INDEX

PREFACE

PREFACE

ASSET PERFORMANCE MANAGEMENT is a web app developed for providing Real Time Performance Management and Monitoring of various assets installed at industrial plants. It provides an Analytic platform where predictive and descriptive analytics can be executed.

Previously there were no software available which provides real time asset management and monitoring along with analytic executor platform in a single software, so administrator must configure and invest in two different software. Since two different software were used, there were high chances of getting inaccurate results.

ASSET PERFORMANCE MANAGEMENT overcomes this issue and takes care of analytic execution as well as real time monitoring. It provides dashboard for data visualization of analytics and real time asset status if anything goes wrong. It also produces alarms if certain analytic is getting failed.

OBJECTIVE OF THE APPLICATION:

- To provide an application to manage and monitor assets in real-time.
- To provide an application that helps to make strategic planning and decisions.
- To provide early knowledge about asset outage.
- To provide real time view of the target output specified.
- To Provide a platform for analytic deployment and execution which gives real time asset performance.

ACKNOWLEDGEMENT

ACKNOWLEDGEMENT

It gives me great pleasure in expressing my sincere gratitude towards my Principal Dr. Mrs. Usha Mukundan for providing me with the valuable and necessary infrastructure to carry out the project work.

I express my sincere gratitude to my Project Guide **Prof. Bharti Bhole** (Professor, Department of Information Technology, R.J.C) for all the encouragement and analytical views on my topic of interest. Their constructive suggestions and insight helped me in exploring my project from all possible perspective.

Their flexible approach, patience and faith in my abilities not only taught me the fundamentals behind the field but also enabled me to develop innovative ideas. They explained the toughest concept with remarkable ease and because of their easy approachability all my doubts were comprehensively answered. It is their knowledge that has been my guiding light through all my work so far.

REQUIREMENT GATHERING PHASE

PROBLEM STATEMENT –

Previously there were no application for real time asset management and monitoring which was integrated with analytic executor. As well the monitoring features were very limited. No application was available which gives facility to perform data analysis using business intelligent tools integrated with the analytic platform.

OBJECTIVE OF THE APPLICATION:

- To provide an application to manage and monitor assets in real-time.
- To provide an application that helps to make strategic planning and decisions.
- To provide early knowledge about asset outage.
- To provide real time view of the target output specified.
- To Provide a platform for analytic deployment and execution which gives real time asset performance.

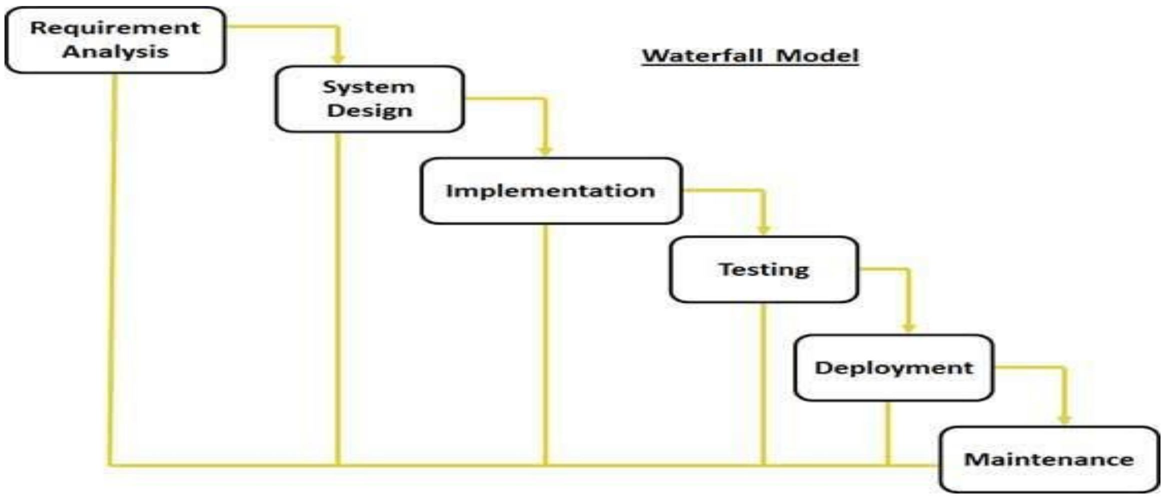
SCOPE & ROLE OF THE PROJECT

- ➢ Administrator can use their credentials to visualize the processed and interpolated data.
- ➢ It maintains records of the analytic execution status whether they got passed or failed.
- ➢ Provide admin a feature with which admin can manage and monitor real time data generated by the assets as well he can interpolate it.
- ➢ Minimizes the time consumption right from analytic execution to data visualization as everything is automated so there is no manual work needed.
- ➢ Admin can view each analytic which he/she has deployed for execution. He/ She gets notified with the status of analytic.

METHODOLOGY
AND
PROCESS MODEL

PROCESS MODEL USED IN MY PROJECT

For my project, planned to use waterfall as a process model. The waterfall model is a sequential design process, often used in software development processes, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Conception, Initiation, Analysis, Design, Construction, Testing and Maintenance.



The sequential phases in Waterfall model are:

- **Requirement Gathering and analysis:** All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification doc.
- **System Design:** The requirement specifications from first phase are studied in this phase and system design is prepared.

- **Implementation:** With inputs from system design, the system is first developed in small programs called units, which are integrated in the next phase.
- **Integration and Testing:** All the units developed in the implementation phase are integrated into a system after testing of each unit.
- **Deployment of system:** Once the functional and non functional testing is done, the product is deployed in the customer environment or released into the market.
- **Maintenance:** There are some issues which come up in the client environment. To fix those issues patches are released.

ANALYSIS

REQUIREMENT ANALYSIS

Requirement analysis is a Software engineering task bridging the gap between system requirements engineering and software design. Software Requirement analysis may be divided into five areas of effort:

- Problem recognition
- Evaluation and synthesis
- Modeling
- Specification
- Review

Requirement Analysis Process Steps

- Domain understanding:

  -Understanding of application domain.

- Requirements collections:

  - The process of interacting with customers, users to discover the requirements for the system.

- Requirements classification:

  - Group and classify the gathered requirements.

- Conflict resolution:

  - Resolve the conflict requirements.

- Prioritization:

  - Identify and list requirements according to their importance

- Requirements validation:

- Check and validate the gathered requirements to see if they are complete, correct, and sound.

DRAWBACKS OF THE EXISTING SYSTEM

The challenges in the Existing System are:

- ➢ No single application for real time asset performance and management.
- ➢ No analytic executor platform integrated with web application which can visualize analytic status.
- ➢ There was no application integrated with business intelligent tools.

PROPOSED SYSTEM

- ➢ To provide an application to manage and monitor assets in real-time.
- ➢ To provide an application that helps to make strategic planning and decisions.
- ➢ To provide early knowledge about asset outage.
- ➢ To provide real time view of the target output specified.
- ➢ To Provide a platform for analytic deployment and execution which gives real time asset performance.

Functional requirement: -

- The functional requirements describe the core functionality of the application. This section includes the data and functional process requirements.
- There is two type of user who uses these app

<u>Asset Admin:</u>

- ▪ The application allows admin view the real time data and status for the analytic execution.
- ▪ Admin can analyze real time data on the visual graphs.
- ▪ Admin can create graphs and data statistics using business intelligent tool integrated with the application.

Analytic Admin:

- The Analytic Admin can use Analytic Executor web application to deploy and validate analytics being developed.

NON-FUNCTIONAL REQUIREMENT ONE AS FOLLOWS:

- Performance: -

It is deployed in AWS, so it's performance is fast.

- Portability: -

Used Bootstrap and make it available for all platforms

- Safety: -

VPA security groups are used for data security.

- Availability: -

It is easy to available for all user from anywhere.

FEASIBILITY STUDY

FEASIBILITY STUDY

This is an evaluation and analysis of the potential of the proposed project which is based on extensive investigation and research to support the process of decision making. The feasibility study is intended to be a preliminary review of the facts to see if it is worthy of proceeding to the analysis phase. From the systems analyst perspective, the feasibility analysis is the primary tool for recommending whether to proceed to the next phase or to discontinue the project. There are aspects in the -feasibility study portion of the preliminary investigation:

1. Technical Feasibility
2. Operation Feasibility
3. Economical Feasibility
4. Schedule Feasibility

1. Technical Feasibility

The technical issue usually raised during the feasibility stage of the investigation includes the following:

- Does the necessary technology exist to do what is suggested?
- Can the system be upgraded if developed?
- Are there technical guarantees of accuracy, reliability, ease of access and data security?
- Is the project feasible within the limits of current technology?
- Is it available within given resource constraints?
- Is it a practical proposition?
- Is there enough manpower- programmers, testers & debuggers?
- Do the required software and hardware exist?
- Are the current technical resources sufficient for the new system?

- Can the technology be easily applied to current problems?
- Does the technology have the capacity to handle the solution?

2. <u>Operational Feasibility</u>

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. Some of the important issues raised are to test the operational feasibility of a project includes the following: -

- Is there sufficient support for the management from the users?
- Will the system be used and work properly if it is being developed and implemented?
- Will there be any resistance from the user that will undermine the possible application benefits?
- Is the project feasible to operate or not?
- Does current mode of operation provide adequate throughput and response time?
- Could there be a reduction in cost and or an increase in benefits?
- Does current mode of operation offer effective controls to protect against fraud and to guarantee accuracy and security of data and information?
- Does it agree with the government regulations?
- Will the proposed system really benefits the organization?

3. <u>Economic Feasibility</u>

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economic feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs.

4. <u>Schedule Feasibility</u>

A project will fail if it takes too long to be completed before it is useful. Typically, this means estimating how long the system will take to develop, and if it can be completed in a given time period using some methods like payback period. Schedule feasibility is a measure of how reasonable the project timetable is. Given our technical expertise, are the project deadlines reasonable? Some projects are initiated with specific deadlines. You need to determine whether the deadlines are mandatory or desirable.

TOOLS AND TECHNOLOGIES USED

Hardware Requirements

The system should be tested on different Configuration and Platforms. For optimum performance of the project, the requirement is as shown below: -

- Processor - Intel Core Duo 2.0 GHz or more
- RAM - 2 GB MB or More
- Hard disk – 30 GB or more
- Monitor - 15" CRT, or LCD monitor
- Keyboard - Normal or Multimedia
- Mouse - Compatible mouse
- AWS – EC2 Instance – 2 GB RAM, 1 GHZ Processor
- RDS Storage – 30 GB

Tools and software:

FRONT END:

- HTML
- CSS
- Bootstrap
- AngularJS
- PloymerJS
- JavaScript

BACK END:

- Java JDK 1.8
- Python 2.7
- PostgreSQL
- Proficy Historian
- Git
- Apache Maven 3.4 or above
- NodeJS v6.11.3
- npm
- Cassandra 2.2.8
- Grafana 4.5.2
- Kairos DB 0.9.3

SERVER:

Windows server 2012 R2

HTML

HTML was created by Berners-Lee in late 1991 but "HTML 2.0" was the first standard HTML specification which was published in 1995.

HTML is a computer language devised to allow website creation. These websites can then be viewed by anyone else connected to the Internet. It is relatively easy to learn, with the basics being accessible to most people in one sitting; and quite powerful in what it allows you to create.

The definition of HTML is Hyper Text Markup Language.

Hyper Text is the method by which you move around on the web — by clicking on special text called hyperlinks which bring you to the next page. The fact that it is hyper just means it is not linear — i.e. you can go to any place on the Internet whenever you want by clicking on links — there is no set order to do things in.

Markup is what HTML tags do to the text inside them. They mark it as a certain type of text (italicized text, for example).

HTML is a Language, as it has code-words and syntax like any other language.

CSS

CSS Stands for "Cascading Style Sheet." Cascading style sheets are used to format the layout of Web Pages. They can be used to define text styles, table sizes, and other aspects of Web pages that previously could only be defined in a page's HTML.

CSS helps Web developers create a uniform look across several pages of a Web site. Instead of defining the style of each table and each block of text within a page's HTML, commonly used styles need to be defined only once in a CSS document. Once the style is defined in cascading style sheet, it can be used by any page that references the CSS file. Plus, CSS makes it easy to change styles across several pages at once. For example, a Web developer may want to increase the default text size from 10pt to 12pt for fifty pages of a Web site. If the pages all reference the same style sheet, the text size only needs to be changed on the style sheet and all the pages will show the larger text.

AngularJS

AngularJS is a structural framework for dynamic web apps. It lets you use HTML as your template language and lets you extend HTML's syntax to express your application's components clearly and succinctly. AngularJS's data binding and dependency injection eliminate much of the code you would otherwise have to write. And it all happens within the browser, making it an ideal partner with any server technology.

AngularJS is what HTML would have been, had it been designed for applications. HTML is a great declarative language for static documents. It does not contain much in the way of creating applications, and as a result building web application is an exercise in *what do I have to do to trick the browser into doing what I want?*

The impedance mismatch between dynamic applications and static documents is often solved with:

a library - a collection of functions which are useful when writing web apps. Your code is in charge and it calls into the library when it sees fit. E.g., jQuery.

frameworks - a particular implementation of a web application, where your code fills in the details. The framework is in charge and it calls into your code when it needs something app specific. E.g., durandal, ember, etc.

AngularJS takes another approach. It attempts to minimize the impedance mismatch between document centric HTML and what an application needs by creating new HTML constructs. AngularJS teaches the browser new syntax through a construct we call *directives*. Examples include:

Data binding, as in {{}}.

DOM control structures for repeating, showing and hiding DOM fragments.

Support for forms and form validation.

Attaching new behavior to DOM elements, such as DOM event handling.

Grouping of HTML into reusable components.

PolymerJS

As a team of front-end engineers embedded in the Chrome team, our mission is to make the web better, by helping developers unlock the web's full potential and by spurring the web platform to evolve and improve.

We work on libraries, tools and patterns to help developers build modern Progressive Web Apps, taking advantage of cutting-edge features like Web Components, Service Workers and HTTP/2.

We also participate in the standards process, doing our part to ensure that front-end developers have a strong voice in the evolution of the platform.

JavaScript

Javascript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

JavaScript was first known as **LiveScript,** but Netscape changed its name to JavaScript, possibly because of the excitement being generated by Java. JavaScript made its first appearance in Netscape 2.0 in 1995 with the name **LiveScript**. The general-purpose core of the language has been embedded in Netscape, Internet Explorer, and other web browsers.

- The ECMA-262 Specification defined a standard version of the core JavaScript language.
- JavaScript is a lightweight, interpreted programming language.
- Designed for creating network-centric applications.
- Complementary to and integrated with Java.
- Complementary to and integrated with HTML.
- Open and cross-platform

Java

Java is a programming language and computing platform first released by Sun Microsystems in 1995. There are lots of applications and websites that will not work unless you have Java installed, and more are created every day. Java is fast, secure, and reliable. From laptops to datacenters, game consoles to scientific supercomputers, cell phones to the Internet, Java is everywhere!

Principles

- There were five primary goals in the creation of the Java language
- It must be "simple, object-oriented, and familiar".
- It must be "robust and secure".
- It must be "architecture-neutral and portable".
- It must execute with "high performance".
- It must be "interpreted, threaded, and dynamic".

Pyhon

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

PostgreSQL

PostgreSQL, is an open source relational database management system. It is based on the structure query language (SQL), which is used for adding, removing, and modifying information in the database. Standard SQL commands, such as ADD, DROP, INSERT, and UPDATE can be used with PostgreSQL.

Many database-driven websites that use PostgreSQL also use a programming language like Java to access information from the database. PostgreSQL commands can be incorporated into the java code, allowing part or all a Web page to be generated from database information. Because both PostgreSQL and java are both open source (meaning they are free to download and use), the Java/ PostgreSQL combination has become a popular choice for database-driven websites.

AWS

Amazon Web Services (AWS) is Amazon’s cloud web hosting platform that offers flexible, reliable, scalable, easy-to-use, and cost-effective solutions. This tutorial covers various important topics illustrating how AWS works and how it is beneficial to run your website on Amazon Web Services. In my project, used concept of RDS, EC2 instance, VPC, security group.

RDS- Amazon Relational Database Service (Amazon RDS) is a web service that makes it easier to set up, operate, and scale a relational database in the cloud. It provides cost-efficient, resizable capacity for an industry-standard relational database and manages common database administration tasks.

DB Instances- The basic building block of Amazon RDS is the DB instance. A DB instance is an isolated database environment in the cloud. A DB instance can contain multiple user-created databases, and you can access it by using the same tools and applications that you use with a stand-alone database instance. You can create and modify a DB instance by using the AWS Command Line Interface, the Amazon RDS API, or the AWS Management Console.

Each DB instance runs a DB engine. Amazon RDS currently supports the MySQL, MariaDB, PostgreSQL, Oracle, and Microsoft SQL Server DB engines. Each DB engine has its own supported features, and each version of a DB engine may include specific features.

Amazon EC2 - Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, resizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers.

Amazon EC2's simple web service interface allows you to obtain and configure capacity with minimal friction. It provides you with complete control of your computing resources and lets you run on Amazon's proven computing environment. Amazon EC2 reduces the time required to obtain and boot new server instances to minutes, allowing you to quickly scale capacity, both up and down, as your computing requirements change. Amazon EC2 provides developers the tools to build failure resilient applications and isolate them from common failure scenarios.

Amazon VPC- Amazon Virtual Private Cloud (Amazon VPC) enables you to launch AWS resources into a virtual network that you've defined. This virtual network closely resembles a traditional network that you'd operate in your own data center, with the benefits of using the scalable infrastructure of AWS.

Security Group Basics

The following are the basic characteristics of security groups for your VPC:

You have limits on the number of security groups that you can create per VPC, the number of rules that you can add to each security group, and the number of security groups you can associate with a network interface. For more information, see Amazon VPC Limits.

You can specify allow rules, but not deny rules.

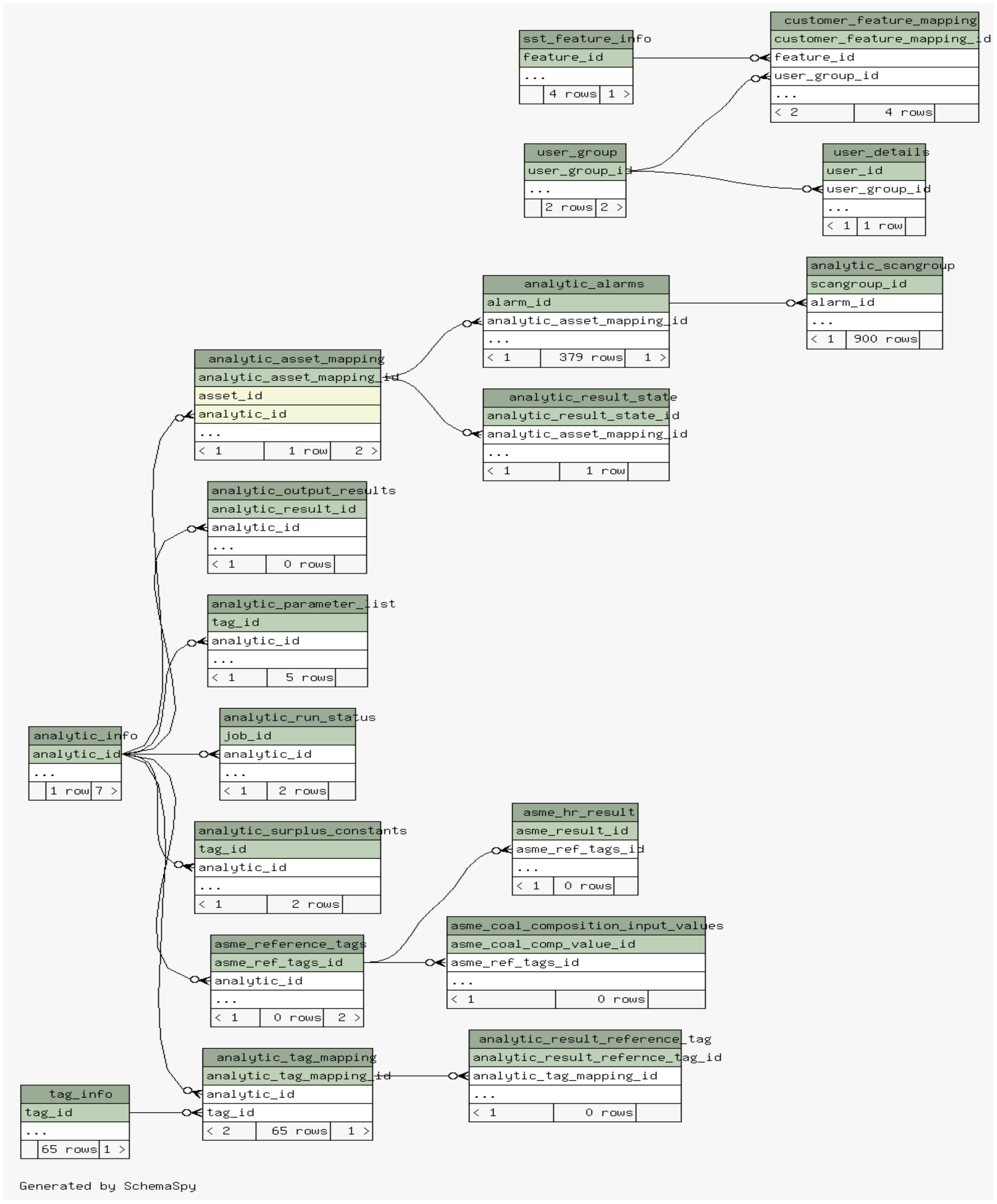
You can specify separate rules for inbound and outbound traffic.

