Quiz 2

Project management, UX and UI, Javascript, Rails/SaaS review

*Take-home, due at 11:59 pm on November 19 (submit as word document on Canvas). Each question is 2 pts.*

The goal of this assignment is to review fundamental concepts related to SaaS, Rails, Project management, UX and UI in order to help you identify weak spots that might hinder your progress with project development. You can use any resource (starting from the textbook) to answer the questions of this quiz, using the following guideline: if you’re learning, you can use the resource. If you avoid learning (e.g. find a forum where the answer is posted but not discussed, or find a classmate who knows the answer), you cannot use the resource. Googling for answers on student forums and similar activities of passing someone’s work as your own are not permitted and will be treated as Honor Code violation.

You are allowed to collaborate with your classmates on deriving and understanding the answers. You are not allowed to share answers. Each student should submit their own version of this take-home quiz.

# SaaS and Agile

1. If a software project is behind schedule, it may not be a good idea to add people in order to catch up because:
2. it takes time to bring new programmers up to speed

2. Compared to plan-and-document, Agile/XP places less emphasis on formal code reviews because:

1. pair programming is a form of code review that can be practiced constantly

3. The **single best predictor** that a software project is likely to come in excessively over budget or late schedule is:

* 1. project uses P&D methodology rather than Agile

4. Select ALL that apply. In Agile/XP, when a user story is delivered, who may sign off on the story as ’complete’?

* 1. A team member acting as the voice of the customer, such as the product owner – No, PO’s mark as accepted, not complete
  2. The developer whose code and tests constitute the delivered story

# Git

5. How do you compare two revisions of a file, or your current file and a previous revision? (just a command is sufficient)

git diff HEAD [filename]

# Javascript

6. Explain this code. What will it print and why (detailed explanation is required for full credit)?

function foo(x) {  
 var baz = 2;  
 return function (y) {  
   console.log(x + y + (baz++));  
 }  
}  
var bar = foo(4);  
bar(11);

17 will be the output. Bar is returned as the function inside foo. We’re passing the parameter 11 there, which will be our value of y. Baz is equal to 2, as it’s defined in foo, and x will be 4 since that is what was passed through foo. Baz will increase after the print statement executes (due to the placement of ++) so 17 is printed because we’re adding 11 + 4 + 2.

7. Event listeners: Write a click handler for the foobar div in jQuery that changes the background color of the div to #FF0000.

<div class="foobar">Change my background</div>

$(“#foobar”).click(function() {

$(this).css(‘background-color”, “#FF0000);

};

# Ruby and Rails

8. Suppose the example site rotsee.com wants to launch m.rotsee.com, an alternate version of the site for mobile clients. If the site was built using an MVC architecture, the majority of the work will go into writing new:

1. views

9. You want to add a new action "get\_new\_groups" to StudyGroupFinder, to list only groups created last week. We MUST create which of the following:  (1) a route, (2) a controller method, (3) a view?

1. Only (2)

10. Which statements are TRUE regarding Rails RESTful routes and the resources to which they refer? (more than one answer is expected)

1. A "resource" may be existing content or a request to modify something.
2. One common set of RESTful actions is the CRUD actions on models.

# Architecture

11. Why is cloud computing able to provide more computing power for a lower cost than traditional datacenters? (1-3 sentence answer)

Having a data center requires costs beyond just the software and hardware. You must pay for the physical building’s amenities in addition to labor costs for employees that are required to work at that data center. In addition, with cloud computing, most of the servers run similar applications on similar hardware, which is not how traditional data centers are designed.

12. Why is three-tier Shared-Nothing Architecture is a good fit for large-scale SaaS applications? (1-3 sentence answer)

A shared-nothing architecture is distributed and so there is no conflict between different storage spaces. This is good for large-scale applications so that a change in one node will not directly change other nodes, resulting in different changes where we don’t actually want them. There is no central node that will bottleneck the data, we can just add nodes on.