Intelligent Auto Scaler for Micro-Services

Design Document

(Web api usage pattern analysing and prediction module)

Project ID: 17-021

Author's Name: Firnas A.M
Author's ID: IT14064432

…………………………………….
Dr. Dharshana Kasthurirathne (Supervisor)

Bachelor of Science in Information Technology special in Software Engineering
Sri Lanka Institute of Information Technology

May, 2017

Declaration

I declare that this is my own work and this Project SRS does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of our knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Signature: ___________________________

Aleem Mohamed Firnas

Table of contents

List of Figures

List of Tables

Design Document (DD)

1 Introduction

1.1 Purpose

The purpose of the document is providing the information about the requirements and the understanding on the proposed design for the component of **Web api usage pattern analyzing and prediction module** of the research project 'Intelligent Auto Scaler for Micro-Services'. This document will be used to getting the clear idea on the research component by make use of functional, non-functional, proposed designs diagrams and technological concerns mentioned in this document.

1.2 Scope

This document mainly covers the requirements of the component "web api usage pattern analyzing and prediction module" of the research project titled "Intelligent Auto Scaler for Micro-Services". The entire project is creating an application which could be able to automatically scaled up or down the Docker containers which uses Micro-Services architectural patterns. The scaled up or down decision is not just making by considering only the request load or only the resource allocation but it considers both factors.

The Web API pattern analyzing and prediction module is for performing the analytics on the HTTP requests to the application and then the parameter cleaning prior to the machine learning model build. This module is predicting the future load to the application for a short period of time. The importance factor for any kind of the application is the performance so the analyzing the load of application is very much useful for any monolith or micro-service architectural patterns applications. [1]

1.3 Definitions, Acronyms, and Abbreviations

HTTP	Hyper Text Transfer Protocol
Model	Machine Learning mathematical model
API	Application Programming Interfaces
CLFF	Common Log File Format
httpd	Hypertext transfer protocol daemon
ml	Machine Learning
URL	Uniform Resource Locator
RMSE	Root Mean Squad Error
MSE	Mean Squad Error
RNN	Recurrent Neural Network

Table 1.0 Definitions, Acronyms and Abbreviations list

1.4 References

References are available at the end of the document.

1.5 Overview

The application is capable of automatically scale up and down the instances by make use of the modules will be developed in future. There are four 4 modules in the application such are (1) web api usage pattern analyzing and prediction module which is analyzing and the prediction of the request load patterns and will be detailed in the following sections the second is (2) absolute resource utilization analyzing and micro service tagging module which is for analyzing the resource consumption of each individual microservices by make use of the profiling system and the tagging system would be developed in order to tracking the sequences of execution of the microservices. The other module is (3) micro service instance managing and decision making module which is a decision-making module weather to increase the instances or decrease the instances by getting the results of the above-mentioned modules as the input to this. The prediction of the business scenarios and the sequences of the business scenarios executions and the visualization of the resource consumption will be done by the (4) business scenario prediction and analytics visualization module.

The component (1) mentioned in above will be covered in detail in this document. The web API gateway is the entry point to any hosted web application in server. The API gateway is where all the logs are recorded that means it has the records of each and every http requests to the application, this is called the **loggers**. These loggers are semi structured or unstructured data, in order to make it useful it needs to be analyzed. [3] There are several log analyzers or web analytics tools available in Internet. So, these analytics servers will produce the useful information. The identification of useful parameters from analytical server is very much important since analytical server produce so many parameters. At first, the analytics server should be plugged in to the API gateway so that the data will be collected and then preprocessed and then machine learning model will be build which could be able to predict usage patterns, most importantly the number of requests for typical time period and the response time prediction for upcoming HTTP request calls.

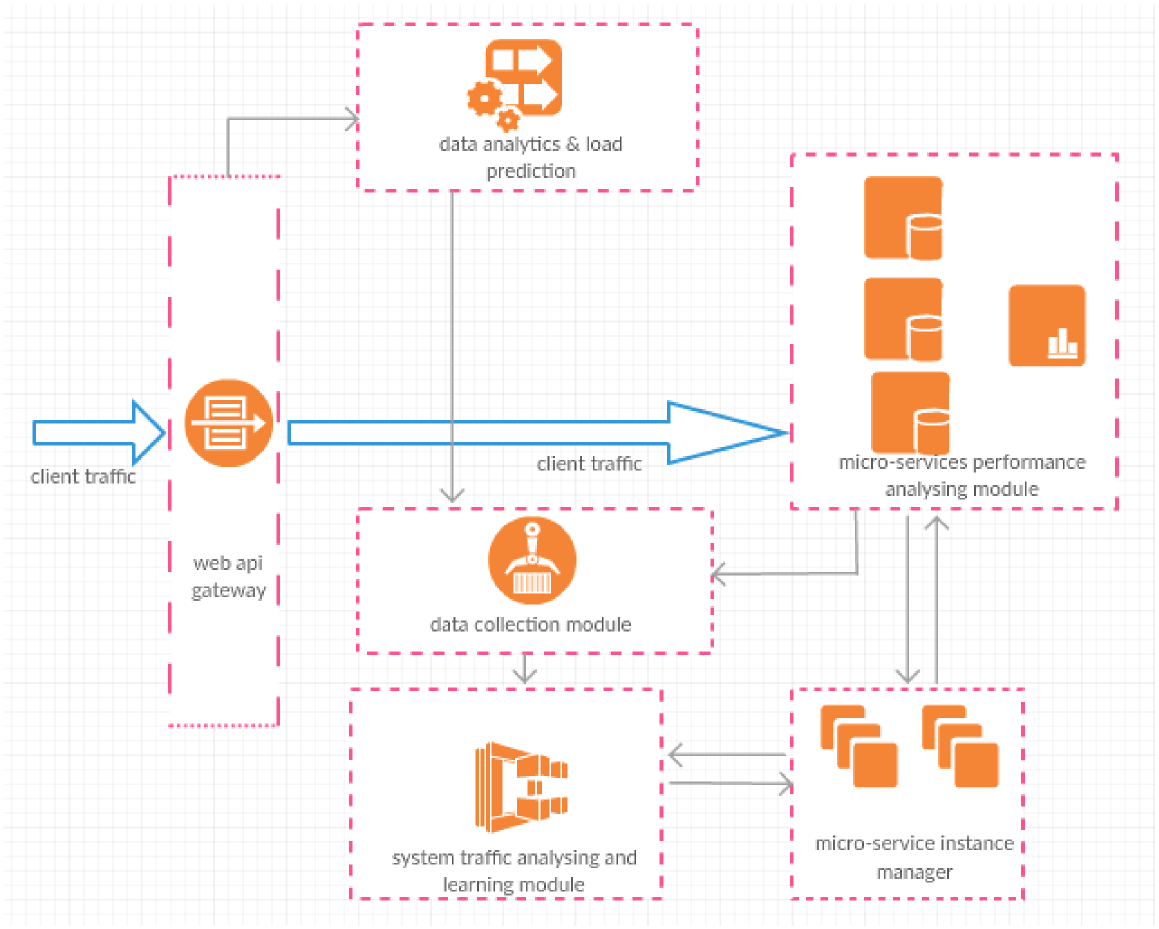

data analytics & load prediction
micro-services performance analysing module
client traffic
client traffic
web api gateway
data collection module
system traffic analysing and learning module
micro-service instance manager

Figure 1.0 – High level system architecture diagram

The estimation will be done on top of the collected data from analytics server. The collected data with extracted analytics parameters will be stored in to a central query location. The time-based machine learning model will be built with effective data visualization using that collected dataset.

The other components as mentioned by (2), (3), (4) are as follows resource utilization analyzing and micro service tagging module, Micro service instance managing and decision making module and Business scenario prediction and analytics visualization module.

Resource managing module which is used to concerned on the resource allocation for each request calls on the micro service instance such as I/O usage, Memory Usage, Network Usage etc. The profiling mechanism would be used in order to track the resource allocation for individual micro services. And also, the tagging system would be implemented to track the sequence of micro services calls in a business scenario. These profiling and tags will be used to identify each services' resource usages and the sequence of execution plan.

