CLOUD INSIGHTS

A PROJECT REPORT

Submitted by

NITHIN S V (TKM19MCA017)

to

The APJ Abdul Kalam Technological University

In partial fulfillment of the requirements for the award of the degree of

MASTER OF COMPUTER APPLICATIONS

Thangal Kunju Musaliar College of Engineering
Kerala

DEPARTMENT OF COMPUTER APPLICATIONS

MAY 2022

DECLARATION

I undersigned hereby declare that the project report on **CLOUD INSIGHTS**, submitted for partial fulfillment of the requirements for the award of degree of Master of Computer Applications of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by me under supervision of Dr. Fousia M Shamsudeen. This submission represents my ideas in my own words and where ideas or words of others have been included,I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in our submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University..

Kollam

16-05-2022

NITHIN S V

DEPT. OF COMPUTER APPLICATIONS TKM COLLEGE OF ENGINEERING

KOLLAM

2021 - 22

CERTIFICATE

This is to certify that the report entitled **CLOUD INSIGHTS** submitted by **NITHIN S V** (TKM19MCA017) to the APJ Abdul Kalam Technological University in partial fulfillment of the Masters degree in Computer Applications is a bonafide record of the project work carried out by him under our guidance and supervision. This report in any form has not been submitted to any other University or Institute for any purpose.

Internal Supervisor | Head of the Department | External Examiner

Acknowledgement

First and foremost I thank GOD almighty and my parents for the success of this project. I owe sincere gratitude and heart full thanks to everyone who shared their precious time and knowledge for the successful completion of my project.

I am extremely grateful to **Dr. Fousia M Shamsudeen**, Head of the Department, Department of Computer Applications, for providing me with best facilities.

I would like to thank my coordinator and project guide **Prof. Vaheetha Salam**, Department of Computer Applications, who motivated me throughout the project .

I would like to thank my external coordinator **Mr. Amal G Jose** and **Mr. Chandrakanth Gottam**, Knowledge Lens, who guided me throughout my work.

I profusely thank all other faculty members in the department and all other members of TKM College of Engineering, for their guidance and inspirations throughout my course of study.

I owe my thanks to my friends and all others who have directly or indirectly helped me in the successful completion of this project.

NITHIN S V

ABSTRACT

CLOUD INSIGHTS, is a web platform that tracks and evaluates all of a company's cloud resources. Although all platforms offer consumer monitoring, there is no way to keep track of several resources in one place. This application solves the issues by providing a single platform for monitoring several resources.

The primary goal of development is to keep track of the company's cloud usage. Because my company is increasingly using the Azure platform, and all of our applications use it, we created the initial version of Azure platform support. We can add many Azure accounts and manage them all from one location. Everything in this application has been designed in accordance with the SRS requirements. It displays all of the subscriptions' detailed consumption as well as resource level usage. All cloud platforms will be supported in the forthcoming edition. We are in the process of planning the next version.

Cloud Insights is a self-adjusting monitoring server that may be used for both agent-based and centralised monitoring. Custom monitors and agents can be added to Cloud Insights. When a specific event occurs or a specific threshold is crossed, the monitors have an idea of an implicit action to be executed. These actions enable cloud applications to self-heal when they encounter problems. Manual involvement may be required for some actions. In these circumstances, an action is nothing more than a notification.

Contents

List of Figures

Chapter 1

Introduction

CLOUD INSIGHTS is a web-based software built to meet the company's needs for analysing many cloud platform services in one place. Although cloud services include a monitoring and analysis dashboard, it does not include the capability of displaying the status of numerous cloud services on the same platform.It also offers notification services, allowing customers to select a maximum threshold value and receive notifications through email and phone when resource usage exceeds the stated threshold. You may provide people custom access to different subscriptions and resources with this application. It has access control at the resource level. We primarily concentrated on monitoring Azure accounts in the first version of the application. Multiple Azure accounts have been added to the application. We have a dashboard in cloud insights that shows a graphical representation of overall cost, total resource usage, daily resource, and a customizable graphical depiction of usage.It has daily, monthly, weekly, and custom date ranges in the customisable graphical representation, and we can even show individual resource consumption. The usage data we're displaying was obtained from the Azure consumption API. The usage data will be returned as JSON, and the Account ID and Subscription ID will be given in the payload.

1.1 Objective

The goal is to accomplish the following:

- Capacity and Resource Planning
- Capacity and Resource Management
- Billing
- Monitoring the usage of cloud services
- Increase productivity
- Controlling the use of cloud services effectively
- Different cloud services are monitored under one roof
- Customized graphical representation of cloud service consumption at the resource level
- Custom access control at the resource level for cloud service assessment and tracking

1.2 Company Profile

Knowledge Lens provides a collection of Lens that automates and simplifies the discovery of hidden insights from Big Data .Our mission is to turn the dark data to meaningful business insights. We are Big Data Technology Geeks with extensive Industry expertise and a wide range of Big Data Projects ranging from Big Data Engineering to Data Science.

1.2.1 Products

- **iLens(Intelligent Lens)**

  iLens provides a single platform for smart integration with various devices or sensors in large enterprises, manufacturing industry, home, commercial properties etc.

  iLens provides an MQTT interface for seamless integration of various sensor devices in the field to capture time series data in real time. Based on pre-configured rules, iLens is able to generate alerts, alarms based on the rules.

- **Mlens**

  MLens is a one-step solution which enables you to manage disaster recovery for your big data and platforms

  Features of MLens :

  1. Big Data Backup Migration
  2. Automated Disaster Recovery
  3. Data Encryption, compression Archival
  4. High Speed Batch Data Ingestion
  5. Monitoring Scheduling
  6. Secured Access controls

- **AiLens**

  Next Generation Ai platform that offers a collaborative workspace with experiment designer, modelling feature engineering work bench, AI/ML assets repository integrations for enterprise security and DevOps.

  AiLens is an intelligent assistant for Artificial Intelligence crafted with a unified graphical interface for building Data Engineering and AI/ML pipelines.AiLens includes a unified AI Orchestrator which triggers model execution runs on any runtimes like Tensorflow, SparkML, H2O, MxNet, Theano, PyTorch, AWS / Azure from a console. AiLens is quite flexible as the user experience will be the same.

  Irrespective of any new technological advancements because of the meta model-driven platform. Intuitive job submission and monitoring framework, secured integration with external entities and inbuilt encryption and rolebased access control support make our product stand out with a huge margin.

  Key features are:

  1. Any AI Stack, Any AI Algorithm, Anywhere
  2. Unified AI Orchestrator
  3. Simplified User Experience
  4. Intelligent Assistant for AI
  5. Integrated Data Preparation AI Modelling Environment

6. Seamless Enterprise Security Integration

- **GLens**

  GLens is a Real-Time Data Acquisition, Monitoring and Analytics suite of Products for Industrial Emissions, Effluent Discharges and Ambient Air Monitoring. GLens DAS Software, GLens Server Platform, GLens Environ Data Logger provides a comprehensive solution for all Industry Environmental needs. The platform connects to any analyser, sensor or device in a plug and play model acquiring data in real time.

  The key features of GLens are::

  1. Rest based open protocol for multi-client deployment.
  2. Real time alerts and alarms with SMS and Email integration.
  3. Remote calibration and configuration of analyzers.
  4. Plug and play complete protocol integration with any analyzer make and model. – Integrated and data quality codes as per ISO 7168.
  5. Integrated analytics and predictive models for effective pollution control.
  6. Live consolidated industry dashboards.

1.2.2 Services

- **Big Data Engineering Services**

  We provide end to end Architecture, Design, Development, Testing and Deployment of Big Data Protects.

- **Big Data Security Services**

  We are one of the niche consulting companies to provide specialized Big Data Services.

- **Big Data Analytics Services**

  We deliver hidden insights from a wide variety of data sources using our pre-build analytical Lens.

- **Big Data Competency Development**

  Without unique Big Data expertise, we provide one of the best Big Data Competency Development programs for the enterprise.

Chapter 2

Literature Survey

Literature review is the comprehensive study and interpretation of literature that relates to a particular topic. When one uses literature review research questions are identified, then one seek to answer this research questions by searching for and analyzing relevant literature. Some importance of literature reviews is that new insights can be developed by the re-analyzing the results of the study. A literature review is both a summary and explanation of the complete and current state of knowledge on a topic as found in academic books and journal articles. There are two kinds of literature reviews you might write at university: one that students are asked to write as a stand-alone assignment in a course, and the other that is written as part of an introduction to, or preparation for, a longer work, usually a thesis or research report. The focus and perspective of your review and the kind of hypothesis or thesis argument you make will be determined by what kind of review you are writing. One way to understand the differences between these two types is to read published literature reviews or the first chapters of theses and dissertations in your own subject area. Analyses the structure of their arguments and note the way they address the issues.

2.1 Purpose of the Literature Review

1. It gives readers easy access to research on a particular topic by selecting high quality articles or studies that are relevant, meaningful, important and valid and summarizing them into one complete report.

2. It provides an excellent starting point for researchers beginning to do research in a new area by forcing them to summarize, evaluate, and compare original research in that

specific area.

