**A**

**PROJECT REPORT**

**ON**

**“AWS HOUSE-KEEPING TOOL”**

**SUBMITTED TO**

**SHIVAJI UNIVERSITY, KOLHAPUR**

**IN THE PARTIAL FULFILLMENT OF REQUIREMENT FOR THE AWARD OF DEGREE BACHELOR OF ENGINEERING IN COMPUTER SCIENCE AND ENGINEERING**

**SUBMITTED BY**

**MISS. ANUJA MAHESH CHIVATE 13CMPN02**

**MISS. KIRTI ANIL DEVADIGA 13CMPN08**

**MISS. MARGARET SURESH DONE 13CMPN10**

**MISS. JIDNYASA JAYPRAKASH JANKAR 13CMPN17**

**MISS. POOJA KAILASHCHAND KABRA 13CMPN19**

**UNDER THE GUIDANCE OF**

**Prof. U. A. Nuli**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**DKTE SOCIETY’S TEXTILE AND ENGINEERING INSTITUTE, ICHALKARANJI**

**(AN AUTONOMOUS INSTITUTE)**

**ACCREDITED WITH ‘A’ GRADE BY NAAC**

**An ISO 9001-2015 Certified**

**SHIVAJI UNIVERSITY, KOLHAPUR**

**2016-2017**

**D.K.T.E.SOCIETY’S**

**TEXTILE AND ENGINEERING INSTITUTE, ICHALKARANJI**

**(AN AUTONOUMOUS INSTITUTE)**

**ACCREDITED WITH ‘A’ GRADE BY NAAC**

**An ISO 9001-2015 Certified**

**SHIVAJI UNIVERSITY, KOLHAPUR**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**



**CERTIFICATE**

**This is to certify that, project work entitled**

**“AWS HOUSE-KEEPING TOOL”**

**is a bonafide record of project work carried out in this college by**

**MISS. ANUJA MAHESH CHIVATE 13CMPN02**

**MISS. KIRTI ANIL DEVADIGA 13CMPN08**

**MISS. MARGARET SURESH DONE 13CMPN10**

**MISS. JIDNYASA JAYPRAKASH JANKAR 13CMPN17**

**MISS. POOJA KAILASHCHAND KABRA 13CMPN19**

**is in the partial fulfillment of award of degree Bachelor in Engineering in Computer Science & Engineering prescribed by Shivaji University, Kolhapur for the academic year 2016-2017.**

**Prof. U. A. NULI**

**(PROJECT GUIDE)**

**Prof. (Dr.) D.V.KODAVADE Prof. (Dr.) P.V.KADOLE**

**(HOD CSE DEPT.) (PRINCIPAL)**

**EXAMINER: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**DECLARATION**

We hereby declare that, the project work report entitled “AWS House-Keeping Tool” which is being submitted to D.K.T.E. Society’s Textile and Engineering Institute Ichalkaranji, affiliated to Shivaji University, Kolhapur is in partial fulfillment of degree B.E. (CSE). It is a bonafide report of the work carried out by us. The material contained in this report has not been submitted to any university or institution for the award of any degree. Further, we declare that we have not violated any of the provisions under Copyright and Piracy / Cyber / IPR Act amended from time to time.

MISS. ANUJA MAHESH CHIVATE 13CMPN02 …………………

MISS. KIRTI ANIL DEVADIGA 13CMPN08 …………………

MISS. MARGARET SURESH DONE 13CMPN10 …………………

MISS. JIDNYASA JAYPRAKASH JANKAR 13CMPN17 …………………

MISS. POOJA KAILASHCHAND KABRA 13CMPN19 …………………

**ACKNOWLEDGEMENT**

With great pleasure we wish to express our deep sense of gratitude to Prof. U. A. Nuli for his valuable guidance, support and encouragement in completion of this project report.

Also, we would like to take opportunity to thank our head of department Prof. (Dr.) D. V. Kodavade for his co-operation in preparing this project report.

We feel gratified to record our cordial thanks to other staff members of Computer Science and Engineering Department for their support, help and assistance which they extended as and when required.

Thank you,

MISS. ANUJA MAHESH CHIVATE 13CMPN02

MISS. KIRTI ANIL DEVADIGA 13CMPN08

MISS. MARGARET SURESH DONE 13CMPN10

MISS. JIDNYASA JAYPRAKASH JANKAR 13CMPN17

MISS. POOJA KAILASHCHAND KABRA 13CMPN19

**ABSTRACT**

Amazon Web Services (AWS) offers reliable, scalable and inexpensive cloud computing services. Free to join, pay only for what you use.

AWS compute the cost of resources. Even if the resources are not used, they are in running state which increases the cost. Looking at the high expenses on AWS account, a tool is created which will work as an agent to notify users about their unused AWS cloud resources, which will help in cutting down expenses to a large extent. There will be a look over the security breaches to secure the cloud instances. The AWS resource can be sorted according to user and region using this tool. The historic records of user and the resources can be seen. It automatically generates the summary report of the whole day and notify to all users.

AWS offers a pay-as-you-go approach for pricing for over 70 cloud services. With AWS one can pay only for the individual services as per need, for as long as usage, and without requiring long-term contracts or complex licensing. AWS pricing is similar to how to pay for utilities like water or electricity. Only pay for the services which are consumed, and once stop using them, there are no additional costs or termination fees.

**INDEX**

1. **Introduction 6** 
   1. Problem definition 7
   2. Aim and objective of the project 8
   3. Scope and limitation of the project 8
   4. Timeline of the project 9
   5. Project Cost 9
2. **Background study and literature overview 13**
   1. Literature overview 14
   2. Critical appraisal of other people’s work 14
   3. Investigation of current project and related work 15

1. **Requirement analysis 17**
2. **System design 21**
   1. Architectural Design 22
   2. User Interface Design 23
   3. Algorithmic description of each modules 24
   4. System Modeling 27
      1. Class Diagram 28
      2. Dataflow Diagram 29
      3. Usecase Diagram 30
      4. Activity Diagram 31
      5. Sequence Diagram 32

* + 1. Deployment Diagram 33

1. **Implementation 34**

a. Environmental Setting for Running the Project 35

b. Detailed Description of Methods 36

c. Implementation Detail 37 **<<PgNo>>**

1. **Integration and Testing 39**
   1. Description of the Integration Modules 40
   2. Testing 41
2. **Performance Analysis 45**

1. **Applications 46**
2. **Installation Guide and User Manual 47**
3. **Publication Details 48**
4. **Industry Sponsorship Letters 49**
5. **References 53**

Bibliography 54

1. **Introduction**
   1. **Problem Definition:**

Amazon Web Services (AWS) users who use on-demand cloud computing services leave their AWS resources in running state on the weekends, when the resources are rarely used. The tool helps users to cut down their high expenses by notifying the users to shut down their AWS resources when not in use.

* 1. **Aim and objective of the project:**

- To develop a tool that generates a summary report for the AWS account.

- The tool notifies the user about their unused resources.

- The user can view the historic records by using the tool.

* 1. **Scope and limitation of the project:**

The main purpose of the tool is to cut down the higher expenses of the AWS account users by notifying the users about their unused resources. Thus, the details of the individual user account are monitored within the constraints imposed by the need to optimize Quality of Service.

The parameters which are monitored of the AWS account users are as follows:

* Account Holder Name
* Email ID of AWS account holder
* Login time of the user
* Logout time of the user
* Number of resources utilized by the user.
* Time period of individual resource utilized by the user.

However, the tool will be driven according to Quality of Service, it will aim to minimize the cost and to generate a summary report within user-imposed constraints. Thus, user satisfaction is the prime concern.

* 1. **Timeline of the project:**

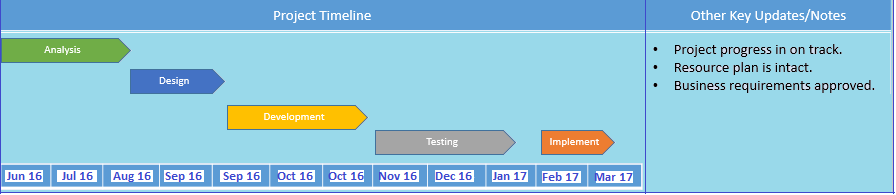
**PROJECT SCHEDULE:**

We have used classic life cycle paradigm for software engineering which is sequential approach to software development that begins at the system level and progress through Analysis, Design, Coding, Testing and Maintenance. The establishment and use of sound engineering principles in order to obtain economically software that is reliable and works efficiently on real machines.