When you create a security group, it has no inbound rules. Therefore, no inbound traffic originating from another host to your instance is allowed until you add inbound rules to the security group.

By default, a security group includes an outbound rule that allows all outbound traffic. You can remove the rule and add outbound rules that allow specific outbound traffic only. If your security group has no outbound rules, no outbound traffic originating from your instance is allowed.

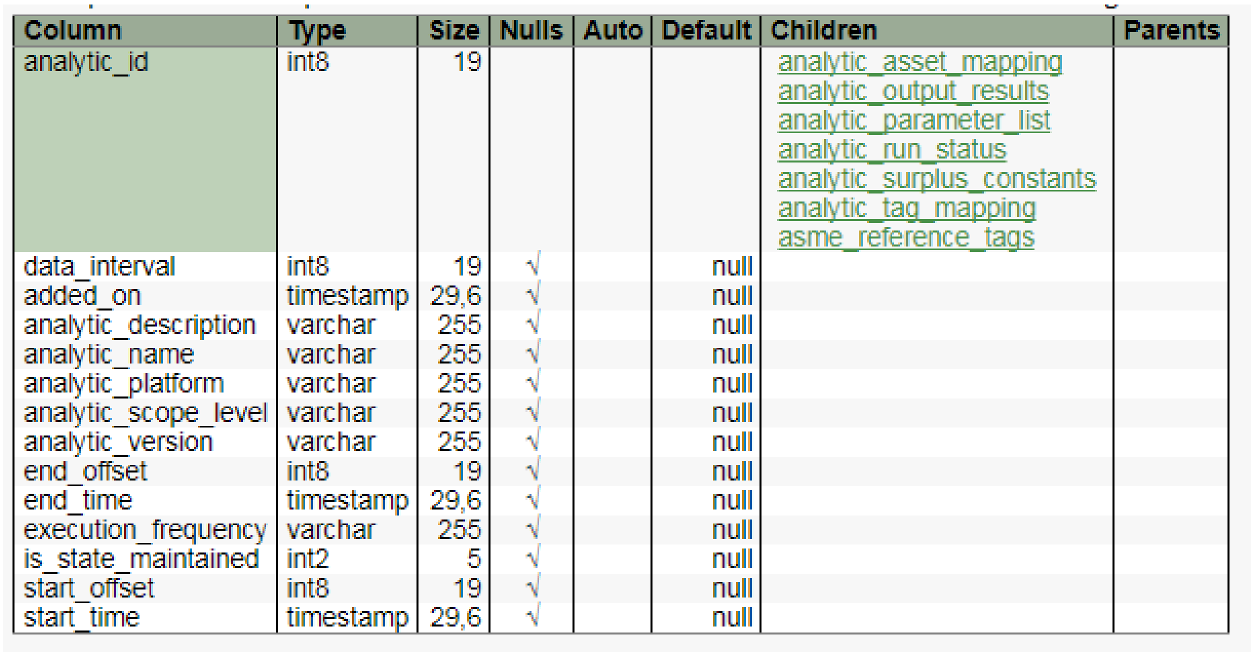
DATABASE DESIGN

Database table Relationship



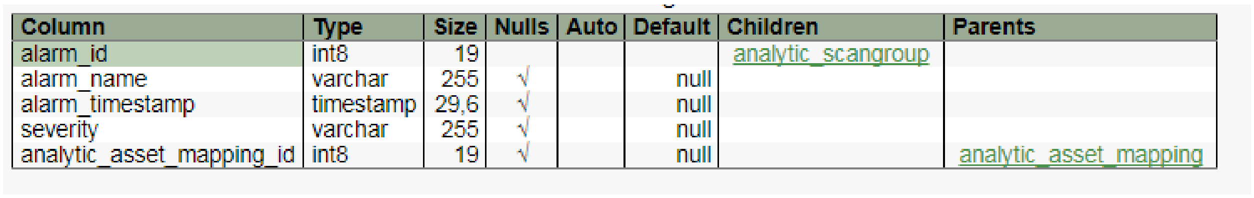
Database Tables

analytic_info



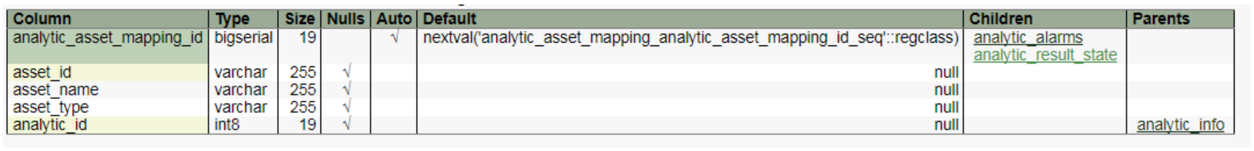
Column	Type	Size	Nulls	Auto	Default	Children	Parents
analytic_id	int8	19				analytic_asset_mapping analytic_output_results analytic_parameter_list analytic_run_status analytic_surplus_constants analytic_tag_mapping asme_reference_tags	
data_interval	int8	19	√		null		
added_on	timestamp	29,6	√		null		
analytic_description	varchar	255	√		null		
analytic_name	varchar	255	√		null		
analytic_platform	varchar	255	√		null		
analytic_scope_level	varchar	255	√		null		
analytic_version	varchar	255	√		null		
end_offset	int8	19	√		null		
end_time	timestamp	29,6	√		null		
execution_frequency	varchar	255	√		null		
is_state_maintained	int2	5	√		null		
start_offset	int8	19	√		null		
start_time	timestamp	29,6	√		null		

analytic_alarms



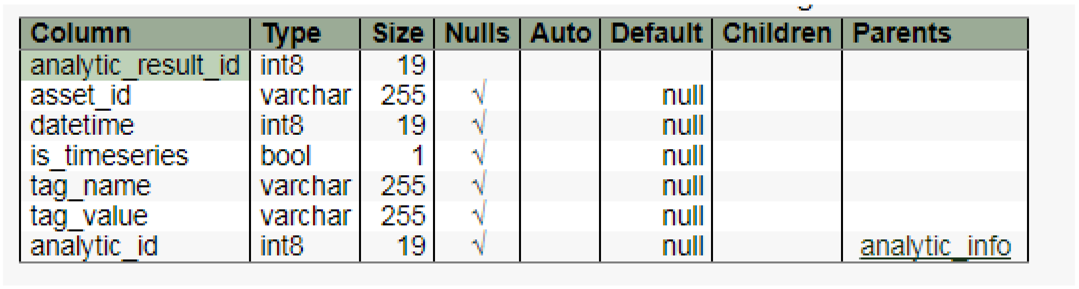
Column	Type	Size	Nulls	Auto	Default	Children	Parents
alarm_id	int8	19				analytic_scangroup	
alarm_name	varchar	255	√		null		
alarm_timestamp	timestamp	29,6	√		null		
severity	varchar	255	√		null		
analytic_asset_mapping_id	int8	19	√		null		analytic_asset_mapping

analytic_asset_mapping



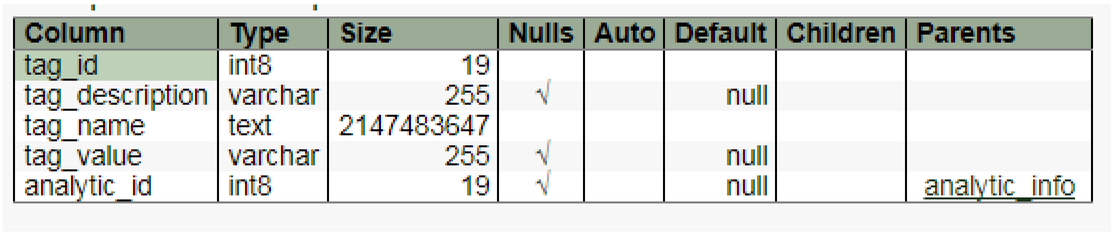
Column	Type	Size	Nulls	Auto	Default	Children	Parents
analytic_asset_mapping_id	bigserial	19		√	nextval('analytic_asset_mapping_analytic_asset_mapping_id_seq'::regclass)	analytic_alarms analytic_result_state	
asset_id	varchar	255	√		null		
asset_name	varchar	255	√		null		
asset_type	varchar	255	√		null		
analytic_id	int8	19	√		null		analytic_info

analytic_output_results



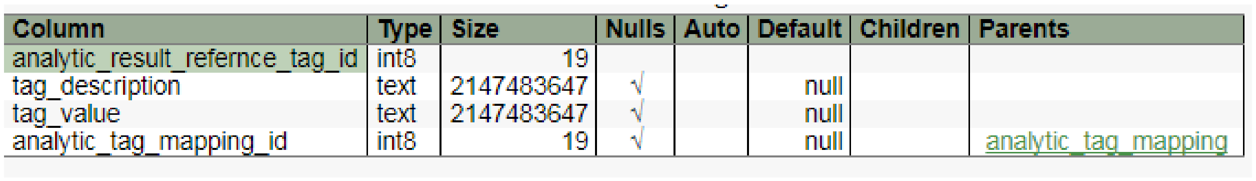
Column	Type	Size	Nulls	Auto	Default	Children	Parents
analytic_result_id	int8	19					
asset_id	varchar	255	√		null		
datetime	int8	19	√		null		
is_timeseries	bool	1	√		null		
tag_name	varchar	255	√		null		
tag_value	varchar	255	√		null		
analytic_id	int8	19	√		null		analytic_info

analytic_parameter_list



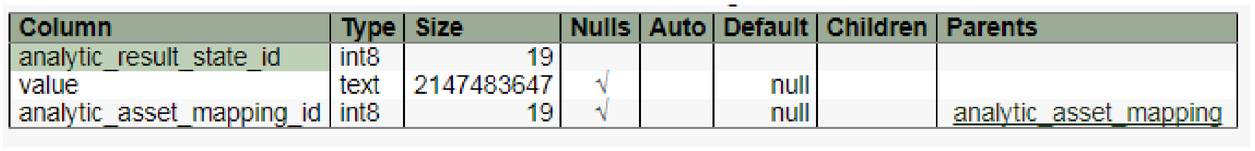
Column	Type	Size	Nulls	Auto	Default	Children	Parents
tag_id	int8	19					
tag_description	varchar	255	√		null		
tag_name	text	2147483647					
tag_value	varchar	255	√		null		
analytic_id	int8	19	√		null		analytic_info

analytic_result_reference_tag



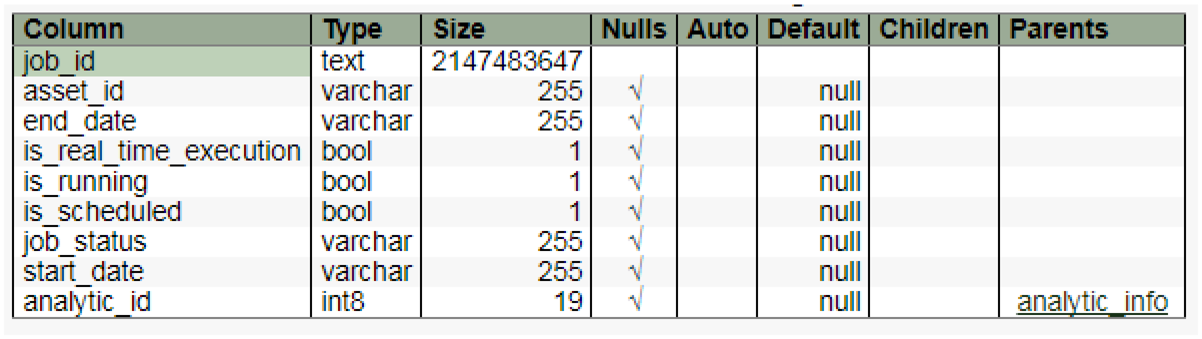
Column	Type	Size	Nulls	Auto	Default	Children	Parents
analytic_result_refernce_tag_id	int8	19					
tag_description	text	2147483647	√		null		
tag_value	text	2147483647	√		null		
analytic_tag_mapping_id	int8	19	√		null		analytic_tag_mapping

analytic_result_state



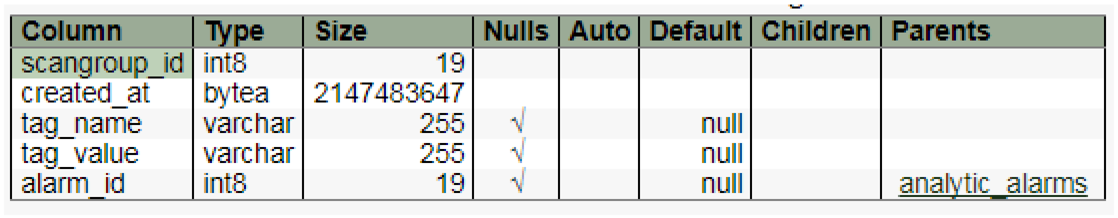
Column	Type	Size	Nulls	Auto	Default	Children	Parents
analytic_result_state_id	int8	19					
value	text	2147483647	√		null		
analytic_asset_mapping_id	int8	19	√		null		analytic_asset_mapping

analytic_run_status



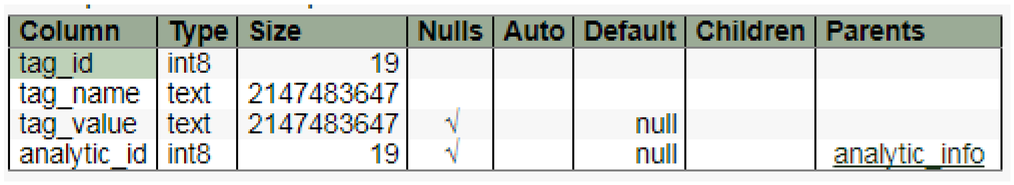
Column	Type	Size	Nulls	Auto	Default	Children	Parents
job_id	text	2147483647					
asset_id	varchar	255	√		null		
end_date	varchar	255	√		null		
is_real_time_execution	bool	1	√		null		
is_running	bool	1	√		null		
is_scheduled	bool	1	√		null		
job_status	varchar	255	√		null		
start_date	varchar	255	√		null		
analytic_id	int8	19	√		null		analytic_info

analytic_scangroup



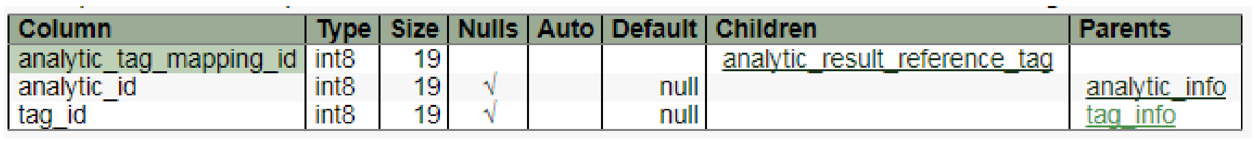
Column	Type	Size	Nulls	Auto	Default	Children	Parents
scangroup_id	int8	19					
created_at	bytea	2147483647					
tag_name	varchar	255	√		null		
tag_value	varchar	255	√		null		
alarm_id	int8	19	√		null		analytic_alarms

analytic_surplus_constants



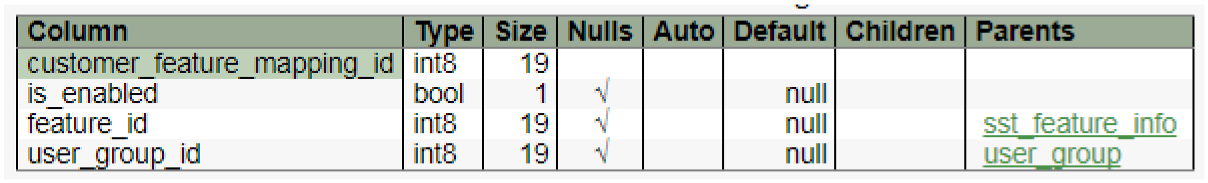
Column	Type	Size	Nulls	Auto	Default	Children	Parents
tag_id	int8	19					
tag_name	text	2147483647					
tag_value	text	2147483647	√		null		
analytic_id	int8	19	√		null		analytic_info

analytic_tag_mapping



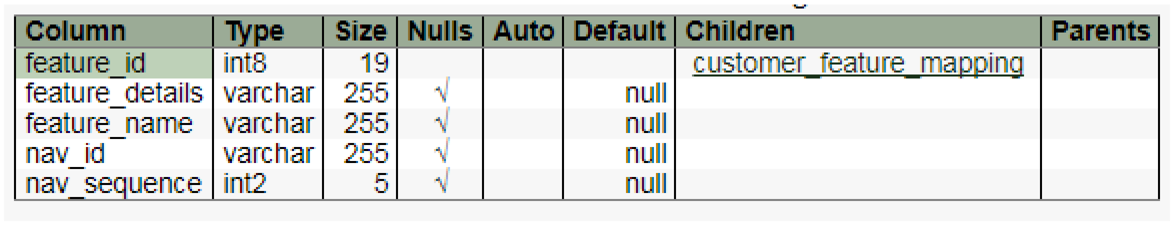
Column	Type	Size	Nulls	Auto	Default	Children	Parents
analytic_tag_mapping_id	int8	19				analytic_result_reference_tag	
analytic_id	int8	19	√		null		analytic_info
tag_id	int8	19	√		null		tag_info

customer_feature_mapping



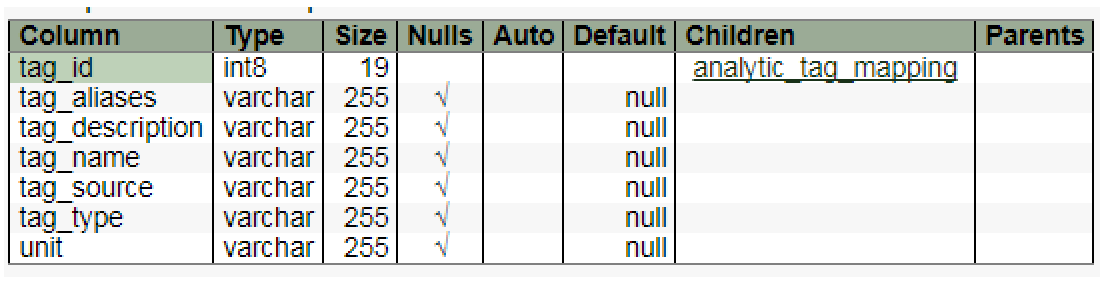
Column	Type	Size	Nulls	Auto	Default	Children	Parents
customer_feature_mapping_id	int8	19					
is_enabled	bool	1	√		null		
feature_id	int8	19	√		null		sst_feature_info
user_group_id	int8	19	√		null		user_group

