4/30/2019

Text Annotation Tool (CSIRO)

System Design Document

Presented By:

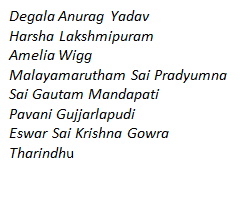
****

Table of contents

0 PREFACE

0.1 Purpose of this document 1

0.2 Use of this document 1

0.3 Overview 2

0.4 Basis of this Document 2

1 Introduction

1.1 Purpose 3

1.2 Scope 3

1.3 References 4

1.4 Roles and Responsibilities…………………………………………………………………………………..…7

1.5 Sprint Management…………………………………………………………………………………………..….7

1.6 Overview…………………………………………………………………………………………………………….16

2 System OvervieW

2.1 System Architecture 17

2.2 Technical Architecture……………………………………………………………………………………..….18

2.3 Infrastructure Services 19

3 UML Diagrams

3.1 Sequence diagram 22

3.2 Activity Diagram 23

3.3 Use Case Diagram 24

4 System DesiGN

4.1 Proposed Technologies 25

4.2 Functional Characteristics……………………………………………………………………………..…..26

0 Preface

0.1 Purpose of this document

1. This document is a System Design document for Text Annotation Tool. It provides guidance and template material which is intended to assist the relevant management or technical staff, whether client or supplier, in producing a project specific‑ Design Document. It is also useful background reading for anyone involved in developing the Text Annotation Tool for CSIRO. (En.wikipedia.org, 2019)

**0.2 USE OF THIS DOCUMENT**

#1 A software design description (a.k.a. software design document is just a design document and Software design specification) is a written description of a software product, that a software designer writes to give a Software Development team overall guidance to the architecture of the software project. An SDD usually accompanies an architecture diagram with pointers to detailed feature specifications of smaller pieces of the design. Practically, the description is required to coordinate a large team under a single vision, needs to be a stable reference, and outline all parts of the software and how they will work.

The SDD usually contains the following information:

1. The Data Design describes structures that reside within the software. Attributes and relationships between Data objects dictate the choice of Data structures
2. The architecture design uses information flowing characteristics and maps them into the program structure. The transformation mapping method is applied to exhibit distinct boundaries between incoming and outgoing data. The data flow diagrams allocate control input, processing and output along three separate modules.
3. The interface design describes internal and external program interfaces, as well as the design of human interface. Internal and external interface designs are based on the information obtained from the analysis model.
4. The procedural design describes structured programming concepts using graphical, tabular and textual notations.

These design mediums enable the designer to represent procedural detail, that facilitates translation to code. This blueprint for implementation forms the basis for all subsequent software engineering work.

**0.3 OVERVIEW**

1. This preface is for information only.
2. This preface will therefore not be retained in the project Design Document.
3. The remaining sections (numbered 1, 2, 3,…) constitute a template that should be used to construct the project-specific document.

**0.4 BASIS OF THIS DOCUMENT**

1. This following introductory sections set out an approach to designing systems that may be developed under CSIRO. It attempts to set standards and create a consistent approach to the design and development of systems across the Annotator Tool. It will enable the Programme to benefit from ‘economies of scale’ and a consistency in the approach to building and deploying systems. Important issues that need to be considered include the architecture of systems, links to legacy systems, contemporary approaches to design (Node.js), aims for code re-use and the need to develop systems that will work on an operational basis over many years and the associated desire to make such systems easily supportable and affordable.
2. The main point is to develop a web service that annotates text with definitions, taken from controlled online vocabulary services. Make it easier, simpler and more accurate to get a definition, which has been provided by a collaboration of online vocabulary services. A quick understanding which is enabled by hyperlinking technical terms to their corresponding definitions.
3. The concept of a **Reference Architecture** is also introduced as part of the process of creating an interoperable environment which facilitates the exchange of information between administrations through setting out several standard building blocks around which solutions can be assembled. These building blocks or ‘components reflect the emerging technologies that should form the technical basis of the Text Annotator Tool. The Biportal and lightag are the similar tools are taken as the reference architecture that highlight code re-use, scalability and the creation of interoperable architectures around legacy environments. (Project Management Docs, 2019)
4. **INTRODUCTION**

