**CS673S16 Software Engineering** 

**Team 4 - Job Skills Statistics Website**

**Project Proposal and Planning**

|  |  |  |  |
| --- | --- | --- | --- |
| Team Member | Role(s) | Signature | Date |
| Patrick Ryan | Design Leader | *Patrick Ryan* | 2/1/2018 |
| Matthew Scott | Team Leader | *Matthew Scott* | 2/1/2018 |
| John Staley | Requirements Lead | *John Staley* | 2/1/2018 |
| Peter DiMaria | Environment Lead | *Peter DiMaria* | 2/1/18 |
| Amal Krishna | QA/Implementation | *Amal Krishna* | 2/1/18 |
| James Christensen | Security Lead | *James Christensen* | 2/1/18 |
| Siwei He | Configuration Lead | *Siwei He* | 2/1/18 |
|  |  |  |  |

**Revision history**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Author** | **Date** | **Change** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

[Overview](#_87t9hln2vjz0)

[Related Work](#_mps353x5ezyl)

[Detailed Description](#_fg3z0hpd4q9v)

[Management Plan](#_ds8oyr75pnh1)

[Process Model](https://docs.google.com/document/d/107bVcXdAG-ogRr90PquFB8-aWGvTwSua8pu_O4Kmz6c/edit#heading=h.27177f40uci)

[Risk Management](https://docs.google.com/document/d/107bVcXdAG-ogRr90PquFB8-aWGvTwSua8pu_O4Kmz6c/edit#heading=h.a4oqwntk3mw)

[Monitoring and Controlling Mechanism](https://docs.google.com/document/d/107bVcXdAG-ogRr90PquFB8-aWGvTwSua8pu_O4Kmz6c/edit#heading=h.ywdoc2clc9yt)

[Schedule and deadline](#_tadq5mb0pici)

[Quality Assurance Plan](https://docs.google.com/document/d/107bVcXdAG-ogRr90PquFB8-aWGvTwSua8pu_O4Kmz6c/edit#heading=h.72e1f4uawy2r)

[Metrics](#_b2haznn3yyz2)

[Standard](https://docs.google.com/document/d/107bVcXdAG-ogRr90PquFB8-aWGvTwSua8pu_O4Kmz6c/edit#heading=h.vc72k6dweldv)

[Inspection/Review Process](#_f1c69ifi68h7)

[Testing](https://docs.google.com/document/d/107bVcXdAG-ogRr90PquFB8-aWGvTwSua8pu_O4Kmz6c/edit#heading=h.r5d5mhtlf0kq)

[Defect Management](https://docs.google.com/document/d/107bVcXdAG-ogRr90PquFB8-aWGvTwSua8pu_O4Kmz6c/edit#heading=h.54a4wuncjg1c)

[Process improvement process](#_jhct37ebxxpn)

[Configuration Management Plan](https://docs.google.com/document/d/107bVcXdAG-ogRr90PquFB8-aWGvTwSua8pu_O4Kmz6c/edit#heading=h.hw41vg4ykxen)

[Configuration items and tools](https://docs.google.com/document/d/107bVcXdAG-ogRr90PquFB8-aWGvTwSua8pu_O4Kmz6c/edit#heading=h.bwlb4d4vdox2)

[code commit guidelines](#_yyauft6zr9hw)

[References](https://docs.google.com/document/d/107bVcXdAG-ogRr90PquFB8-aWGvTwSua8pu_O4Kmz6c/edit#heading=h.8mva2050iy7t)

[Glossary](#_ty3i2nqffhtc)

# Overview

The purpose of this project is to present information about available employment opportunities and the skills posted as required or desired for those positions. This information could be used by educational institutions, students, and job seekers to make them more desirable candidates for the job types that they are interested in.

# Related Work

(describe any similar software systems, and the differences from them)

<https://insights.stackoverflow.com/survey/2017> This stack overflow survey is similar to what we are trying to accomplish. Where they are giving information from a voluntary survey, we’d like to accumulate data from actual job postings, but display very similar information.

# Proposed High level Requirements

* 1. Functional Requirements
     1. Essential Features (the core features that you definitely need to finish)  
        Percentage of skills by region and job type (at least one combination region/job type) displayed on a website

- web scraping through Beautiful Soup

- Visualization of scraped statistics through Plotly

- Visualized statistics of job type, skills associated with job, salary, location

- SQL Lite database

- Homepage/UI

* + 1. Desirable Features (the nice features that you really want to have too)
* Data for many different regions/job types available for display  
  More insight than percentage of skills
* Data from several job posting sites
* Load screen (UI)
* Extended location of job statistics
* Animated visualization of statistics on website
  + 1. Optional Features (additional cool features that you want to have if there is time)
* Skill clusters (which skills appear together frequently)
* Data sliced by historical periods and trends displayed
* Map of US with State shading based on usage of certain skills in Job Titles
  + 1. Existing Features (Not applied to a brand new project)
  1. Nonfunctional Requirements (example: Compatible with all browsers, response time)