apm_feature_info



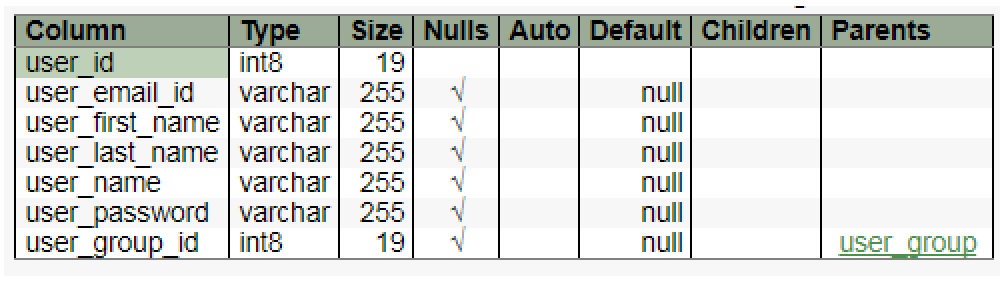
Column	Type	Size	Nulls	Auto	Default	Children	Parents
feature_id	int8	19				customer_feature_mapping	
feature_details	varchar	255	√		null		
feature_name	varchar	255	√		null		
nav_id	varchar	255	√		null		
nav_sequence	int2	5	√		null		

tag_info



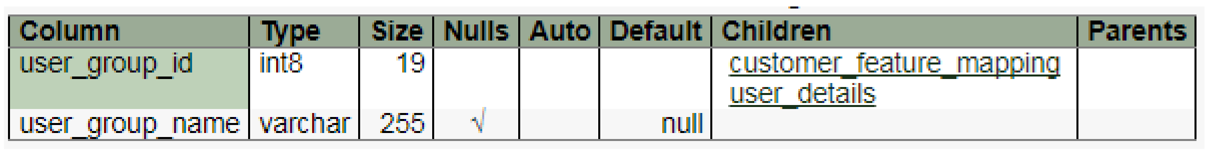
Column	Type	Size	Nulls	Auto	Default	Children	Parents
tag_id	int8	19				analytic_tag_mapping	
tag_aliases	varchar	255	√		null		
tag_description	varchar	255	√		null		
tag_name	varchar	255	√		null		
tag_source	varchar	255	√		null		
tag_type	varchar	255	√		null		
unit	varchar	255	√		null		

user_details



Column	Type	Size	Nulls	Auto	Default	Children	Parents
user_id	int8	19					
user_email_id	varchar	255	√		null		
user_first_name	varchar	255	√		null		
user_last_name	varchar	255	√		null		
user_name	varchar	255	√		null		
user_password	varchar	255	√		null		
user_group_id	int8	19	√		null		user_group

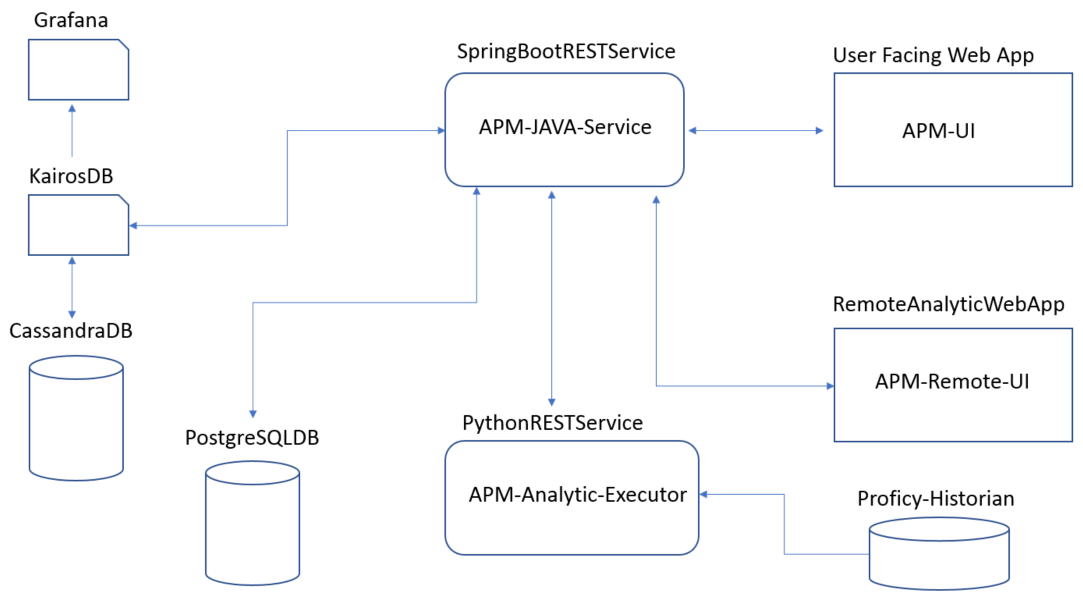
user_group

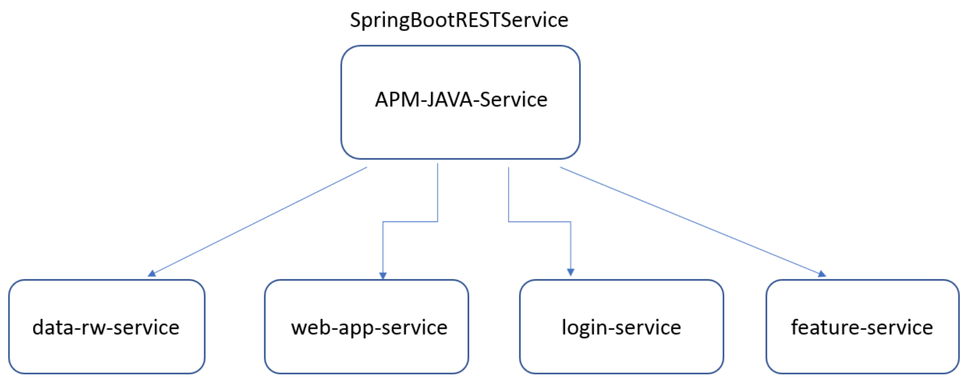


Column	Type	Size	Nulls	Auto	Default	Children	Parents
user_group_id	int8	19				customer_feature_mapping user_details	
user_group_name	varchar	255	√		null		

SYSTEM DESIGN

Technical Architecture:





Technical Component and Services

APM-JAVA-Service:
A Java SpringBoot REST service which will be having below sub-service.

data-rw-service:
A Java SpringBoot REST Service which will be responsible for exposing APIs for read and write data from and to python REST service.

web-app-service:
A Java SpringBoot REST Service which will be responsible for exposing APIs for data visualization on APM-UI.

login-service:
A Java SpringBoot REST Service which will be responsible for exposing APIs for authentication and authorization of users.

features-service:
A Java SpringBoot REST Service which will be responsible for exposing APIs for role based access to application features.

APM-Analytic-Executor:
A python REST service responsible for exposing APIs for analytic execution and data flow between Java and Python REST services.

APM-UI:
A web application for data visualization.

APM-Remote-UI:
A web application for deploying analytics.

PostgreSQL DB:
A relational database for storing and manipulating transactional and configuration data related to APM application.

Cassandra DB:
A NoSQL database use to store time-series data returned from Analytics.

KairosDB:
A time-series database working as an interface for CassandraDB to store and retrieve time-series data, also useful for interpolation of time-series data.

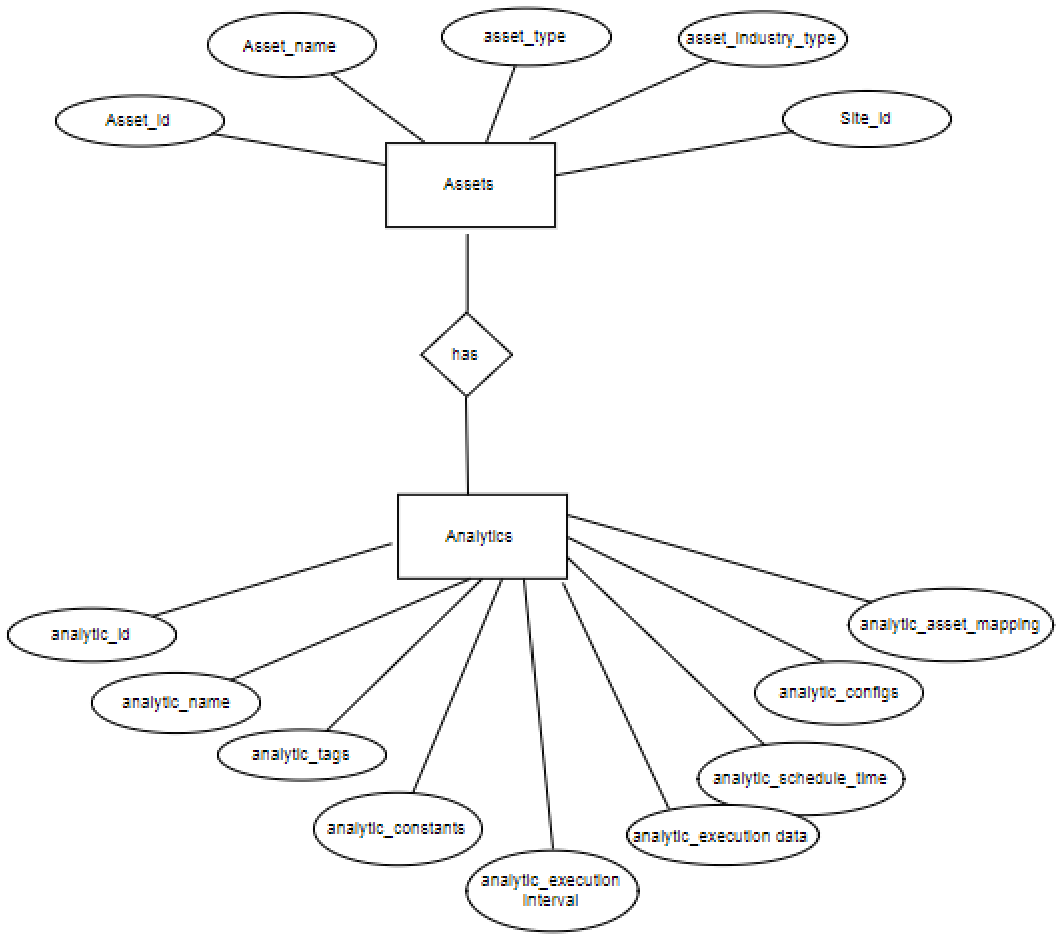
Proficy-Historian:
A type of data storage which stores real time sensor data and provide this data to analytic on demand.

Grafana:
An intelligent tool for visualization of data in various formats, Can be easily used by non-technical management leaders and managers.

Entity–relationship model (ER model)

In software engineering is an abstract way to describe a database. Describing a database usually starts with a relational database, which stores data in tables. An ERD is a logical representation of an organization's data, and consists of three primary components:

- Entities
- Attributes
- Relationships

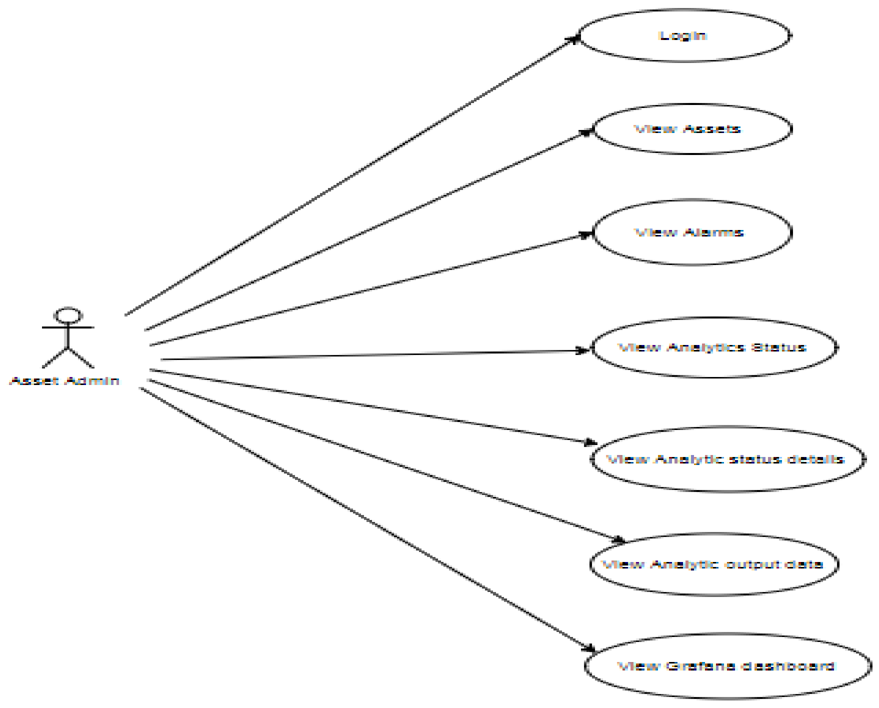


USE CASE DIAGRAM

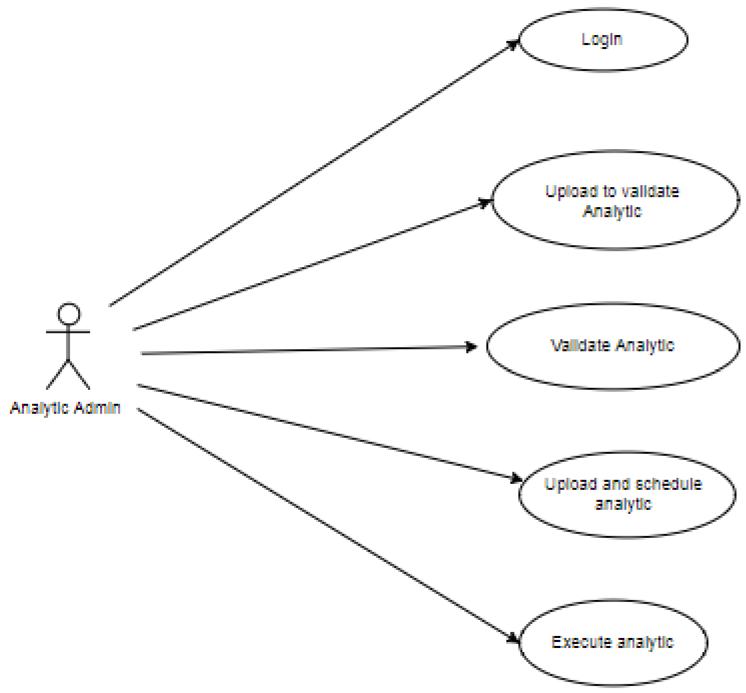
A **use case** diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well.Use case diagrams depict:

1. Use cases
2. Actors
3. Associations
4. System boundary boxes (optional)
5. Packages (optional)

Asset Admin



Analytic Admin



ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency.

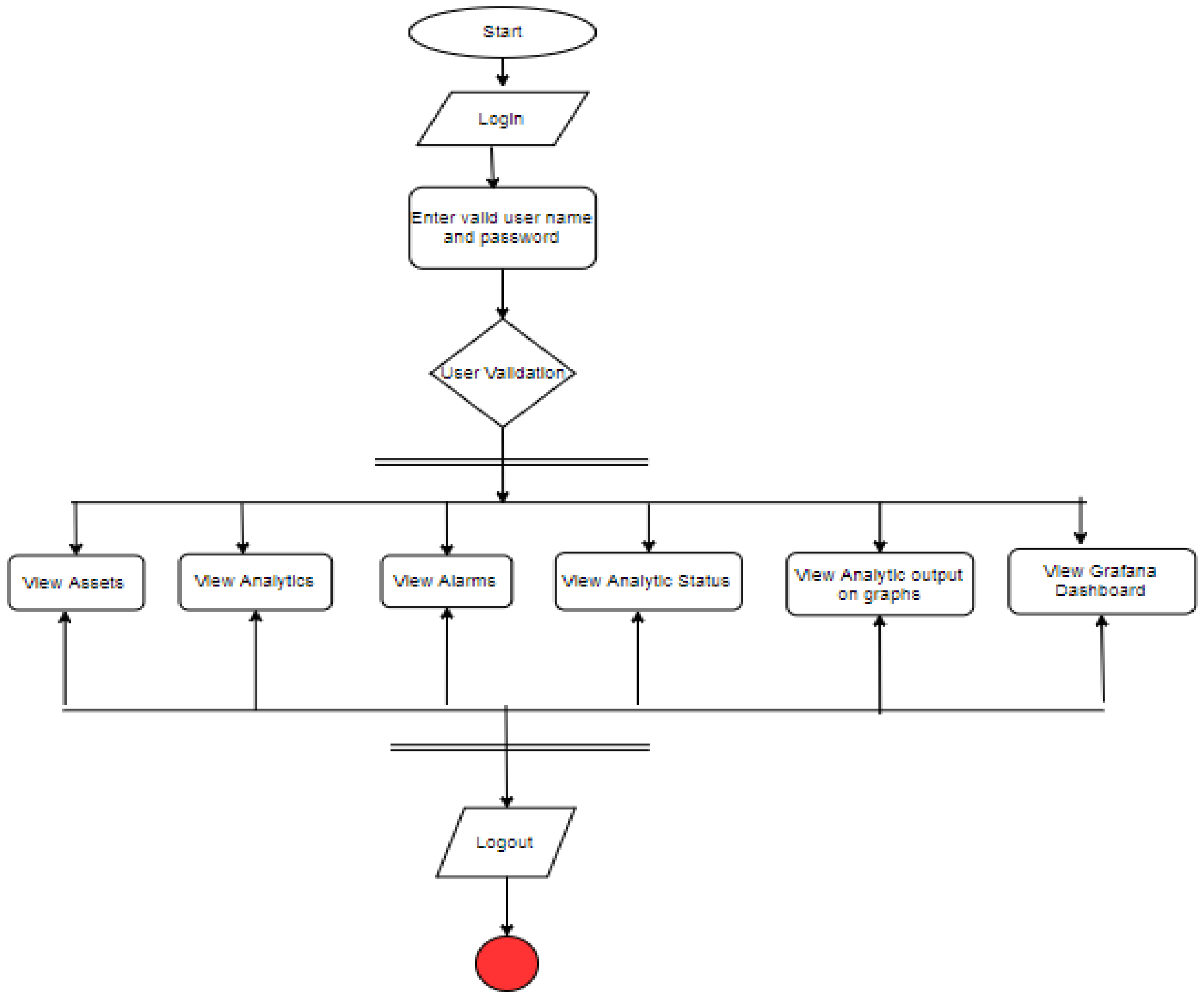
In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

Activity diagrams are constructed from a limited number of shapes, connected with arrows. The most important shape types:

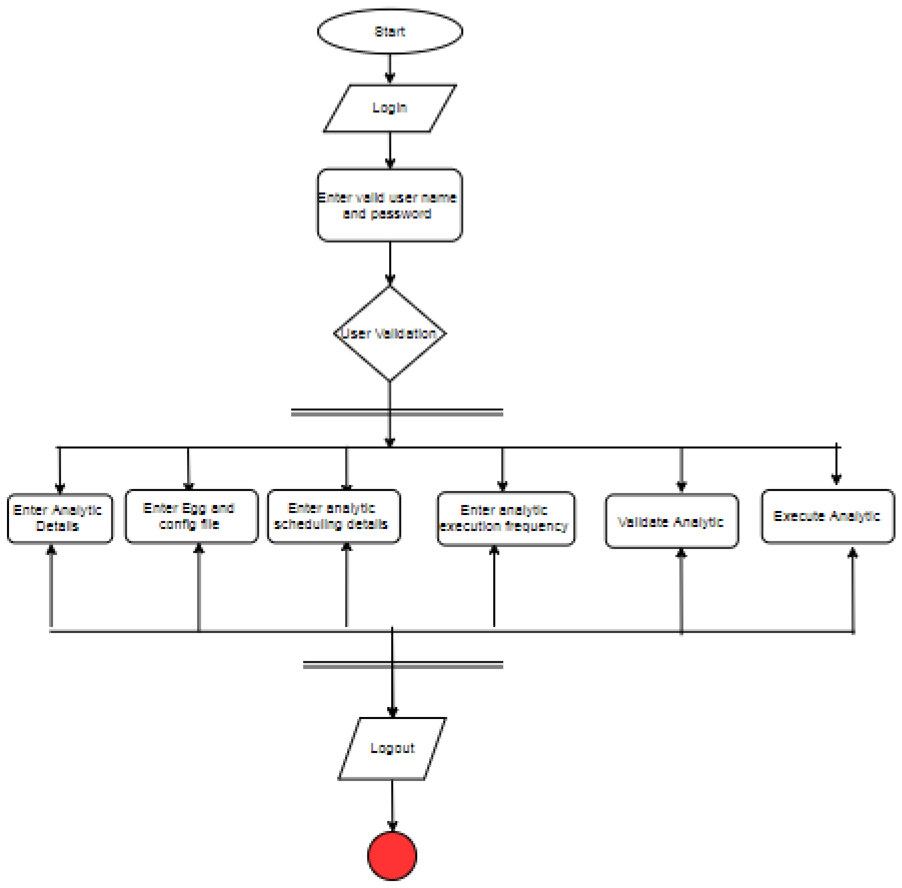
- rounded rectangles represent actions;
- diamonds represent decisions;
- bars represent the start (split) or end (join) of concurrent activities;
- a black circle represents the start (initial state) of the workflow;
- An encircled black circle represents the end (final state).