3. It ensures that researchers do not duplicate work that has already been done.
4. It can provide clues as to where future research is heading or recommend areas on which to focus.
5. It highlights the key findings.
6. It identifies inconsistencies, gaps and contradictions in the literature.
7. It provides a constructive analysis of the methodologies and approaches of other researchers.

2.2 Related Works

Monitoring of Cloud is a task of paramount importance for both Cloud Service Providers (called Providers in the following) and Cloud Service Consumers (called Consumers in the following). On the one side, it is a key tool for controlling and managing hardware and software infrastructures; on the other side, it provides information and Key Performance Indicators (KPI) for both platforms and applications. The continuous monitoring of the Cloud and of its Service Level Agreements (SLAs), - for example, in terms of availability, delay, etc. - supplies both the Providers and the Consumers with information such as the workload generated by the latter or the performance and Quality of Service (QoS) offered through the Cloud, also allowing to implement mechanisms to prevent or recover violations, for both the Provider and Consumers. Cloud Computing involves many activities for which monitoring is an essential task. The most important ones are:

- **Capacity and Resource Planning.** One of the most challenging tasks for application and service developers, before the large scale adoption of Cloud Computing, has always been resource and capacity planning.
- **Capacity and Resource Management.** The first step to manage a complex system like a Cloud consists in having a monitoring system able to accurately capture its state.
- **Data Center Management.** Cloud services are provided through large scale data centers, whose management is a very important activity. Data center management

includes two fundamental tasks: (i) monitoring, that keeps track of desired hardware and software metrics; (ii) data analysis, that processes such metrics to infer system or application states for resource provisioning, troubleshooting, or other management actions.

- **SLA Management.** The unprecedented flexibility in terms of resource management provided by Cloud Computing calls for new programming models in which Cloud applications can take advantage of such new feature, whose underlying premise is monitoring

- **Billing.** In order to offer "measured services" allowing the Consumer to pay proportionally to a metered parameter, monitoring is fundamental, not only -trivially- for the Provider (or the Auditor), but also for the Consumer, in order to verify its effective usage of the Cloud services, and also to compare pricing over different providers (a nontrivial monitoring task).

- **Troubleshooting.** The complex infrastructure of a Cloud represents a big challenge for troubleshooting (e.g. root cause analysis), as the cause of the problem has to be searched in several possible components (e.g. network, host, etc.), each of them made of several layers. Monitoring is therefore needed for Providers to understand where to locate the problem inside their complex infrastructure and for Consumers to understand if any occurring performance issue or failure is caused by the Provider or by other causes

- **Performance Management.** Being the hardware infrastructure maintenance delegated to the Providers, the Cloud Computing model is attractive for most Consumers (primarily medium sized enterprises and research groups). However, despite the attention paid by Providers, some Cloud nodes may attain performance orders of magnitude worse than other nodes.

- **Security Management.** Cloud security is very important for several reasons. It is one of the most significant obstacles to the spread of Cloud Computing, especially considering certain kinds of applications (e.g. business-critical ones) and Consumers (e.g. governments) [1].

Cloud Monitor is a self-adjusting, multi-threaded monitoring server that supports both agent-based and centralized monitoring modes. CM allows users to add custom monitors and agents.

The monitors have a concept of an implicit action to be taken when a particular event happens or a particular threshold is reached. These actions allow cloud application to self-heal in case of failures. Some actions may require manual intervention. In such cases, an action is just a notification. Cloud Monitor monitors specified cluster nodes for errors. On encountering an error, CM tries to auto-repair the node if the corresponding monitor has a tagged auto-heal action. Failure to repair the node or absence of an auto-heal action leads to a notification to support team. These notifications are send in the increasing level of priority. Currently CM supports sending the notifications via email, SMS and voice. CM server is redundant. There is a seamless transition to the backup notification server when the primary server fails [2].

Cloud computing services are rapidly gaining in popularity.They allow the user to rent, only at the time when needed,only a desired amount of computing resources s(processingability and storage capacity) out of a huge mass of distributed computing resources swithout worrying about the locations orinternal structures of these resources. It isanticipated sthat enterprises will accelerate their migrationfrom building and owning their own systems to srenting cloudcomputing services because cloud computing services are easyto use, and can sreduce both business costs and environmentalloads. The cloud computing environments srequire a huge amount of ICT equipment such as servers, storage devices, communication snetwork devices and client terminals. Therefore, it is clear that the widespread use of cloud scomputing services will greatly contribute to a rapid increase on ICT power consumption [3].

Cloud Computing has rapidly become a widely adopted paradigm for delivering services over the Internet. This is due to a number of technical reasons, including: improvement of energy efficiency, optimization of hardware and software resources utilization, elasticity, performance isolation, flexibility, and on-demand service schema. In addition to such technical benefits, the literature has shown how the Cloud Computing model provides several economical benefits including minimal capital and operational expenditures (CAPEX and OPEX). For all these reasons, the number of organizations adopting Cloud solutions and subscribers accessing Cloud services has rapidly increased, exceeding the optimistic initial plans, and so has done the complexity of Cloud systems. Cloud services are on-demand, elastic and scalable, and the following main features are therefore needed by a Cloud system: availability, concurrency, dynamic load balancing, independence of running applications, security, and intensiveness. To provide these features, advanced virtualization techniques, robust and dynamic scheduling approaches, advanced security measures and disaster recovery mechanisms are implemented

and operated in Cloud Computing systems. Data centers for Cloud Computing continue to grow in terms of both hardware resources and traffic volume, thus making Cloud operation and management more and more complex

In this scenario, accurate and fine-grained monitoring activities are required to efficiently operate these platforms and to manage their increasing complexity.

In literature, there is a large number of works proposing surveys and taxonomies of Cloud Computing in general, of Virtualization technologies, and of Cloud Security. To the best of our knowledge, however, there are no specific surveys on platforms, techniques, and tools for monitoring Cloud infrastructures, services, and applications. This is what we define as Cloud monitoring [4].

Chapter 3

Methodology

CLOUD INSIGHTS is a web-based software built to meet the company's needs for analysing many cloud platform services in one place. Although cloud services include a monitoring and analysis dashboard, it does not include the capability of displaying the status of numerous cloud services on the same platform.It also offers notification services, allowing customers to select a maximum threshold value and receive notifications through email and phone when resource usage exceeds the stated threshold. You may provide people custom access to different subscriptions and resources with this application. It has access control at the resource level. We primarily concentrated on monitoring Azure accounts in the first version of the application. Multiple Azure accounts have been added to the application. We have a dashboard in cloud insights that shows a graphical representation of overall cost, total resource usage, daily resource, and a customizable graphical depiction of usage.It has daily, monthly, weekly, and custom date ranges in the customisable graphical representation, and we can even show individual resource consumption. The usage data we're displaying was obtained from the Azure consumption API. The usage data will be returned as JSON, and the Account ID and Subscription ID will be given in the payload.

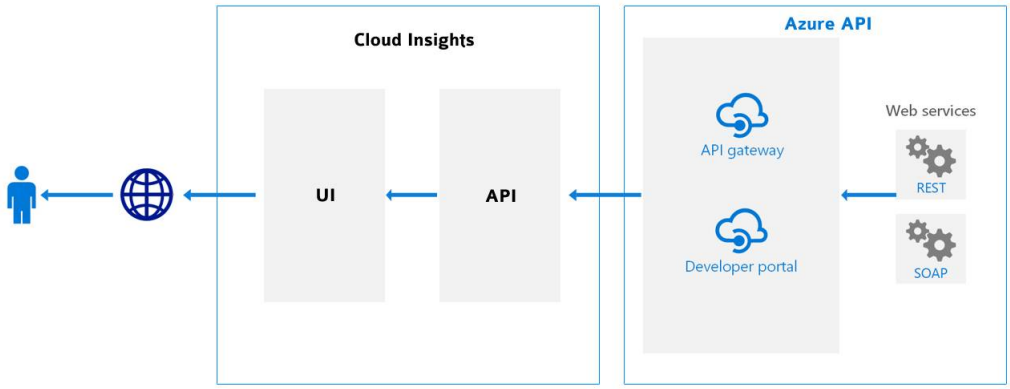

Figure 3.1: Work flow

We focused on monitoring Azure cloud services because this is the first version of the application. We acquire the consumption details using the REST API given by Azure in the developer portal as shown in the figure 3.1, and we can get an exact and detailed consumption status by giving all of the required inputs. We get the usage status from the back-end and add the appropriate details before passing it to the cloud insights API. We employ charts in the user interface to display usage status, and we used the echarts package to do so. We can also add users to resource level with custom access restrictions.

Azure consumption API

The Azure Consumption APIs give you programmatic access to cost and usage data for your Azure resources. These APIs currently only support Enterprise Enrollments and Web Direct Subscriptions (with a few exceptions). The APIs are continually updated to support other types of Azure subscriptions.