The target market of our tool includes anyone who is undertaking online research using the internet. There are no gender specifications of our target market, nor is there a specific age group due to the tool being aimed at anyone undertaking research. However, the target market must be old enough to be undertaking research. The text annotation tool is particularly helpful for reading and understanding research papers and complex term, therefor the market is targeted at those likely to examining research papers for as part of their job, education or personal research. (Its.dot.gov, 2019)

* 1. **PURPOSE**

Commonly in research papers there are a set of typical words which can derive multiple meanings, easily causing confusion or misunderstanding. This highlights the need for a tool that provides users with the assistance of hyperlinking text which can provide easy definitions or the term. Additionally, the user can be redirected to the webpage which can assist in finding a meaning or search for terms in a field.   (Its.dot.gov, 2019)

* 1. **SCOPE**

There a small number of existing tools which assist in annotating words and providing links to users. There is high demand for this type of tools, due to the complexity and array of meaning in research papers. There are a lot of scientific terms which user may require assistance understanding or comprehending. A tool could have a range of available ontologies and therefore could be useful in a range of different fields.

Quite often technical documents have complex terminology and specific word, or phrases can denote precise definitions of some concept. In most of the cases a single term has more than one meaning. By hyperlinking the text with precise and accurate meaning, we can develop clarity and specificity of terms and meanings. This project will develop a tool and web micro service to enable semi-automated annotation of text with hyperlinks to the complex definitions of terminology used in research papers.  (People.eecs.berkeley.edu, 2019)

* 1. **REFERNECES**

Bioportal

***Description of the tool***

Bioportal is a bio ontology website which assists in annotating words by providing links for various different websites, in which the user can obtain meanings and definitions of words. The website includes an ontologies library where the user can explore different libraries dependant on their specific area, examples include National Cancer Institute Thesaurus (NCIT) and other medical libraries.

The below image depicts the process where text can be entered and searched in the NCIT ontology search library. The user needs make certain selections before clicking the ‘get annotations’ tab, to derive results.

**Benefits of the Tool**

* Free web page annotation tool
* Advance search options available based on databases
* Customisation options according to user queries
* Easy to use and handle
* Get recommendations for the input query we provided

**Limitations of the Tool**

* We can search only ontology & Biomedical bases keywords
* Limited to Biomedical uses not compatible all fields

**User Scenarios**

* The user needs to open the bio-portal browser and just search the annotated word.
* The user doesn’t need to login or signup for annotated words.

Lightag

**Description of the tool**

Lightag is a useful text annotation website/tool. It is predominantly used as text annotation platform for data scientists creating AI training data. LightTag aims to solve one of the main bottlenecks of the deep learning-based AI development.

**Benefits of the Tool**

* Can annotate in any language
* Measures inter annotator agreement to understand the quality of annotation
* Continuously boot straps the projects and provides suggestions
* Can add new meaning from our own library and upload it to LightTag with Rest API

**Limitations of the Tool**

* Provides hourly rate for engineers to build own labelling tools
* Comes with only in on-premise version

**User Scenarios**

* The user needs to sign in first to use the tool.
* The user can see various options like upload JSON, write or paste text, use demo data base and paste conllu data.
* The user can choose any option regarding the project and annotate the words.

* 1. **ROLES AND RESPONSIBILITIES**

The entire team of Eight members are involved in various responsibilities. Here are roles and responsibilities of the team:

**SPRINT MANAGEMENT**

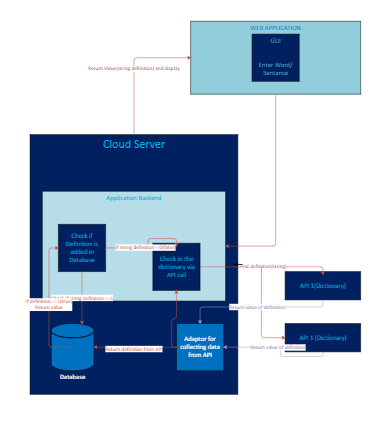
|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Activity Done** | **Working** | **Hours (Sprint 1)** |
| **Amelia Wigg** | * Established team roles * Organising and assisting group * Upskilling and created Trello board * Upskilling and created slack group * Upskilling and joined GitHub * Completed sprint increment report * Attended and contributed to team meetings * Communicating progress to supervisor via email | * Team leader upskilling * GitHub upskilling * JavaScript upskilling * Began market study for market analysis | 18 hours |
| **Degala Anurag Yadav** | * Understanding about Slack workplace – what it is meant and used for. Created an account and joined the required groups. * Learned about GitHub and Trello board from Lynda. Developed the skill in it and created a repository for our project and a basic initial Trello board with some random example. * Created a Board in Trello for our project with initial backlogs and the work must be done. * Met Our supervisor and understood about the project what is required and what must be delivered. | * Started upskilling the Design part which is required for our project. * Simultaneously I was doing the upskilling of the programming language based our requirement. * Started to do some market study on similar tools. | 17 Hours |
| **Eswar Sai Krishna Gowra** | * Understanding the purpose and usage of GitHub and Trello. * Attended the supervisor meeting to know about the project in detail. * To take suggestions from the supervisor that we are according to the project or not. | * Started upskilling on Lynda for GitHub and Trello. * Started research on the similar applications to understand the what we must do. * Working on infrastructure Document. | 16 Hours |
| **Harsha Lakshmipuram** | * Contacting The team & arranging initial meeting * Upskilling in Trello & created a Trello board with Name CSIRO1 * Upskilling in Git Hub & created a group so that we team can share code or related files. * Assigned individual roles in the group and make sure everyone is working towards their individual goals * Work towards sprint increment that has been submitted by team leaders. * Done team leader upskilling in Leadership skills in Lynda. * First supervisor meeting & understanding the project scope, Deliverables that are required for the product owner. * Contacting& analysing C2 regarding the previous work that has been done in this project. | * Started working towards Market Research analysis. * Started working on Java script programming fundamentals. * Upskill in both NodeJS & Node Express as they were important for our project. * Creating a slack channel & adding C1 & C2 team. * Started counting individual contributions | 18 Hours |
| **Malayamarutham Sai Pradyumna** | * Got an idea on how to use trello and git hub * Created a slack channel how for group communication about the project * Every week after the completion of supervisor meeting prepared a document of Minutes of Meeting * Researched on project scope and market need and about the diigo website which is a similar tool we are developing | * Started upskilling Lynda courses for GitHub & trello * Preparing a documentation of Minutes of Meeting every week after supervisor meeting * Research about the market need of our project * Worked on project scope document and researched about the similar tool of our project * Java script upskilling | 17 hours |
| **Pavani Gujjarlapudi** | * Got the idea on how to use GitHub and Trello * Slack channel is created to communicate with the group * Worked on project scope document * Met the supervisor for clear understanding of the roles and project | * Started upskilling on Lynda courses for GitHub and Trello boards * Researching about the market tools required for annotation as by the supervisor guidance * Worked on the project scope document | 17 Hours |
| **Sai Gautam Mandapati** | * Developing the skill of using GitHub and creation of Trello boards. * Created a slack channel for communication for discussing the project details. * Meeting the supervisor and previous squad and understanding the project. * Understanding the project and requirements of the project. * Searched for the templates for scope document. | * Started development of project scope document. * Started upskilling programming languages based on the requirements of the project. * Started doing the research on market for similar tools and as guided by the supervisor. * Developing the detailed report on the research. | 16 Hours |
| **Tharindu Chamara Nauththuduwa Liyanage** | * Studied about GitHub and slack * Researched about tag tog and added summarised to the document. * Compared Tag Tog and other online tools. * Researched about Monday.com and added summarised to main document. * Research about Asana and added summarised to the document. | * Found some valuable information about market research and analysing all the data. * Researching about similar tools which are related to our project. * Analysing about similar tools. * Updating work process. * JavaScript upskilling | 17 hours |