Micro service instance managing and decision making module is take the decision of scaling the micro services using the results of (1), (2) modules. That means the consideration of future request load pattern to the application and the consumption of the resources of each individual micro services. The parameters from both modules will be collected and analyzed then the complex model will be built to make the decision of scaling up and down the instances.

The other module is Business scenario prediction and analytics visualization module which is for identify the patterns of the business scenario execution in the deployed micro service architecture and visualization of the resource utilization of the micro service instances by retrieving the details of (2) module. This module is mainly useful for identifying the business scenarios that are there in the application and the prediction of the sequences of the business scenarios.

So the combination all these four 4 modules will be a complete solution for addressed problems in the micro service architectural pattern. [2] The application “Intelligent Auto Scaler for Micro services” which will be deployed in the client organizational environment which follows the Micro Service architecture pattern in their software development. The application will be use by operational level privileges person (ex: Dev-Ops) in the organization.

2 Overall Descriptions

This section will be covered more details about the module which is Web API usage pattern analyzing and the prediction module. The WEB API is the interface for the application that has been deployed in the server. That is the entry point where all the requests come to the application, this information will be kept and stored in the separate location in the server. These are called access logs of the applications which are nothing but loggers. These loggers record the each and every information on the application. A log file can be located in three different places i) web servers ii) web proxy servers and iii) client browsers. Our focus in this application is only on the web server access logs, the other two are much complicated scenarios and there are several researches going on. [3] These server side access logs generally supply the most complete and accurate usage data. Web server logs are plain text (ASCII) files, that is independent from the server platform.

A web log [4] file records the information of the request which is being for the resource from the application. The logs are in the common log file format (CLFF).

```
192.168.100.160 - - [21/Jun/2016:10:08:17 +0530] "POST
/pension_serverside/rest/employees/authenticate/login HTTP/1.1" 404 1098
```

This reflects the information as follows.

Client-id - a unique identifier for the client that issued the request (this may be a proxy)

Timestamp – the date and the time of the request

Method of the request – ex: GET, POST etc.

Resource – the URL path for the requesting resource

Status – status code of the responses

Byte – The content length of the document transferred

Since these logs files should be analyzed in order to make use for the application so that the analytics tools such as Google Analytics or Web Interface would be used. The way it is going to use is plugging in to the server and register the tool by giving the required fields. Then the analytics parameters will be collected and this will be the sample dataset for the application which having much more fields than the access logs which has been collected previously. So then the there will be the processes of the cleaning the dataset in order to satisfy the requirements of the application.

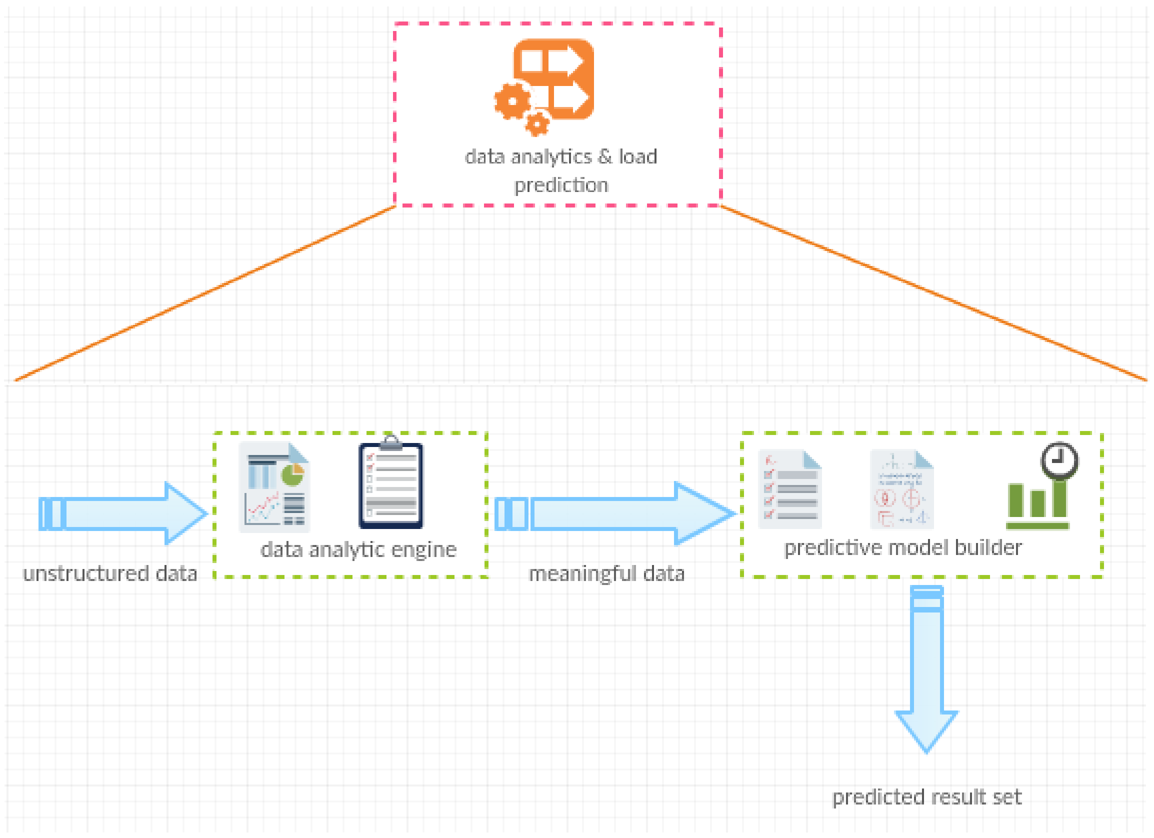


Figure 2.0 – Architectural diagram of the component "data analytics and load prediction module"

The mining of the access logs and identify the usage patterns are called web mining. The web mining processes should be done in order identify the usage patterns in the log files. [4] In order to do that the analyzer tool will be plugged and collected the data. The tool will give the more detailed information with efficient data visualization graphs. This information will be collected and cleaned for the purposes of the model building.

Since these processes are much more deal with the time so that we will be building a time-based machine learning model which predict the number of HTTP request load and the response time for time and requests respectively.

The prediction graphs will be drawn as number of requests vs time for short period of time and the response time vs request for the instances there in the micro service architecture.

2.1 Product perspective

Since the project is giving priority for the research component, this section will cover on the researches have been made specific to the area of log file analyzing, usage pattern identifying and request load visualizations.

There are several researches have been done related to the log analysis using the access logs collected from the server [4][5][6][3]

The research or the solution which is done by Dr.R.Krishnamoorthy and K.R.Suneetha from India Anna University and Bangalore Institute of Technology is that about "Identifying User Behavior by Analyzing Web Server Access Log File" [4] they have applied the web usage mining in order to identify the user behavior which help system administrator and web designers of the application to improve their system. And also, they have worked on the analyzing of system errors, the entries which are irrelevant in data analyzing and mining are removed in the data cleaning process. They have analyzed NASA server log files and various analysis has been done in order to identify the user behavior pattern. In the future work section, they have mentioned to track only the interested users' behaviors to find the access patterns than doing for interested and uninterested users, to come up with high accuracy and performance with less time consumption.

So, this research is mainly focused on identify the user behavior from the logs and increase the performance of the web site this is not focused on the server level rather it more focused on the user level. That means not much concerned on the request load and the patterns of the load for the application that we are going to do in our application.

The another one is "A Study of Web Log Analysis Using Clustering Techniques" [5]. In this paper, same as before the web usage mining is performed which deals with the extraction of interesting knowledge from web log information produced by web servers. The three different algorithms reviewed for generating clusters such are k-means, k-means using neural network, and self-organization map (SOM). The three main tasks have been performed in web mining such as preprocessing, pattern discovery and pattern analysis. The research has been concluded as it is always difficult to determine the number of clusters so that auto clustering feature of SOM is more effective than k-means method.

In "Web Server Workload Characterization" paper, the researchers examined the request and response types and characterize the traffic distribution on the basis of those factors. By work load the researches meant both request stream presented by the clients (the work) as well as the server response to the requests (the load). They have described in detail the requests to WWW server and the characteristics of system's responses. In their architecture, the client web browser make the connection and request to the parent httpd process, the parent httpd process gives the request to one of child process. They have analyzed monthly, weekly and hourly traffic and also about user requests characterization , server response characterization. The research has been concluded with recommending some performance tools. The future work section they mentioned the next phase of study is about the system resource utilization in order understand the system behavior efficiently.