Azure Consumption APIs provide access to:

- Enterprise and Web Direct Customers
  - Usage Details
  - Marketplace Charges
  - Reservation Recommendations
  - Reservation Details
  - Reservation Summaries

- Enterprise Customers Only
  - Price sheet
  - Budgets
  - Balances

Usage Details API

Use the Usage Details API to get charge and usage data for all Azure 1st party resources. Information is in the form of usage detail records which are currently emitted once per meter per resource per day. Information can be used to add up the costs across all resources or investigate costs / usage on specific resource(s).
The API includes:

- **Meter Level Consumption Data** - See data including usage cost, the meter emitting the charge, and what Azure resource the charge pertains to. All usage detail records map to a daily bucket.
- **Azure role-based access control (Azure RBAC)** - Configure access policies on the Azure portal, the Azure CLI or Azure PowerShell cmdlets to specify which users or applications can get access to the subscription's usage data. Callers must use standard Azure Active Directory tokens for authentication. Add the caller to either the Billing Reader, Reader, Owner, or Contributor role to get access to the usage data for a specific Azure subscription.
- **Filtering** - Trim your API result set down to a smaller set of usage detail records using the following filters: - Usage end / usage start - Resource Group - Resource Name
- **Data Aggregation** - Use OData to apply expressions to aggregate usage details by tags or filter properties
- **Usage for different offer types** - Usage detail information is currently available for Enterprise and Web Direct customers.

Marketplace Charges API

Use the Marketplace Charges API to get charge and usage data on all Marketplace resources (Azure 3rd party offerings). This data can be used to add up costs across all Marketplace resources or investigate costs / usage on specific resource(s).

Balances API

Enterprise customers can use the Balances API to get a monthly summary of information on balances, new purchases, Azure Marketplace service charges, adjustments, and overage charges. You can get this information for the current billing period or any period in the past. Enterprises can use this data to perform a comparison with manually calculated summary charges. This API does not provide resource-specific information and an aggregate view of costs.

Budgets API

Enterprise customers can use this API to create either cost or usage budgets for resources, resource groups, or billing meters. Once this information has been determined, alerting can be configured to notify when user-defined budget thresholds are exceeded.

Reservation Recommendations API

Use this API to get recommendations for purchasing Reserved VM Instances. Recommendations are designed to allows customers to analyze expected cost savings and purchase amounts.

Reservation Details API

Use the Reservation Details API to see info on previously purchased VM reservations such as how much consumption has been reserved versus how much is actually being used. You can see data at a per VM level detail.

Reservation Summaries API

Use this API to see aggregate information on previously purchased VM reservations such as how much consumption has been reserved versus how much is actually being used in the aggregate.

Price Sheet API

Enterprise customer can use this API to retrieve their custom pricing for all meters. Enterprises can use this in combination with usage details and marketplaces usage info to perform cost calculations using usage and marketplace data.

3.1 System Specifications

The application development architecture recognized for this project is specified in this section on the basis of requirements.

3.1.1 Software Specification

- Programming Language : Python
- Designing tools : Angular
- Cloud service used : Azure
- Web server : Nginx
- Web Browser : Any web browser
- Database : PostgreSQL
- Backend: Python FastAPI

3.1.2 Software Description

- **Python**

  Python is a widely used general purpose, high level programming language.Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of cod than would be possible in language such as C++ or Java. The language provides constructs intended to enable clear programs on both a small large scale. Python supports multiple programming paradigms, including object-oriented, imperative and functional programming or procedural styles. It features a dynamic type system and automatic memory management and has a large and comprehensive standard library. Python interpreters are available for installation on many operating systems, allowing Python code execution on a wide variety of systems. Using third party tools, such as Py2exe or PyInstaller, Python code can be packaged into standalone executable programs for some of the most popular operating systems, allowing for the distribution of python based software for use on those environments without requiring the installation of a python interpreter.

- **Angular**

  Angular is a platform that makes it easy to build applications with the web. Angular combines declarative templates, dependency injection, end to end tooling and integrated

best practices to solve development challenges. Angular empowers developers to build applications that live on the web, mobile, or the desktop.

Features:

- Components : Components help to build the applications into many modules. This helps in better maintaining the application over a period of time.
- TypeScript :This is a superset of JavaScript and is maintained by Microsoft.
- Services :Services are a set of code that can be shared by different components of an application. So for example if you had a data component that picked data from a database, you could have it as a shared service that could be used across multiple applications. Environment setup.
- Editor :There are many editors that can be used for Angular development such as Visual Studio code.

- **PostgreSQL**

  PostgreSQL is a powerful, open source object-relational database system that uses and extends the SQL language combined with many features that safely store and scale the most complicated data workloads. The origins of PostgreSQL date back to 1986 as part of the POSTGRES project at the University of California at Berkeley and has more than 30 years of active development on the core platform.

  PostgreSQL has earned a strong reputation for its proven architecture, reliability, data integrity, robust feature set, extensibility, and the dedication of the open source community behind the software to consistently deliver performant and innovative solutions. PostgreSQL runs on all major operating systems, has been ACID-compliant since 2001, and has powerful add-ons such as the popular PostGIS geospatial database extender. It is no surprise that PostgreSQL has become the open source relational database of choice for many people and organisations.

3.2 System Design

Design has been described as a multi-step process that is a representation of data structure, program struc tures, interface characteristics and procedural detail synthesized from information

requirements. Design serves as a foundation for all software engineering and maintenance step that follows. It is an activity concerned with making decision, often of a structural nature. Design builds coherent well-planned rep resentations of programs that concentrate on the Interrelations of parts at the higher level and the logical operations involved at the lower levels. Depending on the applications and project requirements a good design is one, which allows efficient code to be produced and whose implementation is compact as possible.

Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input design.

The fundamental design concept specifies the following methods to develop a project are:

- Abstraction
- Modularity
- Software Architecture
- Structural Partitioning
- Data Structure
- Software Procedure

There are two levels of system design:

- Logical design
- Physical design

In logical design, the designer produces a specification of the major features of the system which meet the objectives. The physical design gives the actual design of the system

- **Design Concept**

  The design concepts provide the software designer with a foundation from which more sophisticated methods can be applied. A set of fundamental design concepts has evolved.They are:

- Abstraction - Abstraction is the process or result of generalization by reducing the information con tent of a concept or an observable phenomenon, typically in order to retain only information which is relevant for a particular purpose.
- Refinement - It is the process of elaboration. A hierarchy is developed by decomposing a macroscopic statement of function in a stepwise fashion until programming language statements are reached. In each step, one or several instructions of a given program are decomposed into more detailed instructions. Abstraction and Refinement are complementary concepts.
- Modularity - Software architecture is divided into components called modules.
- Software Architecture - It refers to the overall structure of the software and the ways in which that structure provides conceptual integrity for a system. Good software architecture will yield a good return on investment with respect to the desired outcome of the project, e.g. in terms of performances, quality, schedule and cost.
- Control Hierarchy - A program structure that represent the organization of a program Component and implies a hierarchy of control.
- Structural Partitioning - The program structure can be divided both horizontally and vertically. Horizontal partitions define separate branches of modular hierarchy for each major program function vertical partitioning suggests that control and work should be distributed top down in the program structure.
- Data structure - It is a representation of the logical relationship among elements of data • Software Procedure - It focuses on the processing of each module individually
- Information Hiding - Modules should be specified and designed so that information contained within a module ¡s inaccessible to other modules that have no need for such information

Major Activities Carried out during Design Phase includes the following:

- Logical Design
- UMI Diagrams
- Architectural Design

- Module Design
- Database Design
- User Interface Design
- Input Design
- Output Design

3.3 Model Design

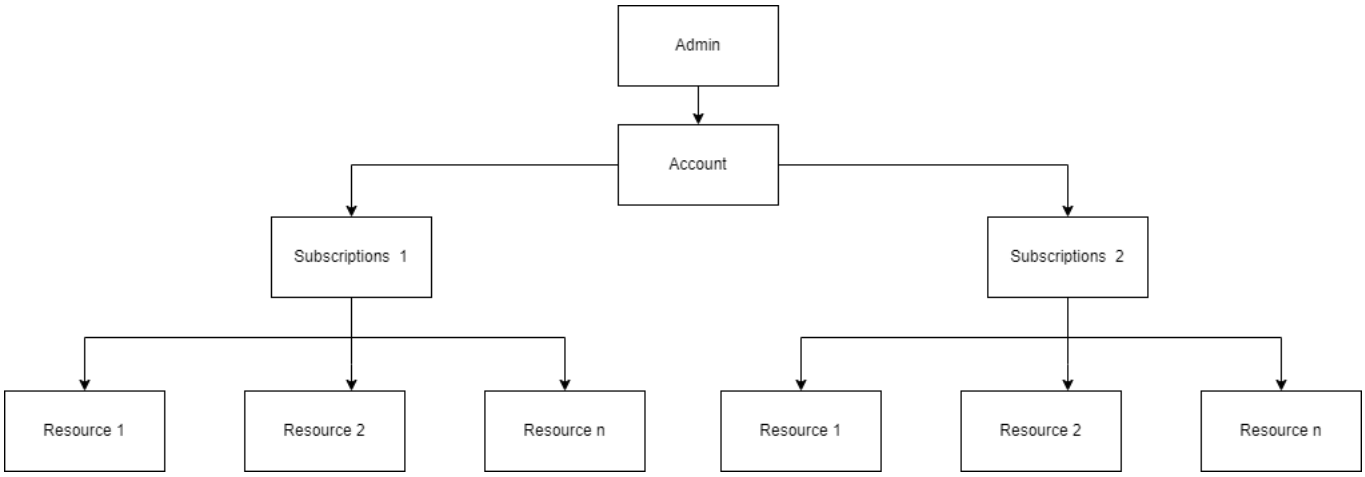

Figure 3.2: Hierarchy of Accounts

Login Hierarchy:

- Super Admin
  - User logins are created by super admin
  - Has ability to add or remove users.
  - Can configure accounts
  - Access to all dashboards
  - Maintain all the notification and thresholds
- Admin
  - Access to all dashboards
  - Maintain his accounts and users

  - Can create threshold and also he can maintain the thresholds created by the users he added to the account.
  - Can create notifications and also he can maintain the notifications created by the users he added to the account.