Arrows run from the start towards the end and represent the order in which activities happen. Hence they can be regarded as a form of flowchart.

Asset Admin


Start
Login
Enter valid user name and password
User Validation
View Assets
View Analytics
View Alarms
View Analytic Status
View Analytic output on graphs
View Grafana Dashboard
Logout

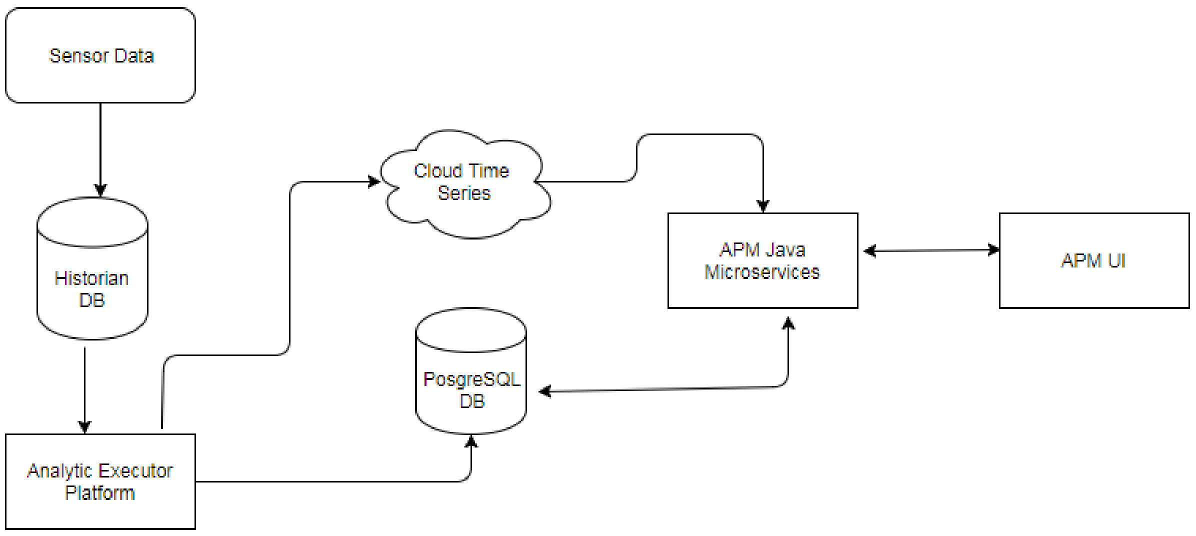
Analytic Admin


Start
Login
Enter valid user name and password
User Validation
Enter Analytic Details
Enter Egg and config file
Enter analytic scheduling details
Enter analytic execution frequency
Validate Analytic
Execute Analytic
Logout

Data Flow Diagram (DFD):

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system. It differs from the flowchart as it shows the data flow instead of the control flow of the program. A data flow diagram can also be used for the visualization of data processing (structured design). Data flow diagrams were invented by Larry Constantine, the original developer of structured design, based on Martin and Estrin's "data flow graph" model of computation.

A DFD shows what kinds of data will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of processes, or information about whether processes will operate in sequence or in parallel



PROJECT SCHEDULE

PROJECT SCHEDULE GANTT CHART

A Gantt chart is a horizontal bar chart used in project management as a tool for graphically representing the schedule of a set of specific activities or tasks. The horizontal bars indicate the length of time allocated to each activity, so the x-axis of a Gantt chart is subdivided into equal units of time, e.g., days, weeks, months. The y-axis of a Gantt chart, on the other hand, simply lists all the activities or tasks being monitored by the Gantt chart. A simple look at a Gantt chart should enable its user to determine which tasks take the longest time to complete, which tasks are overlapping with each other, etc.

A Gantt chart indicates the following:

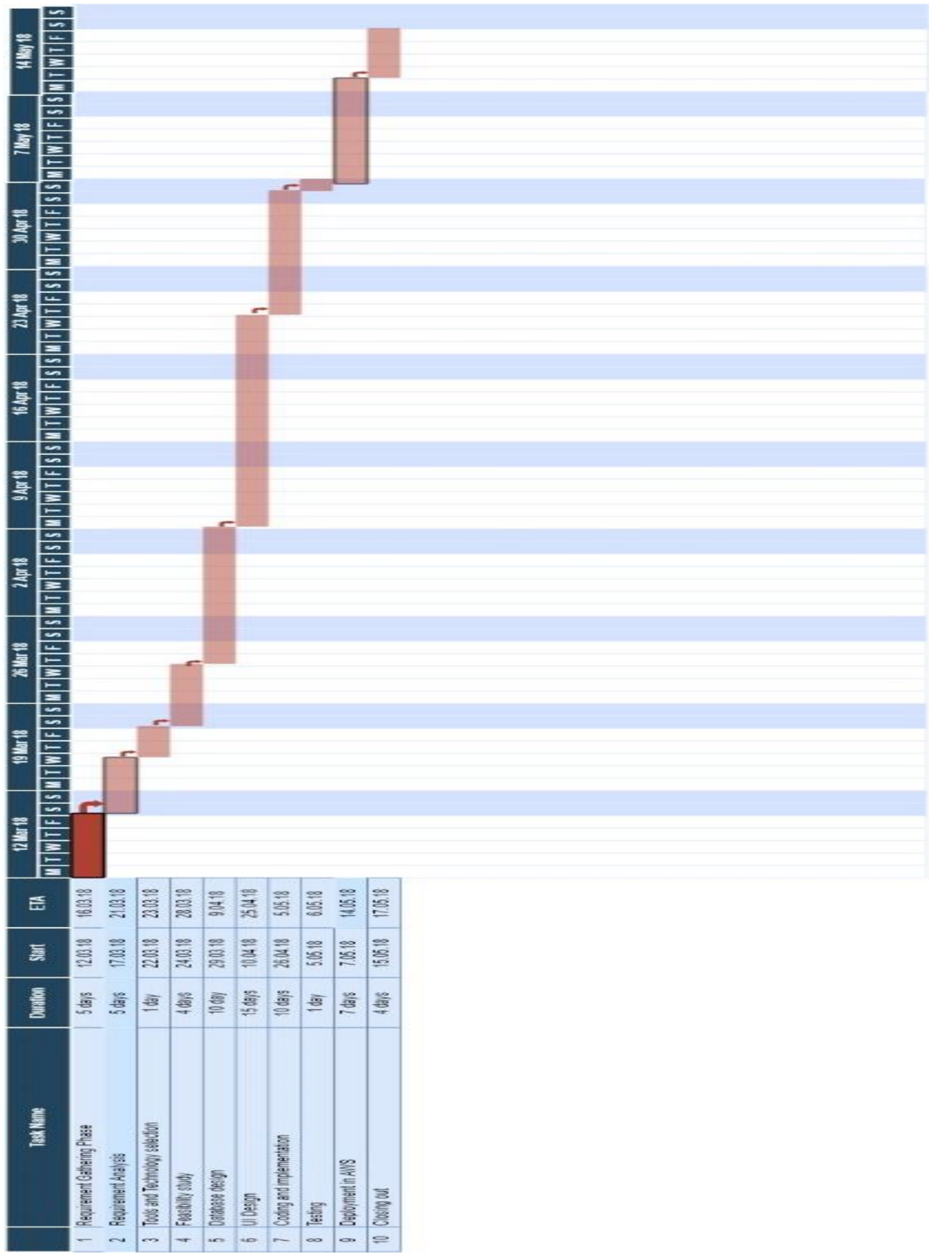
- Durations and timelines of the listed activities;
- The target and actual completion dates of the activities;
- The cost of each activity;
- The person or group of persons responsible for each activity;
- Milestones in the progress of the project.

Symbols Used

Since a Gantt chart is a graphical tool, it employs symbols to represent variety of information about a project. These symbols include:

- The task bar, which is the horizontal bar used to indicate the duration of each activity in the Gantt chart;
- The milestone marker, which denotes a major turning point in the project such as the release of an approved budget or the launching of a new product;
- The link line, which shows the relationship between two tasks, such as the fact that one activity can only begin after another one is complete.

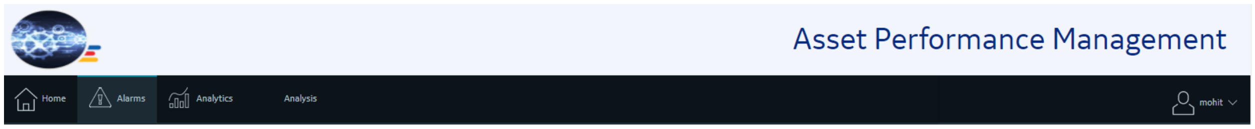
GANTT CHART DIAGRAM



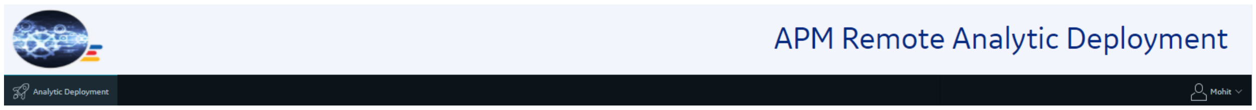
	Task Name	Duration	Start	ETA
1	Requirement Gathering Phase	5 days	12.03.18	16.03.18
2	Requirement Analysis	5 days	17.03.18	21.03.18
3	Tools and Technology selection	1 day	22.03.18	23.03.18
4	Feasibility study	4 days	24.03.18	28.03.18
5	Database design	10 day	29.03.18	9.04.18
6	UI Design	15 days	10.04.18	25.04.18
7	Coding and implementation	10 days	26.04.18	5.05.18
8	Testing	1 day	5.05.18	6.05.18
9	Deployment in AWS	7 days	7.05.18	14.05.18
10	Closing out	4 days	15.05.18	17.05.18

IMPLEMENTATION

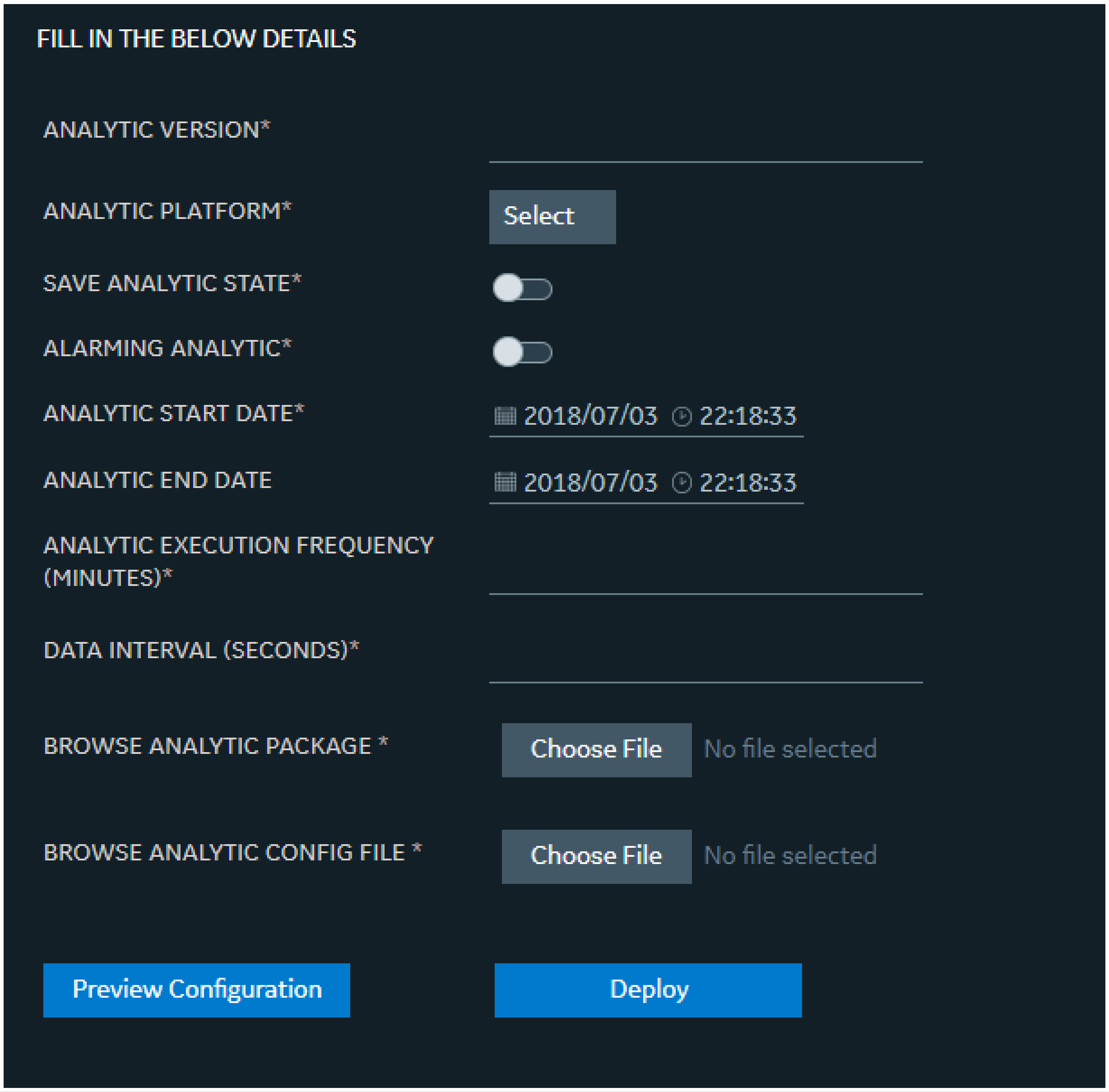
APM Asset Admin UI Header



APM Remote Analytic Deployment UI Header



Form for getting data about analytic



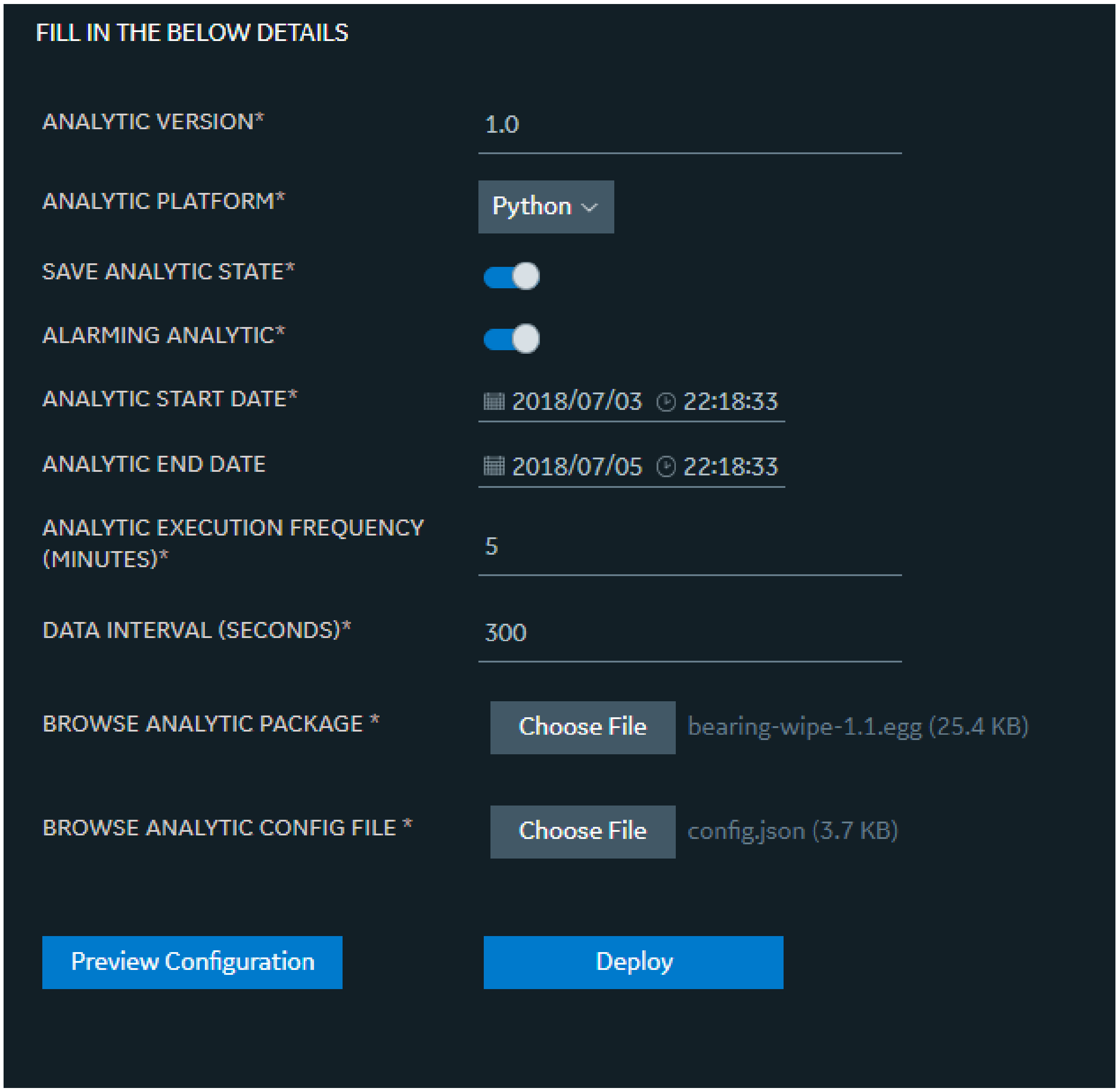
FILL IN THE BELOW DETAILS

ANALYTIC VERSION*	
ANALYTIC PLATFORM*	Select
SAVE ANALYTIC STATE*	
ALARMING ANALYTIC*	
ANALYTIC START DATE*	2018/07/03 22:18:33
ANALYTIC END DATE	2018/07/03 22:18:33
ANALYTIC EXECUTION FREQUENCY (MINUTES)*	
DATA INTERVAL (SECONDS)*	
BROWSE ANALYTIC PACKAGE *	Choose File No file selected
BROWSE ANALYTIC CONFIG FILE *	Choose File No file selected