So, all these researches has been focusing areas where identifying the user behaviors using the access logs and some are analyzing the traffic to the server.

So, this module is different in where the accessing the API gateway and analyzing the request load using the plugged analytics server and building the machine learning model which will predict the request load for period of time and also considering the response time of the server to make the parameter available for the decision making module which will be developed in the application in future.

2.1.1 System interfaces

This will be described in section 2.1.4 and 3.1.3

2.1.2 User interfaces

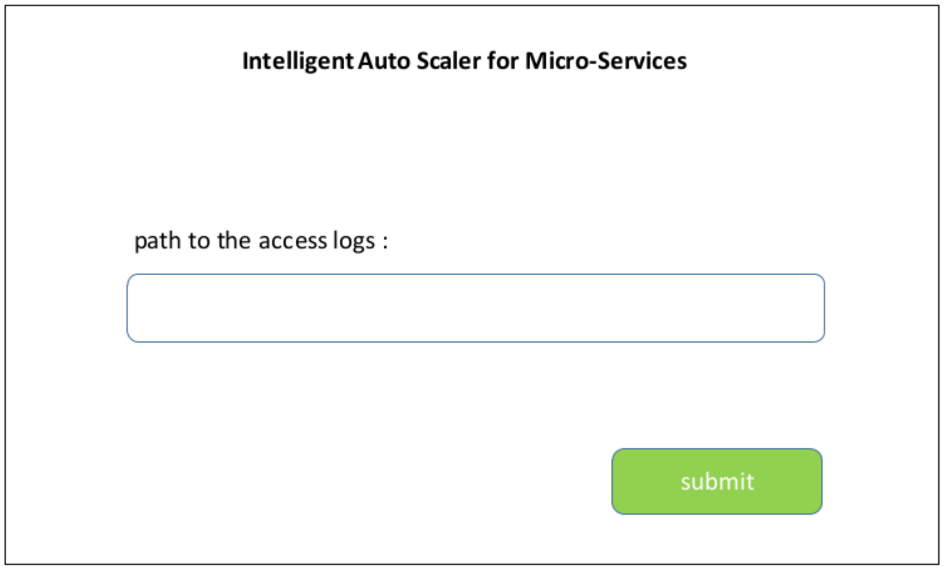

Intelligent Auto Scaler for Micro-Services
path to the access logs :
submit

Figure 3.0 – input path to the access logs of the server

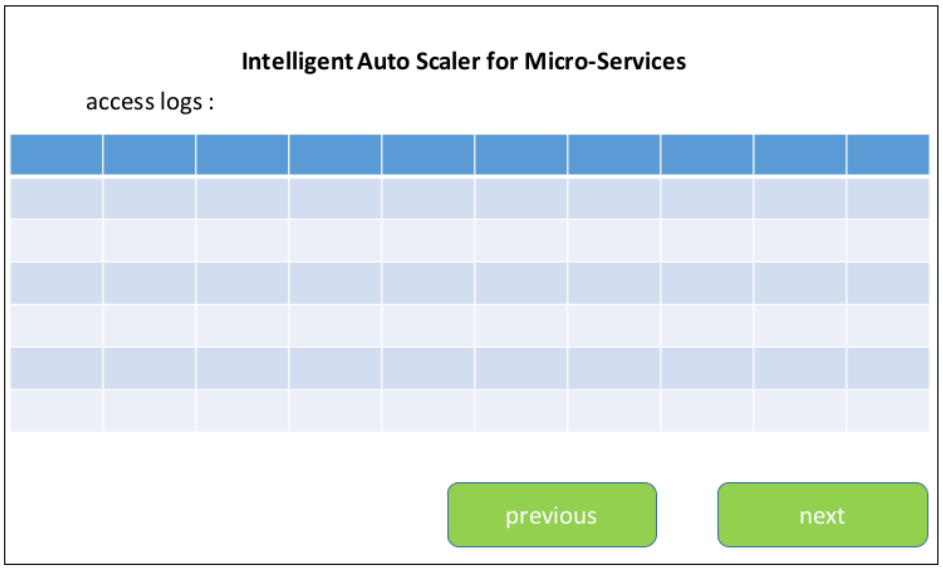


Figure 4.0 – show the access logs in table view

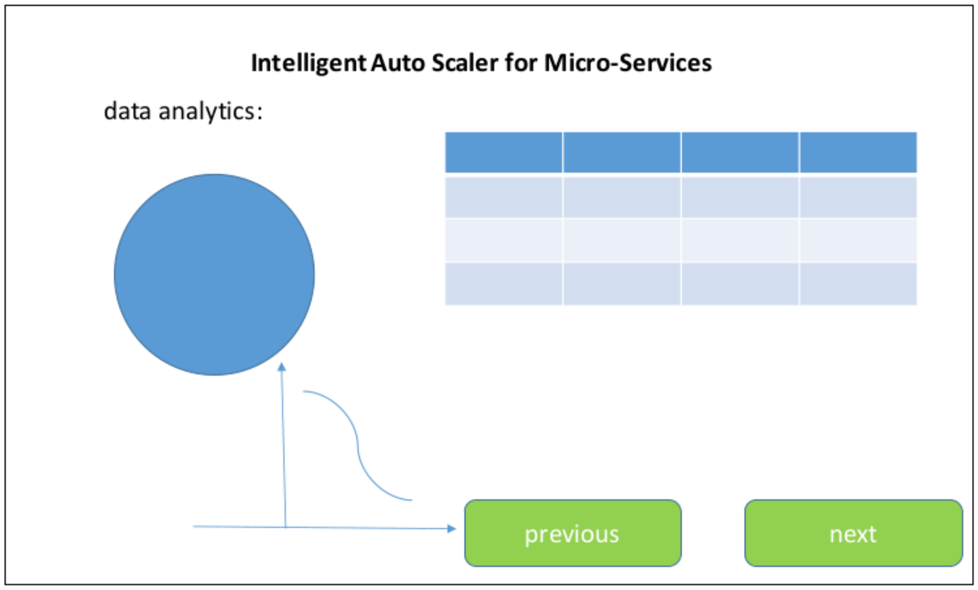


Figure 5.0 – results from the analytics tool for the log file data

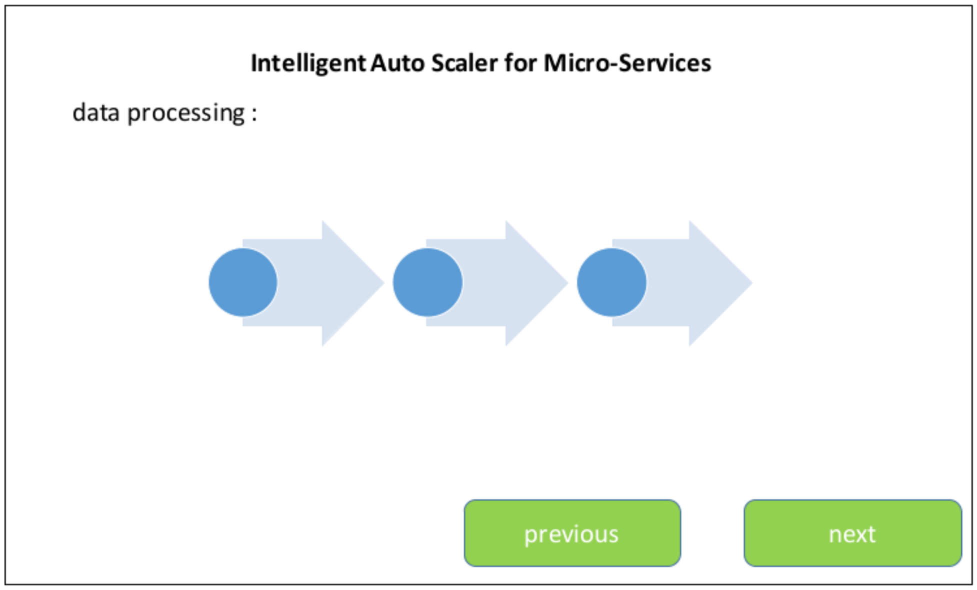


Figure 6.0 – data collecting, preprocessing / cleaning and transforming step

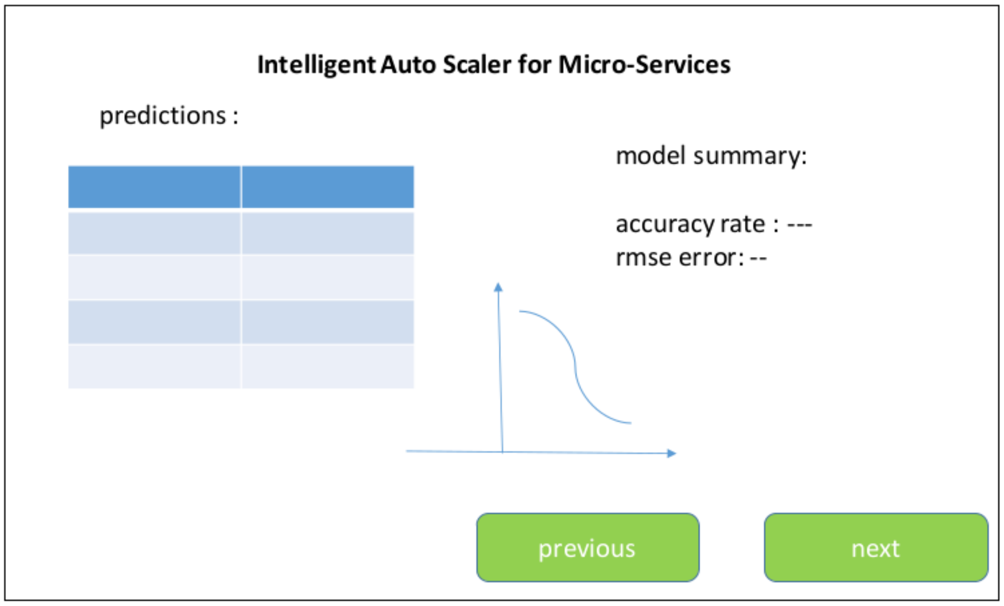


Figure 7.0 – prediction result of the built machine learning model

2.1.3 Hardware interfaces

There are no particularly any hardware devices will be used in this system.

2.1.4 Software interfaces

PyCharm

IntelliJ IDEA

Analytic tool

Node JS

MySQL

2.1.5 Communication interfaces

Internet

Gateway access

API’s

Model builder interfaces

2.1.6 Memory constraints

Minimum of 2 GB memory is expected in order to get the maximum use of the system which means that to working with efficiently and output with minimal response time.

The lack of memory may cause the system to fails in performance.

2.1.7 Operations

User provides the permission to access the log files

User provides information to work with analytics api’s.

User accepts the results and act accordingly.

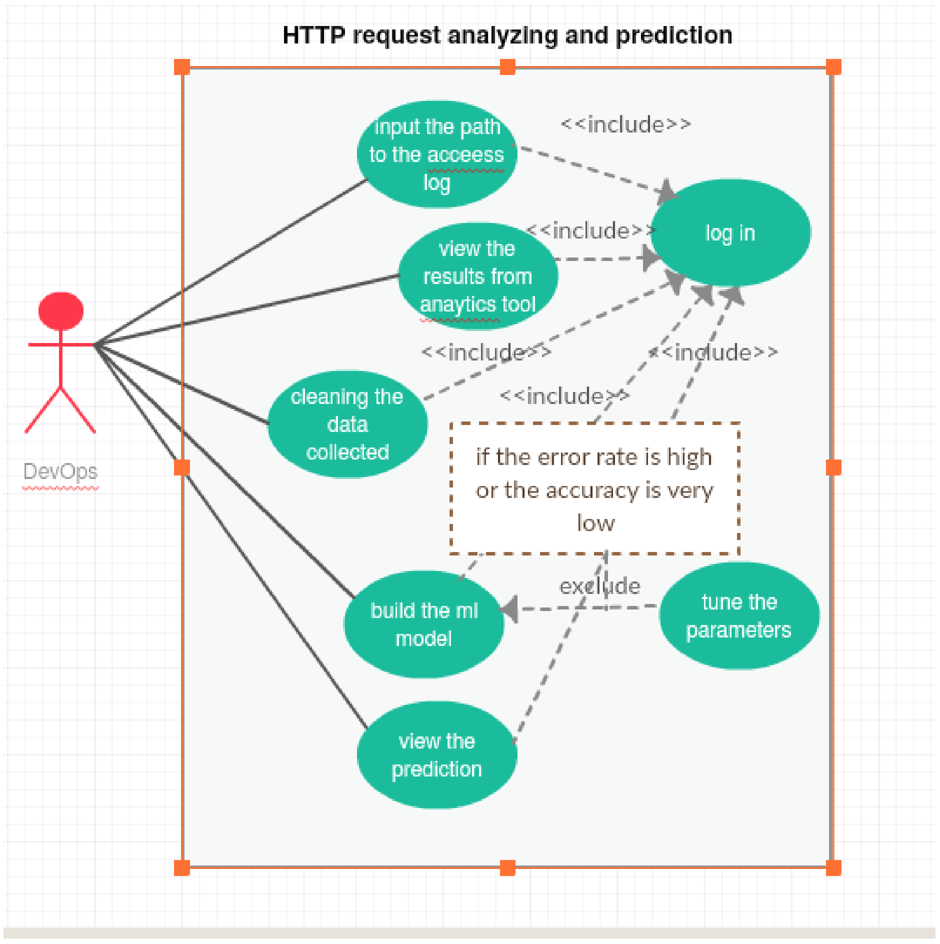
2.1.8 Site adaptation requirements

The port should be configured in order to run this web application

The web browser version should be up to date.

The versions of other product should be compatible.

2.2 Product functions



HTTP request analyzing and prediction
input the path to the acceess log
<<include>>
log in
view the results from anaytics tool
<<include>>
DevOps
cleaning the data collected
<<include>>
<<include>>
<<include>>
if the error rate is high or the accuracy is very low
build the ml model
exclude
tune the parameters
view the prediction

Figure 8.0 use case diagram for the module