- user Login
  - Access to the resources allocated to him.
  - Can create thresholds and notifications for resources he has authority
  - Has limited access and restricted to stations of that state

Chapter 4

RESULT AND DISCUSSION

Testing is the major quality measures employed during the software development.After the coding phase, computer programs available are executed for testing purpose. Testing not only has to uncover errors introduced during coding, but also locates errors committed during the previous phase. Thus the aim of testing is to uncover requirements, design or coding errors in the program.

- Testing is a process of executing a program with the intention of finding an error.
- A good test case is one that has a highest probability of finding an as yet undiscovered error.
- A successful test is one that uncovers an as yet undiscovered error.

Our objective is to design tests that systematically uncover different classes of errors and to do so with minimum amount of time and effort. Testing demonstrate that software functions appear to be working according to specification, that performance requirements appears to have been met. Data collected as testing is conducted provide a good indication of software reliability and some indication of software quality as a whole. But there is one thing that testing cannot do: Testing cannot show the absence of defects it can only show that software defects as present.

4.1 Testing methods

There are different types of testing methods available.

4.1.1 Validation Testing

At the culmination of black box testing, software is completely assembled as a package, interface errors have been uncovered and corrected and final series of software test, validation test begins. Validation testing can be defined in many ways but a simple definition is the validation succeeds when the software functions in a manner that can be reasonably accepted by the customer. After validation test have been conducted one of the two possible conditions exists.

- The function or performance characteristics are confirmed to the specification and are accepted.
- A deviation from specification is uncovered and a deficiency list is created.

4.1.2 Output Testing

After performing the validation testing, the next step is output testing of the proposed system since no system could be useful if it doesn't produce the required data in the specific format. The output displayed or generated by the system under consideration is tested by, asking the user about the format displayed. The output format on the screen is found to be correct as the format was designed in the system according to the user needs. Hence the output testing doesn't result in any correction of the system.

4.1.3 User Acceptance Testing

User acceptance of the system is the key factor for the success of the system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective systems at the time of developing and making changes wherever required. This is done with regard to the following points:

- Output screen design
- Input screen design
- Menu driven system

4.2 Test plan

A test plan is a systematic approach to test a system. The plan typically contains a detailed understanding of what the eventual workflow will be. Normally testing of any large system will be in two parts.

- The functional verification and validation against the requirement specification
- Performance evaluation against the indicated requirements.

Testing activity is involved right from the beginning of the project. At the very first stage of testing, the goals and objectives are set. This simplifies the limits or borders of the testing process. Before testing, the tester should plan what kind of data he is giving for the test. Give data inputs as functional, boundary, stress, performance, usability values etc.

Characteristics of a Good Test:

- Tests are likely to catch bugs
- No redundancy.
- Not too simple or too complex
- Validity
- Reliability
- Objectivity

4.3 Test cases

A test case is a document, which has a set of test data, preconditions, expected results and postconditions, developed for a particular test scenario in order to verify compliance against a specific requirement.

Test Case acts as the starting point for the test execution, and after applying a set of input values, the application has a definitive outcome and leaves the system at some end point or also known as execution postcondition.

A test case should only test one limited subset of a feature or functionality. Test case documents for each functionality/testing area will be written, reviewed and maintained separately in excel sheets. In system testing, test data should cover the possible values of each parameter based on the requirements. Since testing every value is impractical, a few values should be chosen from each equivalence class. An equivalence class is a set of values that should all be treated the same. Ideally, test cases that check our error conditions are written separately from the functional test cases and should have steps to verify the error messages and logs. Realistically, if error test cases are not yet written, it is OK for testers to check for error conditions when performing normal functional test cases. It should be clear which test data, if any, is expected to trigger errors.

Implementation

Implementation is the process of having the system personnel check out and put new equipment to use, train the users to use the new system and construct any file that are needed to see it. The final and impartment phases in the system life cycle are the implementation of the new system. System implementation refers to the steps necessary to install a new system to put into operation. The implementation has different meaning, ranging from the conversion of a basic application to complete replacement of computer system. Implementation includes all these activities that take place to convert from old system to new one. The new system may be totally new replacing an existing manual or automated system or it may be major modification to an existing system. The methods of implementation and time scale adopted are found out initially. The system is tested properly and at the same time the users are trained in the new procedure.33 Proper implementation is essential to provide a reliable system to meet organizational requirements. Successful implementations may not guarantee improvement in the organization involves the following things:

- Careful planning
- Investigation of the system and constraint
- Design the methods to achieve the change over
- Train the staff in the changed phase
- Evaluation of change over method Implementation methods

There are several methods for handling the implementation and consequent conversation from the old to new automated system. The most secure way for this conversation is to run the old and new system in parallel. This method offers high security but the cost for maintaining the two systems in parallel is very high. An- other method is direct cut over the existing system to an automated system. The chance may take place within a week or within a day.

Implementation Phase

It includes a description of all activities that most occur to implement the new system and put into operation. It consists of the following steps:

- List all files required for the implementation
- Identify all data required to build new files during the implementation
- List all new document and procedure that go to the new system

4.4 Validation

The process of evaluating software during or at the end of the development process to determine whether it satisfies specified requirements.Validation is the process of evaluating the final product to check whether the software meets the customer expectations and requirements. It is a dynamic mechanism of validating and testing the actual product.

Validation is the process of checking whether the specification captures the customer's needs. "Did I build what I said I would?".Validation uses methods like black box (functional) testing, gray box testing, and white box (structural) testing etc.It can catch errors that verification cannot catch. It is High Level Exercise.target is actual product-a unit, a module, a bent of integrated modules, and effective final product.Validation is carried out with the involvement of a testing team.

4.5 Output Screens and Results

1. Login page:

   When user hits the URL this page is loaded

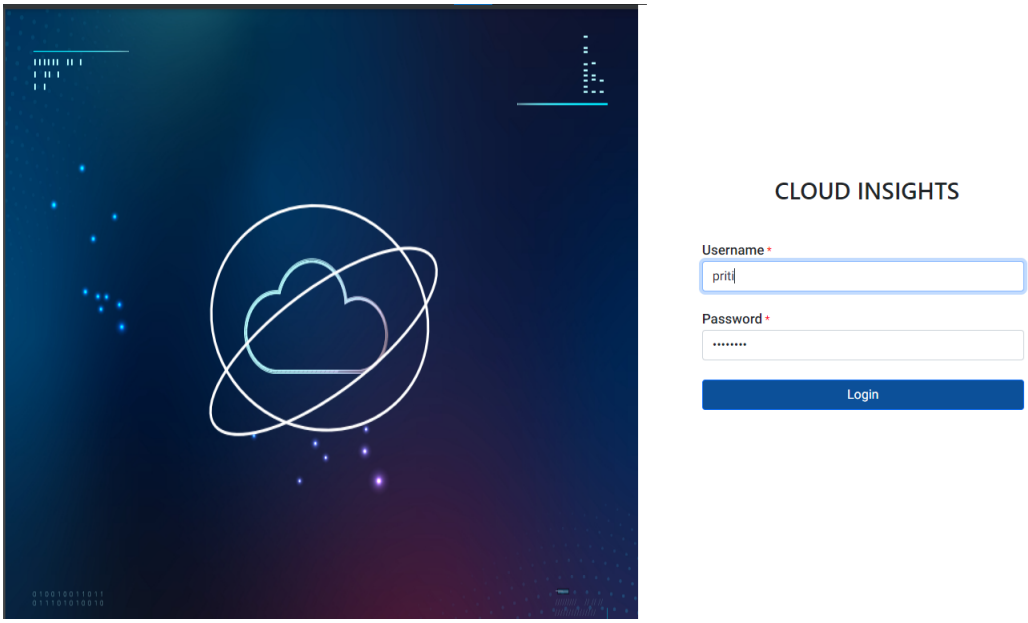

Figure 3.1: Login Page

2. Landing page

   Landing page is divided into Three parts:

   - Overall cost Dashboard
   - Bar chart with filters usage Cost Dashboard
   - Daily usage Cost Dashboard
   - Total Accumulated Cost Dashboard

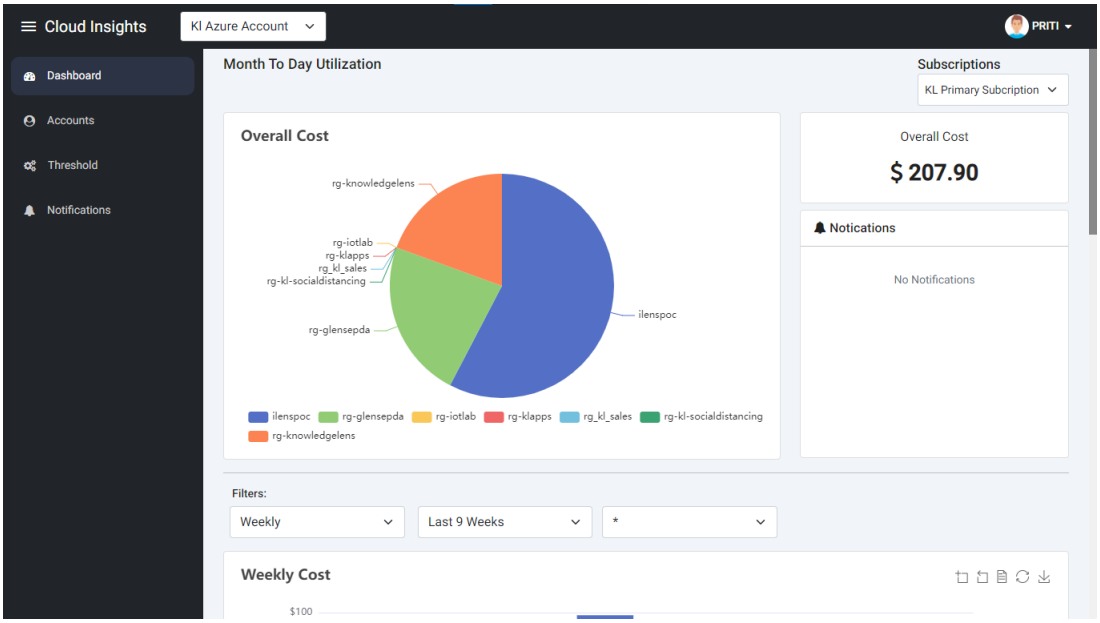

Figure 3.2: Overall cost Dashboard

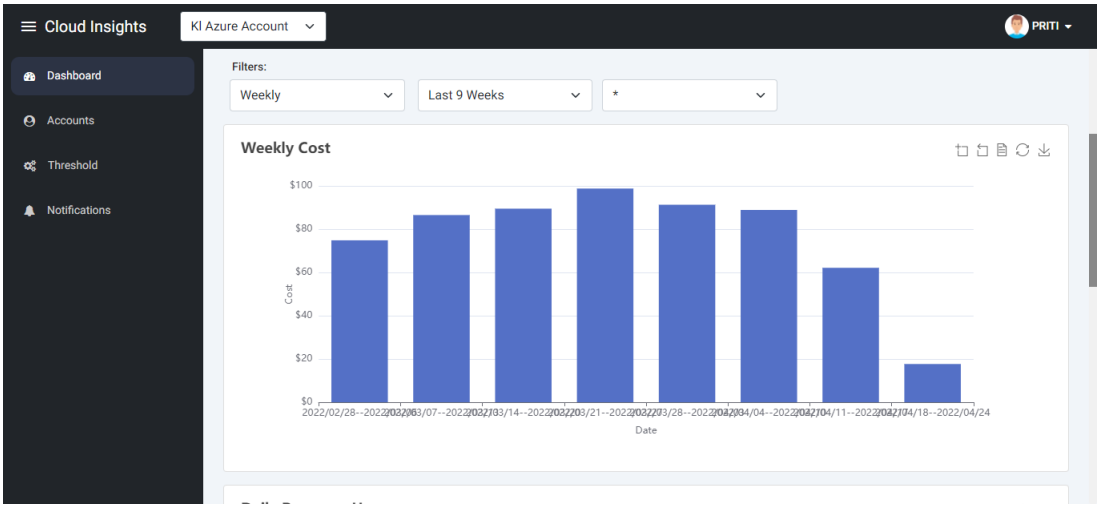

Figure 3.3: Bar Chart With filters

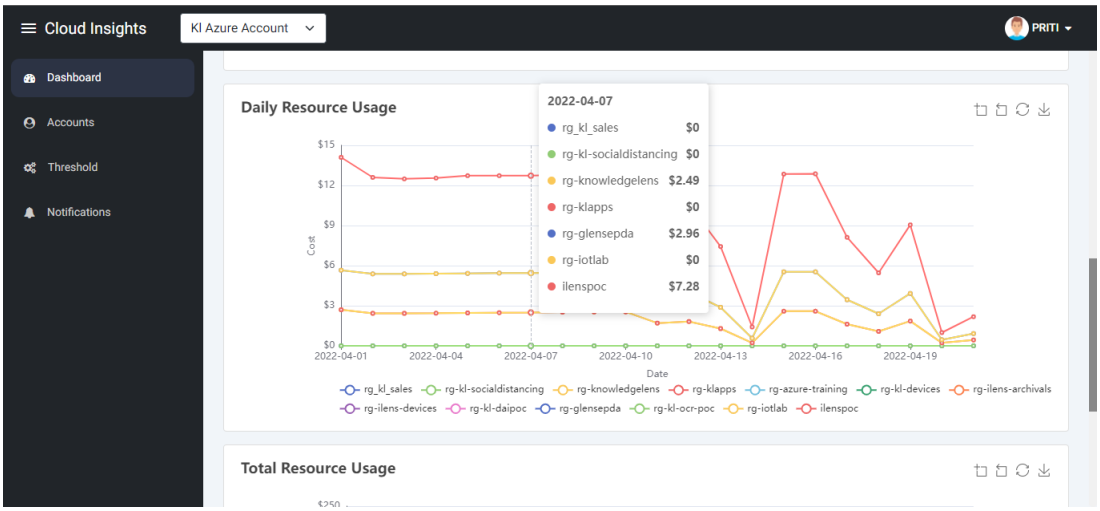

Figure 3.4: Daily usage

Figure 3.5: Total usage

The dashboard lists all of the user's accounts usage, and there is a drop down on the top navbar that lists them all. He has the ability to switch between accounts.

We have three filters in the charts: month, date, and day, number of weeks, number of months, and custom date range. After that, sort by resource.

3. Thresholds Page

   - The thresholds page is where you can add, delete, and update thresholds.
   - Users and administrators can set the threshold. The admin can see all thresholds, whether they were created by the user or by the admin. He has complete control over all of the account's thresholds.

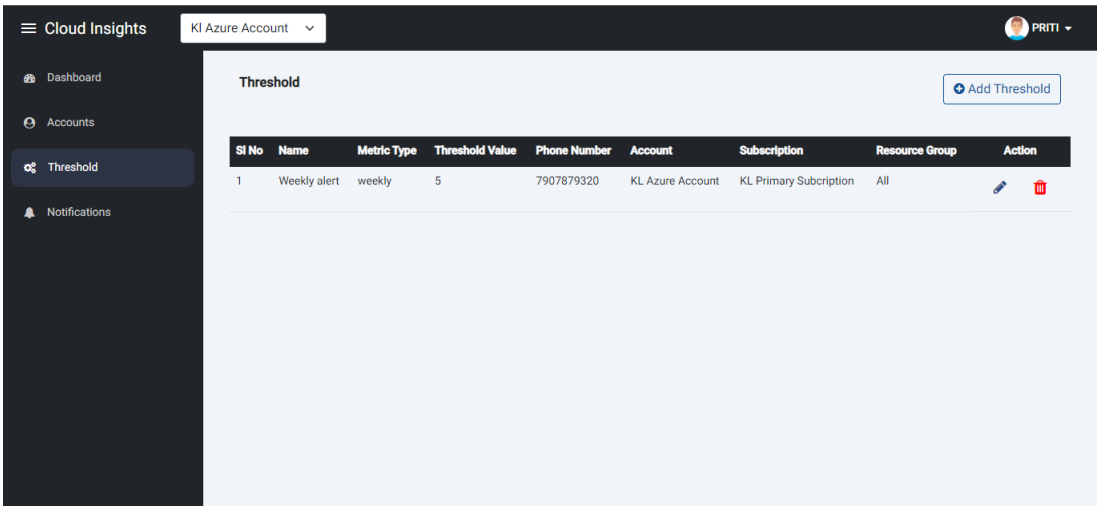

Figure 3.6: Thresholds

4. Notification Page

   - The Notification page is where you can add, delete, and update notification.
   - Users and administrators can set the notification. The admin can see all notification, whether they were created by the user or by the admin. He has complete control over all of the account's notification.