Preview Configuration

Deploy

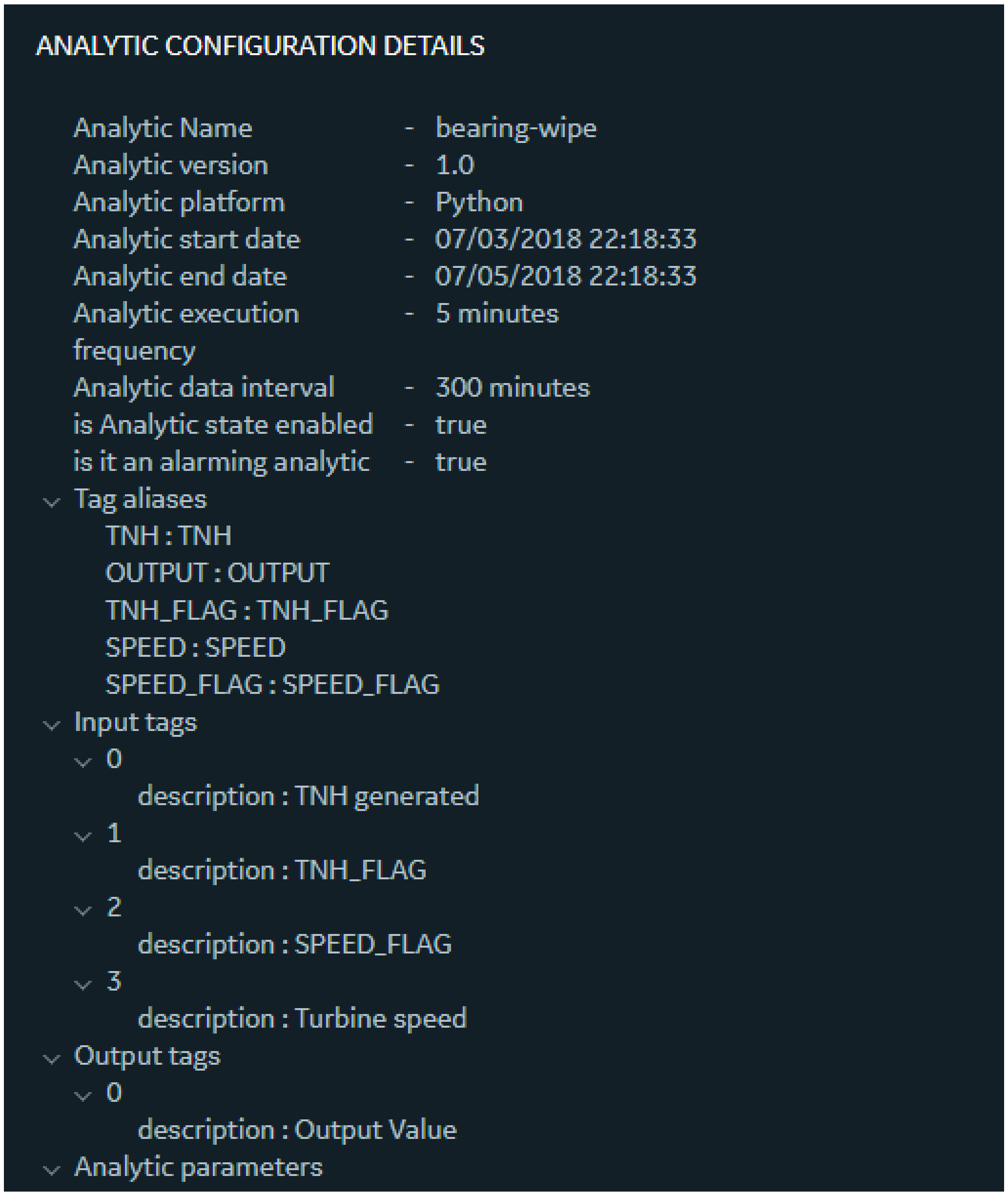
Sample data for analytic



FILL IN THE BELOW DETAILS

ANALYTIC VERSION*	1.0
ANALYTIC PLATFORM*	Python
SAVE ANALYTIC STATE*	✓
ALARMING ANALYTIC*	✓
ANALYTIC START DATE*	2018/07/03 22:18:33
ANALYTIC END DATE	2018/07/05 22:18:33
ANALYTIC EXECUTION FREQUENCY (MINUTES)*	5
DATA INTERVAL (SECONDS)*	300
BROWSE ANALYTIC PACKAGE *	Choose File bearing-wipe-1.1.egg (25.4 KB)
BROWSE ANALYTIC CONFIG FILE *	Choose File config.json (3.7 KB)

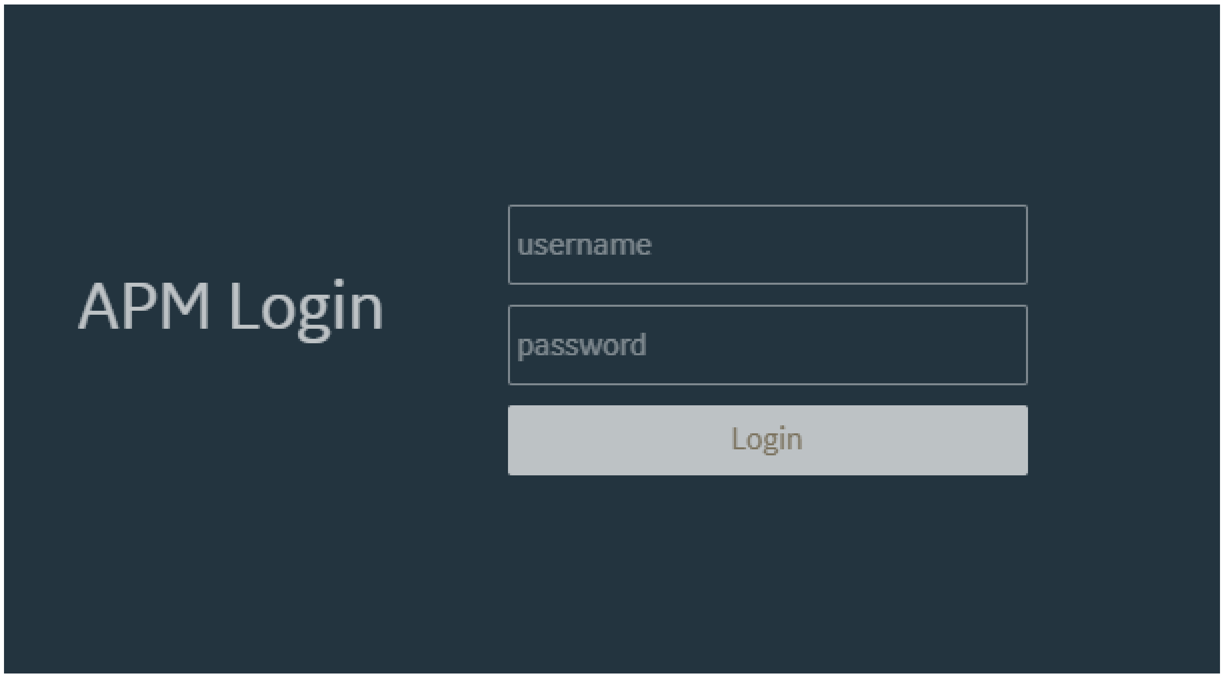
Configuration preview



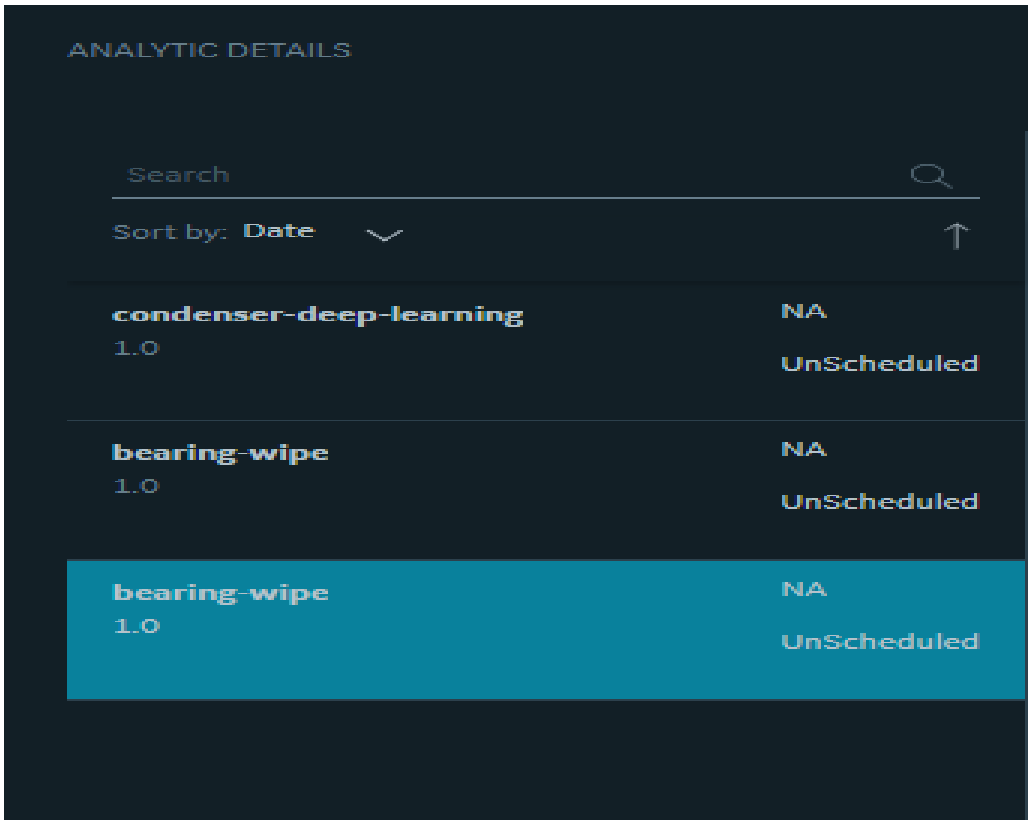
ANALYTIC CONFIGURATION DETAILS

```
    Analytic Name               -  bearing-wipe
    Analytic version            -  1.0
    Analytic platform           -  Python
    Analytic start date         -  07/03/2018 22:18:33
    Analytic end date           -  07/05/2018 22:18:33
    Analytic execution          -  5 minutes
    frequency
    Analytic data interval      -  300 minutes
    is Analytic state enabled   -  true
    is it an alarming analytic  -  true
  ⌄ Tag aliases
       TNH : TNH
       OUTPUT : OUTPUT
       TNH_FLAG : TNH_FLAG
       SPEED : SPEED
       SPEED_FLAG : SPEED_FLAG
  ⌄ Input tags
     ⌄ 0
          description : TNH generated
     ⌄ 1
          description : TNH_FLAG
     ⌄ 2
          description : SPEED_FLAG
     ⌄ 3
          description : Turbine speed
  ⌄ Output tags
     ⌄ 0
          description : Output Value
  ⌄ Analytic parameters
```

APM Login Page



Analytic Status Details

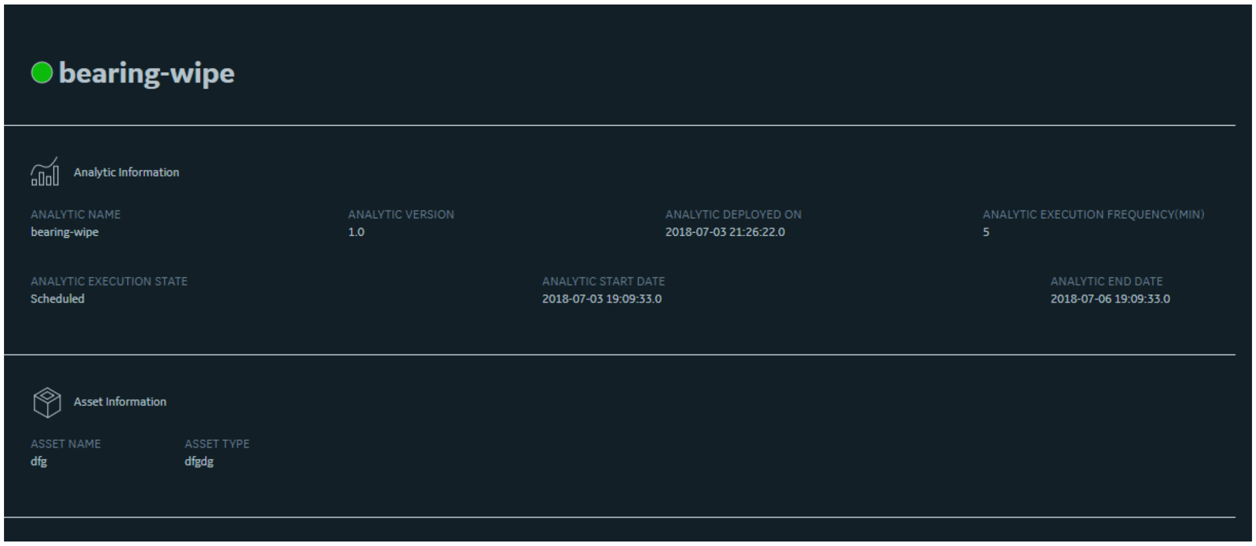


condenser-deep-learning
1.0
NA
UnScheduled

bearing-wipe
1.0
NA
UnScheduled

bearing-wipe
1.0
NA
UnScheduled

Analytic execution details



● bearing-wipe

Analytic Information

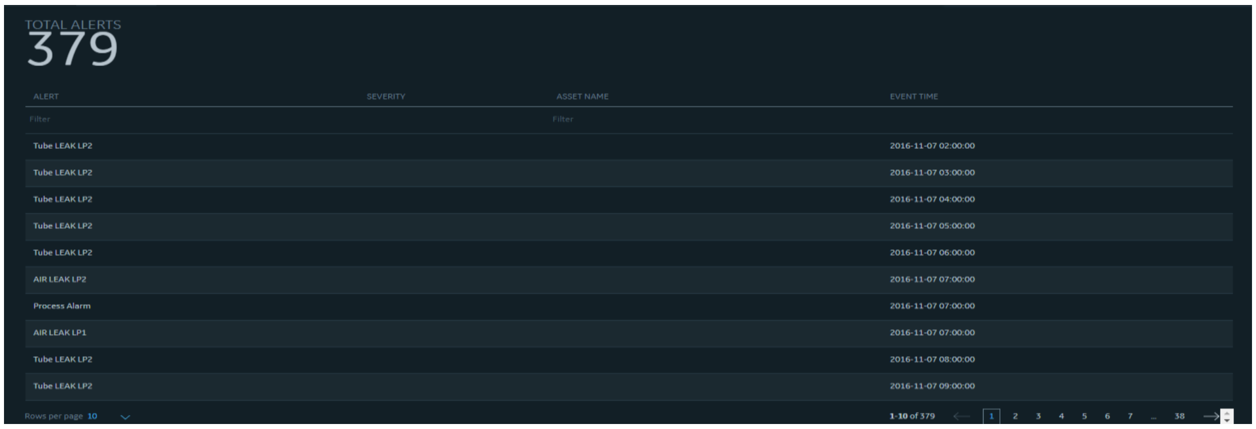
ANALYTIC NAME	ANALYTIC VERSION	ANALYTIC DEPLOYED ON	ANALYTIC EXECUTION FREQUENCY(MIN)
bearing-wipe	1.0	2018-07-03 21:26:22.0	5

ANALYTIC EXECUTION STATE	ANALYTIC START DATE	ANALYTIC END DATE
Scheduled	2018-07-03 19:09:33.0	2018-07-06 19:09:33.0

Asset Information

ASSET NAME	ASSET TYPE
dfg	dfgdg

Analytic Alarm Details

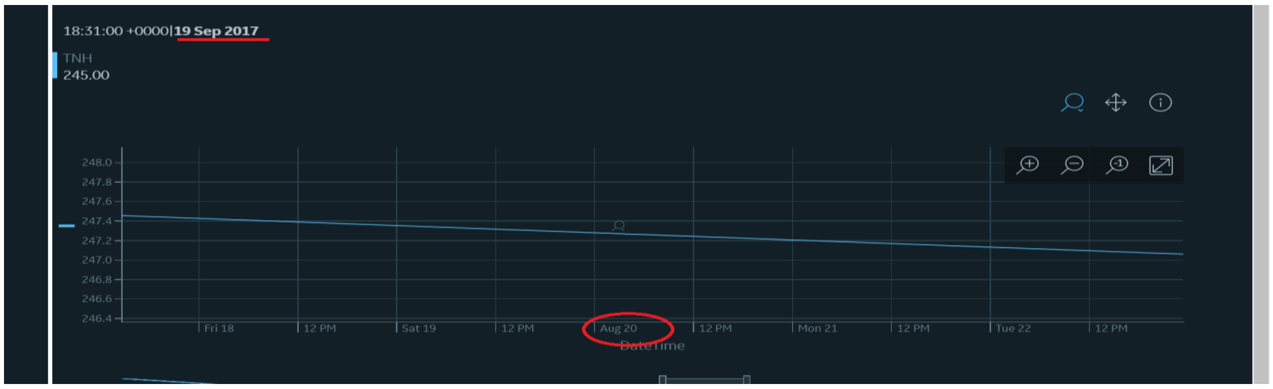


TOTAL ALERTS
379

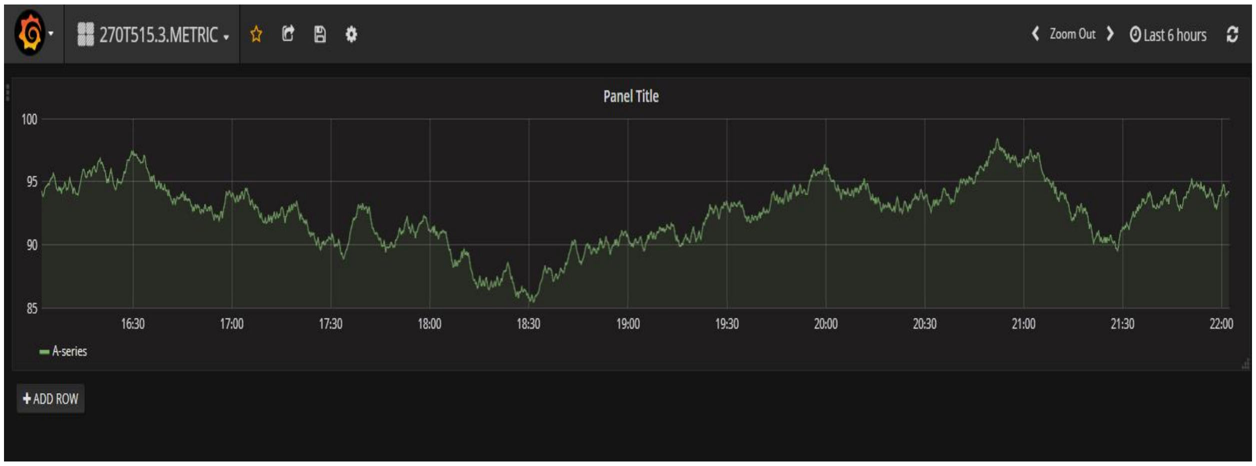
ALERT	SEVERITY	ASSET NAME	EVENT TIME
Filter		Filter	
Tube LEAK LP2			2016-11-07 02:00:00
Tube LEAK LP2			2016-11-07 03:00:00
Tube LEAK LP2			2016-11-07 04:00:00
Tube LEAK LP2			2016-11-07 05:00:00
Tube LEAK LP2			2016-11-07 06:00:00
AIR LEAK LP2			2016-11-07 07:00:00
Process Alarm			2016-11-07 07:00:00
AIR LEAK LP1			2016-11-07 07:00:00
Tube LEAK LP2			2016-11-07 08:00:00
Tube LEAK LP2			2016-11-07 09:00:00

