**CS673S16 Software Engineering**

**Team 1 - Project Name**

**Project Proposal and Planning**

|  |  |  |  |
| --- | --- | --- | --- |
| Team Member | Role(s) | Signature | Date |
| Rohit Agrawal | Team Leader | *Rohit Agrawal* | 9/26/2017 |
| Yashvardhan Nanavati | Backup Team Leader/ Configuration Leader/ Implementation Leader | *Yashvardhan Nanavati* | 9/24/2017 |
| Wenjun Shen | Requirement Leader/Environment and Integration Leader | *Wenjun Shen* | 9/26/2017 |
| Tianqi Xu | Requirement Leader/ Security Leader | *Tianqi Xu* | 9/26/2017 |
| Anirvan Maiti | Environment & Integration Leader  Design Leader | *Anirvan Maiti* | 09/26/2017 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Revision history**

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

[Overview](#_3whwml4)

[Related Work](#_2bn6wsx)

[Detailed Description](#_qsh70q)

[Management Plan](#_2et92p0)

[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](#_1t3h5sf)

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

[Metrics](#_3as4poj)

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

[Inspection/Review Process](#_35nkun2)

[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](#_1pxezwc)

[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](#_4i7ojhp)

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

[Glossary](#_1ci93xb)

# Overview

Motivation for this project comes mainly from the inability of the existing visualization tools like ‘Kibana’ to parse multi-level data in the form of JSON and their outdated, lack-lustered visualizations. Although powerful, Kibana lacks to produce interactive visualizations defeating its own purpose.

The name Ukubuka is inspired from a unique language spoken in South Africa and roughly translates to *"View"* or *"Visualization"* in Zulu which is one of the official languages of South Africa and is spoken by about 9 million people mainly in Zululand.

Ukubuka aims at producing high-quality interactive visualizations. It is powerful, lightweight and easy to use. It allows the users to analyze, manipulate and visualize their datasets in a novel and creative ways with no serious programming.

# 

# 

# Related Work

Kibana is a visualization tool which allows its users to create visualizations from data indexes in elasticsearch. Elasticsearch is a storage system in which large chunks of data can be stored in a user-defined format. Kibana basically extracts the data from Elasticsearch and provides a GUI to create visualizations and extract meaningful metrics.

The main drawback of Kibana is that it is very constrained in the type of data it can process and allows only a handful of simple graphical methods to choose from. For instance, if a user wants to visualize an attribute which contains multiple sub-attributes in the elasticsearch index then he/she has to go through a tedious process by taking help of another open source software named ‘logstash’ in order to make changes and make the data suitable for Kibana to consume. These are some of the challenges we plan to overcome and provide simple and effective solutions.

# Proposed High level Requirements

* 1. Functional Requirements
     1. Essential Features
* Data Import - With Ukubuka, it’s possible to import your data from many sources including the text files, CSV files, XML files, and databases etc.
* Data Manipulation - You can manipulate your data using a huge bunch of functions.
* Data Visualization - You can visualize your data in novel and creative ways.
  + 1. Desirable Features
* Data Export - You can export your analyzed and manipulated data in the form of flat files and databases. You can also export your visualizations and share them with the world.
* Workflows - You can document your workflows and save them for future use.
  + 1. Optional Features
* Data Analysis - You can run various analysis on your data.
  1. Nonfunctional Requirements
     1. Setting up the development environment by installing Eclipse, JDK 1.8, SonarQube, and Git.
     2. Distributing work among team members and ensuring a uniform workflow.
  2. Implemented Features (to be completed at the end of project)

# Management Plan

## Process Model

* + 1. We will be having three one weekly sprints per iteration throughout the course of the project.
    2. Each member will have stories worth 3 points per sprint (per week).
    3. There will be a sprint review and planning meeting at the end of each week to review the work done and plan the work for the next sprint.
    4. During the sprint, team members will submit PR’s which will contain the code they have written; It will be reviewed and merged with the base code.

## Objectives and Priorities

* + 1. Iteration 1: Finishing parsing the raw data(CSV and XML) into json file, which we can process more conveniently
    2. Iteration 2: Developing the visualization of the json data so that we can achieve our core functionalities.
    3. Iteration 3: Design a User Interface for appearance.