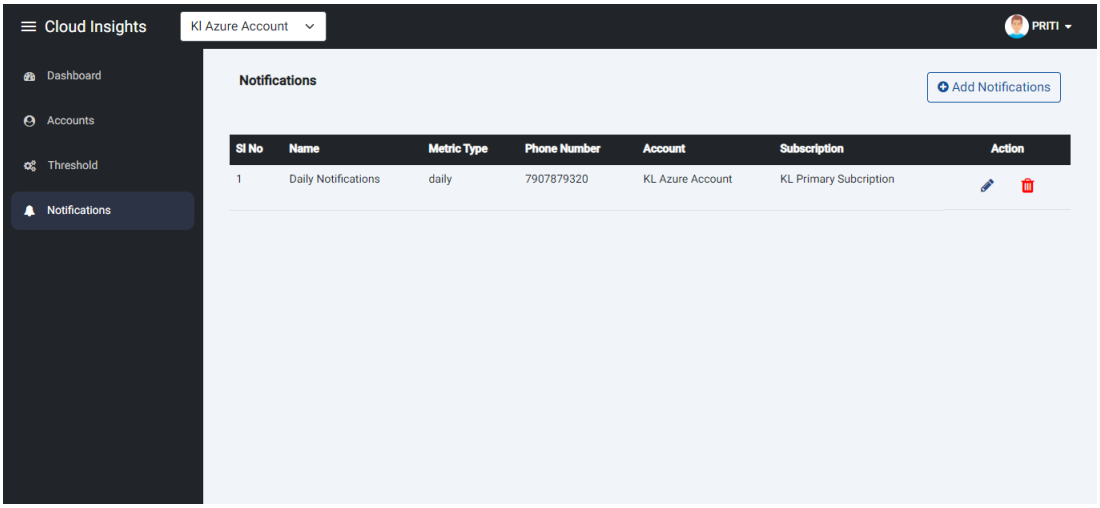

Figure 3.7: Notifications

5. Add Account Page

- Only Admin has the access to the add account.
- He can create a new account with several subscriptions and people to monitor, each with their own level of access.

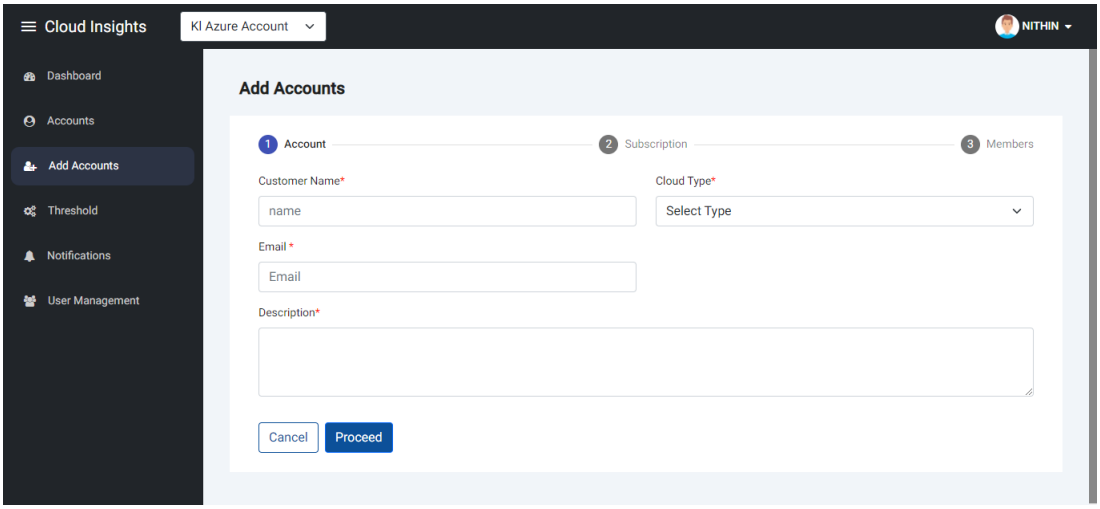

Figure 3.8: Add Account

6. User Management Page

- Only Super Admin has the access to the user management
- He has the authority to add new member to the users list delete and update

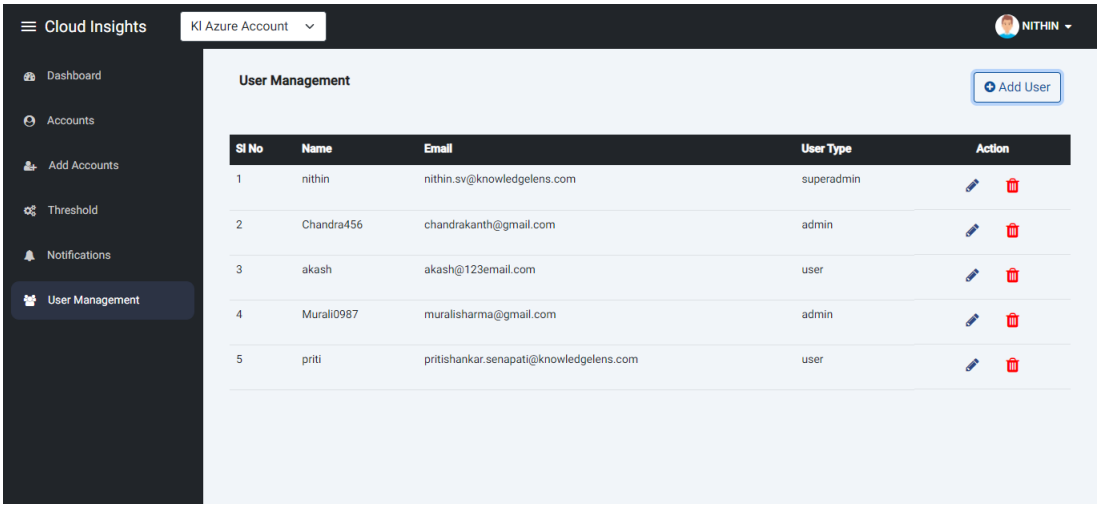

Figure 3.9: Add Account

Chapter 5

CONCLUSION

The project aimed to minimize user efforts to monitor and control pollution as per industrial rules and regulations. First attempt has been made to find the need for the system. To fulfil the user needs, a detailed study has been designed in such a way that it is user friendly and easy to use. This particular system has been designed in an attractive manner, so that even a user with minimum knowledge could operate the system easily.

5.1 Future Enhancement

The system is designed in such a way that addition of new modules can be done without much difficulty. The reconstruction of the system will increase the flexibility of the system. The system has been developed as versatile and user friendly as possible keeping in mind the advanced features in this technology. Now in the system all the SRS requirements are implemented.

References

[1] iuseppe Aceto, Alessio Botta, Walter de Donato, Antonio Pescapè - *Cloud Monitoring: definitions, issues and future directions* - 2012 IEEE 1st International Conference on Cloud Networking

[2] anu Anand - *Cloud Monitor Monitoring applications in cloud* - 2012 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)

[3] iuseppe Aceto, Alessio Botta, Walter de Donato, Antonio Pescapè - *Cloud monitoring: A survey* - April 2013 Survey

[4] ou Yongdnog, Wang Jing, Zhao Zhuofeng, Han Yanbo - *A Scalable and Integrated Cloud Monitoring Framework Based on Distributed Storage* - 2013 10th Web Information System and Application Conference