We had completed software requirement analysis by end of November 2016 which encompasses both system and software requirement gathering.

On the basis of design prepared in the previous stage by mid of January we had completed with the coding stage at end of march. Testing was done simultaneously.

Various criteria’s of testing were taken in account such as unit testing, integration testing, validation testing and system testing. First, each and every module of project was tested under the unit testing. Afterwards integration was carried out by integrating all modules. Then final testing was done.

****

JULY

AUGUST

SEPTEMBER

OCTOBER

NOVEMBER

DECEMBER

JANUARY

FEBRUARY

MARCH

* 1. **Project Cost:**

1. Lines of code – 10,127 lines

Effort and total development cost – 5 members , 480hrs.

1. Hardware cost – Laptop with core i3 processor
2. Proprietary software if used – -
3. Electricity used – 8 months
4. Internet cost (while developing) – 8 months - 4TB
5. Any other charges – resources access, server usage, sonarqube
6. **Background study and literature overview**
   1. **Literature overview**

Amazon Web Services (AWS) is a subsidiary of Amazon.com that offers on-demand cloud computing platforms. These services operate from 16 geographical regions across the world. They include Amazon Elastic Compute Cloud, also known as "EC2", and Amazon Simple Storage Service, also known as "S3". As of 2016 AWS has more than 70 services, spanning a wide range, including compute, storage, networking, database, analytics, application services, deployment, management, mobile, developer tools and tools for the Internet of things.

Amazon markets AWS as a service to provide large computing capacity quicker and cheaper than a client company building an actual physical server farm.

* 1. **Critical appraisal of other people’s work**

**AWS Trusted Advisor.**In 2013, AWS released Trusted Advisor which monitors and inspects AWS customer accounts. It provides optimization recommendations concerning deployment security, performance availability and AWS costs.

**AWS Cost Explorer.**Small businesses can't afford to pay for premium support, so AWS offers Cost Explorer. This tool lets customers go to the billing and cost management section of the AWS Management Console and find the AWS Cost Explorer link on the left menu bar. This feature helps users visualize and analyze trends in their monthly and daily spending.

* 1. **Investigation of current project and related work**

AWS is located in 16 geographical regions-

|  |  |
| --- | --- |
| **Region Name** | **Region** |
| US East (N. Virginia) | us-east-1 |
| US East (Ohio) | us-east-2 |
| US West (N. California) | us-west-1 |
| US West (Oregon) | us-west-2 |
| China (Beijing) | cn-north-1 |
| Asia Pacific (Mumbai) | ap-south-1 |
| Asia Pacific (Seoul) | ap-northeast-2 |
| Asia Pacific (Singapore) | ap-southeast-1 |
| Asia Pacific (Sydney) | ap-southeast-2 |
| Asia Pacific (Tokyo) | ap-northeast-1 |
| EU (Frankfurt) | eu-central-1 |
| EU (Ireland) | eu-west-1 |
| EU (London) | eu-west-2 |
| South America (São Paulo) | sa-east-1 |
| AWS GovCloud (US) | us-gov-west-1 |

AWS has announced another 3 Regions and 7 Availability Zones in China, India, and the United Kingdom coming online throughout 2017. Each Region is wholly contained within a single country and all of its data and services stay within the designated Region. Each Region has multiple "Availability Zones", which are distinct data centers providing AWS services. Availability Zones are isolated from each other to prevent outages from spreading between Zones. Several services operate across Availability Zones e.g., S3. While others can be configured to replicate across Zones to spread demand and avoid downtime from failures. As of December 2014, Amazon Web Services operated an estimated 1.4 Million servers across 28 availability zones.

The global network of AWS Edge locations consists of 54 points of presence worldwide, including locations in the United States, Europe, Asia, Australia, and South America.

In 2014, AWS committed to achieving 100 % renewable energy usage. As part of this effort in the United States, AWS commissioned with Community Energy of Virginia a solar farm coming online in 2016 to support the US East region.In January 2015, AWS announced it has teamed with Pattern Development to construct and operate Amazon Wind Farm Fowler Ridge. In July 2015, AWS announced that it has contracted with Iberdrola Renewables, LLC to construct and operate Amazon Wind Farm US East. In November 2015, AWS announced that it has contracted with EDP Renewables to construct and operate Amazon Wind Farm US Central. AWS is also working with Tesla Motors to apply battery storage technology to address some power needs in the US West (Northern California) region.

**3.Requirement analysis**

1. Housekeeping tool have two users –
2. Super admin user
3. Normal user

**Super admin user** –

Super admin user is the person from the admin team who is responsible for managing all umbrally activity accounts.

This particular person has access to master account access key and secret key.

An account may have one or many super admin user.

**Normal user** -

Normal user is an individual user who uses the AWS account in the organization.

This particular person is admin of sub account.

This person can track own account only.

1. Super admin user can login to the system and logout from the system.

Post / login

Post / registerSuperAdmin

Post / logout

1. CRUD on User information.
2. User can change the profile picture.
3. User can change the password.
4. User can view the description and other information.
5. User can have onboard new AWS account which is already in umbrally account.

CRUD -

1. Store access key and secret key.
2. Store, edit, update and delete.

* On boarding date
* Budget in Dollar
* HSPS Id – This is the other tool reference id
* Project or Account Expiration date
* Free trial Expiration date
* Services information which are during the HSPS request
* Example – T2.micro instance
* Account type
* Example- Enum
* Paid by the client
* Self paid
* HSPS Expiration date
* Email Id of the account holder
* Employee name of the account
* Project Name
* Project Id

1. Super admin can schedule email report

* E-mail report must be stored in the database
* User can browse the historical report
* Adding the scheduling report

1. Choose single or multiple AWS account
2. Two E-mail Ids
3. Scheduling time
4. More information message
5. Option to choose the view

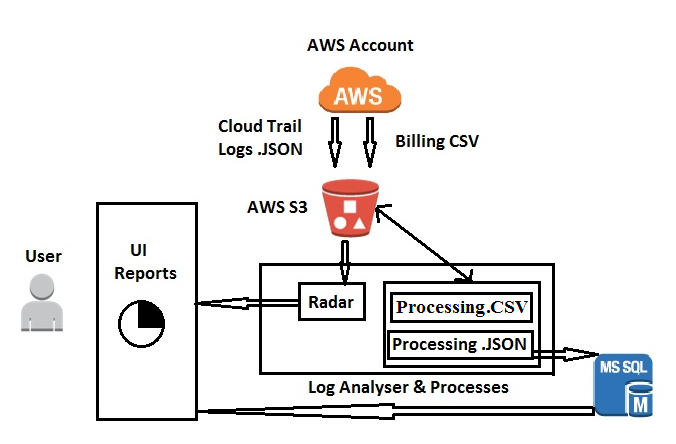
* Graph
* Table

1. User can choose different services
2. User can choose usage report accordingly
3. User can choose and view the billing report
4. If available with the Access key for an account, then option to send the detailed report of running instance
   * + Security information, IAM report
5. User have the option to download the scheduled report in the html format when clicked on download button
6. Cloud trail log processing tool