Use case ID	001
Use case name	Input the path to the access logs
Pre-condition	Log in to the application
Actor	Dev-ops or system performance engineer or any other authorized employee
Main flow	1. Log in to the system by providing user name and password 2. Input the access logs 3. System displays the message
Alternative flow	1.a Success or error message will be prompted 2.a If user permits log files will be accessed or user cancel the process

Table 2.0 Use case scenario 1

Use case ID	002
Use case name	View the results from analytic tool
Pre-condition	1.Log in to the application 2.Access permission accepted to the analytic tool
Actor	Dev-ops or system performance engineer or any other authorized employee
Main flow	1. Access the analytic tool 2. View the analytics result 3. Select the graphs types 4. Print as report if needed
Alternative flow	1.a Success or error message will be prompted

Table 3.0 Use case scenario 2

Use case ID	003
Use case name	Cleaning and Preprocess the data collected
Pre-condition	1. Dataset is available
Actor	Dev-ops or system performance engineer or any other authorized employee
Main flow	1. Use the dataset 2. Preprocess the dataset 3. Transform as it's align with building the machine learning model 4. Output cleansed dataset
Alternative flow	

Table 4.0 Use case scenario 3

Use case ID	004
Use case name	Build the machine learning model
Pre-condition	1. Preprocessed dataset is available
Actor	Dev-ops or system performance engineer or any other authorized employee
Main flow	1. Use the dataset 2. Classify train and test data 3. Use the RNN Time series algorithm 4. View the result
Alternative flow	3.a Use another algorithm 4.a If the error rate is high then tune the parameters and rebuild

Table 5.0 Use case scenario 4

Use case ID	005
Use case name	View the prediction result
Pre-condition	1. Machine learning built model is available
Actor	Dev-ops or system performance engineer or any other authorized employee
Main flow	1. Use the test data 2. Analyze the prediction 3. Show with visualization graphs to the user
Alternative flow	

Table 6.0 Use case scenario 5

2.3 User characteristics

User should have the kind of privileges of checking the performance of the system and responsibility of doing tasks that to increase the system performance and so on. Also, it will be added beneficial that user aware about the Docker management to the system.

E.g.: DevOps, System performance engineer

The user will have the access of view the request load coming in to the system and how does it varies with user, time etc. and the prediction of the load in near future.