APPENDIX

Screenshots

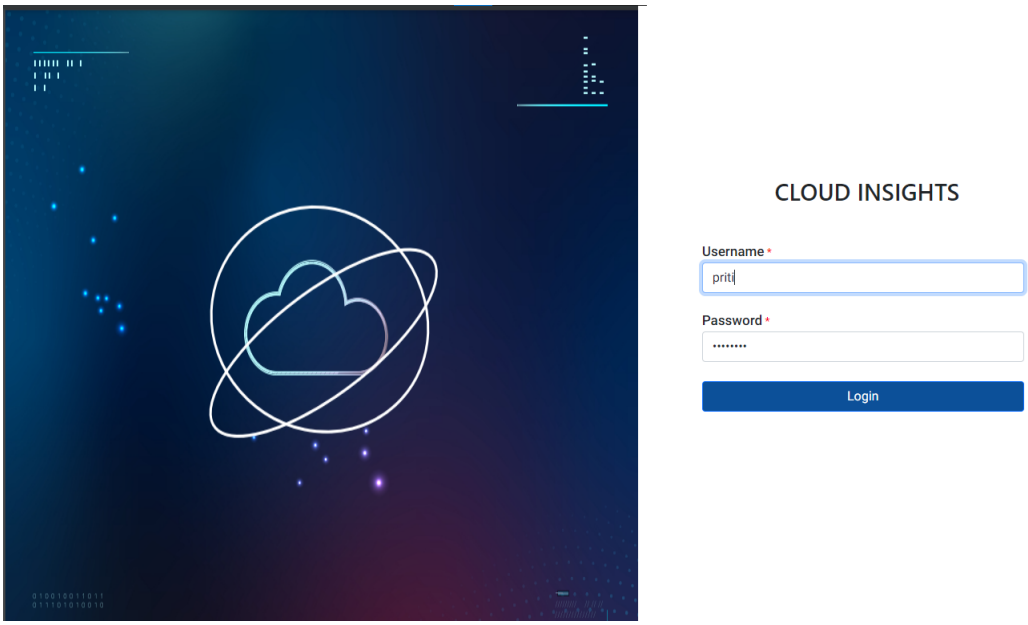

Figure A.1: Login Page

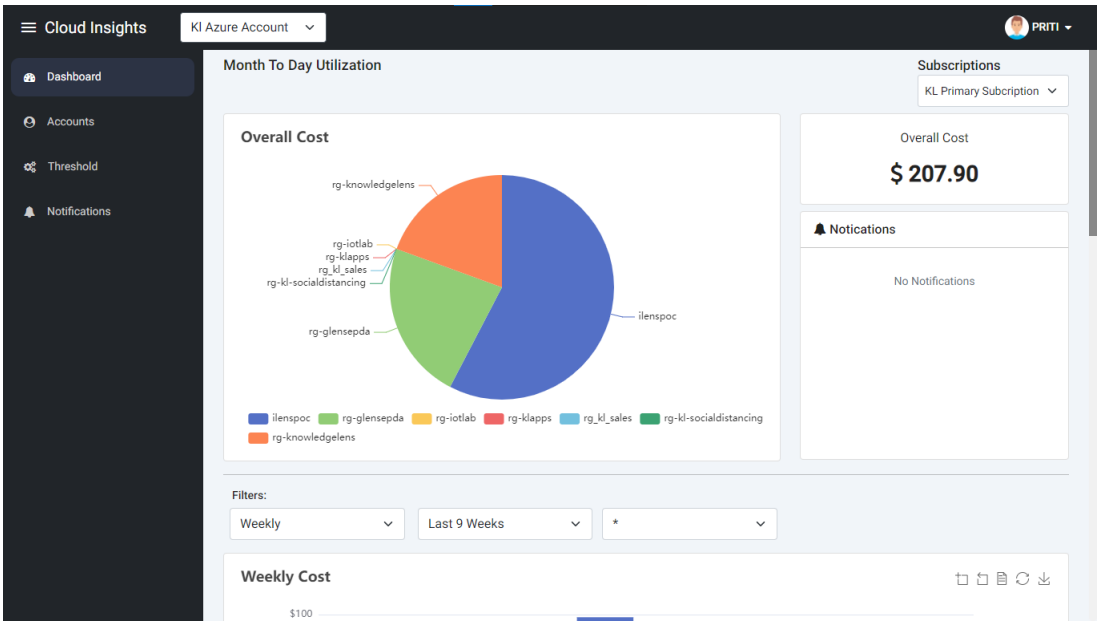

Figure A.2: Dashboard Overall Cost

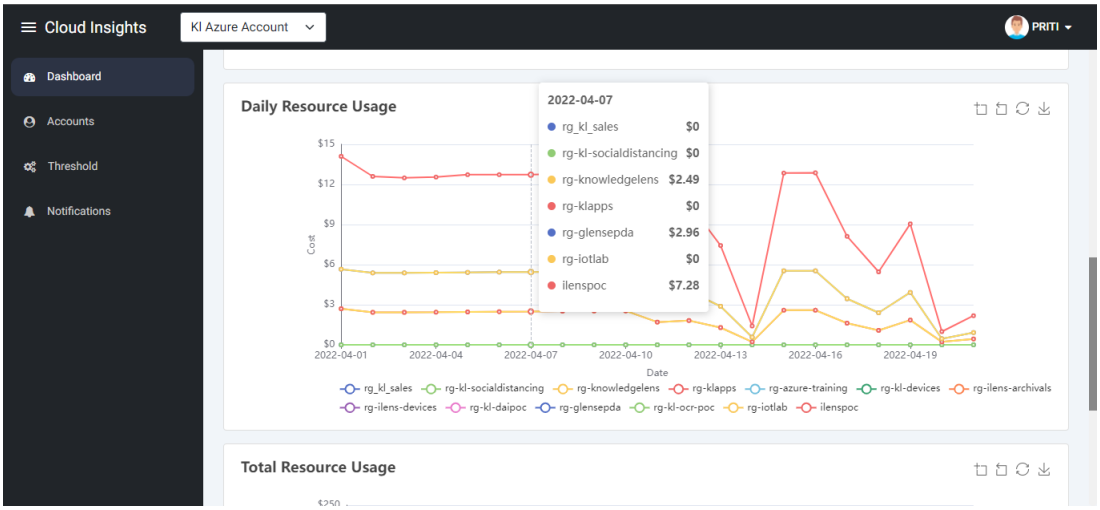

Figure A.3: Dashboard Daily Resource Usage

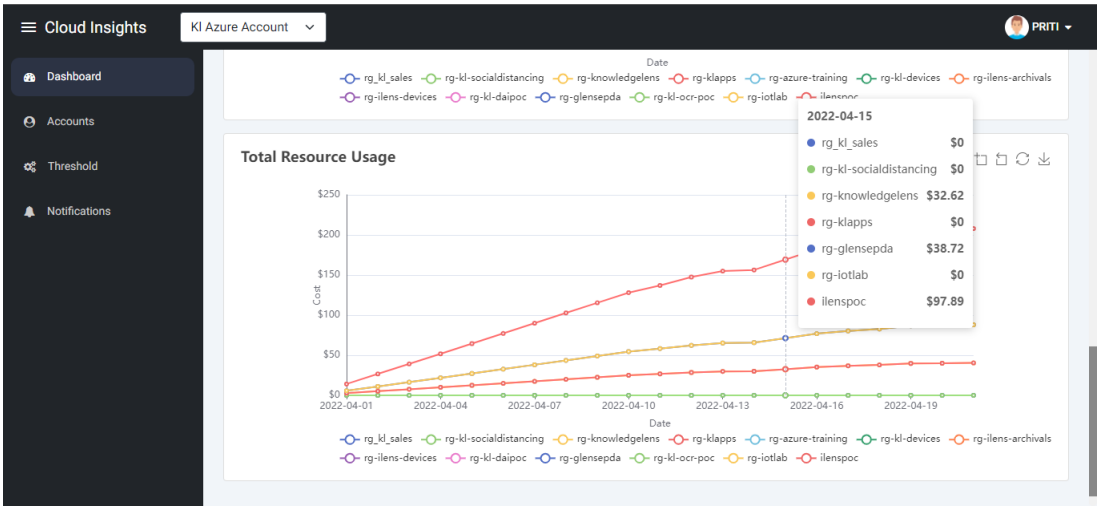

Figure A.4: Dashboard Total Accumulated Cost

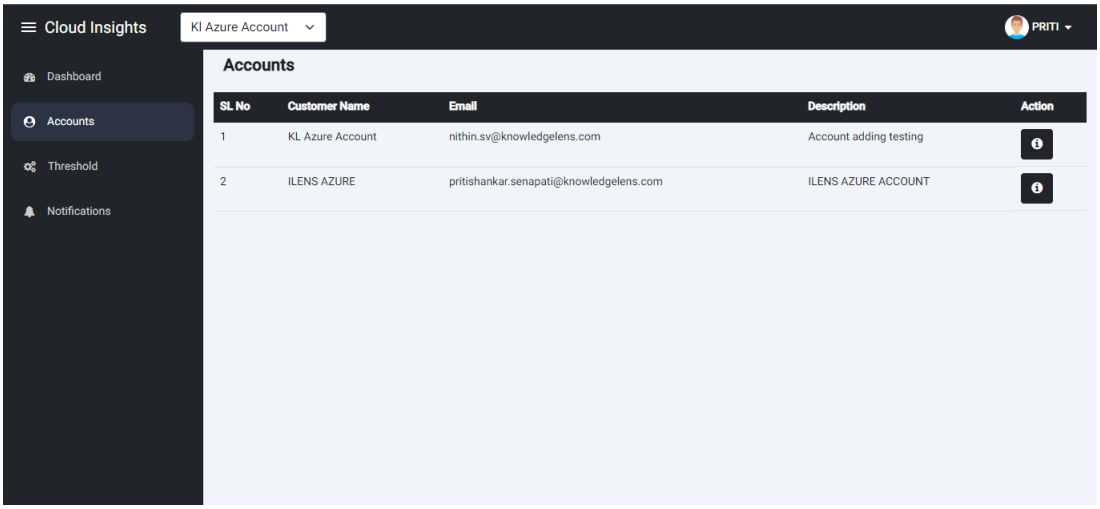