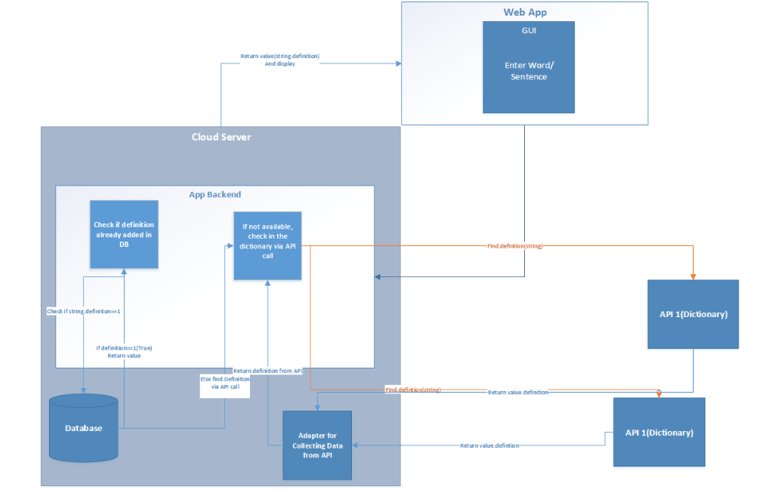
**SPRINT 2**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Activity Done** | **Work in Progress** | **Hours (Sprint 2)** |
| **Amelia Wigg** | * Communicated and divided up group tasks * Contributed to project scope document * Complete group increment report * Undertook market research * Created a template for market research document * Identified similar competitor tools to analyse * Upskilling programming languages * Edited the market analysis template to facilitate required changes * Contributed feasibility analysis section * Edited the market analysis document and ensured it was professional * Started a webpage rough draft of what users will see | * Continue to upskill programming languages such as JavaScript and Node.JS * Continue to edit the market analysis until it meets requirements * Continue to develop front end webpage in html | 25 hours |
| **Degala Anurag Yadav** | * Understood our project scope and developed it by referring the document from our seniors. * Completed the research analysis and feasibility study which describes the outline of our developing tool with comparisons from similar tools. * My Contribution in market research is based on the similar tool named **“Light Tag”.**I have mentioned pros and cons of that tool and how it is different from our developing tool. * Completed the foundations in programming language I.e. JavaScript which is basic requirement to understand the further code used in our project. | * Started upskilling the further knowledge in programming which is a step ahead for JavaScript I.e. Node JavaScript. * Started analysing the code which is developed by our previous squad (written in Node express language). * Creating the design document. * Developing the feature in our tool that identifies the errors that user make while executing. | 17 Hours |
| **Eswar Sai Krishna Gowra** | * Completed infrastructure document. * Completed System Architecture diagram. * Completed courses on GitHub and Trello in Lynda. | * Drawing the sequence diagrams. * Drawing the activity, Use case diagrams. * Working on the Design document. | 17 Hours |
| **Harsha Lakshmipuram** | * Sent a project scope document according to client requirements in a professional way. * Completed the Market research document * Helped in analysing the pros & cons of some competitors. * Helped in analysing similar market research tools that are close to client requirements * Updating & assigning tasks in Trello. Setting up deadlines to individual members. * Helped in Group Increment report. * Done upskilling in JavaScript Fundamentals (5hrs hours) * Have completed web design fundamentals in Lynda for basic web design understanding * Started a basic webpage with annotator & similar functionalities as a mock up. * Started analysing the code that has been sent by project delivery team for better understanding. * Sending Individual contribution report to supervisor. | * Now that we have a market research document we are working towards design documentation. * Help Eshwar regarding the final documentation * Final upskilling in NodeJS. * Creating a mock up design diagrams for the project * Working on prototype webpage annotator & creating a local server * Start working on individual components in the system architecture | 23 Hours |
| **Malayamarutham Sai Pradyumna** | * Completed Minutes of Meeting document every week * Completed courses on GitHub and trello in Lynda * Completed market analysis and feasibility study * Completed project scope document and contributed to market need and “Diigo” which is a similar website of our project | * Continue to upskill programming languages i.e., JavaScript, node.js * Working on design document of the project. | 16 hours |
| **Pavani Gujjarlapudi** | * Completed the project scope document * Completed the market analysis and feasibility study * Completed the programming foundations on Lynda * Contributed to the market research document by giving description to the, target market, “ZipBoard” tool and adding the user scenarios for all the tools. | * Developing the skills on java script and Nodejs to understand the code of the project * Working on the design document for the project | 16 hours |
| **Sai Gautam Mandapati** | * Developed a project scope document based on the requirements. * Developed a detailed report on the market study and feasibility * Completed the essential training for Java script. * Contributed to the market study and feasibility study document by giving a description for “BIOPORTAL”. | * Developing further skill on new programming languages like Nodejs and Nod express to understand the code of the previous squad. * Developing the tool for spell checker (Displaying the message when entered wrong spelling of the word. | 16 hours. |
| **Tharindu Chamara Nauththuduwa Liyanage** | * Read the project scope document and added some points. * I have mentioned about pros and cons of Monday.com * Edited the final stage of similar tools. * Added main categories to similar tools. * Completed online tutorials about JavaScript. | * Researching about project scope. * Continue to upskill programming languages such as Java and python. * Working on the design document for the project * Researching about market research possibilities. | 17 hours |