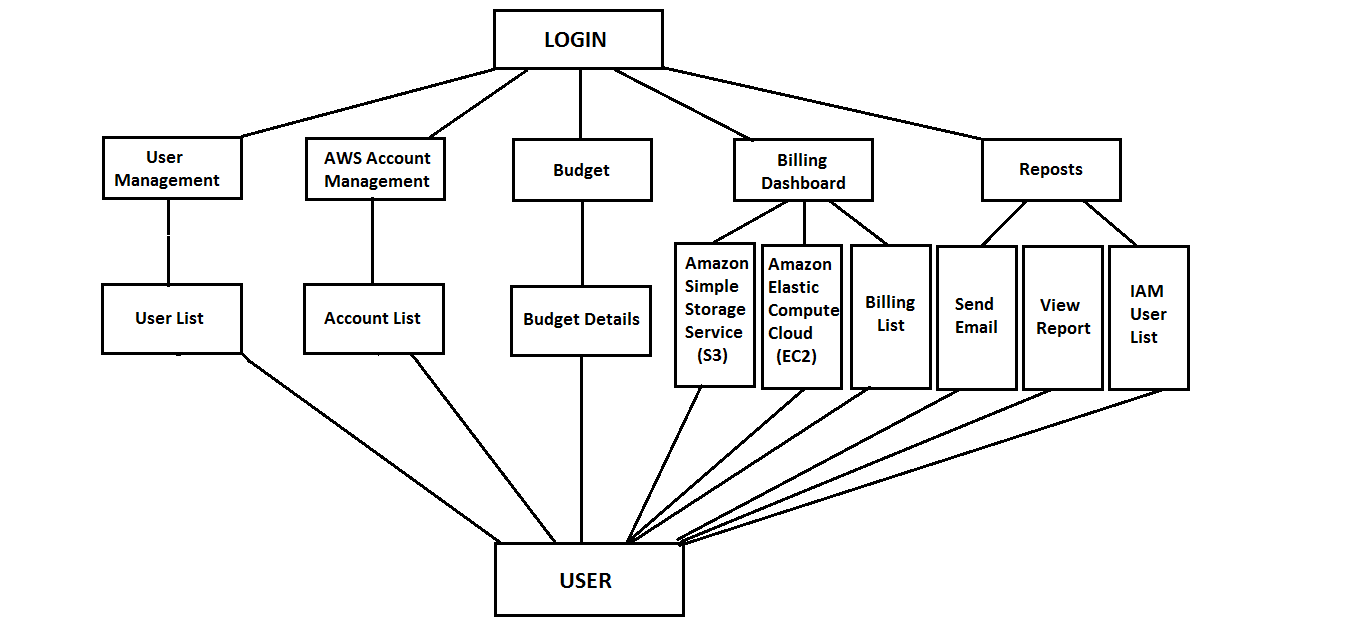
* User can configure cloud trail logs as the input
* Bucket name
* Account number
* Processing enabled/disabled
* CRON timing of log processing

**4.System design**

* 1. **Architectural Design**

****

**b.User Interface Design**



**c.Algorithmic description of each modules**

1. User management module

Algorithm

Step1: Select user management.

Step2: User can view the details of user information.

Step3: User can add new user.

Step4: User can enter all the fields of user management.

2. Account management module

Algorithm

Step1: Select account management.

Step2: User can view the details of account information.

Step3: User can add new account.

Step4: User can enter all the fields of account management.

3.Budget

Algorithm

Step1: Select budget.

Step2: User can view the details of budget information.

Step3: User can add budget.

Step4: User can enter all the fields of budget.

4.Billing dashboard

Algorithm

Step1: Select billing dashboard.

Step2: User can view the details of billing dashboard.

Step 3: Select resource security to view details.

Step 4: User can view the details of the security.

5.Reposts

Algorithm

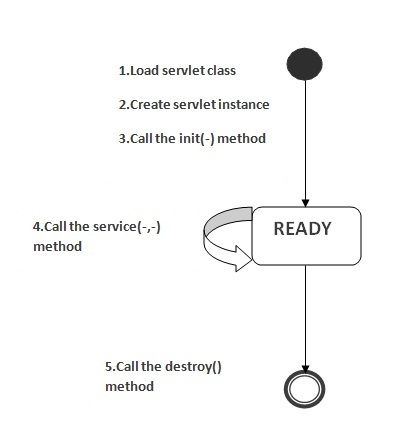
Step1: Select reposts.

Step2: Email can be send to user.

Step3: User can view the IAM list of users.

Step 4: User can view report.

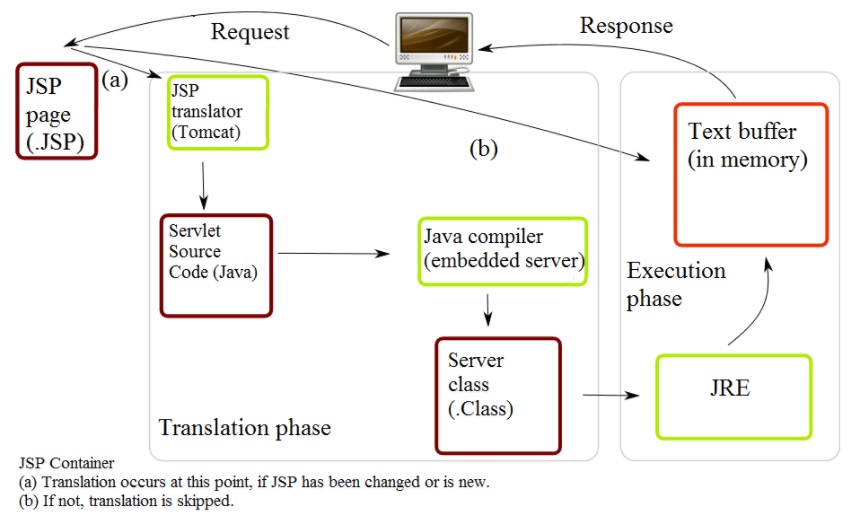
**Diagrammatic view of each individual module**



**Step 1:** Servlet class of each module is loaded by using annotational configuration.

**Step 2:** Each module has attributes defined in database corresponding to the attributes defined for each model.

.

