Cognitive Regulatory Compliance Manager Tool

Project Workbook

SAN JOSÉ STATE
UNIVERSITY

By

Akansha Mehta (akansha.mehta@sjsu.edu)
Shweta Kothari (shwetaajit.kothari@sjsu.edu)
Sreedeep Katragadda(sreedeep.katragadda@sjsu.edu)
Sai Ravi Tejabhishek Sreepada(sairavitejabhishek.sreepada@sjsu.edu)

Spring 2018

Advisor: Professor Rakesh Ranjan

Contents

Abstract

There are thousands of compliances and regulations imposed by federal and state government that large-scale businesses have to conform to and comply in order to do business in a specific country.

Currently, a particular set of individuals called compliance managers are hired by business organizations and are paid millions of dollars to review regulations and to ensure that employees, management, processes and thus the whole organization is adhering to those policies, and if not, enforce the policies set forth by federal or state government or both. The entire process of understanding a new policy, the business, co-relating both and finding out which are applicable to the organization, is time consuming and is becoming expensive. This is where our tool comes into play.

In this project, we propose to build a tool that would solve two major problems. Firstly, relating the regulatory records and articles with business glossaries and terms of an organization. Secondly, creating a Compliance Manager dashboard that would give a summary of company’s current data protection, compliance situation and overall compliance performance. The outcome of our project would be a large reduction in manual effort, time and man power, thus saving significant investment and cost cuttings on compliance for organizations.

Chapter 1. Literature Search, State of the Art

1.1 Literature Search

There has been very little research that has been done on this topic. Out of the very few papers that have been published to reputable journals, we have found few informative articles, two of which are ‘Terminology matching of requirements specification documents and regulations for compliance checking’ and ‘Classifying Natural Language Sentences for Policy’. There are a few methods as per the authors which are used to analyze the documents and classify them. One was creating an intermediate representation and then performing linear regression. [1] The other one uses the technique of case frames to represent the semantics of the natural language. We have found the method which uses case frames to be innovative. [2]

The tool should be able to extract the text from regulatory documents which usually have generic terms applicable to all industries, understand it, process it and should be able to relate it to an organization under test, after training with different large data sets. For the development of our tool we are in consideration of latest machine learning techniques and algorithms. The final product will be able to accurately predict which terms are related to an industry and thus what all regulations an organization should follow and lets the admin/compliance manager assign the compliances to employee or a set of employees.

Steps in our tool

- Data cleaning
- Data Preprocessing including word to vector conversion using various python libraries
- Data Classification using different algorithms (a few in consideration)
- Training the tool with a large varied data set
- Processing actual data

1.2 State-of-the-Art Summary

A good amount of research has been done to the select good algorithms. For the tool to work efficiently, we had to select correct set of technologies, or tools or libraries.

As we already know, a single classifier never works on all given problems. For reducing the randomness in the data and to normalize the data, we need to pre-process it and extract the features. We then need to find out keywords, their synonyms from the government documents and correlate them with the terminology of our organization. So, we are currently considering classification algorithms like nearest neighbors, random forest from python machine learning libraries like scikit-learn, matplotlib etc for doing it. There after we create taxonomies of words by developing a model checker and case frames to classify the sentences and check if they are applicable to the organization.

A neat, simple and clean dashboard will be developed with analytics containing details of regulations and employee compliance. We are regulating access through OAuth2.0 validation to allow entry to only a specific set of individuals (Admins, compliance managers, legal departments etc.,). The completely processed/analyzed regulatory data with identified related regulations highlighted will be displayed to admins and managers. The dashboard will be developed with new technologies like NodeJS and ReactJS

A modern NoSQL database like MongoDB will be used to store the processed data. Also, it would help us in making the application scalable.