2.4 Constraints

The implementation language for the machine learning is decided as Python since more libraries are available in that domain as well as the recommended language by experts.

The web application will be implemented which will run in the client environment with servers in back-end.

The Internet connection should be available to accessing the analytic tool and API's.

2.5 Assumptions and dependencies

There will be so many areas that make the effect on this system since it is more complex and highly involvement and latest technology so the application only could be built in a way with the limited time available with the limited resources.

Assume that building model will produce the better accuracy.

Assume that the integration with other tools will work as expected if something goes wrong the risk transfer plans will be developed like using suitable frameworks and comes up with the working solution.

This particular discussing module's output will be used by the decision-making module so that there will be the dependencies.

2.6 Apportioning of requirements

The order of the requirements implementation is same as the order of the user interfaces provided in section 2.1.2. The overall description for the interfaces is provided in section 3.1.1.

3 Specific requirements

3.1 External interface requirements

3.1.1 User interfaces

This section will describe about the user interfaces provided in section 2.1.2

Figure 3.0 shows that getting the input data to the system to discover usage patterns. The user access log files present very significant information about a web server. There are products available [3] [7] which uses log analysis to understand the usage patterns of the server by make use of log files.

Figure 4.0 shows that the log files data in the UI to increase the visibility of the system. Here user can understand about each request format with the several fields such as client ID, resource URL, method of request, status code, file size.

Figure 5.0 shows that the analytics result for the inputted log files. There are several web analytics tools which could be used to view the patterns of requests through graphs and tables with categorization of several factors like number of users, country, time, number of request from each client, number of request from region etc.

Figure 6.0 shows that the preprocessing step of model building. There will be the processes of data loading, preprocessing or cleaning the data, mapping or transforming the data and rearranging in order to perform prior to the model building process. The dataset needs to be classified as training and testing dataset. The training dataset will be used to train the model and the testing dataset will be used to view the prediction of the model.

Figure 7.0 shows that the prediction result of the built model. The accuracy of the model will be analyzed by using error values such as RMSE, MSE etc.

3.1.2 Hardware interfaces

There are no any hardware interfaces for this system.

3.1.3 Software interfaces

PyCharm : Thy light weight IDE for Python and Scientific development.

IntelliJ IDEA: For Java development

Node JS: For web application development

MySQL: Local database implementation

Analytic tool: Google Analytics or any other web analytic tool

3.1.4 Communication interfaces

Internet: Web application, API access

Gateway access: For data retrieval

API’s: For data analytics tasks

Model builder interfaces: For Machine Learning

3.2 Architectural Design

3.2.1 High level Architectural Design

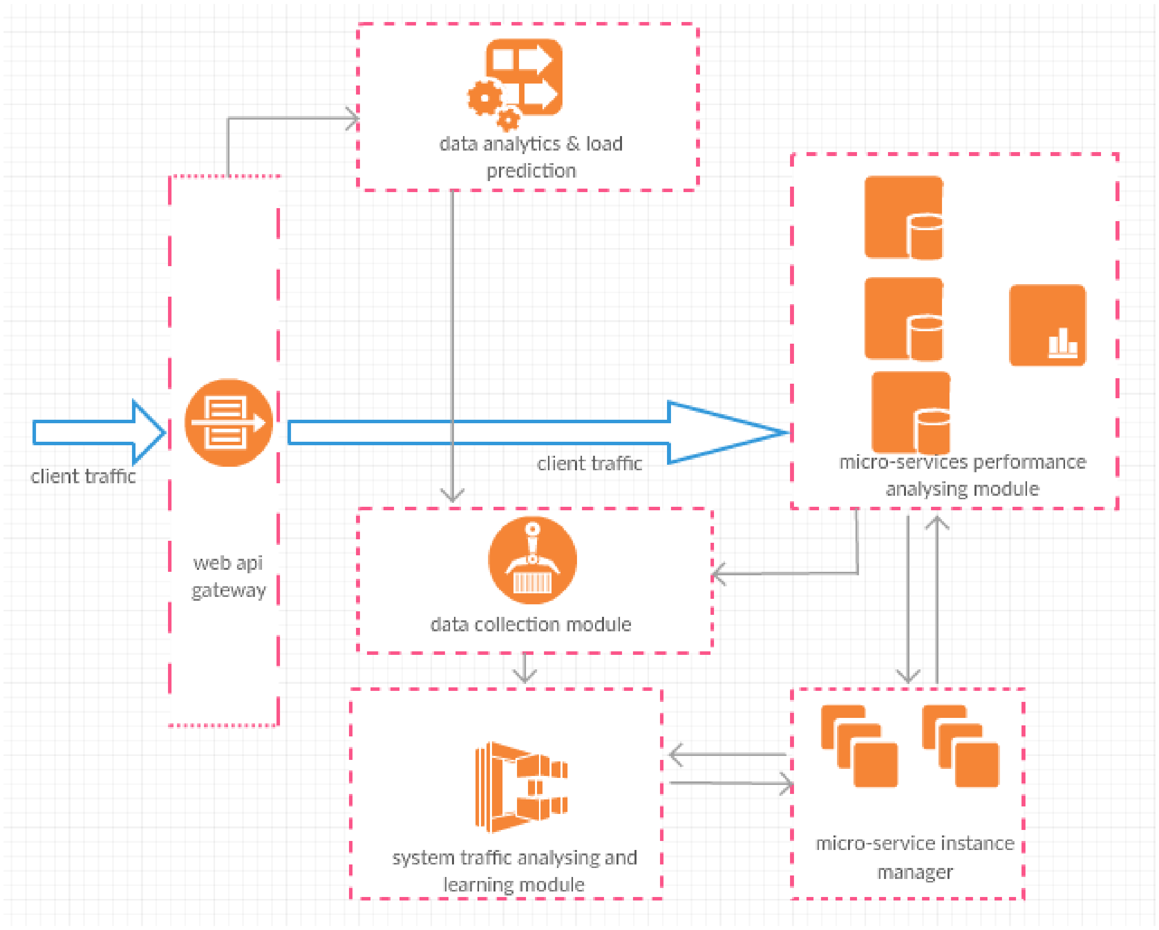

data analytics & load prediction
micro-services performance analysing module
client traffic
client traffic
web api gateway
data collection module
system traffic analysing and learning module
micro-service instance manager