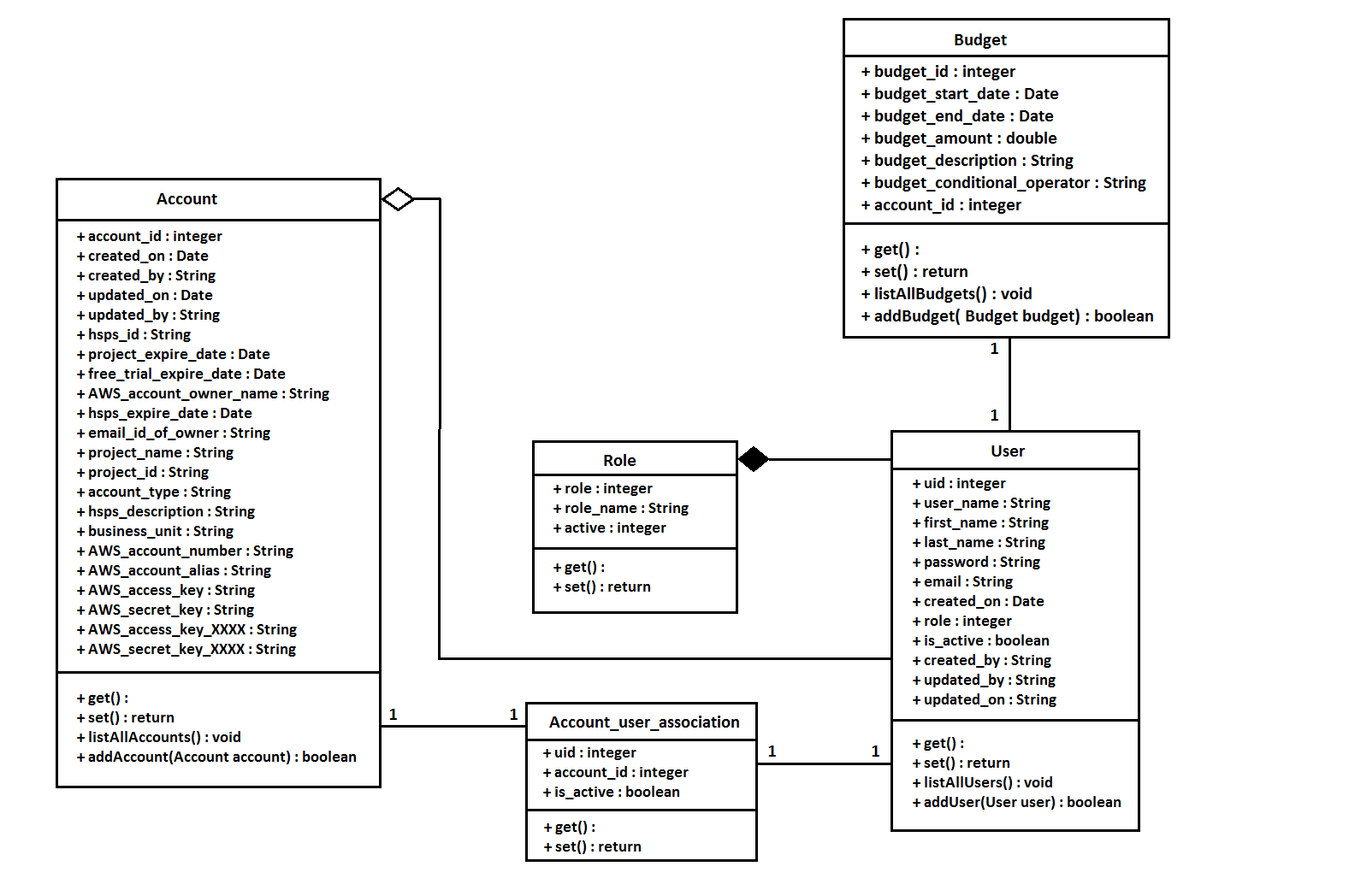
****

**Step 3:** Each module interacts with corresponding service implementation.

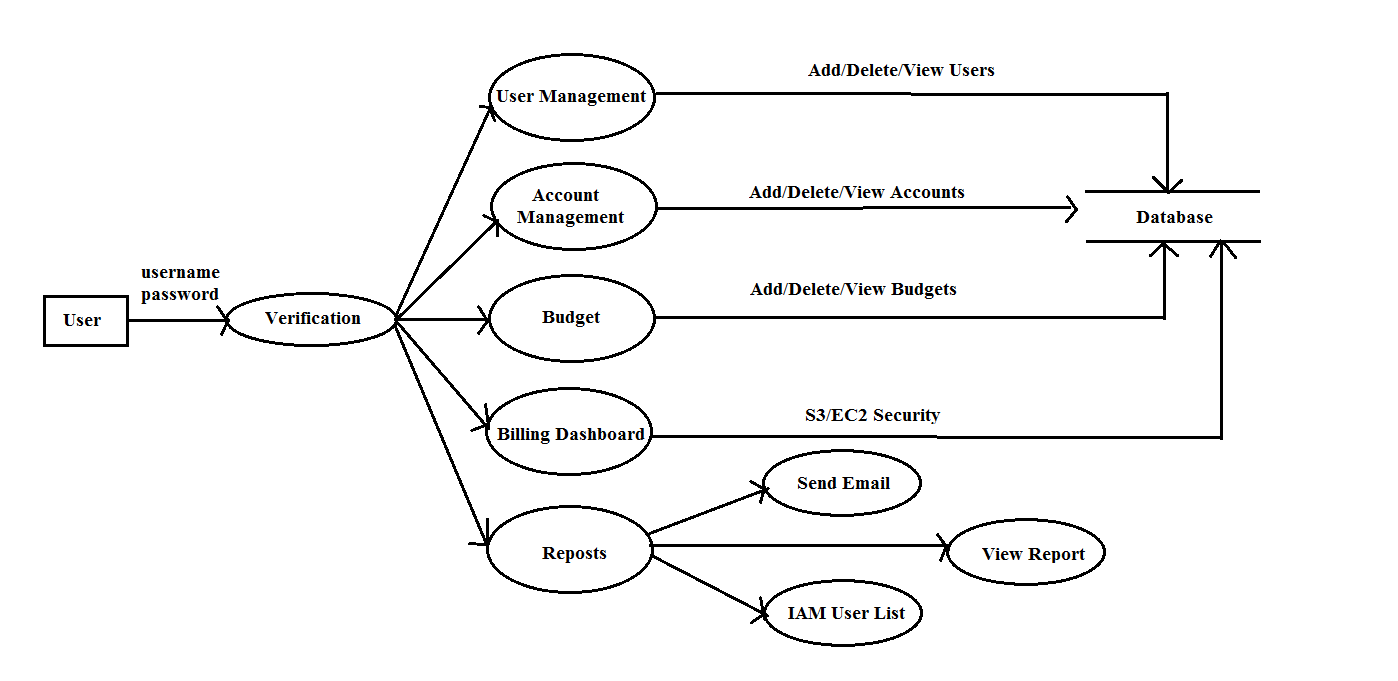
**Step 4**: Each module calls the corresponding Java Server pages JSP page