Rows per page 10 1-10 of 379 1 2 3 4 5 6 7 ... 38

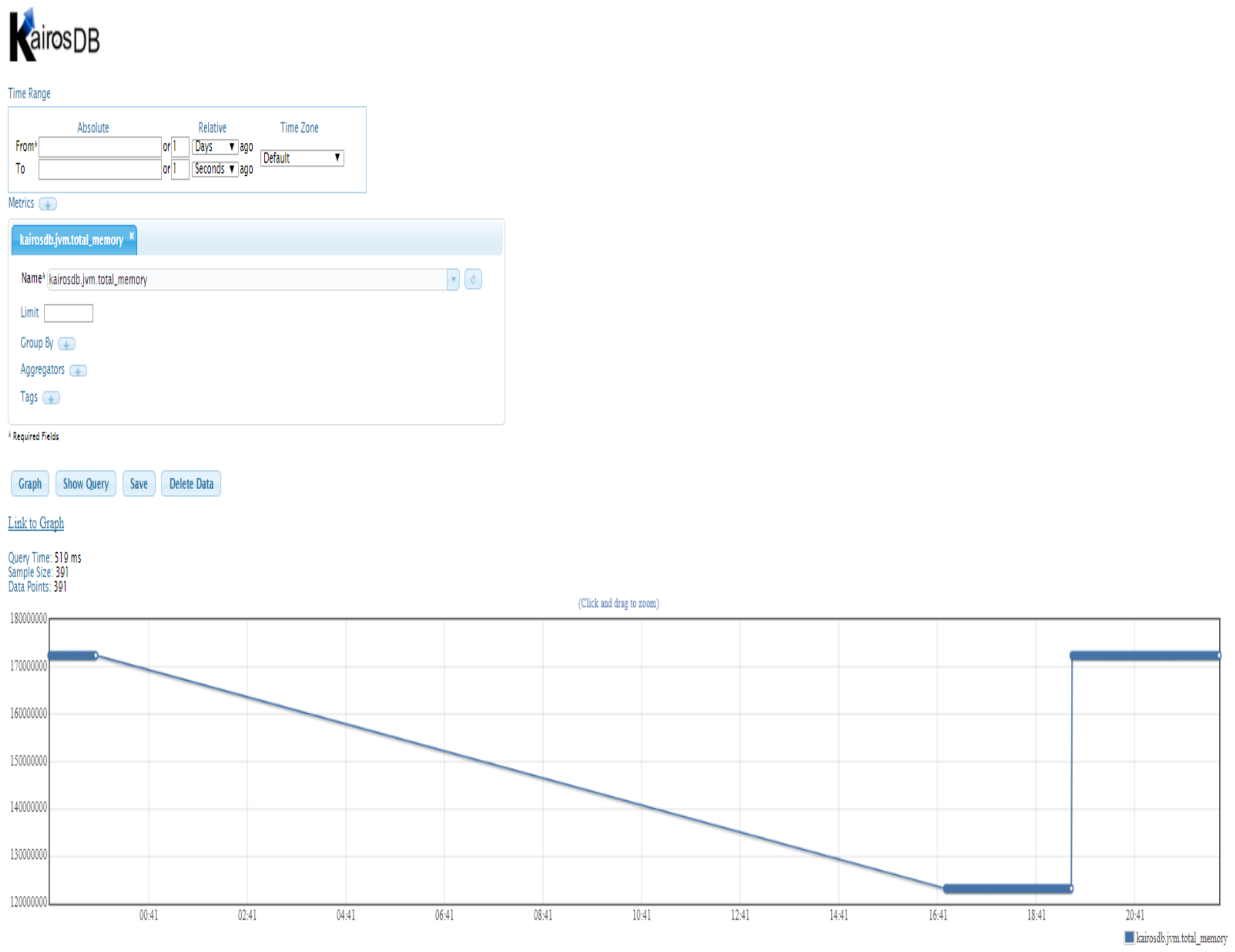
Analytic execution grpah



Grafana Analytic Graph

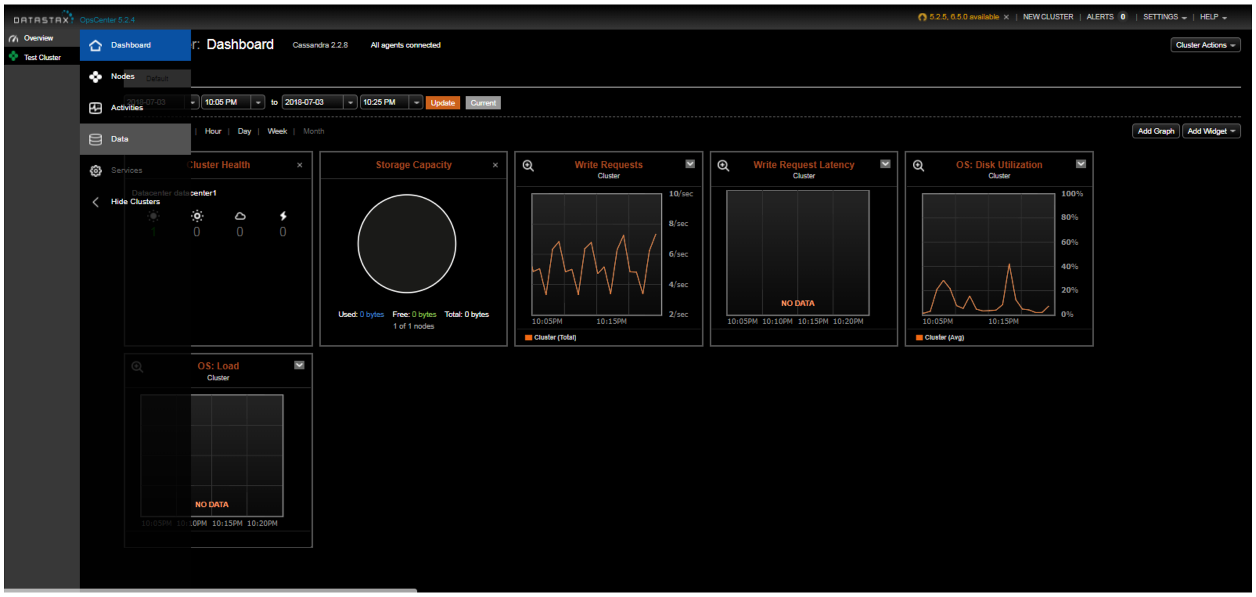


KairosDB interface with graph example

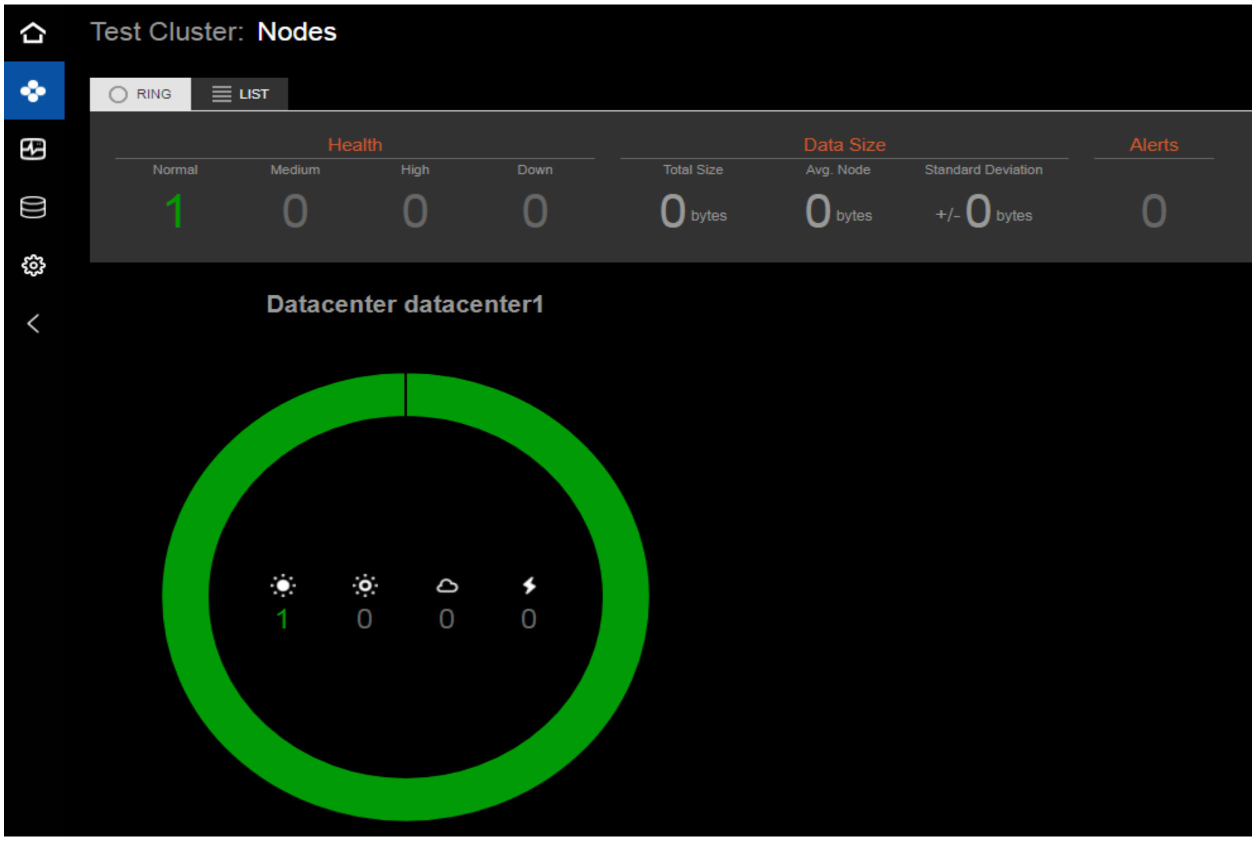


Query Time: 519 ms
Sample Size: 391
Data Points: 391

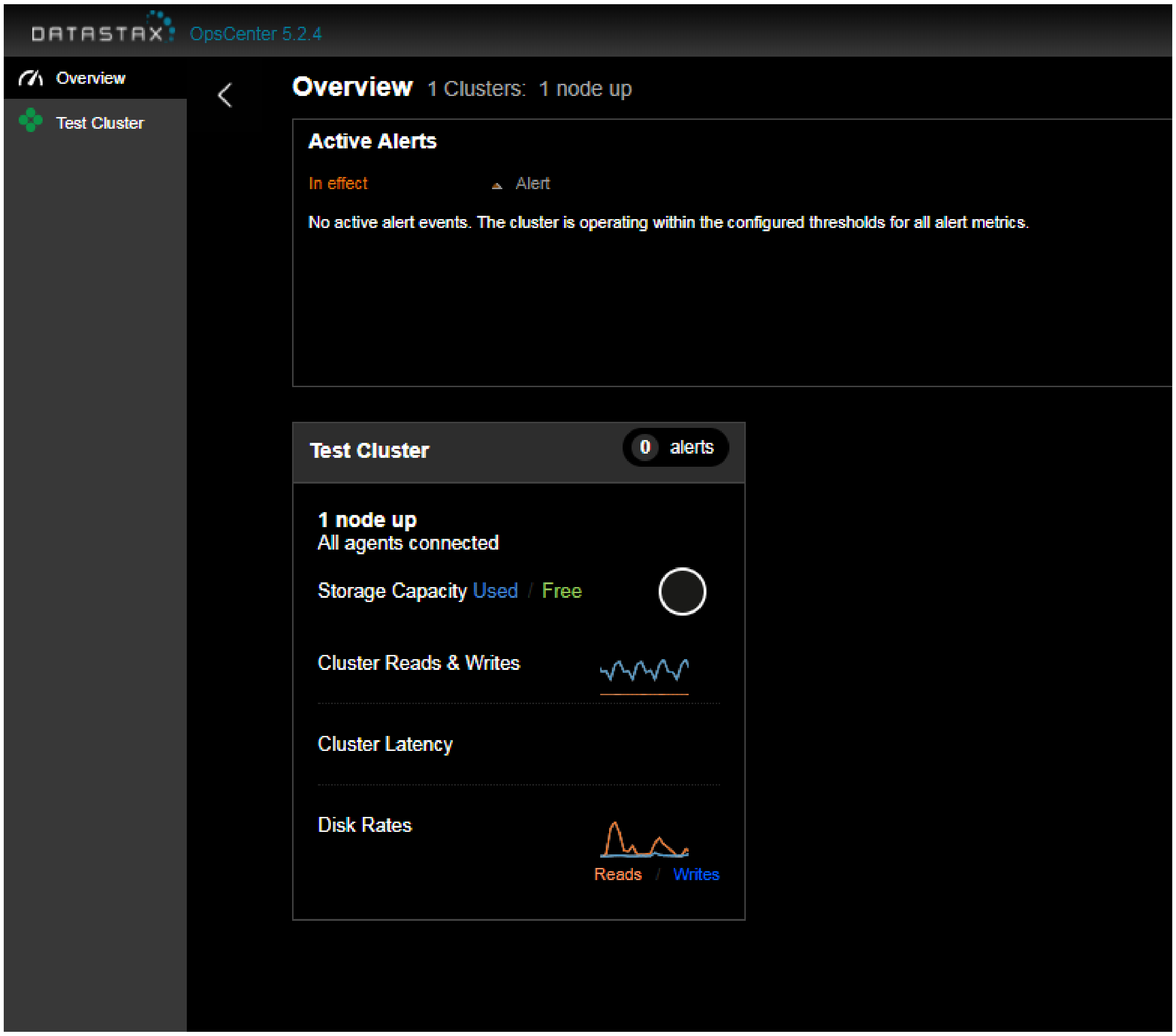
Datastax Opscenter Dashboard



OpsCenter Cluster View



OpsCenter Test Cluster



No active alert events. The cluster is operating within the configured thresholds for all alert metrics.

TESTING

TESTING

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation.

Test techniques include, but are not limited to, the process of executing a program or application with the intent of finding software bugs (errors or other defects).

Software testing can be stated as the process of validating and verifying that a computer program/application/product:

- meets the requirements that guided its design and development,
- works as expected,
- can be implemented with the same characteristics,

- White box testing:

In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs.

- Black box testing:

Black box testing have little or no regard to the internal logical structure of the system, it only examines the fundamental aspect of the system. It makes sure that input is properly accepted and output is correctly produced.

- Function testing:

Functional tests involve exercising the code with nominal input values which gives the expected results and boundary values are known.

- Performance testing:

Performance tests are designed to verify response time. If the wrong data is entered then the system does not allow it and calculations are not performed.

- Integration Testing:

Integration testing is critical to ensure the functional correctness of the integrated system.
Integration testing can be divided into two categories. Integration testing is often the most time consuming and expensive part of testing.

- Unit testing:

The first test in the development process is the unit test. The source code is normally divided into modules, which in turn are divided into smaller units called units. These units have specific behavior. The test done on these units of code is called unit test. Unit test depends upon the language on which the project is developed. Unit tests ensure that each unique path of the project performs accurately to the documented specifications and contains clearly defined inputs and expected results.

- System testing:

Several modules constitute a project. If the project is long-term project, several developers write the modules. Once all the modules are integrated, several errors may arise. The testing done at this stage is called system test. System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

DEPLOYMENT

DEPLOYMENT

Software deployment is all the activities that make a software system available for use. The general deployment process consists of several interrelated activities with possible transitions between them. These activities can occur at the producer site or at the consumer site or both. Because every software system is unique, the precise processes or procedures within each activity can hardly be defined. Therefore, "deployment" should be interpreted as a *general process* that has to be customized according to specific requirements or characteristics.

In my Project I used AWS platform for deploying webapp. There are various benefits of deploying project in cloud environment: -

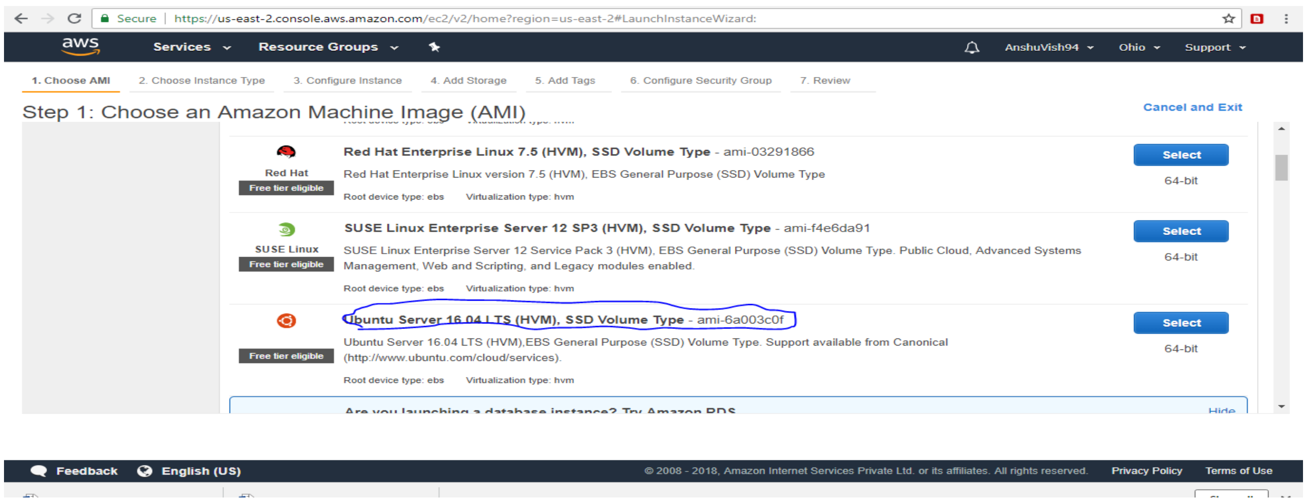
- Lower cost
- Deploy project faster
- Scale as needed
- Lower maintenance cost

Used Winscp, putty and query browser for successful deployment of the project. Below are the steps used in AWS for deploying application

Create an EC2 instance –

1. Create account in AWS (Amazon web service) using personal information and credit card details

- Choose required AMI



Step 1: Choose an Amazon Machine Image (AMI)

Red Hat Enterprise Linux 7.5 (HVM), SSD Volume Type - ami-03291866
Red Hat Enterprise Linux version 7.5 (HVM), EBS General Purpose (SSD) Volume Type
Root device type: ebs Virtualization type: hvm
Red Hat — Free tier eligible — 64-bit

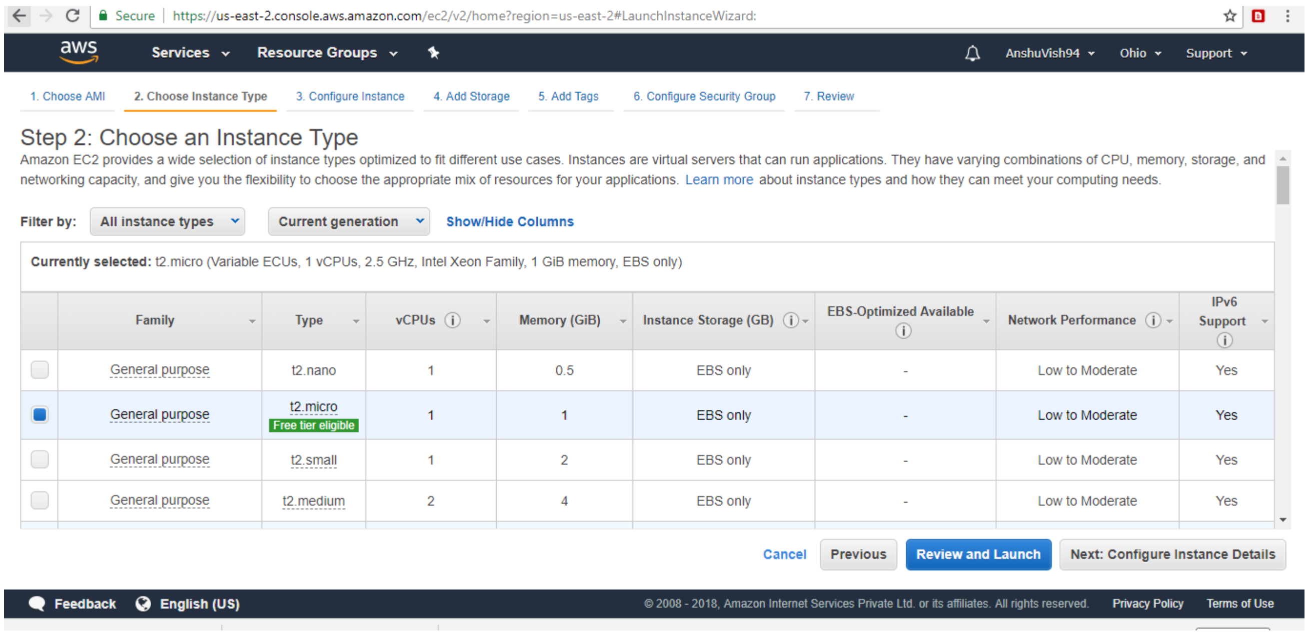
SUSE Linux Enterprise Server 12 SP3 (HVM), SSD Volume Type - ami-f4e6da91
SUSE Linux Enterprise Server 12 Service Pack 3 (HVM), EBS General Purpose (SSD) Volume Type. Public Cloud, Advanced Systems Management, Web and Scripting, and Legacy modules enabled.
Root device type: ebs Virtualization type: hvm
SUSE Linux — Free tier eligible — 64-bit

Ubuntu Server 16.04 LTS (HVM), SSD Volume Type - ami-6a003c0f
Ubuntu Server 16.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).
Root device type: ebs Virtualization type: hvm
Free tier eligible — 64-bit

Are you launching a database instance? Try Amazon RDS

© 2008 - 2018, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.