Figure 9.0 – High level system architecture diagram

3.2.2 High level Component Design

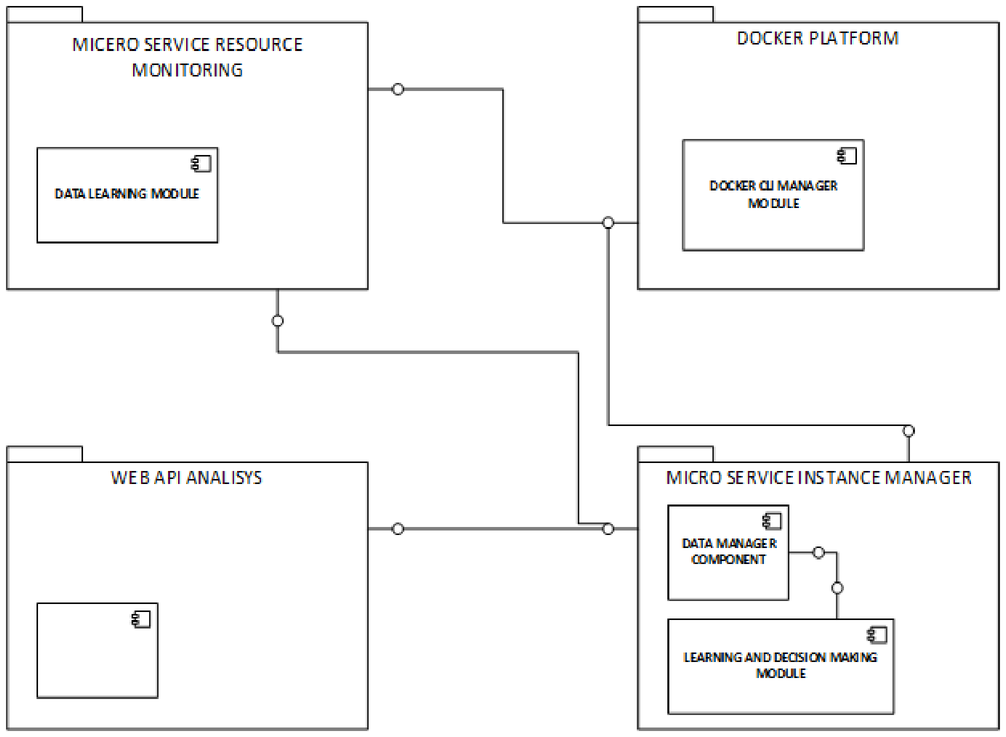
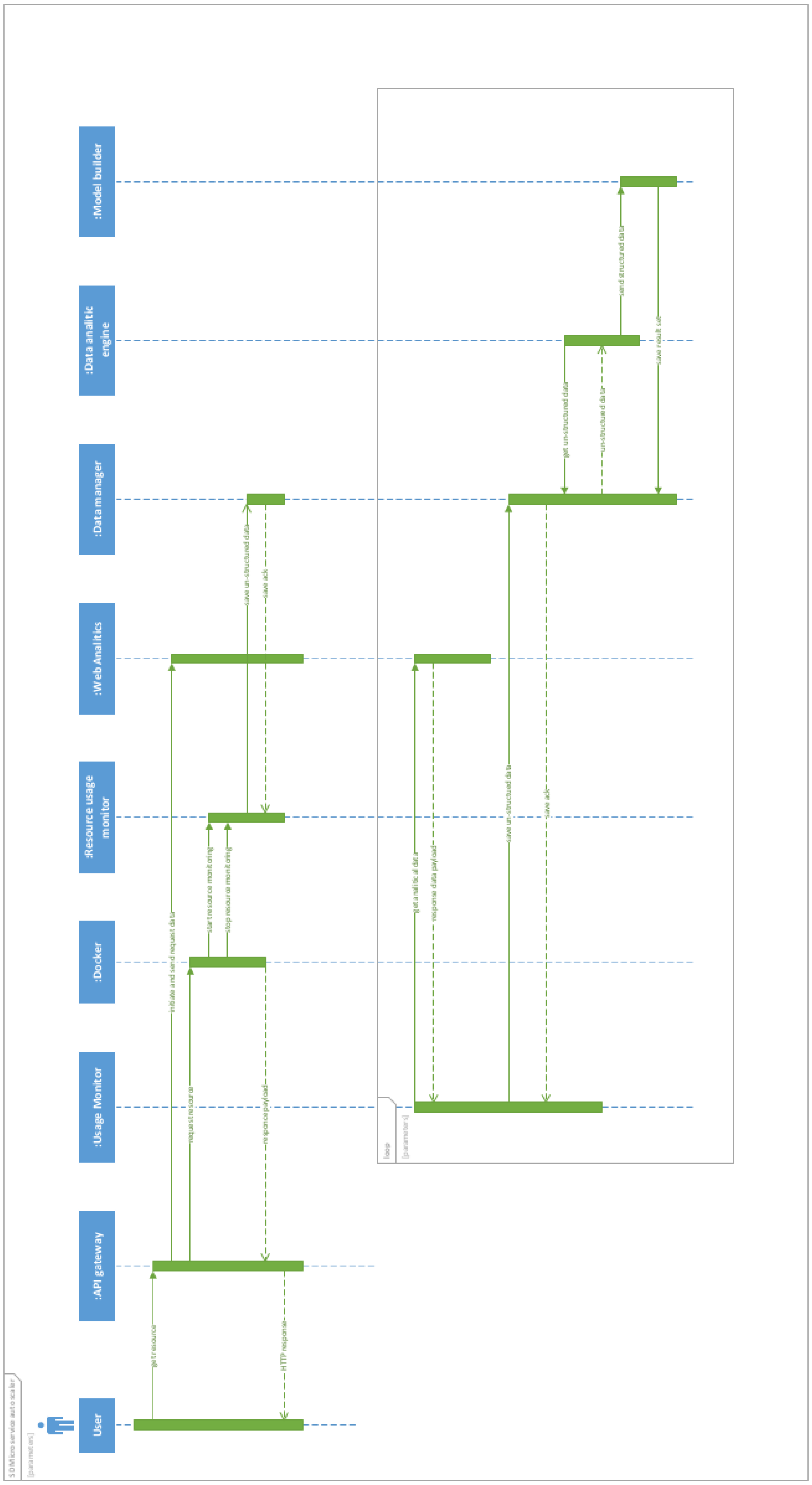

MICERO SERVICE RESOURCE MONITORING
DATA LEARNING MODULE
DOCKER PLATFORM
DOCKER CLI MANAGER MODULE
WEB API ANALISYS
MICRO SERVICE INSTANCE MANAGER
DATA MANAGER COMPONENT
LEARNING AND DECISION MAKING MODULE

Figure 10.0 – system component diagram

3.2.1 High level Sequence Design

Figure 11.0 System sequence diagram



3.2.1 Low level class diagram

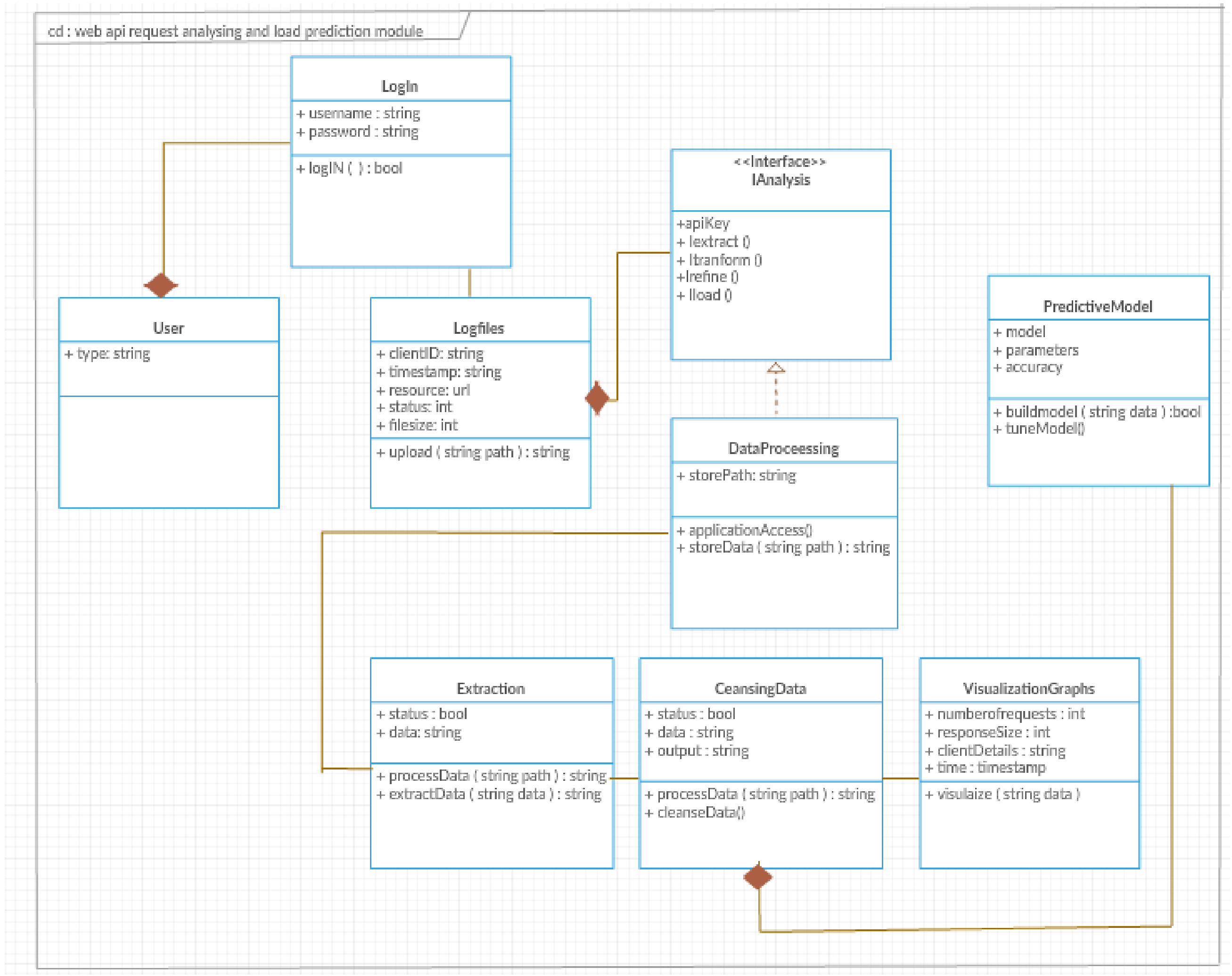


Figure 11.0 – class diagram for the module

3.3 Performance requirements

The system should be able to analyse big data.