1.3 References

1. R Nakamura; Y. Negishi, S. Hayashi, M. Saeki "Terminology matching of requirements specification documents and regulations for compliance checking" *Requirements Engineering and Law (RELAW), 2015 IEEE Eighth International Workshop* on, Ottawa, Ontario, Canada, pp. 10-18. doi: 10.1109/RELAW.2015.7330206
2. J. Slankas and S. Williams, "Classifying Natural Language Sentences for Policy" 2012 IEEE International Symposium on Policies for Distributed Systems and Networks, Chapel Hill, NC, 2012, pp. 33-36. doi:10.1109/POLICY.2012.16

Chapter 2. Project Justification

Industry compliances and regulations set forth by the government can become too complex to interpret. They are difficult to read, difficult to interpret. They usually have words which are not in the usual vocabulary of humans. Therefore, the policies which are actually applicable to the organization might be very difficult for an individual. So, organizations hire Compliance Managers and pay them billions.

The main goal of our project is to build a tool that would parse complex regulatory documents using natural language processing. The tool that is being developed can be used by several organizations in managing their compliance to a good extent. Instead of organizations spending millions on hiring several compliance managers and spending a lot of time on reviewing regulatory documents to find out what all are applicable to the organization, they can input the regulatory documents to the tool. The tool then reads the documents, analyzes it, classifies the sentences and thus lets us know applicable scenarios and lets the compliance manager decide who to assign it to. That's done through an interactive dashboard which uses NodeJS and ReactJS. The main idea behind using NodeJs is to make asynchronous calls that makes it even faster

Hence, we believe that this project would save millions in costs that are spent for compliance managers and also makes the process 80% faster which is very useful for time sensitive regulations.

Chapter 3. Project Requirements

This kind of high-level projects which depend on Machine Learning have many requirement3.1 User Requirements

A user will be provided with a dashboard where one can access and manage their compliances. We are also considering providing a search bar where a user can enter the information about compliance details they want to know. We will provide a web application based on Machine Learning where the user will be provided with dashboard and a ML based tool. To produce a tool with efficiency, we need a huge dataset of compliance documents. The dataset will be split into two sets: training, testing set. We will be needing machine learning libraries which work with language interpretation such as NLP, NLTK etc., These ML libraries will be used in training our algorithm to interpret the new documents and decide if those documents fall under the compliances a company abide to follow. For building the algorithm, we are considering python as our primary programming language, as it supports ML and has many in built libraries. We may also use other languages, if needed. For the visual representation of our algorithm, we will be using HTML, CSS and JavaScript with the help of Flask framework. This is the front-end part of our project where a dashboard is provided to the user where one can manage their compliances. We're also considering implementing a search bar where a user can enter the information about a compliance they want to know. PageRank algorithms can come in handy to implement such features.

3.2 User Story

A user wants to know if his functional requirements are in compliance with the regulations which he needs to follow. He will have to spend a lot of time to understand the terms and may need to study hundreds of articles to find about it. With the help of our application, he can use the dashboard and find out in much less time whether his requirements are in compliance with the regulations or not. The ML algorithm at back-end will do all the necessary work and it will provide the information to the user. He will also be provided with results of highlighted text of what regulations does his requirements come under.

3.3 Use Cases

Machine Learning Regulatory Compliance Tool can provide good number of use cases to the companies:

3.3.1 Compliance Dashboard

The Compliance Dashboards are simple to use and easy to understand. They allow you to quickly see what functional requirements are in compliance with regulation and requirements that are not met.

3.3.2 Regulatory ML Tool

This can also be used for other companies before releasing their product, to make sure that the product follows all the state and federal regulations without spending huge amounts of money and man hours.

3.4 Flow of Events:

Here is a basic flow of events:

User opens the application.
User will be asked to register.
User will be redirected to homepage (dashboard).
User provides the functional requirements along with other compliance details
User will be provided with results along with other visual representation.
User can also search for other regulations using a search bar.
User can later update their requirement details and see the updated results.

3.5 Exit Conditions:

A: Success Guarantee:
A user will have enough insights about the compliance situation of his requirements.

B: Minimal Guarantee:
A user will be given details about what category of compliances his requirements fall under.

