890>
      2. Description: As a user, I want to search the program for a recipe by its name, so that I can see if I have the necessary ingredients.
      3. Tasks and Tests:

Graphical user interface, text, application, email

Description automatically generated

* 1. Story #2 (Display image of recipe):
     1. URL: <https://www.pivotaltracker.com/story/show/179651904>
     2. Description: As the program, I should display an image of the finished recipe, so that the user could see the final product before cooking it.
     3. Tasks and Tests:

Graphical user interface, text, application, email

Description automatically generated

1. EBC
   1. Story #1:
      1. Entities: user(optional), ingredients, recipe, cheffy
      2. Boundaries: search on menu.html, recipe.html
      3. Controls: searchSpoonacularByName, displayRecipe
   2. Story #2:
      1. Entities: recipe, cheffy
      2. Boundaries: recipe.html
      3. Controls: displayRecipe, displayRecipeImage, fetchRecipeImage
   3. Entity necessary attributes
      1. Cheffy – recipeList, userList, givenRecipeName
      2. User – ingredientList
      3. Ingredients – ingredientList (same as ii. If no user logged in)
      4. Recipe – ingredientList (different from above, necessary ingredients for this recipe), name, image
   4. Entity necessary operations
      1. Cheffy – searchforRecipe, searchForRecipeByName, displayRecipe, displayRecipeImage, displayNoRecipeFoundError
      2. User – checkIngredients
      3. Recipe – getImage, getInformation
2. Class Diagram

Diagram

Description automatically generated