The entire log files should load to the interface in less than 2 minutes.

The analytics dashboard should appear in less than 5 minutes.

The time for data pre-processing and machine learning model building should not take more than 5 minutes.

The speed of query from the database should be high

3.4 Design constraints

The given design principles in this document will be followed as it is and the changes should be minimal.

3.5 Software system attributes

3.5.1 Reliability

The system should run along with the running solution in the client environment. The failure of the system should be minimized since the features it produces are very much effect on the system performance analysis.

The system should have engaged with the proper testing procedures prior to the deployment in client environment.

3.5.2 Availability

The availability of the system may align with the processes of the model building and so on, since the system uses historical data and do the prediction for a period of time. If the real time is possible it should be added as a mentionable feature to the system.

The future releases will consider those aspects in deep.

3.5.3 Security

The security of the system should be very high since the system may be deployed in the cloud environment in future, so that with the availability of existing security options in cloud should be covered this application processes as well.

The basic levels of security should be developed as creating the login functionality to the system and for future releases the security aspects should take in place.

3.5.4 Maintainability

The system should be able to easily maintainable by the user whoever responsible for the tasks, the customer support should also be provided in order to solve the issues when it occurs.

The system should be developed in a way of decreasing the complexity of adding new requirements when it arrived. This will be implemented by following the proper coding standards and documentation.

4 Supporting information

4.1 Appendices

4.1.1 High Level Diagram

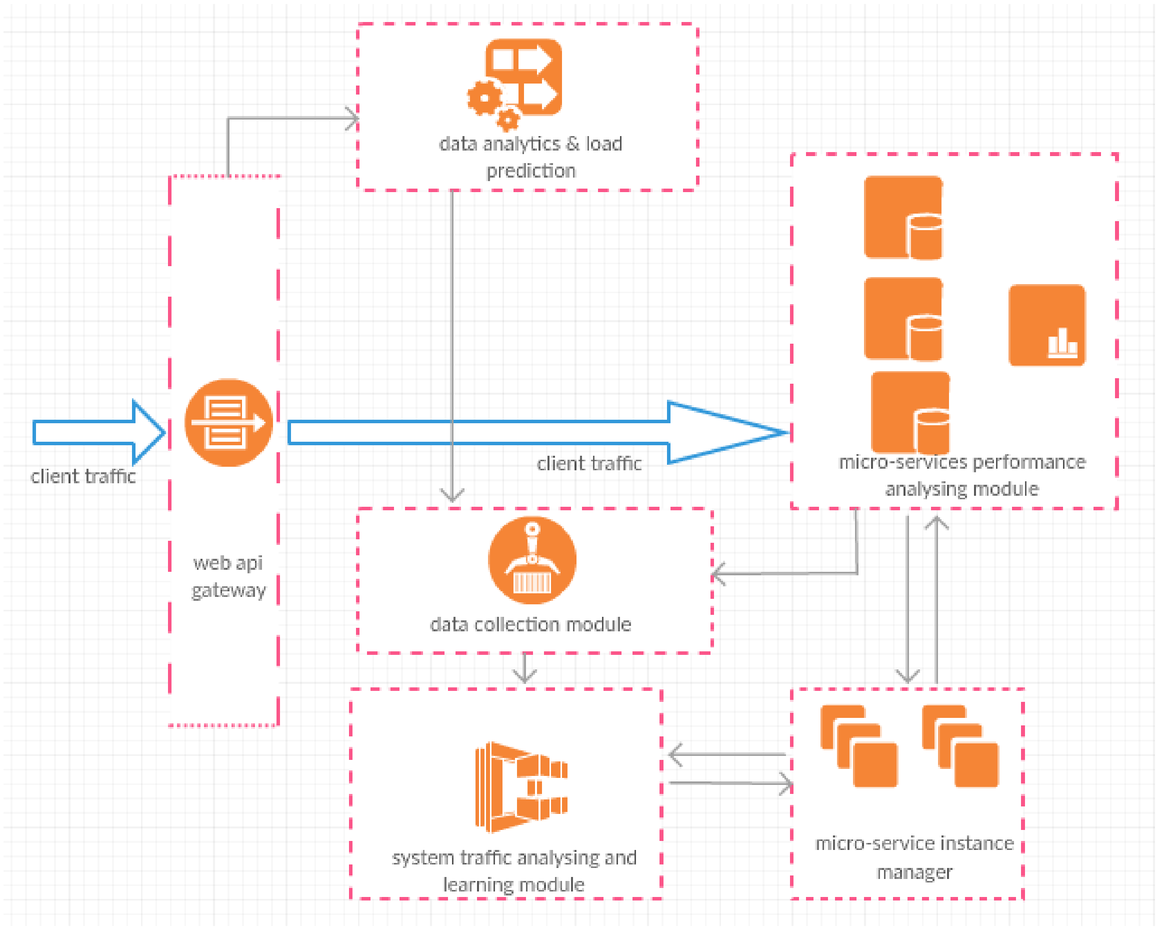
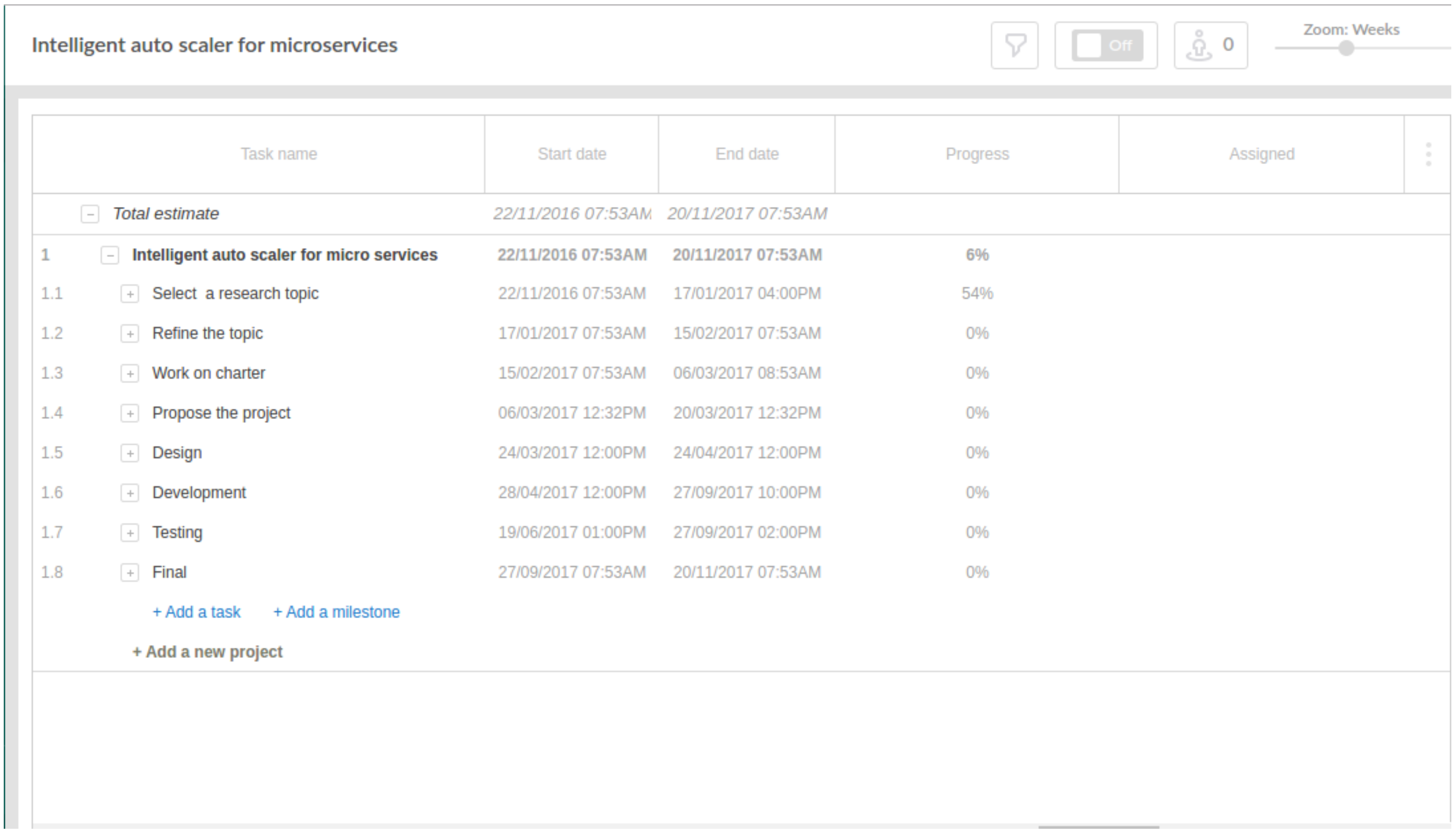