Chapter 4. Dependencies and Deliverables

4.1 Dependencies:

Following are some of the major Dependencies for our project:

- Project is dependent on large number of regulatory documents and articles. As we have to train our software to classify input documents, we need a large number of regulatory documents and articles. We are planning to employ supervised machine learning algorithms for training the software.
- To stay regulatory-ready is an unrelenting challenge, getting it wrong can result in unfavourable situations. As this software will help companies in compliance management, we can not afford give any wrong results to users. We will have to do rigorous testing before delivering the software in the market.
- The amount of documentation can be huge to ingest and relate with business glossaries and terms of an organization. The documents related to regulations and compliance are huge and complex. Performing machine learning on such a huge data is a daunting task.
- Some legal terms can be confusing and strict to deal with and may require an in-depth knowledge of law and norms.

4.2 Deliverables:

- A tool: relating the regulatory records and articles with business glossaries and terms of an organization. This will be a SaaS tool which will be accessible over the internet from anywhere to users.
- A dashboard: giving a summary of company's current data protection, compliance situation and overall compliance performance. This will be also available as a SaaS product. This will be comprehensive display of company's current compliance management.
- The outcome of our project would be a large reduction in manual effort, time and manpower. The whole purpose behind creating this tool is to minimize manual effort and cost involved in compliance management.