**d.System Modeling**

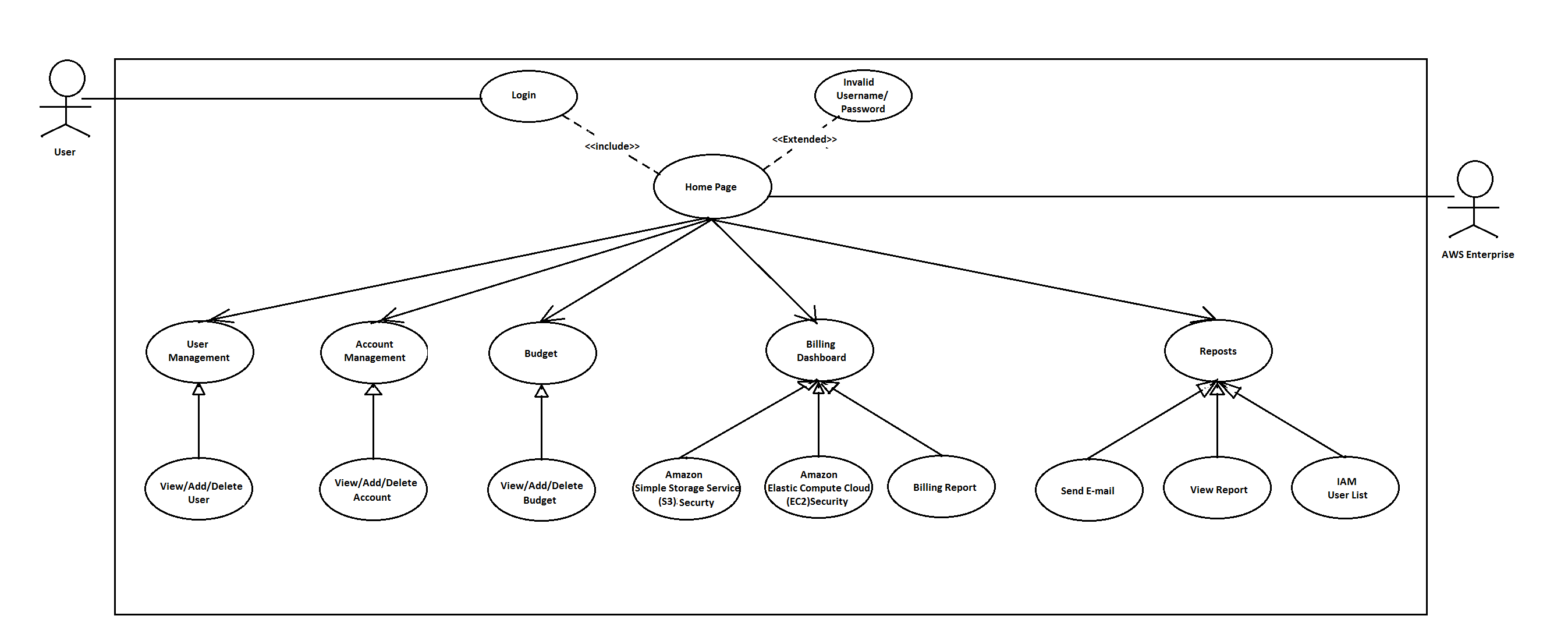
* + 1. **Class Diagram**

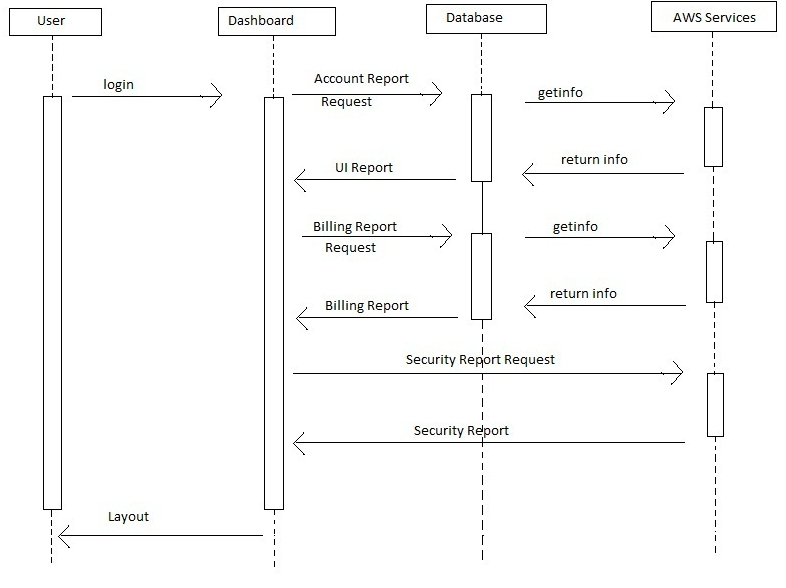
****

* + 1. **Dataflow Diagram**

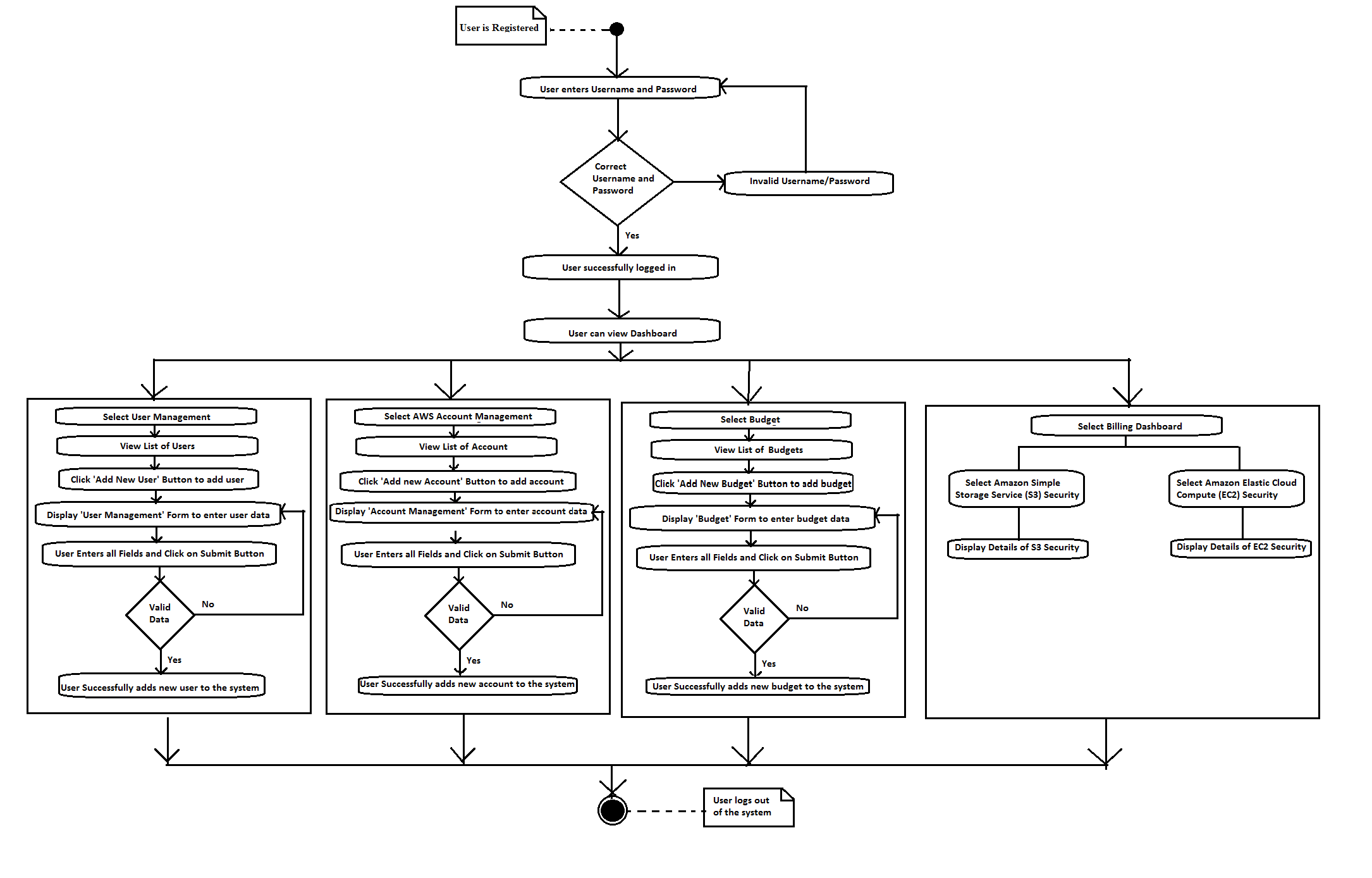
****

* + 1. **Use case Diagram**

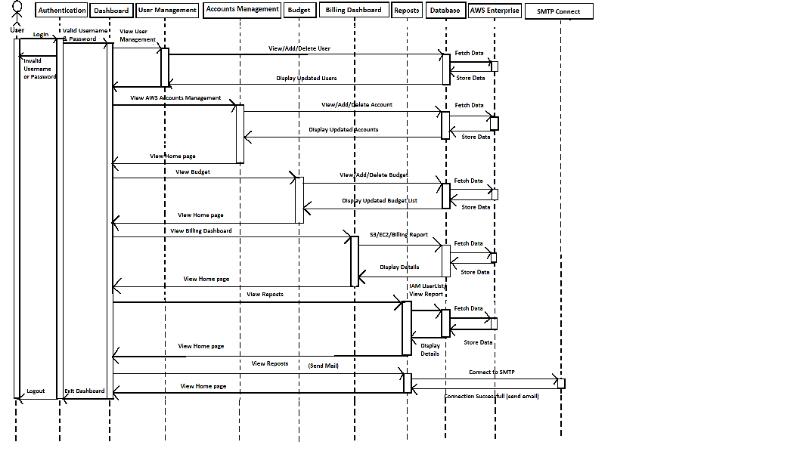


****

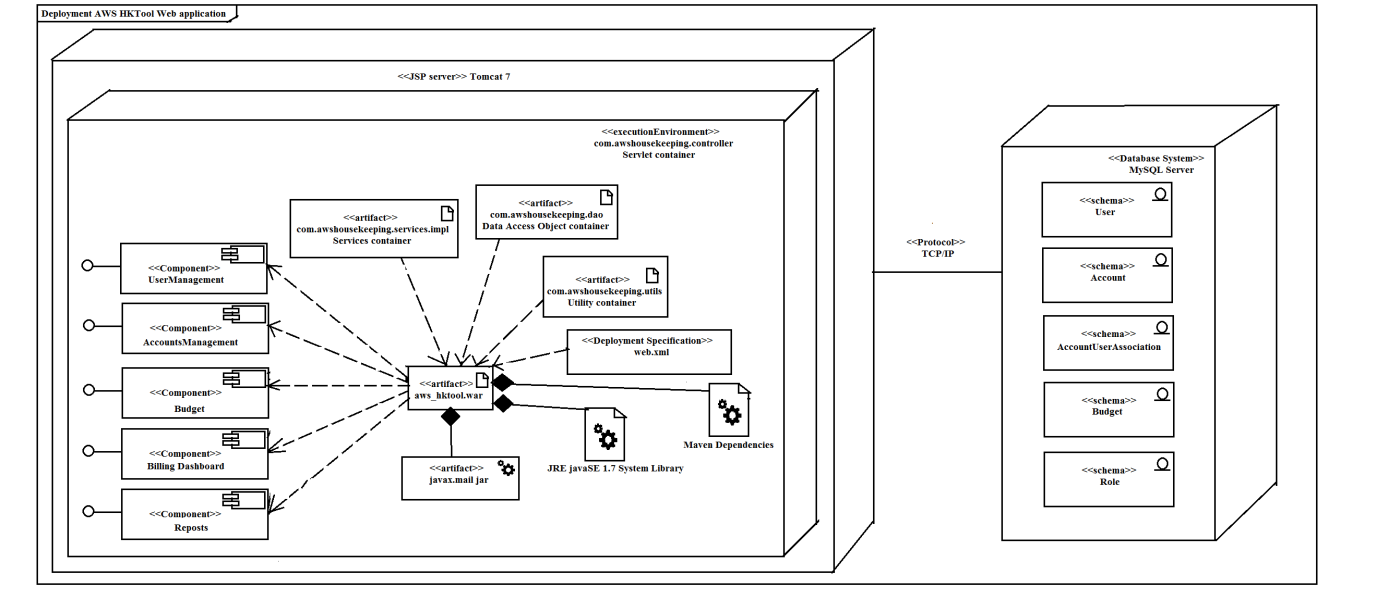
* + 1. **Activity Diagram**

****

* + 1. **Sequence Diagram**

****

* + 1. **Deployment Diagram**

****

**5.Implementation**

* 1. **Environmental Setting for Running the Project**

**IDE**: Ecilipse Java EE Neon

**Database**: MySQL server 5.7.17

**Languages**: Java version 1.8.0.121, Javascript, HTML5, CSS3, JSP, servlet.

**Tool**: Maven dependencies, Javax.mail-1.5.

Maven repository are built automation tool primarily used for java projects.