## Risk Management (need update constantly)

* + 1. The sprint review meetings will make sure that the project is on schedule. Each member will talk about the difficulties faced during the sprint and their work.
    2. We will make use of the weekly burndown charts and pivotal tracker board to keep track of the backlog.

## Monitoring and Controlling Mechanism

* + 1. Github for version control
    2. Pivotaltracker for tracking project progress
       1. Weekly sprints
       2. Burndown charts
       3. Team meetings
    3. Slack for team communications

## Schedule and deadlines (need update constantly)

|  |  |
| --- | --- |
| Iteration 0: Project Proposal | 09 - 27 - 2017 |
| Iteration 1: File parsing RESTFUL API | 10 - 18 - 2017 |
| Iteration 2: Developing visualizations for parsed data | 11 - 08 - 2017 |
| Iteration 3: Design a UI | 11 - 31 - 2017 |

# Quality Assurance Plan

## Metrics

* + 1. Definition:
       1. Product Metrics: Measurements of product quality.
          1. Product Complexity: number of files, number of classes, number of methods, and cost in terms of man hours.
       2. Process Metrics: Measurements of process quality.

* + 1. Results:
       1. Iteration 0: No files added. Drafted software project plan. Cost: 30 man hours.

## Standard

1. Documentation Standards
2. Reversion history. If change is made to document, the author, version number, timestamp, and a short description of the change should be added into reversion history.
3. Coding standards
4. Allocate appropriate variable names.
5. Each file of code should include a short description of the general purpose of the code.For example, what does this file do?
6. Each block of code should be explained specific enough that the function of code can be understand without reading the actual code.

## Inspection/Review Process

## It should be checked that the jobs assigned to every member is completed by them on time

1. Check that the requirements of the project are met by the members and not to deviate
2. Have enough time towards the end of every iteration so that there is time to double check
3. See that the variable names are proper and the code is clean so that the code can later be easily used/reviewed by other members of the team
4. Yashvardhan does the review process and code merging using github as he has the commit rights.

## Testing

## Unit testing - Individual units of source code having control data along with with usage and operation procedures are tested.

1. Mocking frameworks will be used using J units. Eg: mockito, easymock, etc
2. There should be at least 75% of code coverage before review process
3. Integration testing - Every members software modules are combined and tested as a group. Modules that have been unit tested are taken as input in integration testing and tests are applied to those aggregates.
4. Regression testing - In this testing we verify that the software which was developed earlier still works the same after after code integration and that the user requirements are still met

## 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)

1. Minor Defect
   1. Severity: Does not need immediate action. Small bug that does not affect main functionalities of the software.
   2. Priority: Low.
2. Major Defect
   1. Severity:May require activation of plan in the future. Defect that impacts user experience negatively.
   2. Priority: Medium.
3. Urgent Defect
   1. Severity: Require immediate action. Defect that causes software to crash, or leaks information that cause severe security issue.
   2. Priority:High.

# Configuration Management Plan

## Configuration items and tools

* + 1. Github

## Change management and branch management

* + 1. All pull-requests(PRs) submitted by team members will be reviewed by the Configuration Leader (Yashvardhan Nanavati) and Project Leader (Rohit Agrawal)
    2. The code will then be merged to the ukubuka/core branch on Github and the PR will be closed.
    3. Test Coverage will be required to be >=75% in order to get the code approved and merged on the main project repository.
    4. Only the team leader and the configuration leader will have commit rights to the main project repository.

## Code commit guidelines

* + 1. **General Workflow for the first time**:
       1. Fork the project into your own account.
       2. Clone the git repository on your local machine.
       3. Start working on the code.
       4. Submit a pull-request(PR) on the main git repository once you have worked on the code.
       5. Follow the steps mentioned below for subsequent changes.
    2. **Please follow the following steps before you start working again i.e. from the second time**:
       1. Git fetch upstream
       2. Git merge upstream/core
       3. Make changes on your code
       4. Git add <file>
       5. Git commit -m “message”
       6. Git push
       7. Open a PR on the main repository

# References

Dex(<https://github.com/PatMartin/Dex>): The Data Explorer -- A data visualization tool written in Java/Groovy/JavaFX capable of powerful ETL and publishing web visualizations.

Kibana(<https://www.elastic.co/products/kibana>): Kibana gives you the freedom to select the way you give shape to your data. And you don’t always have to know what you're looking for. With its interactive visualizations, start with one question and see where it leads you.

# Glossary