Chapter 5. Project Architecture

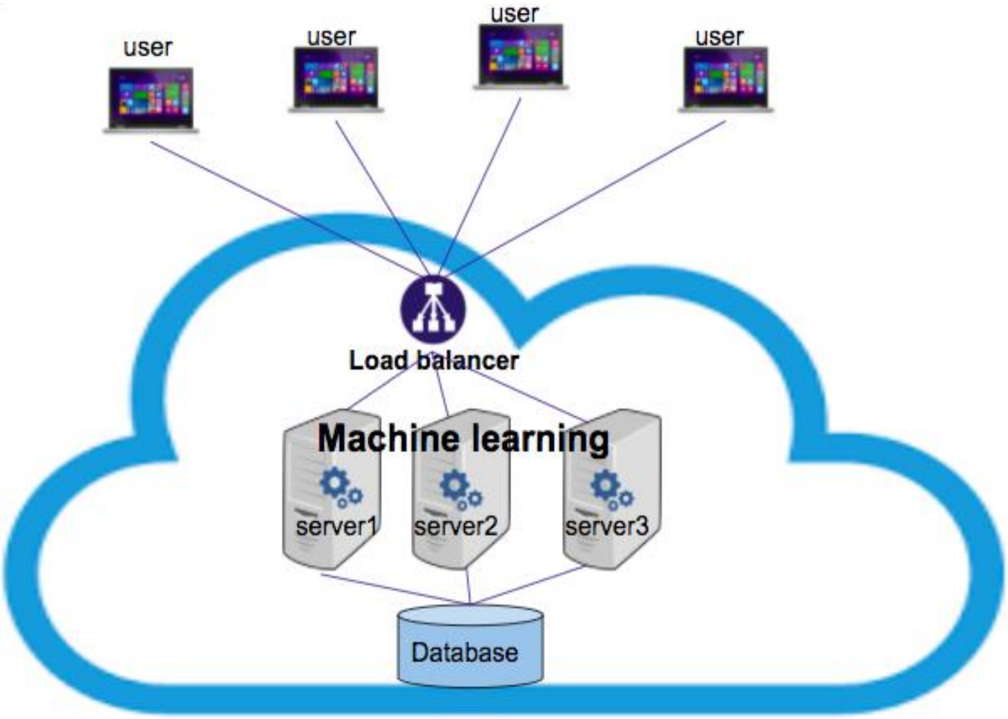


Fig 5.1: Architecture Diagram

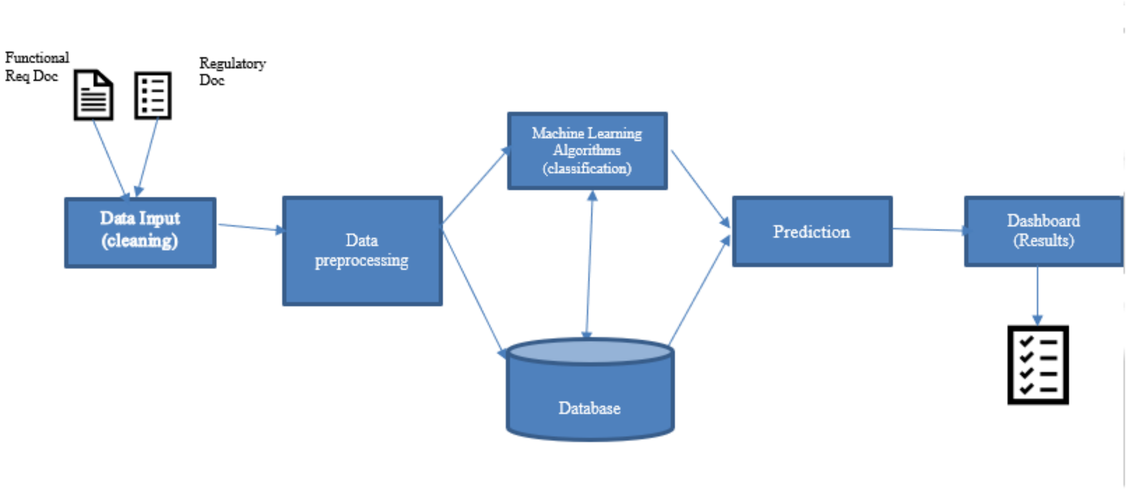


Fig 5.2 Architecture Diagram with Machine learning components and dashboard

5.1 Architecture

This application will be deployed as a SaaS product. Software as a service (SaaS) is a software distribution model in which a third-party provider hosts applications and makes them available to customers over the Internet. SaaS is one of three main categories of cloud computing, alongside infrastructure as a service (IaaS) and platform as a service (PaaS).

Software as a Service (SaaS) has following major advantages:

- SaaS removes the need for organizations to install and run applications on their own computers or in their own data centers. This eliminates the expense of hardware acquisition, provisioning and maintenance, as well as software licensing, installation and support.
- Flexible payments: Rather than purchasing software to install, or additional hardware to support it, customers subscribe to a SaaS offering. Generally, they pay for this service on a monthly basis using a pay-as-you-go model. Transitioning costs to a recurring operating expense allows many businesses to exercise better and more predictable budgeting. Users can also terminate SaaS offerings at any time to stop those recurring costs.
- Scalable usage: Cloud services like SaaS offer high scalability, which gives customers the option to access more, or fewer, services or features on-demand.
- Automatic updates: Rather than purchasing new software, customers can rely on a SaaS provider to automatically perform updates and patch management. This further reduces the burden on in-house IT staff.

- Accessibility and persistence: Since SaaS applications are delivered over the Internet, users can access them from any Internet-enabled device and location.

We are also planning to have load balancer before sending user requests to servers. In computing, load balancing improves the distribution of workloads across multiple computing resources, such as computers, a computer cluster, network links, central processing units, or disk drives. Load balancing aims to optimize resource use, maximize throughput, minimize response time, and avoid overload of any single resource. Using multiple components with load balancing instead of a single component may increase reliability and availability through redundancy. Load balancing usually involves dedicated software or hardware, such as a multilayer switch or a Domain Name System server process.

For the back-end, We are planning to use python libraries to run machine learning and data science algorithms on the server. We can then send the results to client’s browser and render them using front-end development technologies such as HTML, Javascript etc. We want to make the tool user friendly. We will consider aspects of human–computer interaction while building the tool.

Chapter 6. Project Design

Evaluated: Workbook Assignment 2

Chapter 7. QA, Performance, Deployment Plan

Evaluated: Workbook Assignment 2

Chapter 8. Implementation Plan and Progress

- Initiation Phase:
  - ❑ Decide topic
  - ❑ Discuss topic with professor

- Planning:
  - ❑ Determine scope of the project
  - ❑ Idetifying risks and mitigation plan
  - ❑ Define resource plan

- Analysis & Requirements:
  - ❑ Functional & Non-Functional Requirements
  - ❑ Identifying infrastructure requirements
  - ❑ Determining input and output

- Design:
  - ❑ Project Architecture
  - ❑ Create Use-Cases
  - ❑ UML Diagrams(Class Diagram, Sequence Diagram)
  - ❑ UI Mockup
  - ❑ Database Entity Diagram