Java mail API necessary for sending E-mail.

**b. Detailed Description of Methods**

The different modules –

1. **User Management**

To list the User details - List<User> listAllUsers()

To add new user - boolean addUser(User user)

1. **AWS Accounts Management**

To list the AWS Accounts details - List<Account> listAllAccounts()

To add new account - boolean addAccount(Account account)

1. **Budget**

To list the Budget List<Budget> listAllBudgets()

To add new Budget - boolean addBudget(Budget budget)

1. **Billing**

To get S3 Security details -

List<Bucket> listAWSs3Buckets(int accountid)

To get EC2 Security details -

Map<String, List<Instance>> getAllRegionsEc2Resources(int accountId)

1. **Reposts**

// Connect to Amazon SES using the SMTP username and password you-

connect(HOST, SMTP\_USERNAME, SMTP\_PASSWORD)

// Send the email -

sendMessage(msg, msg.getAllRecipients())

**c. Implementation Details <<PgNo>>**

* **com.awshousekeeping.controller**

The Java Servlet program extend the capabilities of the server. Although Servlet can respond to any types of requests, they implement applications hosted on Web servers. Servlets are used to process a Java class in Java EE that conforms to the Java Servlet API, a standard for implementing Java classes that respond to requests. Servlets communicate over client-server protocol, but they are used with the HTTP protocol. Thus "servlet" is used as shorthand for "HTTP servlet".

All the java servlet programs are embedded in com.awshousekeeping.controller.

* **com.awshousekeeping.dao**

 A data access object (DAO) is an object that provides an abstract interface to database. By mapping application calls to the persistence layer, the DAO provides specific data operations without exposing details of the database. This isolation supports the Single responsibility principle. It separates what data access the application needs, in terms of domain-specific objects and data types (the public interface of the DAO), from how these needs can be satisfied with a specific DBMS, database schema. (implementation of the DAO).

All the Data Access Object interfaces are embedded in com.awshousekeeping.dao.

* **com.awshousekeeping.dao.impl**

The design pattern is equally applicable to the following:

1. Programming languages
2. Software with persistence needs
3. Database

It is traditionally associated with Java EE applications and with relational databases.

All the Data Access Object classes are embedded in com.awshousekeeping.dao.impl.

* **com.awshousekeeping.services**

All the services interfaces are embedded in com.awshousekeeping.services.

Interfaces are as follows:

1. LoginService interface
2. UserService interface
3. AccountService interface
4. BudgetService interface
5. AWSListEc2ResourcesService interface
6. AWSS3ListService interface

* **com.awshousekeeping.services.impl**

All the services classes are embedded in com.awshousekeeping.services.impl.

Classes are as follows:

1. LoginServiceImpl interface
2. UserServiceImpl interface
3. AccountServiceImpl interface
4. BudgetServiceImpl interface
5. AWSListEc2ResourcesServiceImpl interface
6. AWSS3ListServiceImpl interface

* **com.awshousekeeping.utils**

All the utilities classes are embedded in com.awshousekeeping.utils.

Classes are as follows:

1. AllAWSClientBuilder class

2. BusinessException class

3. CommonUtility class

4. DatabaseConnect class

5. LoginRequiredFilter class

**6.Integration and Testing**

**a.Description of the Integration Modules**  Integration testing is logical extension of unit testing. In simple form, two units that has been already tested are combined into a unit module. This refers to an integrating multiple unit module into one unit.

The idea is to test combinations of pieces and eventually all the modules making up a process are tested together.

**b.Testing**

Sonarqube- Static Analyser and testing is performed using the tool.

### Overall health

Your project home page shows where you stand in terms of quality in a glimpse of an eye. This main page also shows you an immediate sense of the good results achieved over time.

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Testcase Title** | **Expected Outcome** | **Description** |
| 1 | Login | Login successfully and assign the particular user to own account. | In that allow access for particular user to own account |
| 2 | Unsuccessful user verification due to wrong password | Login to the system with wrong password | Login should fail with an error ‘Invalid Password’ |
| 3 | Unsuccessful User Verification due to invalid login id. | Login to the system with an invalid login id. | Login should fail with an error ‘Invalid User Id’. |
| 4 | View user details | The account detail is inserting into the particular account. | In that user can access the account to view as per requirement |
| 5 | Not accessed any service | Only creating the user account. | No information are stored or not required |
| 6 | Adding new user | Adds the new user to the account | store the information in the database |
| 7 | Deleting existing user | Delete particular user from the account | Remove the information of that user from the database |
| 8 | Adding new account | Adds the new account to the system | store the information in the database |
| 9 | Tracking Budget | Budget should be always tracked by the system | Displays how much from budget user has use the services |
| 10 | Generate Billing Report | Display billing report of the particular user for own account | Allow to view the billing report for registered account |
| 11 | Notifying user | Notify the unused resources and report to the user | Notification has been sent to the user |

