WorkPlan

Name of intern: Rajdeep Mukherjee

Project Title: Frameworks for automation of infrastructure at BigBasket

Analysis of the problem:

- The initial portion of the project consists of an thorough understanding of the production code at BigBasket as this will be essential for work in any domain/role in the company. Documentation of certain undocumented portions of the API is crucial for the understanding of code for future employees of the company and also for the purpose of debugging APIs as and when necessary. Large scale applications are built by a team and it is crucial for all team members to aware of the respective API endpoints of other team members that they are going to use. This is where proper API referencing is necessary for the application.
- The next portion of the project will consist of creation and maintenence of frameworks for automating build, deployment, testing and maintenance tasks necessary for the application. These tasks come under the broad category of **devops**. To start with I will be focussing on continuous integration. Continuous integration dictates that production code be "pushed" into shared repositories and all tasks and testing, building and monitoring of the application would be done automatically through a configured software like jenkins. Building of custom frameworks for tweaking any aspect of continuous integrtion would also fall in this category. When it comes to large software coded by a team of members it is rather easy to overlook certain critical aspects of the functioning of software. Hence tests are written to address these critical aspects. The development team should not be bogged down with the technical difficulties of deployment. This is where the devops team comes in to ensure deployment in a fast, effective and cost-efficient manner. Another aspect of this phase would highlight yet another concept in maintenance of large applications namely stream processing. When large systems are built, maintaining communication between these systems becomes too difficult without using stream pocessing. Stream processing architecture is therefore crucial for maintaining communication between various systems in the entire network.

Project Plan:

- The initial portion of the project will be the documentation of undocumented portions of the APIs of the application. According to python software foundation, python code should be documented with Sphinx. Sphinx is an execellent tool especially for the purpose of documentation of python code. The documentation can be written in restructured text format and Sphinx can then be used for rendering the entire documentation along with API references to html format. Python docstrings written in rst format can be extracted directly from the python scripts using an extension of Sphinx called **autodoc**. Undocumented portions of production APIs of BigBasket will be documented using Sphinx. Since the codebase is large it would be manually impossible to write restructured text for the purpose of extracting docstrings in the application. Hence this portion of the project will also include writing python scripts to autogenerate the restructured text

for extracting docstring s and then rendering that docstring in the form of html. The targets to be achieved in this phase would be an html rendered documentation of the codebase. This phase of the project will constitute the first month of the internship.
- The next portion of project will involve building of frameworks for automation of infrastructure at BigBasket. This involves the use of stream processing softwares like apache kafka and rabbitMQ. For the purposes of continuous integration one popular choice is jenkins. For the initial portion of this phase, tutorials and guides on setting up jenkins and the use of jenkins would be crucial as the only continuous integration software that I have used in one of my projects was Travis CI. Building of custom designed frameworks for automation at various levels is crucial and integration of that with jenkins would compose this phase of the project. This phase of the project will form the next 5 months of the internship. A clear demarkation of commencement and conclusion of project components is difficult to indicate as it will depend on my ability to complete assigned tasks within deadlines.

Project Deliverables

- The initial phase of the project including documentation of the APIs of the production code would be uploaded to the company servers for reference by the end of the first month of internship. The components of this phase as part of deliverables would include html rendered builds for setting up a static website capable of search within the APIs. These deliverables would have to be submitted by the end of the first month of the internship.
- The deliverables for the next phase would simply be1 the uploading of custom automation codes to company servers for use. Once again the strict components of the deliverables are hard to indicate as it will heavily depend of my abilities to submit within deadlines.

Time Line

At this point of time it was not possible to fix a proper timeline for achieving specific targets in the building of components of the project. That would be intimated in a continuous fashion as and when specific targets are cleared.

For now the only possible timeline would be that I am going to document undocumented portions of the APIs in the first month of the internship and for the next 5 months it would be the working with jenkins, kafka and the like.

Relevant skills

1. `Python` as a programming language including a thorough knowledge of certain system python packages especially `sys` and `os`.
2. `Restructured text`. Until now I had been documenting in markdown(md) however with Sphinx it would be most convenient to do it using restructured text. Restructured text has certain directives that enable extraction of docstrings from python scripts for modules, classes and methods. For all purposes of API referencing python docstrings are super important.
3. `Django`, as django is probably the go-to framework for designing large scale web applications in python.
4. `Mysql`. Although Django already comes with a light version sql called sqlite, it is not feasible for building databases with custom features and tweaks in its model. For that we would need a full

featured version of the software as in mysql.
5. `Java`. Since we will be working with Jenkins, it goes without saying that java would be a relevant skill to have.
6. `Apache Kafka`. `Apache Kafka` and hence `Confluent-Kafka` as a stream processing software is essential.
7. `Jenkins`. For the purpose of properly implementing continuous integration it is necessary to learn `Jenkins`.
8. Other skills would be a thorough knowledge of operating systems and their functioning, databases including query transactions and the like. Since everything would be on Linux, a good grasp of popular commands for crisp and clean functioning is necessary.
9. Among other skills some would definitely be discovered as and when I get my hands dirty with the project itself.

Skills to learn

1. `Restructured Text`. Since I have had experience only with markdown for documentation purposes, a knowledge of `Restructured text` would be essential.
2. `Django`. I have had experience in Flask for python web development and not `django`. `Django` being a more complete and full featured web development framework would require clear understanding.
3. `Jenkins`. My prior experience with continuos integration has been with `TravisCI` and testing has been `Nose`. Therefore it is necessary to learn `Jenkins`.
4. `Apache Kafka`. Stream processing is a complete new territory for me and hence it is essential for me to learn and apply it.
5. Once again it is worth mentioning that there will definitely be other skills to master that I will understand once I get my hands dirty in code.