➢ Choose and instance type-


Secure | https://us-east-2.console.aws.amazon.com/ec2/v2/home?region=us-east-2#LaunchInstanceWizard:
aws Services Resource Groups AnshuVish94 Ohio Support
1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. Learn more about instance types and how they can meet your computing needs.

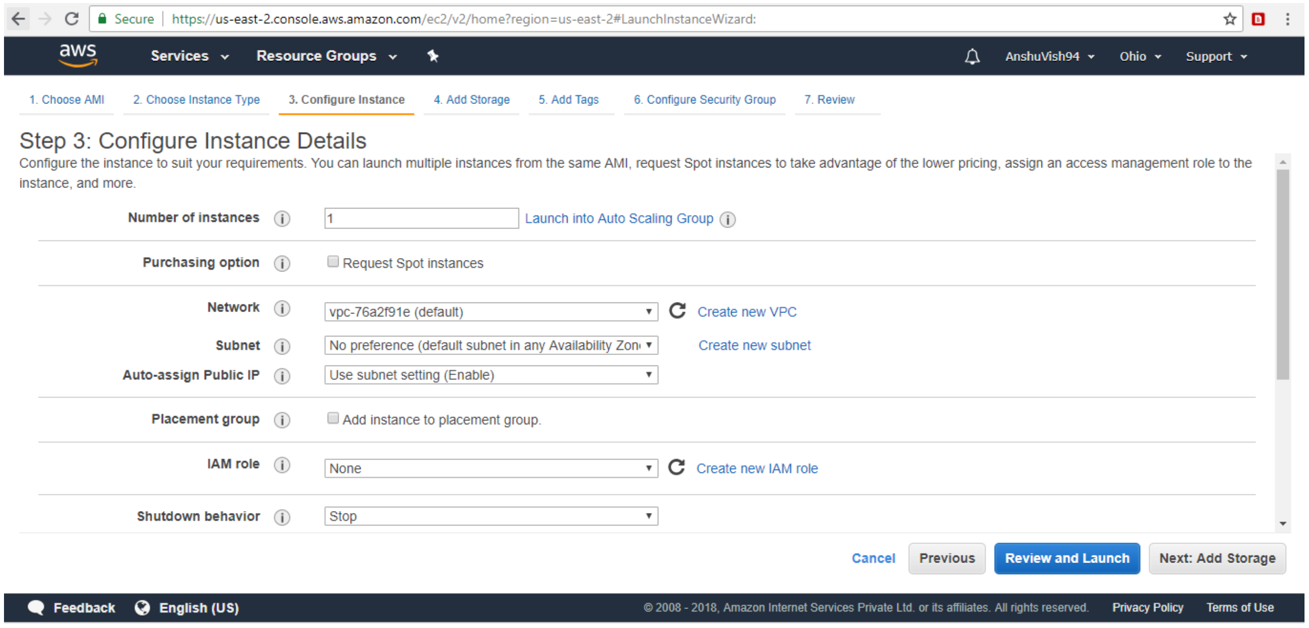
Filter by: All instance types | Current generation | Show/Hide Columns

Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance	IPv6 Support
☐	General purpose	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
■	General purpose	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate	Yes
☐	General purpose	t2.small	1	2	EBS only	-	Low to Moderate	Yes
☐	General purpose	t2.medium	2	4	EBS only	-	Low to Moderate	Yes

Cancel Previous Review and Launch Next: Configure Instance Details
Feedback English (US) © 2008 - 2018, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. Privacy Policy Terms of Use

➢ Configure instance details-


Secure | https://us-east-2.console.aws.amazon.com/ec2/v2/home?region=us-east-2#LaunchInstanceWizard:
aws Services Resource Groups AnshuVish94 Ohio Support
1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

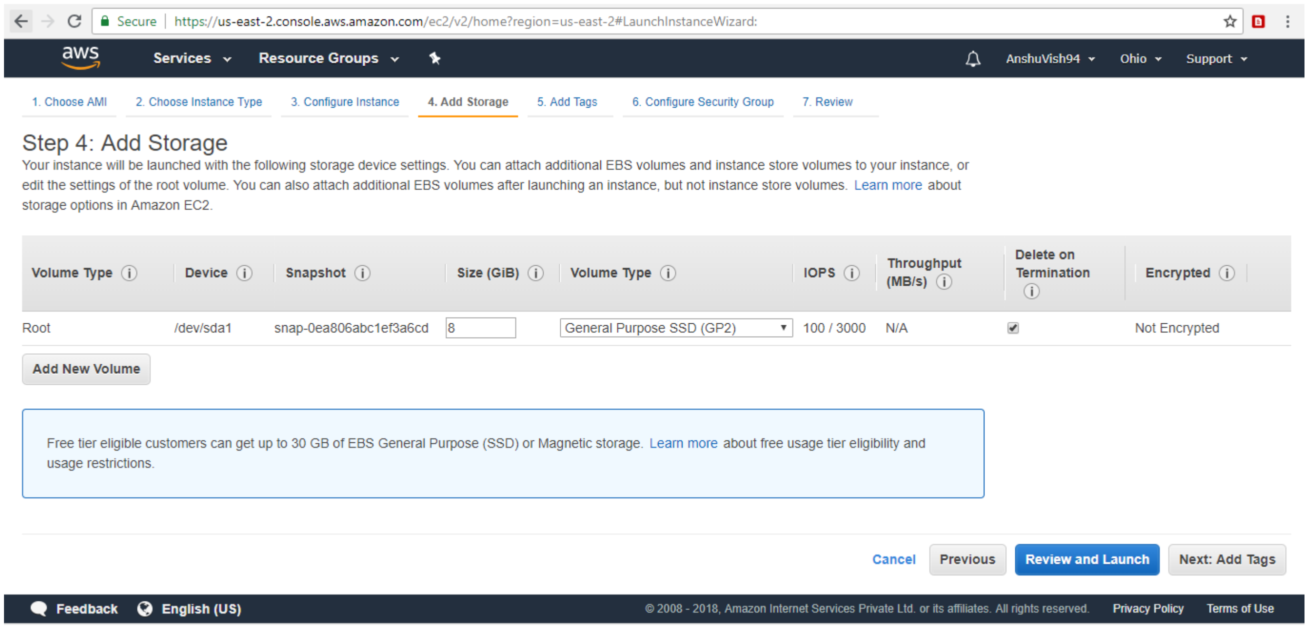
Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.

Number of instances	1 — Launch into Auto Scaling Group
Purchasing option	☐ Request Spot instances
Network	vpc-76a2f91e (default) — Create new VPC
Subnet	No preference (default subnet in any Availability Zon — Create new subnet
Auto-assign Public IP	Use subnet setting (Enable)
Placement group	☐ Add instance to placement group.
IAM role	None — Create new IAM role
Shutdown behavior	Stop

Cancel Previous Review and Launch Next: Add Storage
Feedback English (US) © 2008 - 2018, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. Privacy Policy Terms of Use

- Add security group and storage after that launch the instance.


Secure | https://us-east-2.console.aws.amazon.com/ec2/v2/home?region=us-east-2#LaunchInstanceWizard:

Step 4: Add Storage

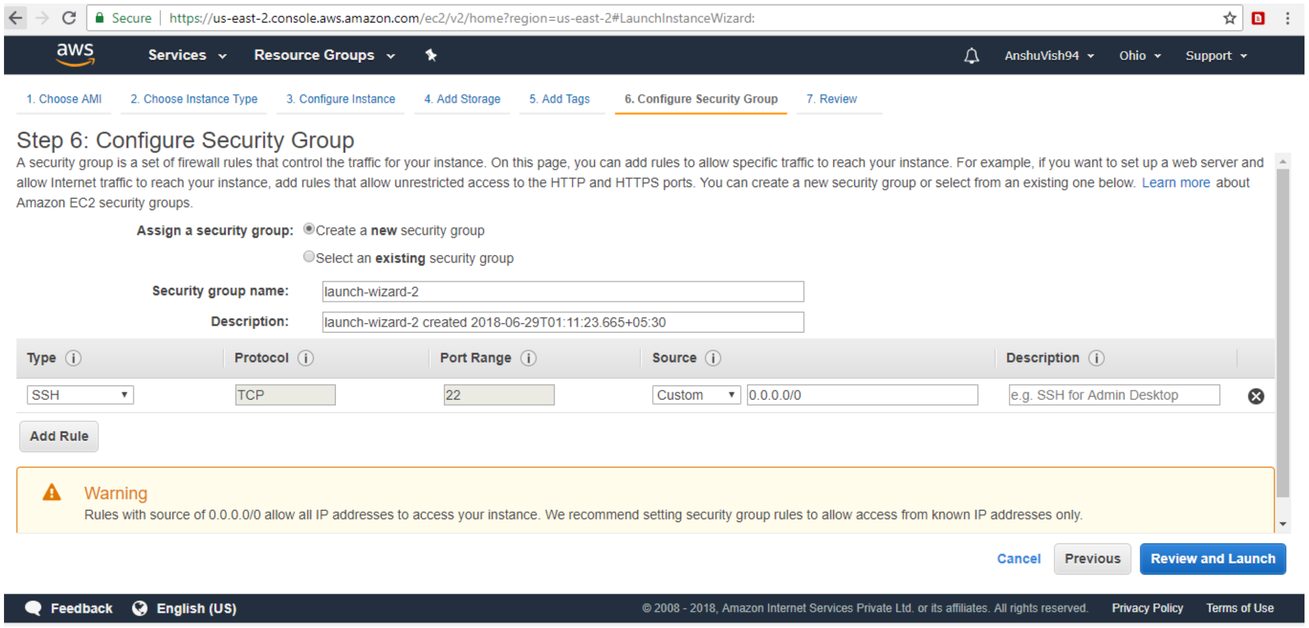
Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. Learn more about storage options in Amazon EC2.

Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Throughput (MB/s)	Delete on Termination	Encrypted
Root	/dev/sda1	snap-0ea806abc1ef3a6cd	8	General Purpose SSD (GP2)	100 / 3000	N/A	✓	Not Encrypted

Add New Volume

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. Learn more about free usage tier eligibility and usage restrictions.

© 2008 - 2018, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. Privacy Policy Terms of Use


Secure | https://us-east-2.console.aws.amazon.com/ec2/v2/home?region=us-east-2#LaunchInstanceWizard:

Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. Learn more about Amazon EC2 security groups.

Assign a security group: [✓ Create a **new** security group] Select an **existing** security group

Security group name: launch-wizard-2

Description: launch-wizard-2 created 2018-06-29T01:11:23.665+05:30

Type	Protocol	Port Range	Source		Description
SSH	TCP	22	Custom	0.0.0.0/0	e.g. SSH for Admin Desktop

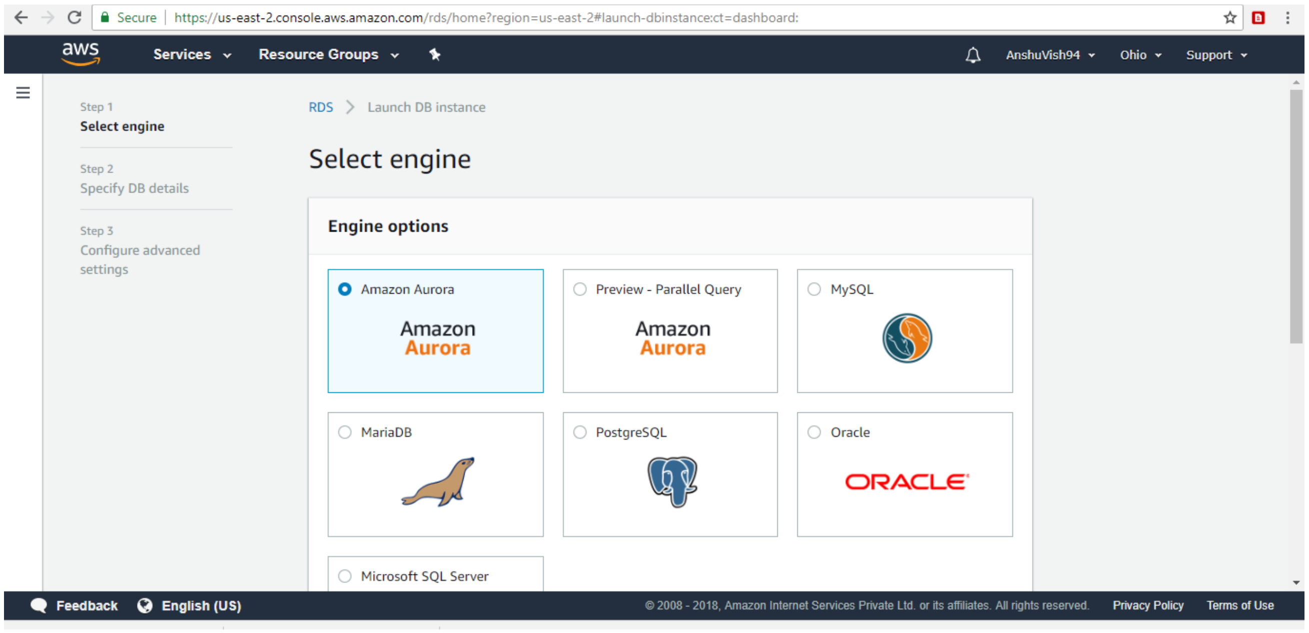
Add Rule

Warning
Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

© 2008 - 2018, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. Privacy Policy Terms of Use

2. Create RDS Instance-

➢ Select DB type



https://us-east-2.console.aws.amazon.com/rds/home?region=us-east-2#launch-dbinstance:ct=dashboard:

Step 1
Select engine

Step 2
Specify DB details

Step 3
Configure advanced settings

RDS > Launch DB instance

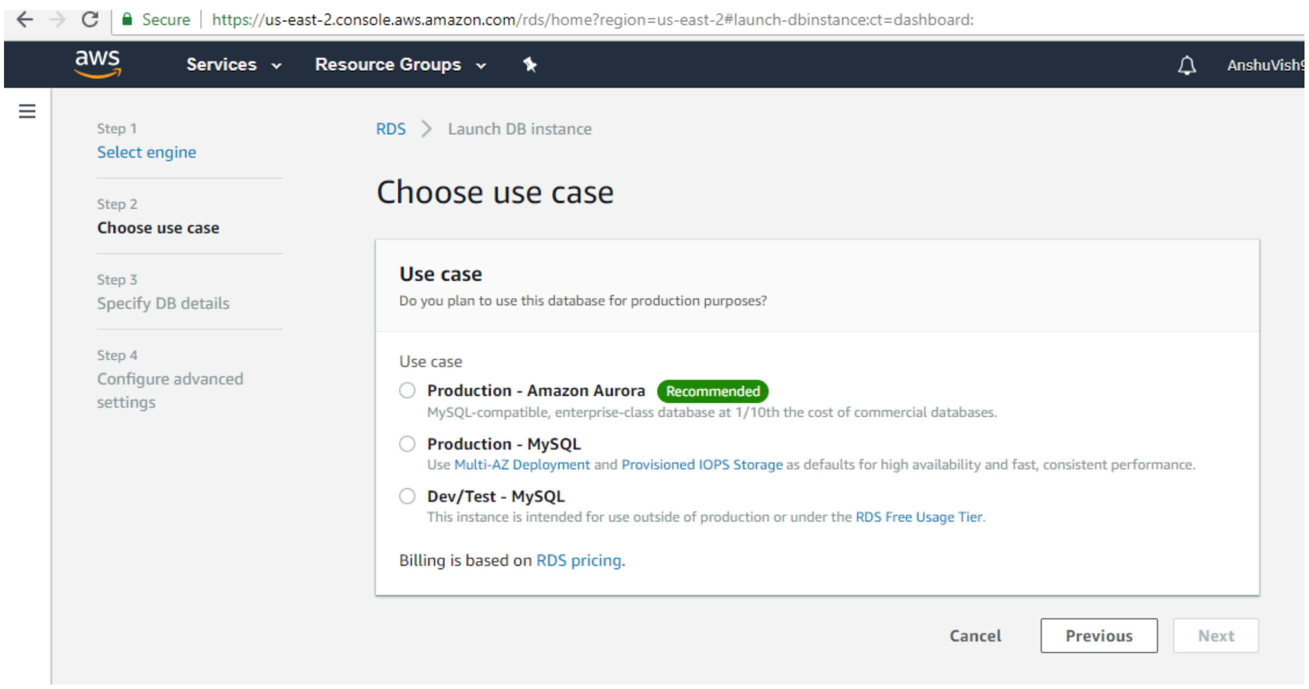
Select engine

Engine options

- (●) Amazon Aurora
- ( ) Preview - Parallel Query
- ( ) MySQL
- ( ) MariaDB
- ( ) PostgreSQL
- ( ) Oracle
- ( ) Microsoft SQL Server

Feedback
English (US)
© 2008 - 2018, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

➢ Create the Use case, which type of DB, you want to select



https://us-east-2.console.aws.amazon.com/rds/home?region=us-east-2#launch-dbinstance:ct=dashboard:

Step 1
Select engine

Step 2
Choose use case

Step 3
Specify DB details

Step 4
Configure advanced settings

RDS > Launch DB instance

Choose use case

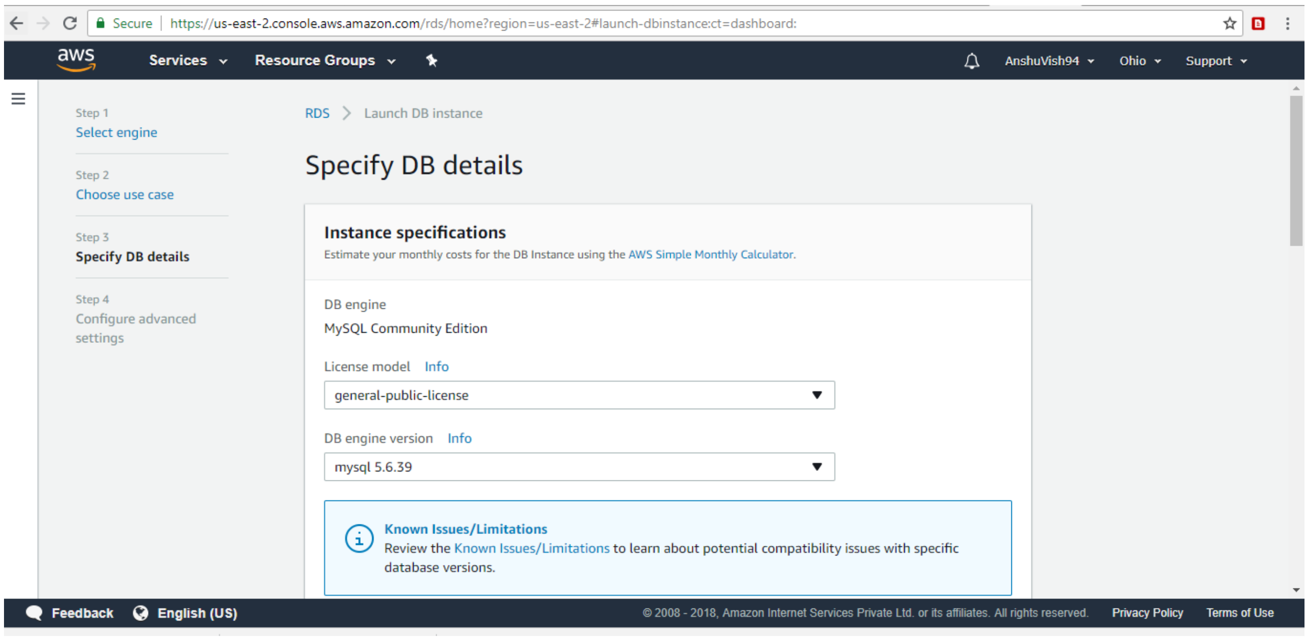
Use case
Do you plan to use this database for production purposes?