**7.Performance Analysis**

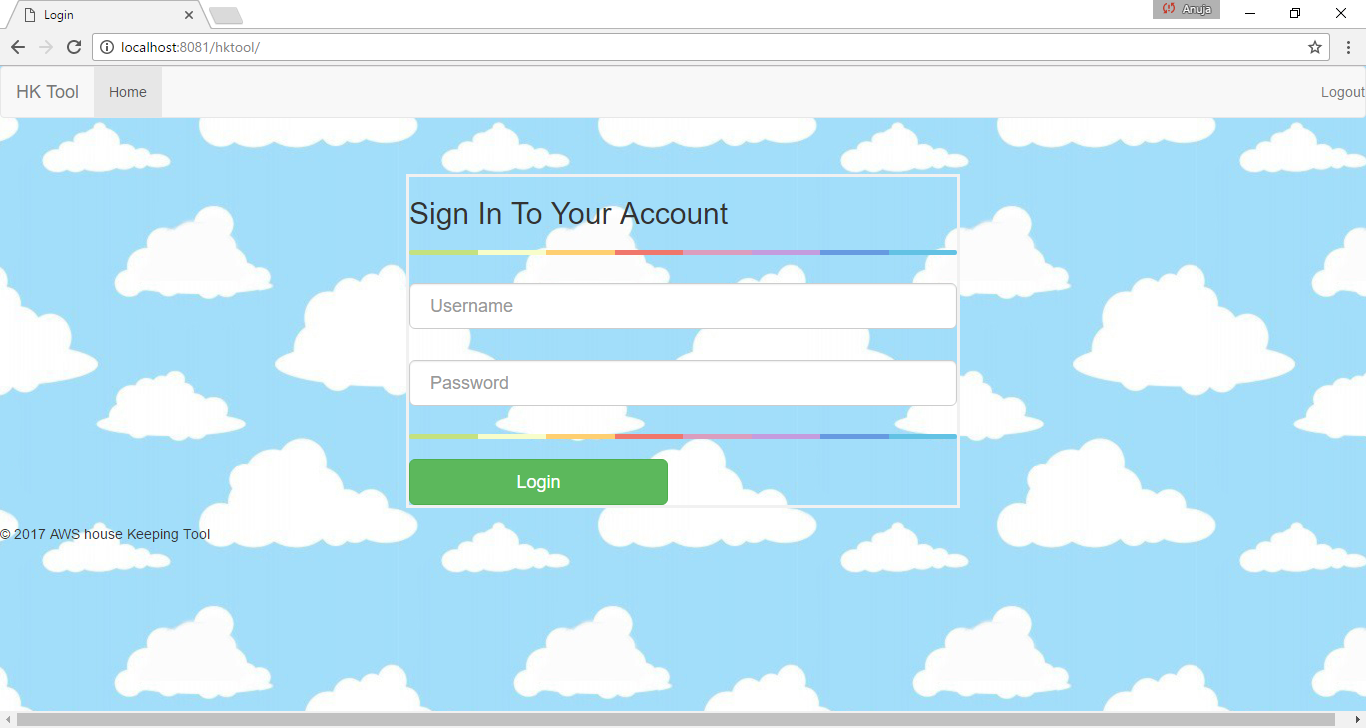
**Performance Analysis**

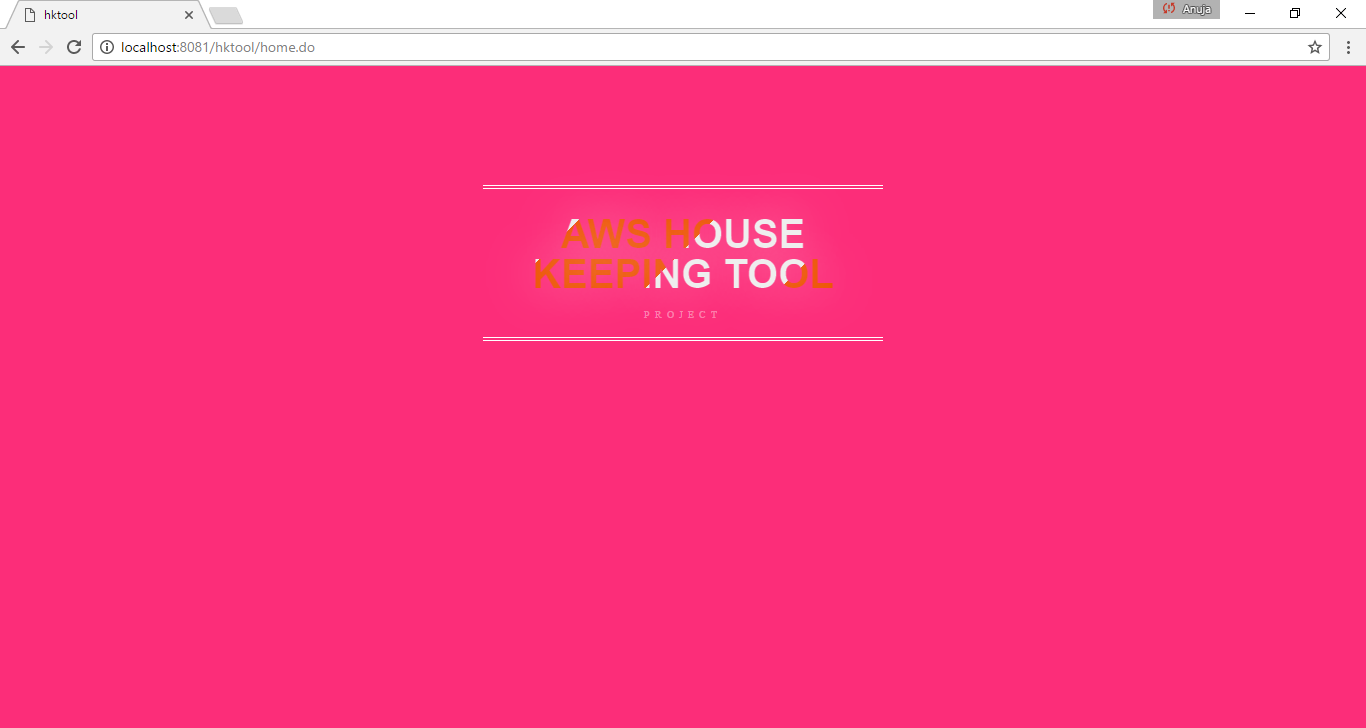
* The input data to the system is in the form of JSON and CSV files.
* The data extracted after processing the files is stored in database.
* There is continuous monitoring and updating in database.
* This system is a standalone system.

**8.Applications**

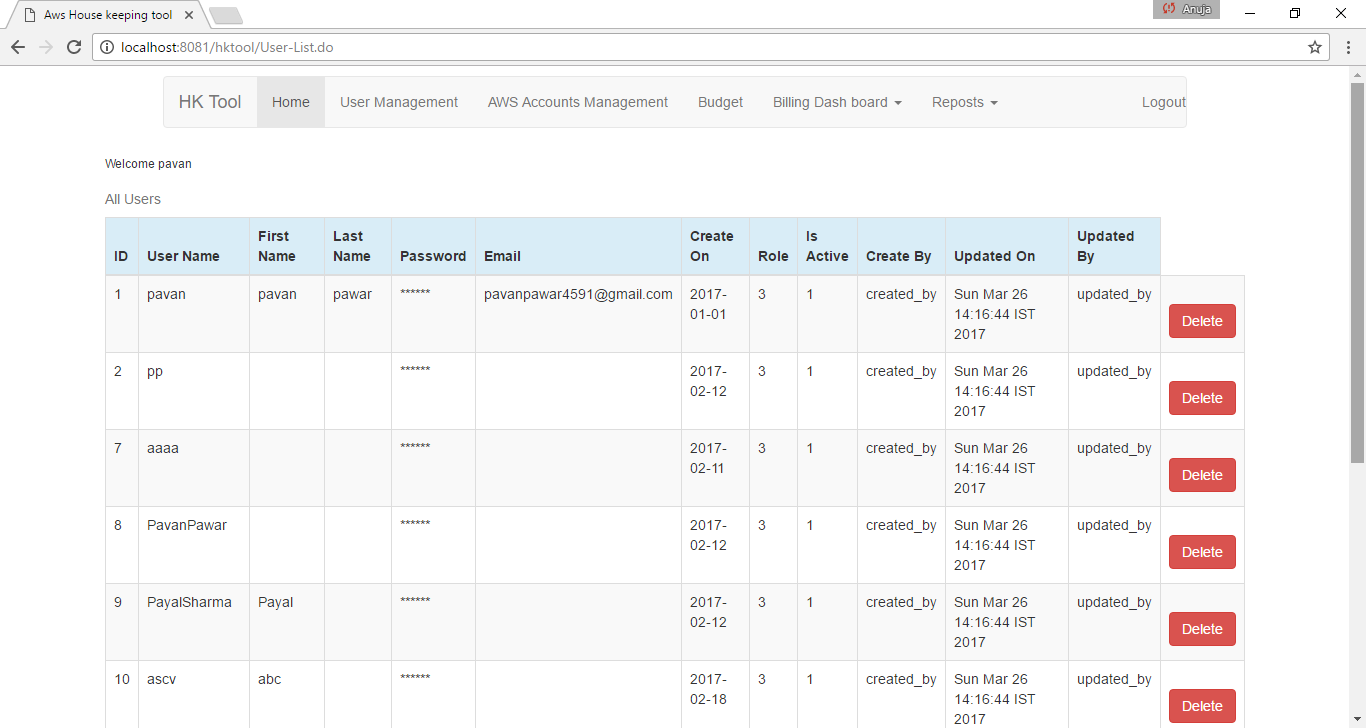
**9.Installation Guide and User Manual**

