Project B

Use eXtreme Programming to begin implementing the system described in your assigned vision statement. This means that you should (1) collect user stories, (2) estimate the effort for those user stories, (3) get your customer to prioritize the stories, (4) organize into pair programming teams, (5) implement unit tests, (6)implement your assigned user stories, and (7) perform acceptance testing. In real XP, the customer would do the acceptance testing; however, for the purpose of this project, your team should do your own acceptance testing.

You have weeks 8, 9 and 10 to do this project. **It is likely that you will have time for only two iterations of (1) through (7), above**. In other words, even though XP uses 1-week cycles, and you will have 3 weeks, you are only expected to get through two 1-week cycles. Use a little of the extra time (up front) to figure out how you'll divide work among the team, and use a little of the extra time (at the end) to write up the document shown below.

Lecture videos from units 6 and 7 will teach you the concepts needed for this project. You do not need to do every possible thing described in this video (e.g. refactoring). You only need to do the steps (1)-(7) above, in order that you can write the PDF described below.

Parts of the PDF you must turn in

From the first XP cycle:

- A list of all the user stories collected (1 page)
- An estimate of the tasks and effort required for each user story (1 page)
- A priority list—indicating the priority assigned by the customer to each user story (1 page)
- A summary of how you organized into pairs, how you worked together in pairs (e.g., how you did screen sharing between drivers and co-pilots), what problems you encountered, and how you overcame those problems (1 page)
- A summary of the unit tests that you wrote and a description of what kinds of bugs, if any, those unit tests enabled you to find (1 page)
- A summary of what acceptance testing you did and a description of what kinds of bugs, if any, your
- acceptance testing enabled you to find (1 page)

From the second XP cycle:

- A summary changes to the list of user stories, changes to your estimates of user story effort, and changes to the priority list. For example, if you were able to complete some user stories in your first XP cycle, then the list of stories should be shorter. But you also might have discovered new stories, revised your estimates of effort, or found from the customer that priorities changed (1 page)
- A summary of how you divided into pairs for the second XP cycle and whether the second cycle experience was in any way different (easier?) than it was during the first cycle (1 page)

Reflection:

- A summary of the ways in which you found XP preferable to waterfall, and the ways in which you found waterfall preferable to XP – strengths and weaknesses of each process, in your experience (1 page). **You also have to submit the code through TEACH**

Grading criteria for Project B

Note that you will not be graded based on how much of the envisioned system you succeed in creating. You are going to be graded on how well you follow XP for two cycles. You do not need to implement a lot of the system; but whatever number of features you have implemented, did you write test case for them and are those actually well thought tests (because it's possible to write a huge number of trivial tests

that actually doesn't add any value). Whatever the number of features you decide to implement has to work correctly and has to be substantial enough according to the number of people in the group.

One example of not substantial implementation would be 6 people over 3 weeks period, implementing only three simple features such as adding, removing and updating record in a database using a frontend written in PHP.

The substantiality is subjective depending on the complexity of the feature, required setup and many other factors. So the instructor will decide whether the implementation is substantial enough or not.

- 10 points: Are all 10 (9 Parts of the pdf and code) of the required parts present?
- 10 points: Are all 10 (9 Parts of the pdf and code) of the required parts fairly easy to understand?
- 10 points: Are all 9 of the required parts consistent with one another?
- 5 points: Were the user stories properly written (concise and focused on requirements)?
- 10 points: Did the team actually divide into pairs when implementing the system?
- 10 points: Did the team actually create unit tests, perform acceptance tests and did those tests succeed in finding bugs?
- 5 points: Based on team member evaluation provided by each member
- 40 points: Is the work completed substantial enough according to the number of people in the group?

Submission guidelines:

- Any format (.tar, .zip) is fine. You can also try some of the online IDE's such as cloud 9 (https://c9.io/).
- Submit the actual code, test case etc. along with the PDF of the report in TEACH.
- A detailed step by step instruction on how to run the project (a working build/install script is also acceptable). If any cloud based solution is used refer that in the report and add the TA's and the instructor as project members.
- An explanation of how to get dependencies
- If it's a web based project, provide the url where it's hosted, etc.
- In short, I should be able to setup the project just by following the step by step instructions.