Code Workstation

Status Report 2

9/23/2015 - 10/7/2015

https://github.com/lrickard/CodeWorkstation/wiki

Lukas Rickard - Project Leader - l_rickard@u.pacific.edu
Jake MacMillan - Integrations Lead - j_macmillan1@u.pacific.edu

Project Schedule

Current Schedule

9/11/2015 - 9/23/2015 - Create outlines for all documents
9/11/2015 - 9/23/2015 - Figure out likely licensing
9/11/2015 - 9/23/2015 - Have considerable content in Requirements Analysis Document
9/11/2015 - 10/7/2015 - Figure out some likely APIs and Libraries
9/24/2015 - 10/7/2015 - Test out one of the APIs or Libraries
9/24/2015 - 10/28/2015 - Begin sketching plan of attack
9/24/2015 - 10/28/2015 - Create Content for Design Document
9/24/2015 - 12/2/2014 - Gather data from peers on which features would be most appreciated
10/8/2015 - 11/13/2015 - Create draft of Design Document
10/8/2015 - 11/13/2015 - Test remaining likely APIs
10/29/2015 - 11/13/2015 - Begin Final draft for Design document
11/13/2015 - 12/2/2015 - Finish 75% of test plan
11/13/2015 - 12/2/2015 - Revise Requirements Analysis documents based on Design
11/13/2015 - 12/2/2015 - Testing API/Library integration and feature compatibility
11/13/2014 - 12/2/2015 - Create plan of attack for implementation based on Design and integration testing

Code Workstation Gantt Chart	9/11	9/18	9/25	10/2	10/9	10/16	10/23	10/30	11/6	11/13	11/20	11/27	12/4
Create docs outlines	🟥	🟥	🟥										
Figure out likely licensing	🟧	🟧	🟧										
Begin filling in RAD	🟨	🟨	🟨										
Figure out likely APIs and libraries	🟩	🟩	🟩	🟩	🟩								
Begin testing APIs and/or libraries			🟦	🟦	🟦								
Begin sketching plan of attack			🟦	🟦	🟦	🟦	🟦	🟦					
Create content for design doc			🟦	🟦	🟦	🟦	🟦	🟦					
Gather data on appreciated features			🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦
Create draft of design doc					🟪	🟪	🟪	🟪	🟪	🟪			
Test remaining likely APIs					🟪	🟪	🟪	🟪	🟪	🟪			
Begin final draft for design doc								🟥	🟥	🟥			
Finish 75% of test plan										🟧	🟧	🟧	🟧
Revise RAD based on design doc										🟨	🟨	🟨	🟨
Testing API/library integration										🟩	🟩	🟩	🟩
Create plan for implementation										🟦	🟦	🟦	🟦

Schedule Changes

Our schedule has not changed at the high level, but we have defined better how we will investigate our coding options. Specifically “Begin sketching plan of attack“ will be us delving into the PyQt API and determining how we will structure the creation of new UI elements during runtime.

Current Status

We spent a lot of time in these past two weeks wrestling with Linux and trying to get it installed properly on our computers. We have also both installed PyQt, an API that we are planning to use for windowing, on our machines and have begun learning how to use it. We found that PyQt comes integrated with the text editing library which we are planning to use, Scintilla. There is also an interface designing tool (installs with PyQt) which saves Qt interface settings. We might be able to use this same or a similar configuration to save settings in our final program. We have started asking people features they would appreciate most and it seems to be the resizable boxes which can be resized and can automatically run scripts at a time interval.

Plan for 10/7/2015 - 10/23/2015

Start on:

1. Start sketching a plan of attack for implementing UI elements which will be customizable at run time. This feature is cardinal in our creation of a dynamic programming environment, so we want to begin assessing the difficulty and exploring methods of implementation with PyQt.
2. Start the Design Document. We should be able to sketch the classes we want based on the features we plan to have. This task will be partially dependant on how much we get done in our other task. Once we are sure PyQt can do everything we want it to, we will be able to start fitting components together.

Work on most:

1. Continue testing and learning how pyqt works.
2. Continue gathering data on which features will be most appreciated

Finish:

1. Determining which Libraries and API's will be necessary/desired.
2. Sequence Diagrams for RAD

Challenges and Concerns

We are not completely sure if the PyQt Library/API will help facilitate the dynamic user interface we want to create. We have been concerned that if we attempt to write our design document we will only need to re-write it once we actually start writing code, so we have shifted our priorities to writing test code. We think writing sloppy test code to see if implementing certain features will allow to understand at a meaningful level how we will implement the features we want to have. We have not found as much time as we wanted to work on senior project. Moving forward we plan to work on senior project Sundays and Wednesdays if time permits.

Updated Artifacts

Included with Update

Project Proposal (Unchanged)
Requirements Analysis Document

- Use Case Diagram consolidated
- Use Cases Named
- 1 to 1 between Use Cases and User Stories created
- Made user stories less dumb
- Fixed Functional and nonfunctional requirements

Updated Wiki and Repository (Linked, not included in zip)

- Added RAD to Repository
- Updated License info
- Updated Wiki a tiny bit

Status Report 1 and 2 (These will not change)

NOT Included in Update

System Design Document - Not really existent
Test Plan - Not started
Implementation Code - Nothing exists we want to share