data analytics & load prediction
micro-services performance analysing module
client traffic
client traffic
web api gateway
data collection module
system traffic analysing and learning module
micro-service instance manager

Figure 12.0 - High level system diagram

4.1.2 Gantt chart



Intelligent auto scaler for microservices

Off · 0 · Zoom: Weeks

	Task name	Start date	End date	Progress	Assigned
	Total estimate	*22/11/2016 07:53AM*	*20/11/2017 07:53AM*		
1	**Intelligent auto scaler for micro services**	**22/11/2016 07:53AM**	**20/11/2017 07:53AM**	**6%**	
1.1	Select a research topic	22/11/2016 07:53AM	17/01/2017 04:00PM	54%	
1.2	Refine the topic	17/01/2017 07:53AM	15/02/2017 07:53AM	0%	
1.3	Work on charter	15/02/2017 07:53AM	06/03/2017 08:53AM	0%	
1.4	Propose the project	06/03/2017 12:32PM	20/03/2017 12:32PM	0%	
1.5	Design	24/03/2017 12:00PM	24/04/2017 12:00PM	0%	
1.6	Development	28/04/2017 12:00PM	27/09/2017 10:00PM	0%	
1.7	Testing	19/06/2017 01:00PM	27/09/2017 02:00PM	0%	
1.8	Final	27/09/2017 07:53AM	20/11/2017 07:53AM	0%	

+ Add a task   + Add a milestone

+ Add a new project

Figure 13.0 – Gannt chart

4.1.3 Work Breakdown Structure (WBS)

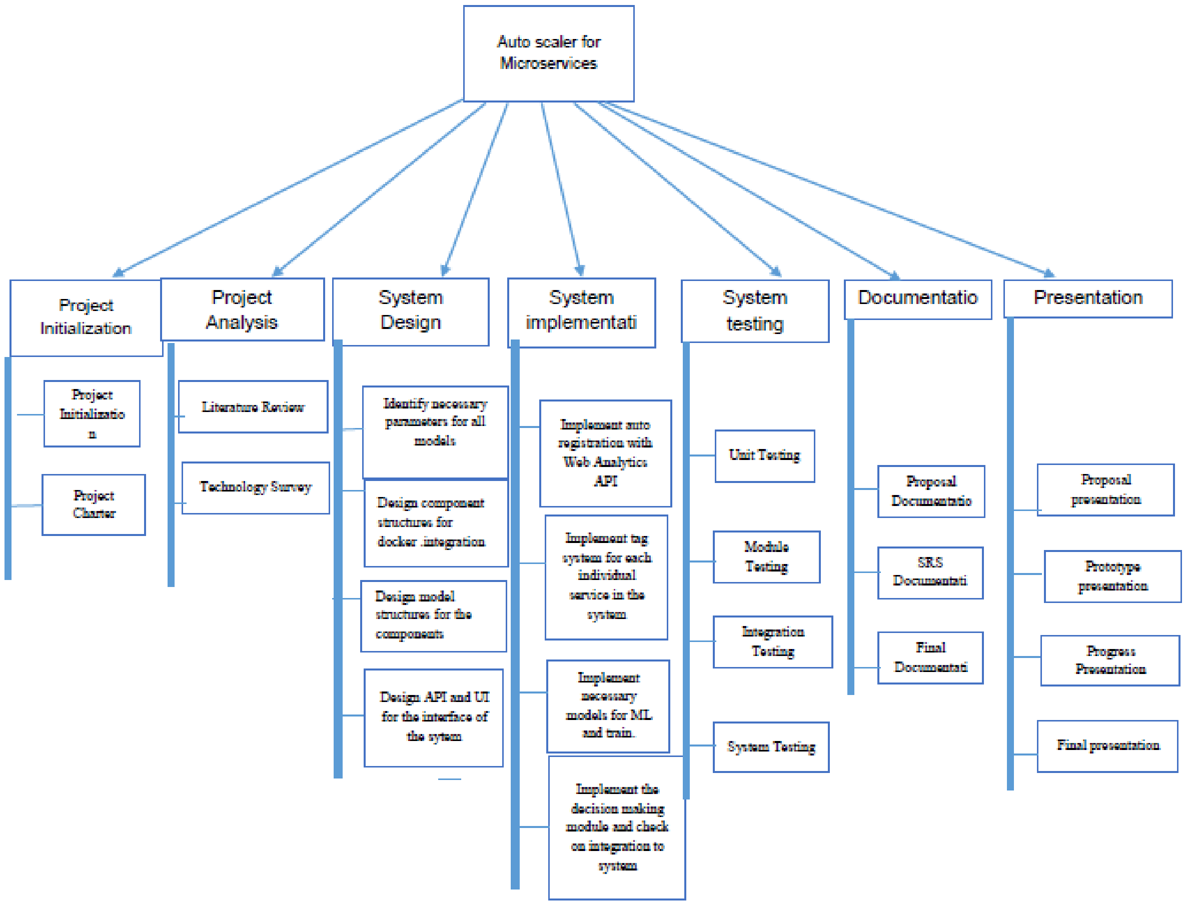

Auto scaler for Microservices
Project Initialization
Project Initializatio n
Project Charter
Project Analysis
Literature Review
Technology Survey
System Design
Identify necessary parameters for all models
Design component structures for docker .integration
Design model structures for the components
Design API and UI for the interface of the sytem
System implementati
Implement auto registration with Web Analytics API
Implement tag system for each individual service in the system
Implement necessary models for ML and train.
Implement the decision making module and check on integration to system
System testing
Unit Testing
Module Testing
Integration Testing
System Testing
Documentatio
Proposal Documentatio
SRS Documentati
Final Documentati
Presentation
Proposal presentation
Prototype presentation
Progress Presentation
Final presentation

Figure 14.0 – Work break down structure diagram

References

[1] Goel, Neha. "Automated Performance Analysis Of Load Tests - IEEE Xplore Document". Ieeexplore.ieee.org. N.p., 2013. Web. 1 May 2017.

[2] The proposal document - https://docs.google.com/document/d/1OqT1LZ5Gvecu4YhT-OPCdu1Uo655s4K_ZqjCA20f2_4/edit

[3] Alspaugh S. , Chen B. , Lin J. “Analyzing Log Analysis: An Empirical Study of User Log Mining – IEEE Xplore Document”. Ieeexplore.ieee.org. N.p 2014. Web. 1 May 2017.

[4] K.R, Suneetha. "Web Usage Pattern Analysis Through Web Logs: A Review - IEEE Xplore Document". Ieeexplore.ieee.org. N.p., 2009. Web. 1 May 2017.

[5] Hemansu Rana , Mayank Patel “A Study of Web Log Analysis Using Clustering Techniques”A Review - IEEE Xplore Document". Ieeexplore.ieee.org. N.p., 2013. Web. 1 May 2017.

[6] Martin Arlitt, Tai Jin . “Workload Characterization of the 1998 World Cup Web Site I EEE Xplore Document". Ieeexplore.ieee.org. N.p., 1999. Web. 1 May 2017.

[7]https://www.novell.com/documentation/groupwise2014r2/gw2014_guide_admin/data/adm_webacc_mon_logs.html [online] last accessed : 1 May 2017