2 System OvervieW

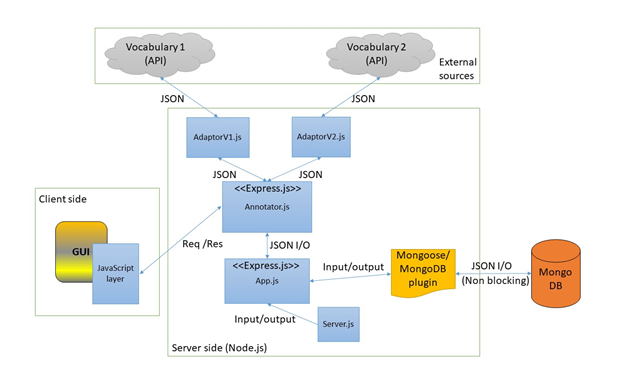
**2.1 SYSTEM ARCHITECTURE**

****



The above diagram describes about the brief outline of the system. This is all about how the proposed system works. Above diagram shows the architecture of proposed web application. It consists of three main areas which are Client side, Server side and External sources. Client side is the area which consists of GUI (Graphical User Interface) and corresponding JavaScript layer to communicate with server side. As per the meaning of GUI, it will be the interface which can be seen from the client browser and it will be stacked with text fields, buttons etc.

**2.2 TECHINICAL ARCHITECTURE**



Above diagram shows the architecture of proposed web application. It consists of three main areas which are Client side, Server side and External sources. Client side is the area which consists of GUI (Graphical User Interface) and corresponding JavaScript layer to communicate with server side. As per the meaning of GUI, it will be the interface which can be seen from the client browser and it will be stacked with text fields, buttons etc. The corresponding JavaScript layer is here for Dynamic content loading, otherwise the client page will be static and always must reload itself. This JS layer will communicate with server side’s Annotator.js (JavaScript file) which will be the API (Application Programming Interface) of this web application developed with Node.js’s most popular web application platform, Express.js. This file will be consisting of all the functionalities in this web app.

When client side ask a definition for a given text as a HTTP Request, Annotator.js will automatically check for the previously cached data in the connected (NoSQL) MongoDB database, if that request is already been responded. If it exists, the answer will be pulled out from the database. If not, it will connect with external vocabularies by using corresponding server information (Ports etc.) from the Server.js file and application requirements from App.js file, in order to find the answer. The response data from these vocabularies will be in the structure of JSON (JavaScript Object Notation). The API is built to receive and respond to requests using JSON, we have used an external library “bodyparser” to handle json data.

In order to make connections with external sources, Adaptor V1.js, Adaptor V2.js files will be used. The advantage of this class structure is, if the client wants more external vocabularies to connect with this application, he can easily do it by just designing an Adaptor.js class without changing the Annotator.js class or any file in the system.

Upon the receiving of required result from external sources, Annotator.js will save it (Caching) in MongoDB for future references through the Mongoose JS/MongoDB plugin. Moreover, since MongoDB is based on JSON objects, this process will be much easier to implement. After that Annotator.js will send the HTTP response with result data to the client-side JS layer and it will be shown in the GUI dynamically.

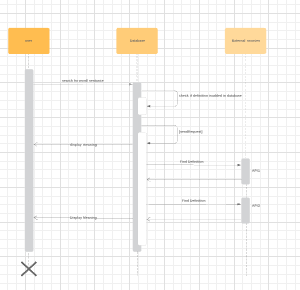
**2.3 INFRASTRUCTURE SERVICES**

**Functional Characteristics of Annotator API: [**Text Wrapping Break]Annotator.js file will be act as the API in this application which holds all the functions which is being used to accomplish the task of connecting external resources and find the explanations to the given text. Following are proposed functions of Annotator API.