Compatibility: cross-browser, mobile breakpoints

* Deployment: by Iteration 3
* Reusability: through coding standards (Python, JS, HTML/CSS) Team Lead - Amal
* Documentation: robust markup file posted on GitHub Team Lead - John
* Open Source (GitHub)
* Quality (See No. 6b)
  1. Implemented Features (*to be completed at the end of each iteration*)
* 0: Complete SPPP, delineate leadership and tasks
* 1: Bones of Homepage/UI
* 2: Working Scraper and Visualization
* 3: Fix all bugs (finish QA), manage desirable and optimal features, deploy, polish UI

# Management Plan

# (For more detail, please refer to SPMP document for encounter example)

## Process Model Due to some of the requirements of the project, we will be most closely following the Scrum methodology. Unfortunately, we will not be able to hold daily meetings, but we will also not all be working on this project on a daily basis, so that portion would be unnecessary. We are, however, on near 30 day cycles. We are also maintaining a product backlog on the website, pivotal tracker, which we will be updating every sprint and as tasks are completed. We will have sprint planning sessions at the beginning of every iteration in order to populate the iteration backlog. Since we will be presenting a defect report at the end of each iteration, we will need a suite of unit and functional tests that can be run on each branch of the code base on an as needed basis. We will also run this suite after each merge from individual branches to the dev branch and from the dev branch to the main branch or as deemed necessary by the QA leader, Amal Krishna. We will have our sprint review in the form of our iteration presentation in class at the end of each iteration.

## Objectives and Priorities

Our top priority is that the project is completed on schedule. Our second priority is completing all essential features. Our third priority is that the software has no known bugs. Our fourth priority is our non functional requirements. Our fifth priority is to finish as many desired functional requirements as possible. Our sixth priority is to finish as many optional features as possible.

## Risk Management (need update constantly)

## Monitoring and Controlling Mechanism Our entire team will have scheduled meetings in person on Thursdays after class. Additionally, we will have a scheduled virtual meeting every Saturday at 1 PM on Slack. Additional meetings can be held if necessary. Meetings can be cancelled if there is nothing to discuss. When these meetings coincide with the beginning of an iteration, the agenda will include iteration planning.

## Schedule and deadlines (need update constantly)

# Quality Assurance Plan

# (For more detail, please refer to SQAP document for encounter example)

## Metrics

* + 1. Definition (e.g. define what metrics will be used, , how to keep track of metrics, and how to analyze the metrics for process improvement. Two types of metrics should be included: product metrics and process metrics. Particularly include product complexity (LOC, # of files, # of classes, # of methods etc.) cost (in terms of man hours), defect and defect fix rate etc.

Total number of test cases: anywhere between 20-30 test cases. Implement 10 per iteration from 1,2 and 3.

Time spend on coming up with test cases

Time spend on implementing test cases

Test case run time before deployment

The coverage of test cases in terms of overall functionality of the web app

* + 1. Results (to be completed at the end of each iteration)

## Standard (e.g. documentation standard, coding standard)

Different IDE’s like Eclipse, Visual Code etc are being used according to each individual’s comfortability. I believe there is no need to settle on a single IDE since we don’t use any specific plugins or functionality of any individual IDE.

For coding standard it was decided to go with the Django coding style.

<https://docs.djangoproject.com/en/dev/internals/contributing/writing-code/coding-style/>

## Inspection/Review Process

## (e.g. describe what are subject to review, when to conduct review, who do the reviews and how ?)

Individual code review should be made by each contributing member while pushing code to the development branch, in order to avoid merge conflicts.

High level code review will be made once every week and make sure all the individual components are still integrated, working correctly and clears all test cases.

## Testing

## (e.g. who, when and what type of testing to be performed? How to keep track of testing results?)

A separate document about testing result should be linked here.

1. Unit testing for individual components.
2. End to end integration testing through the pages.
3. Functional testing for scrapping functions.
4. Regression testing should be added as new scrapping functions are added.
5. Data integrity has to be tested between the real-time data and SQLite database.

## Defect Management

(e.g. describe the criteria of defect, also in terms of severity, extend, priority, etc. The tool used to management defect, actions or personnel for defect management)

# Configuration Management Plan

(For more detail, please refer to SCMP document for encounter example)

## Configuration items and tools

i. Git, Github

ii. IDE: Eclipse, PyCharm

iii. Pivotal Tracker: https://www.pivotaltracker.com/n/projects/2146687

## Change management and branch management

i. Branch management:

● master branch: Product version.

● development branch: Developers can feel free to pulling code for their own branches.

ii. Developers use their own github account to fork the project repository.

iii. Developers can create branches on their own github and local machine.

iv. Developers can push to their own github account freely .

## Code commit guidelines

i. The changes are always committed with the appropriate message, describing the changes made to the code/document in comparison to the previous version.

ii. The committed code should be neatly indented and properly commented on for understanding by other developers.

iii. Before commit (merge), it should be tested first.

# References

(For more detail, please refer to encounter example in the book or the software version of the documents posted on blackboard. )

# Glossary