- Environment Setup:
  - ❑ Acquire development tools
  - ❑ Acquire input data(regulatory documents, functional requirement data)

  - ❑ Build prototype
  - ❑ Decide and apply algorithm to a small dataset

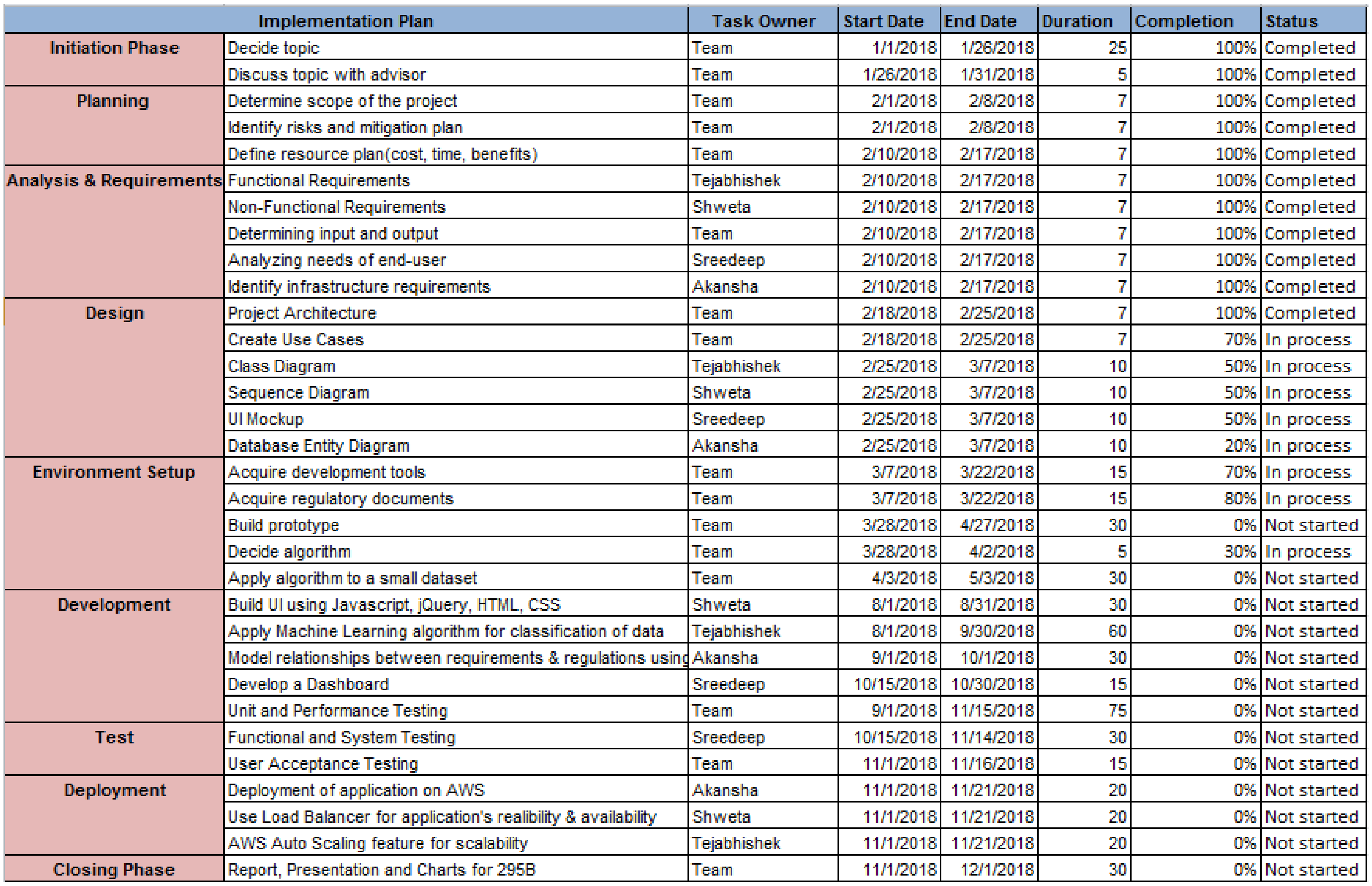
- Development
  - ❑ Build UI using Javascript, jQuery, HTML, CSS
  - ❑ Apply Machine Learning algorithm for classification of data
  - ❑ Apply techniques such as Model Checker and Case Frame for modeling relationships between requirements and regulations
  - ❑ Develop a Dashboard
  - ❑ Unit and Performance Testing