Figure A.5: Accounts Listing

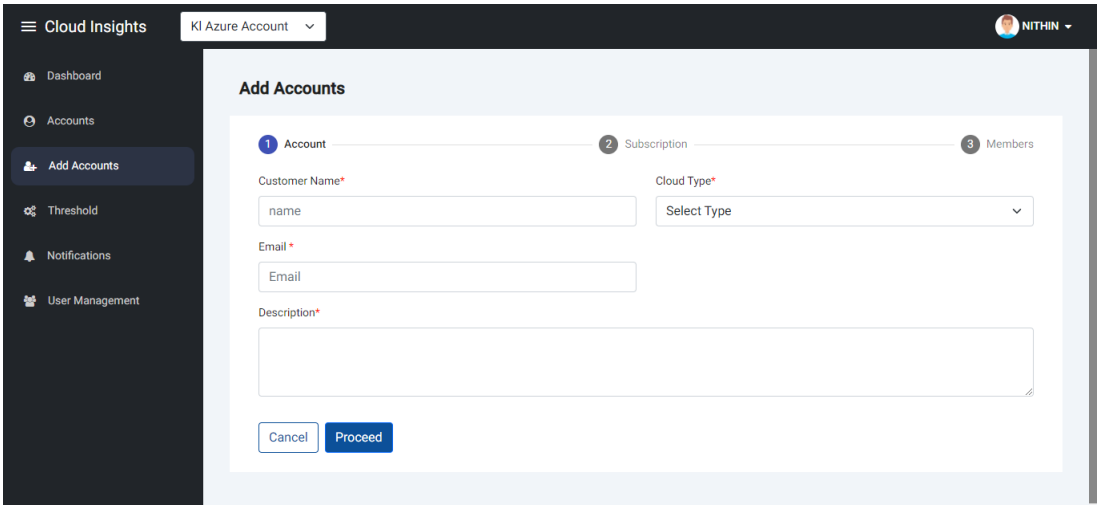

Figure A.6: Accounts Adding

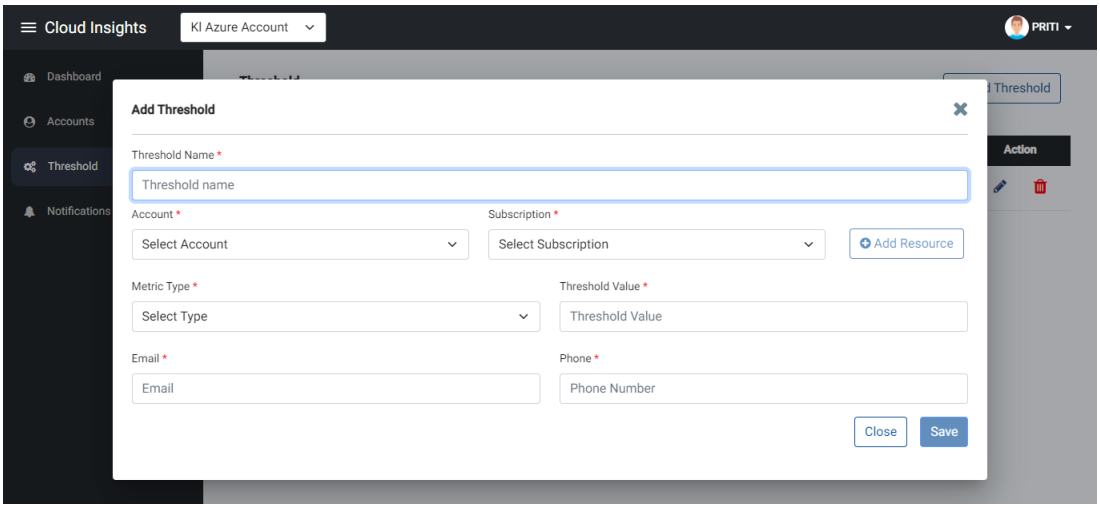

Figure A.7: Add Threshold

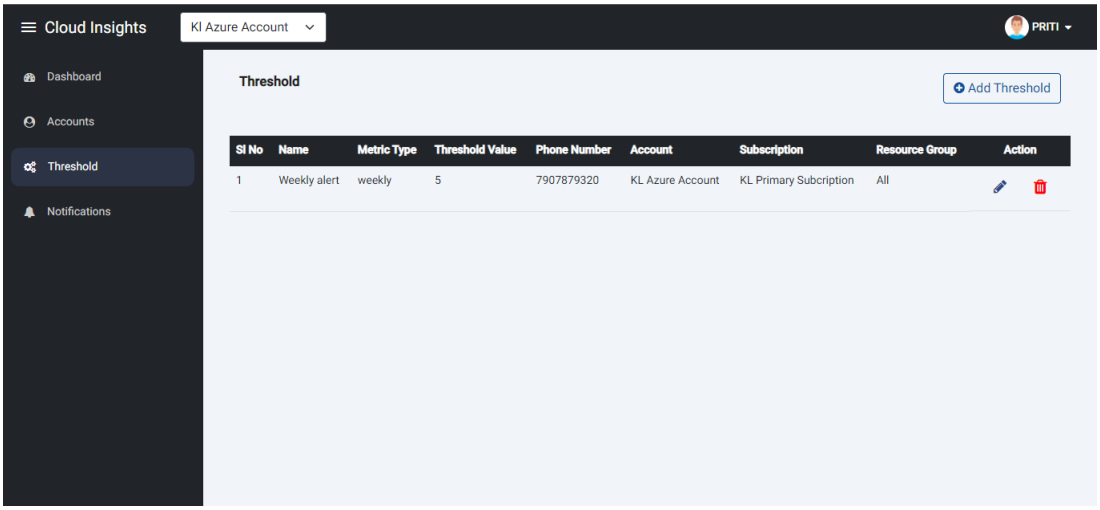

Figure A.8: Threshold Listing

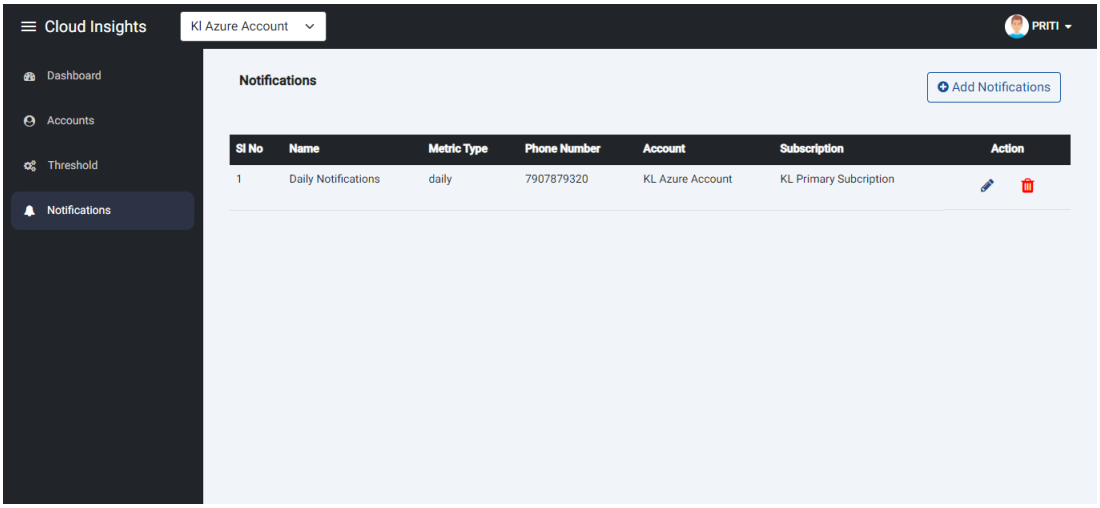

Figure A.9: Notification Listing

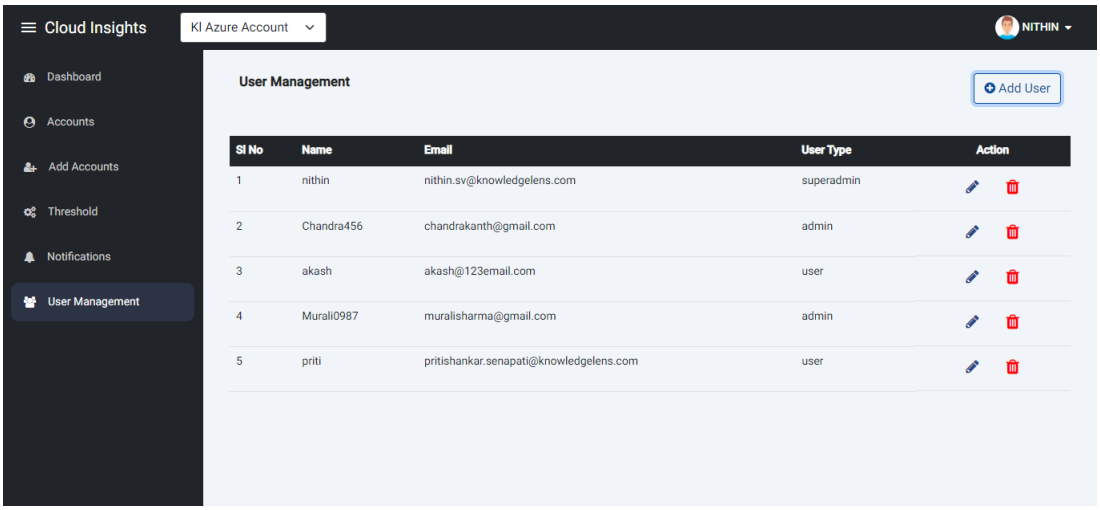

Figure A.10: User Management