1. **Check Ontology Connection (URL, API KEY);**
2. This function will be used to check the external vocabulary’s server status so that the GUI can show which Vocabularies are available to continue with the searching process.
3. This function accepts two arguments which are URL of the ontology (Vocabulary) and API KEY for the authentication purpose if it needs any.
4. Output of this function will be a Boolean which will be true if the connection successful or false if not.
5. **getAllOntologyNames()**
6. This function will be used to find all the ontology names in order to show them in the GUI.
7. Output of this function will be a name collection of saved ontologies.
8. **getTextDefinitions (URL, API KEY, Text string, Ontologies, Match longest, Partial word matching, include mapping, Exclude Numbers, Exclude Synonyms)**
9. This function will be used to find the definitions of given texts from specified ontologies.
10. The arguments for this function will be,
11. URL of the ontology
12. API KEY for authentication purposes
13. Text string of client’s search
14. Collection of client selected ontologies
15. Boolean value of match longest option
16. Boolean value of partial words match option
17. Boolean value of include mapping option
18. Boolean value of exclude numbers option
19. Boolean value of exclude synonyms option
20. Output of this function will be a JSON object which includes all the definitions for the given text string from specified vocabularies.
21. **insertToDataBase(Collection name, Data)**
22. This function will be used to store newly found results from a vocabulary search in the attached database.
23. This function accepts two arguments.
24. A name for the data collection to identifying purposes
25. Data that need to be stored. (JSON objects)
26. Output of this functions would be a success alert or an error.
27. **updateFromDataBase(Collection name, current Values,  new Values)**
28. This function will be used to update an existing data on the database.
29. This function accepts three arguments.
30. A name for the data collection to identifying purposes
31. Existing data that need to be updated.
32. New data that need to be saved instead.
33. Output of this functions would be a success alert or an error.
34. **deleteFromDataBase(Collection name, Query Object)**
35. This function will be used to delete an existing record/ object is the database.
36. This function accepts two arguments.
37. A name for the data collection to identifying purposes
38. Query object which defines the record to delete from database.
39. Output of this functions would be a success alert or an error.
40. **searchFromDataBase(Collection name, Query Object)**
41. This function will be used to find previously cached data from the database.
42. This function accepts two arguments.
43. A name for the data collection to identifying purposes
44. Query object which defines the record to search from database.
45. Output of this functions would be JSON object which consists of corresponding results.
46. **getOntologyByID(Collection name, Query object)**
47. This function will be used to find a stored ontology by its ID from the database.
48. This function accepts two arguments.
49. A name for the data collection to identifying purposes.
50. Query object which defines the record to search from database.
51. Output of this functions would be JSON object which consists of corresponding results.
52. **disableExistingOntology(Collection name,  Query object, currentValues,  newValues)**
53. This function will be used to disable an existing ontology from the database.
54. This function accepts four arguments.
55. A name for the data collection to identifying purposes
56. Query object which defines the record to search from database.
57. Existing data that need to be updated.
58. New data that need to be saved instead.
59. Output of this functions would be a success alert or an error.
60. **enableExistingOntology(Collection name,  Query object, currentValues,  newValues)**
61. This function will be used to enable an existing ontology from the database.
62. This function accepts four arguments.
63. A name for the data collection to identifying purposes
64. Query object which defines the record to search from database.
65. Existing data that need to be updated.
66. New data that need to be saved instead.
67. Output of this functions would be a success alert or an error.