- Test
  - ❑ Functional and System Testing
  - ❑ User Acceptance Testing

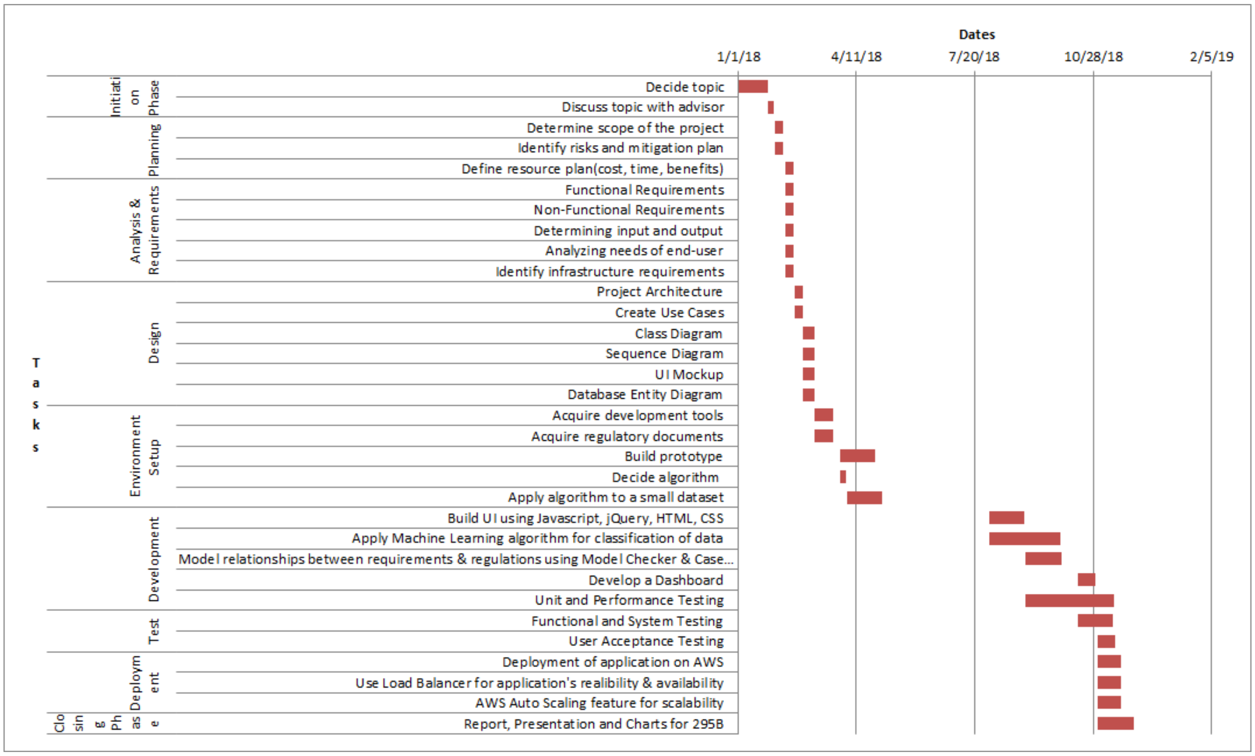
- Deployment
  - ❑ Deployment of application on Cloud(AWS)
  - ❑ Using Load Balancer in AWS for application's realibility & availability
  - ❑ AWS Auto Scaling feature for scalability