Use case

- ( ) **Production - Amazon Aurora** Recommended
  MySQL-compatible, enterprise-class database at 1/10th the cost of commercial databases.
- ( ) **Production - MySQL**
  Use Multi-AZ Deployment and Provisioned IOPS Storage as defaults for high availability and fast, consistent performance.
- ( ) **Dev/Test - MySQL**
  This instance is intended for use outside of production or under the RDS Free Usage Tier.

Billing is based on RDS pricing.

➢ Specify all the DB details and launch the instance.


Secure | https://us-east-2.console.aws.amazon.com/rds/home?region=us-east-2#launch-dbinstance:ct=dashboard:
aws
Services
Resource Groups
AnshuVish94
Ohio
Support
Feedback
English (US)
© 2008 - 2018, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

Step 1
Select engine

Step 2
Choose use case

Step 3
Specify DB details

Step 4
Configure advanced settings

RDS > Launch DB instance

Specify DB details

Instance specifications

Estimate your monthly costs for the DB Instance using the AWS Simple Monthly Calculator.

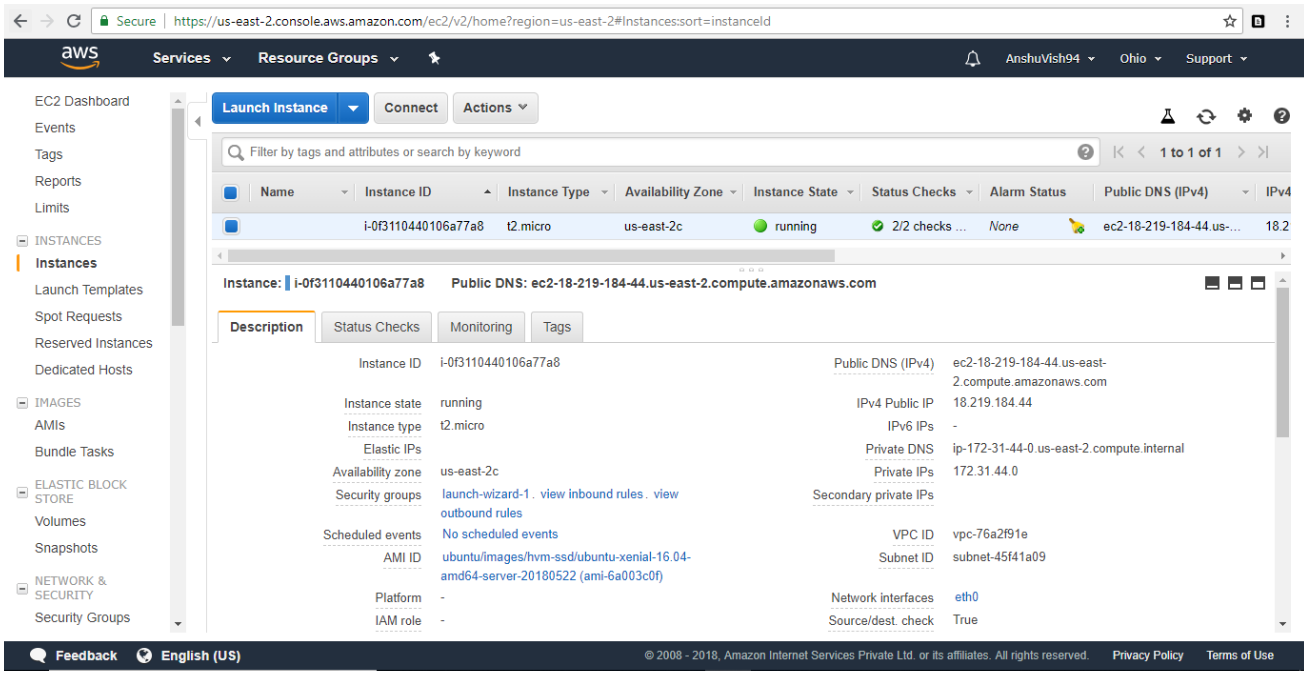
DB engine
MySQL Community Edition

License model Info
general-public-license

DB engine version Info
mysql 5.6.39

Known Issues/Limitations
Review the Known Issues/Limitations to learn about potential compatibility issues with specific database versions.

➢ EC2 Instance

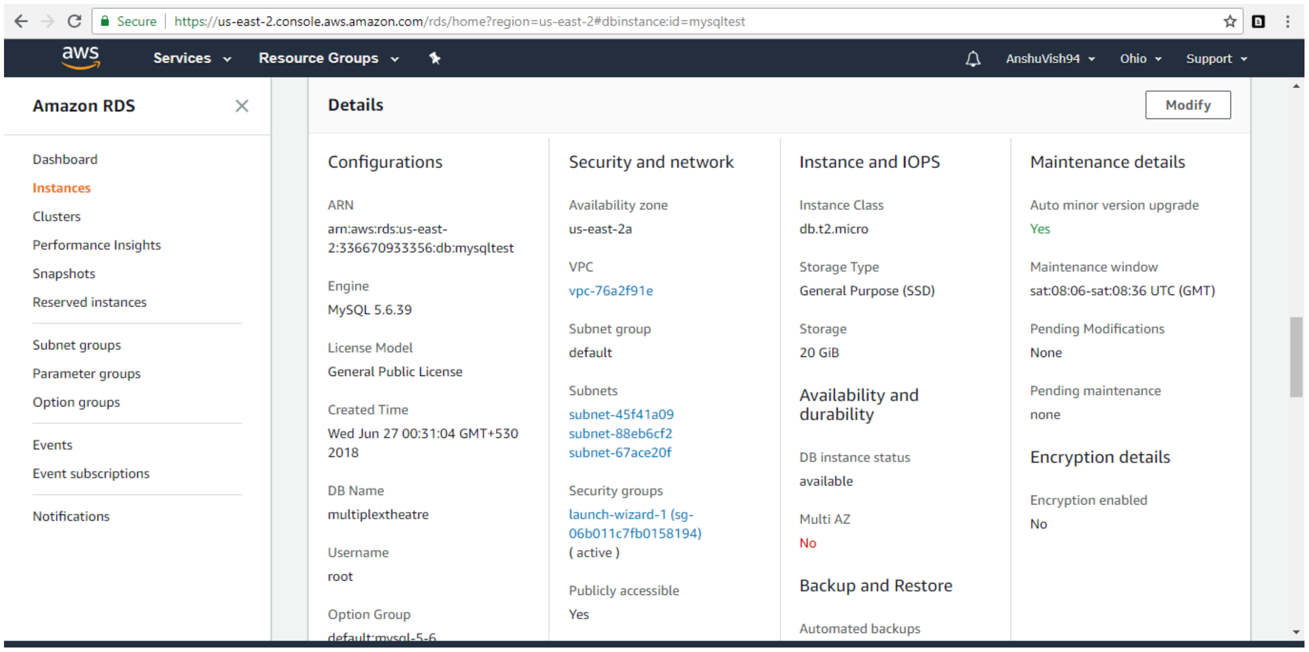


Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4
	i-0f3110440106a77a8	t2.micro	us-east-2c	running	2/2 checks ...	None	ec2-18-219-184-44.us-...	18.2

Instance: i-0f3110440106a77a8 Public DNS: ec2-18-219-184-44.us-east-2.compute.amazonaws.com

Instance ID	i-0f3110440106a77a8	Public DNS (IPv4)	ec2-18-219-184-44.us-east-2.compute.amazonaws.com
Instance state	running	IPv4 Public IP	18.219.184.44
Instance type	t2.micro	IPv6 IPs	-
Elastic IPs		Private DNS	ip-172-31-44-0.us-east-2.compute.internal
Availability zone	us-east-2c	Private IPs	172.31.44.0
Security groups	launch-wizard-1 . view inbound rules . view outbound rules	Secondary private IPs	
Scheduled events	No scheduled events	VPC ID	vpc-76a2f91e
AMI ID	ubuntu/images/hvm-ssd/ubuntu-xenial-16.04-amd64-server-20180522 (ami-6a003c0f)	Subnet ID	subnet-45f41a09
Platform	-	Network interfaces	eth0
IAM role	-	Source/dest. check	True

➢ RDS instance details: -


https://us-east-2.console.aws.amazon.com/rds/home?region=us-east-2#dbinstance:id=mysqltest

Amazon RDS — Details

Configurations
ARN
arn:aws:rds:us-east-2:336670933356:db:mysqltest
Engine
MySQL 5.6.39
License Model
General Public License
Created Time
Wed Jun 27 00:31:04 GMT+530 2018
DB Name
multiplextheatre
Username
root
Option Group

Security and network
Availability zone
us-east-2a
VPC
vpc-76a2f91e
Subnet group
default
Subnets
subnet-45f41a09
subnet-88eb6cf2
subnet-67ace20f
Security groups
launch-wizard-1 (sg-06b011c7fb0158194)
(active)
Publicly accessible
Yes

Instance and IOPS
Instance Class
db.t2.micro
Storage Type
General Purpose (SSD)
Storage
20 GiB

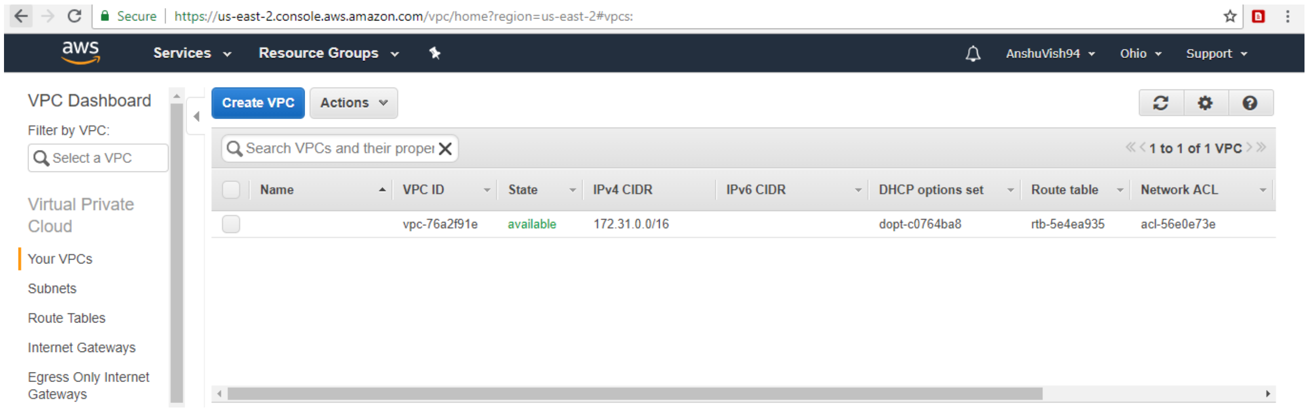
Availability and durability
DB instance status
available
Multi AZ
No

Backup and Restore
Automated backups

Maintenance details
Auto minor version upgrade
Yes
Maintenance window
sat:08:06-sat:08:36 UTC (GMT)
Pending Modifications
None
Pending maintenance
none

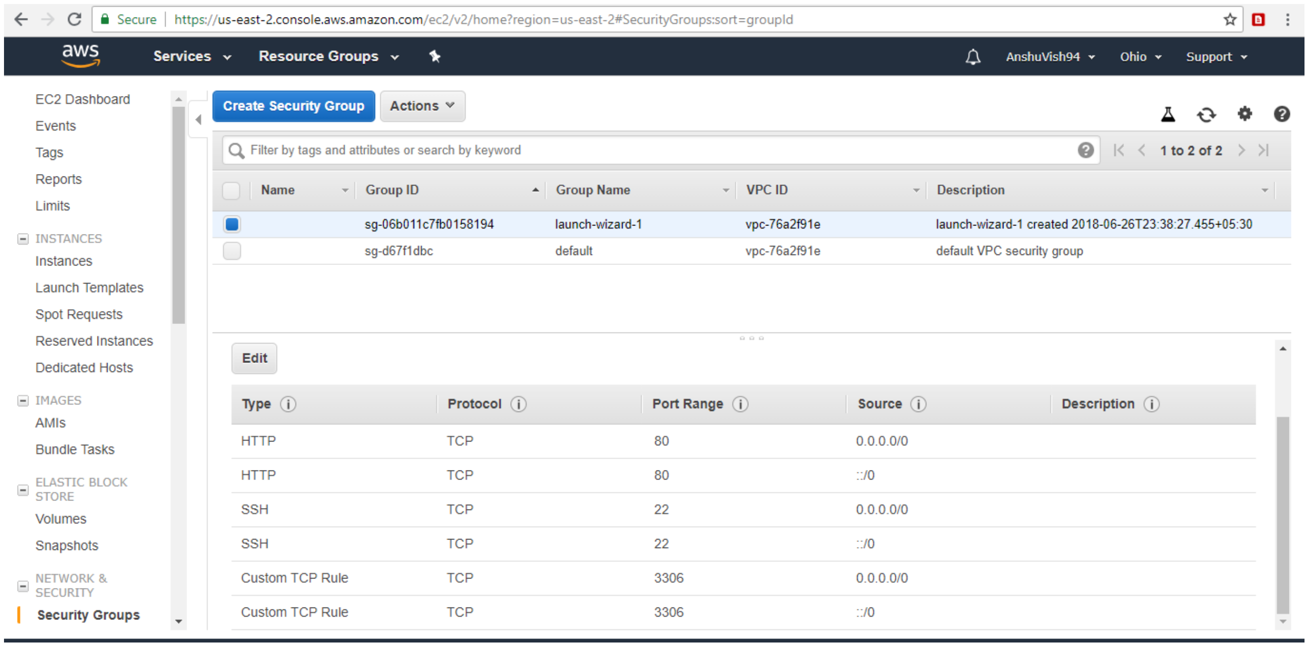
Encryption details
Encryption enabled
No

➢ Also create the VPC for security purpose


https://us-east-2.console.aws.amazon.com/vpc/home?region=us-east-2#vpcs:

Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR	DHCP options set	Route table	Network ACL
	vpc-76a2f91e	available	172.31.0.0/16		dopt-c0764ba8	rtb-5e4ea935	acl-56e0e73e

➢ Security group details



Secure | https://us-east-2.console.aws.amazon.com/ec2/v2/home?region=us-east-2#SecurityGroups:sort=groupId

Name	Group ID	Group Name	VPC ID	Description
	sg-06b011c7fb0158194	launch-wizard-1	vpc-76a2f91e	launch-wizard-1 created 2018-06-26T23:38:27.455+05:30
	sg-d67f1dbc	default	vpc-76a2f91e	default VPC security group

Type	Protocol	Port Range	Source	Description
HTTP	TCP	80	0.0.0.0/0	
HTTP	TCP	80	::/0	
SSH	TCP	22	0.0.0.0/0	
SSH	TCP	22	::/0	
Custom TCP Rule	TCP	3306	0.0.0.0/0	
Custom TCP Rule	TCP	3306	::/0	

MAINTENANCE

MAINTENANCE

Software maintenance in software engineering is the modification of a software product after delivery to correct faults, to improve performance or other attributes. **Software maintenance** is a very broad activity that includes error corrections, enhancements of capabilities, deletion of obsolete capabilities, and optimization. Because change is inevitable, mechanism must be developed for evaluation, controlling and making modifications. So any work done to change the software after it is in operation is considered to be maintenance work. The purpose is to preserve the value of software over the time. The value can be enhanced by expanding the customer base, meeting additional requirements, becoming easier to use, more efficient and employing newer technology

This section describes the six software maintenance processes as:

1. The implementation process contains software preparation and transition activities, such as the conception and creation of the maintenance plan; the preparation for handling problems identified during development; and the follow-up on product configuration management.
2. The problem and modification analysis process, which is executed once the application has become the responsibility of the maintenance group. The maintenance programmer must analyze each request, confirm it (by reproducing the situation) and check its validity, investigate it and propose a solution, document the request and the solution proposal, and finally, obtain all the required authorizations to apply the modifications.
3. The process considering the implementation of the modification itself.
4. The process acceptance of the modification, by confirming the modified work with the individual who submitted the request to make sure the modification provided a solution.
5. The migration process (platform migration, for example) is exceptional, and is not part of daily maintenance tasks. If the software must be ported to another platform without any change in functionality, this process will be used and a maintenance project team is likely to be assigned to this task.
6. Finally, the last maintenance process, also an event which does not occur on a daily basis, is the retirement of a piece of software.

CONCLUSION

CONCLUSION

It has been a great pleasure for me to work on this exciting and challenging project. This project proved good for me as it provided practical knowledge of not only programming in Java, Python, AngularJS, PolymerJS, Javascript, PostgreSQL, cloud deployment, but also about all handling procedure related with **"ASSET PERFORMANCE MANAGEMENT".** It also provides knowledge about the latest technology used in developing web enabled application and client server technology that will be great demand in future. This will provide better opportunities and guidance in future in developing projects independently.

Benefits

The project is identified by the merits of the system offered to the user. The merits of this project are as follows: -

- The Asset Performance Management has been developed to meet the present requirement of customer.
- The web application has been developed using water fall model.
- This web application is very useful and user friendly.
- Any user can access this software without any difficulty
- It is doesn't have any tedious or lengthy process

BIBLIOGRAPHY

Python reference

https://www.python.org/doc/essays/blurb/

PloymerJS reference

https://www.polymer-project.org/

Agile methodology reference

https://en.wikipedia.org/wiki/Agile_software_development

AngularJS reference

https://docs.angularjs.org/guide/introduction

Java reference

https://java.com/en/download/faq/whatis_java.xml