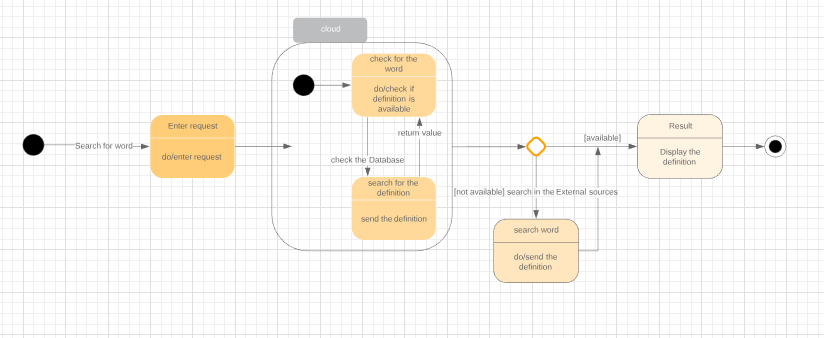
3 UML Diagrams

**3.1 Sequence Diagram**



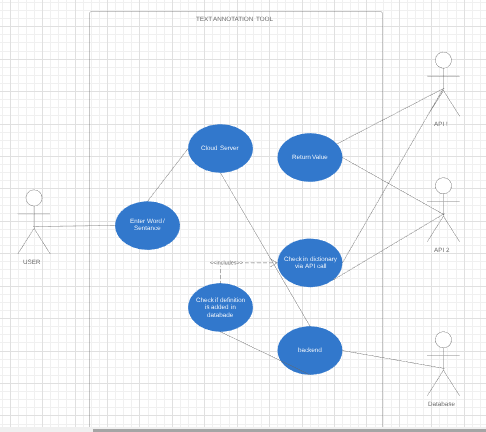
This diagram contains three major sections. The user will search for the required word or sentence. Then it directs to the database where the search function is done for the user typed word. It sends the request to search for the word. If the definition is found in the database itself then it sends the meaning which is displayed to the user. If the word is not found in the database, then it searches in the external sources. In the external source's sections, there are two API’s. It first searches in the API1. If the word definition is found it displays to the user. If there also the word is not found it directs to the API2. There it searches for the word definition and if it's found it redirects to the user page and displays the meaning. If there also the definition of the word is not found it shows no such word definition exists.

**3.2 Activity Diagram**



The activity diagram for the text annotation tool is provided. This diagram explains the process to the user. If the user wants to search for the annotation of a word, then the user enters a request. The request is directed to the cloud and checks whether the word is there. It also checks the definition of the meaning. It checks in the data base as well for the definition. If it is found, then it returns the value. If it is not available in cloud, then it starts searching in the external sources. If the definition is found it sends back the definition to result which displays the definition.

**3.3 Usecase Diagram**



Here is the Use case diagram for our Text Annotation tool. We can see here the actual process that takes part inside the tool and input given by the user and output observed by the user. Let’s get into the picture stated above:

* Firstly, user tries to search by typing the word / phrase / sentence in our tool.
* Then the tool begins to start its work and it analyses the provided phrase/word in its cloud server.
* Cloud server searches from the backend database and checks if the definition is available in the database.
* Now API comes into picture:

- API (Application Program Interface) after checking the meaning of the word or phrase in backend database the cloud server calls the API to check in their dictionary if we did not find the meaning in the local database.

* From API’s after finding the exact and preferred meaning the server provides the user with the proper output.

4 System DesiGN

4.1 Proposed Technologies

This text annotating web application will be developed by using following technologies.

1. HTML/ CSS
2. Bootstrap
3. HTTP/HTTPS
4. JavaScript/ Node.JS (JavaScript runtime)
5. Express.JS (Web application platform)
6. MongoDB (NoSQL database repository)

* Node.JS is ideal for this type of projects because there will be no heavy CPU calculations. Since Node.JS has a single thread to support all the client’s requests it will be a faster mechanism.
* Express.JS is the most popular web application platform of Node.JS and it is easier to build a faster API by using this technology.
* MongoDB is ideal for the database repository, since it is based on JSON objects and it’s easier to connect with Node.JS and do CRUD (Create, Read, Update and Delete) operation.

**4.2 Functional Characteristics**

**References:**

1. Bioportal.bioontology.org. (2019). Welcome to the NCBO BioPortal | NCBO BioPortal. [online] Available at: <https://bioportal.bioontology.org/> [Accessed 5 May 2019]
2. Medium. (2019). Announcing LightTag — The easy way to Annotate text. [online] Available at: <https://medium.com/@TalPerry/announcing-lighttag-the-easy-way-to-annotate-text-afb7493a49b8> [Accessed 6 May 2019].
3. En.wikipedia.org. (2019). Software design description. [online] Available at: https://en.wikipedia.org/wiki/Software\_design\_description [Accessed 08 May 2019].
4. Its.dot.gov. (2019). [online] Available at: https://www.its.dot.gov/research\_archives/msaa/pdf/MSAA\_SystemDesignFINAL.pdf [Accessed 09 May 2019].
5. Its.dot.gov. (2019). [online] Available at: https://www.its.dot.gov/research\_archives/msaa/pdf/MSAA\_SystemDesignFINAL.pdf [Accessed 10 May 2019].
6. People.eecs.berkeley.edu. (2019). How to Write a Design Document. [online] Available at: https://people.eecs.berkeley.edu/~kubitron/courses/cs162-F06/design.html [Accessed 10 May 2019].
7. Project Management Docs. (2019). *System Design Document Template - Project Management Docs*. [online] Available at: https://www.projectmanagementdocs.com/template/project-documents/system-design-document/#axzz5oW86xixj [Accessed 21 May 2019].