- Closing Phase
  - ❑ Report, Presentations, Charts

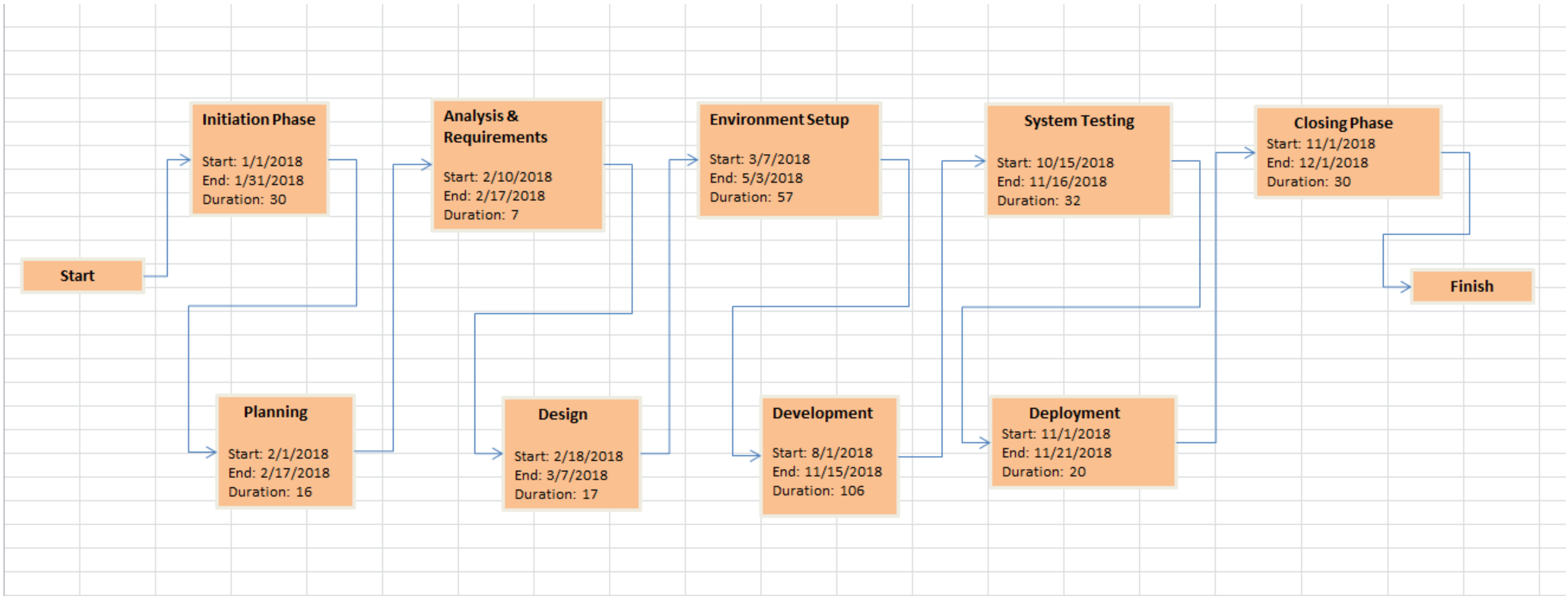
Chapter 9. Project Schedule



Implementation Plan		Task Owner	Start Date	End Date	Duration	Completion	Status
Initiation Phase	Decide topic	Team	1/1/2018	1/26/2018	25	100%	Completed
	Discuss topic with advisor	Team	1/26/2018	1/31/2018	5	100%	Completed
Planning	Determine scope of the project	Team	2/1/2018	2/8/2018	7	100%	Completed
	Identify risks and mitigation plan	Team	2/1/2018	2/8/2018	7	100%	Completed
	Define resource plan(cost, time, benefits)	Team	2/10/2018	2/17/2018	7	100%	Completed
Analysis & Requirements	Functional Requirements	Tejabhishek	2/10/2018	2/17/2018	7	100%	Completed
	Non-Functional Requirements	Shweta	2/10/2018	2/17/2018	7	100%	Completed
	Determining input and output	Team	2/10/2018	2/17/2018	7	100%	Completed
	Analyzing needs of end-user	Sreedeep	2/10/2018	2/17/2018	7	100%	Completed
	Identify infrastructure requirements	Akansha	2/10/2018	2/17/2018	7	100%	Completed
Design	Project Architecture	Team	2/18/2018	2/25/2018	7	100%	Completed
	Create Use Cases	Team	2/18/2018	2/25/2018	7	70%	In process
	Class Diagram	Tejabhishek	2/25/2018	3/7/2018	10	50%	In process
	Sequence Diagram	Shweta	2/25/2018	3/7/2018	10	50%	In process
	UI Mockup	Sreedeep	2/25/2018	3/7/2018	10	50%	In process
	Database Entity Diagram	Akansha	2/25/2018	3/7/2018	10	20%	In process
Environment Setup	Acquire development tools	Team	3/7/2018	3/22/2018	15	70%	In process
	Acquire regulatory documents	Team	3/7/2018	3/22/2018	15	80%	In process
	Build prototype	Team	3/28/2018	4/27/2018	30	0%	Not started
	Decide algorithm	Team	3/28/2018	4/2/2018	5	30%	In process
	Apply algorithm to a small dataset	Team	4/3/2018	5/3/2018	30	0%	Not started
Development	Build UI using Javascript, jQuery, HTML, CSS	Shweta	8/1/2018	8/31/2018	30	0%	Not started
	Apply Machine Learning algorithm for classification of data	Tejabhishek	8/1/2018	9/30/2018	60	0%	Not started
	Model relationships between requirements & regulations using	Akansha	9/1/2018	10/1/2018	30	0%	Not started
	Develop a Dashboard	Sreedeep	10/15/2018	10/30/2018	15	0%	Not started
	Unit and Performance Testing	Team	9/1/2018	11/15/2018	75	0%	Not started
Test	Functional and System Testing	Sreedeep	10/15/2018	11/14/2018	30	0%	Not started
	User Acceptance Testing	Team	11/1/2018	11/16/2018	15	0%	Not started
Deployment	Deployment of application on AWS	Akansha	11/1/2018	11/21/2018	20	0%	Not started
	Use Load Balancer for application's realibility & availability	Shweta	11/1/2018	11/21/2018	20	0%	Not started
	AWS Auto Scaling feature for scalability	Tejabhishek	11/1/2018	11/21/2018	20	0%	Not started
Closing Phase	Report, Presentation and Charts for 295B	Team	11/1/2018	12/1/2018	30	0%	Not started


Dates
1/1/18
4/11/18
7/20/18
10/28/18
2/5/19
Tasks
Initiation Phase
Decide topic
Discuss topic with advisor
Planning
Determine scope of the project
Identify risks and mitigation plan
Define resource plan(cost, time, benefits)
Analysis & Requirements
Functional Requirements
Non-Functional Requirements
Determining input and output
Analyzing needs of end-user
Identify infrastructure requirements
Design
Project Architecture
Create Use Cases
Class Diagram
Sequence Diagram
UI Mockup
Database Entity Diagram
Environment Setup
Acquire development tools
Acquire regulatory documents
Build prototype
Decide algorithm
Apply algorithm to a small dataset
Development
Build UI using Javascript, jQuery, HTML, CSS
Apply Machine Learning algorithm for classification of data
Model relationships between requirements & regulations using Model Checker & Case...
Develop a Dashboard
Unit and Performance Testing
Test
Functional and System Testing
User Acceptance Testing
Deployment
Deployment of application on AWS
Use Load Balancer for application's realibility & availability
AWS Auto Scaling feature for scalability
Closing Phase
Report, Presentation and Charts for 295B

Pert Chart:


Start
Initiation Phase
Start: 1/1/2018
End: 1/31/2018
Duration: 30
Planning
Start: 2/1/2018
End: 2/17/2018
Duration: 16
Analysis & Requirements
Start: 2/10/2018
End: 2/17/2018
Duration: 7
Design
Start: 2/18/2018
End: 3/7/2018
Duration: 17
Environment Setup
Start: 3/7/2018
End: 5/3/2018
Duration: 57
Development
Start: 8/1/2018
End: 11/15/2018
Duration: 106
System Testing
Start: 10/15/2018
End: 11/16/2018
Duration: 32
Deployment
Start: 11/1/2018
End: 11/21/2018
Duration: 20
Closing Phase
Start: 11/1/2018
End: 12/1/2018
Duration: 30
Finish