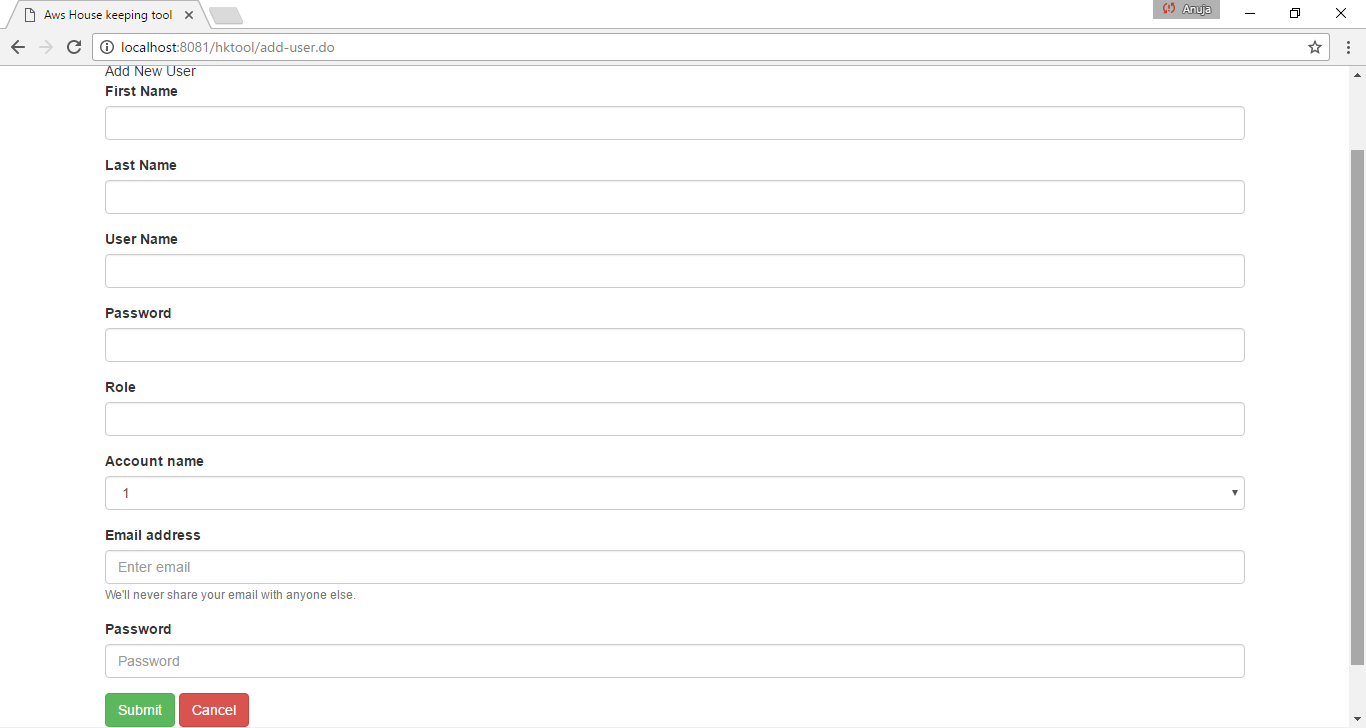
**-Login Page**



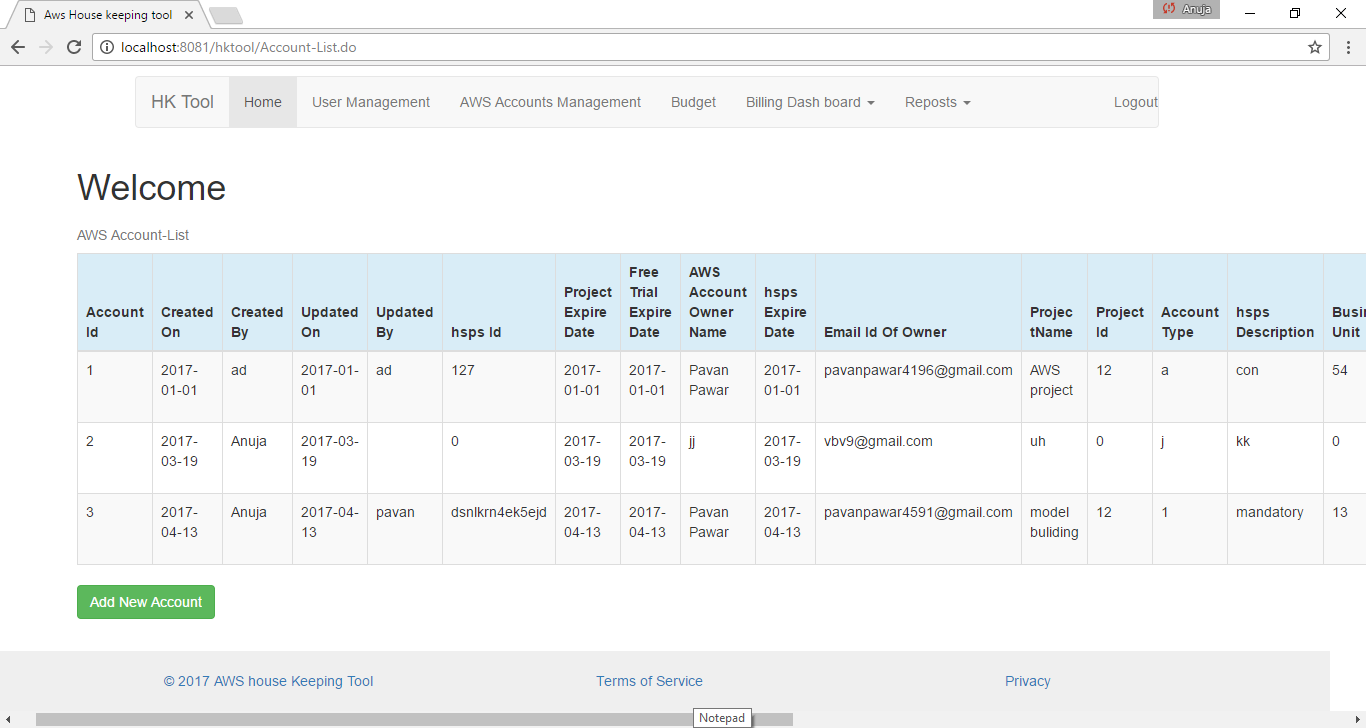
**-Home page **

**-User Management page**

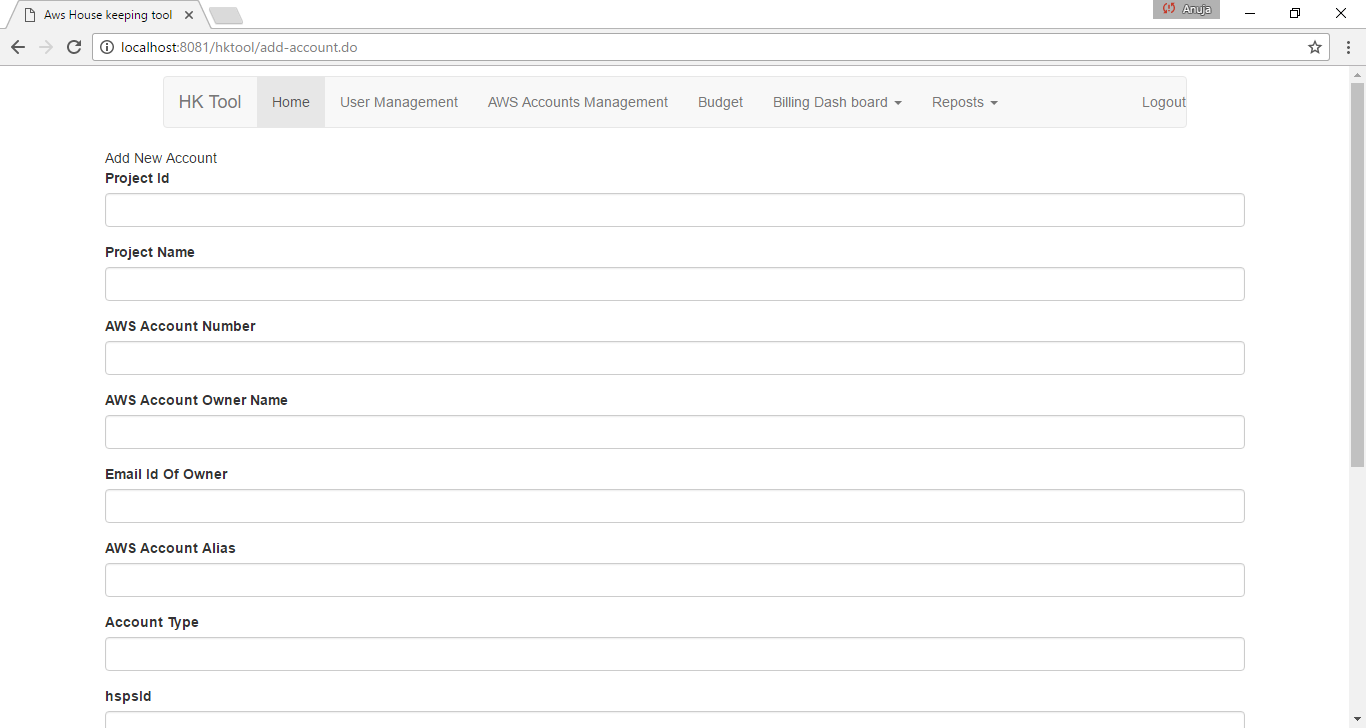
****

**-Add new user page **

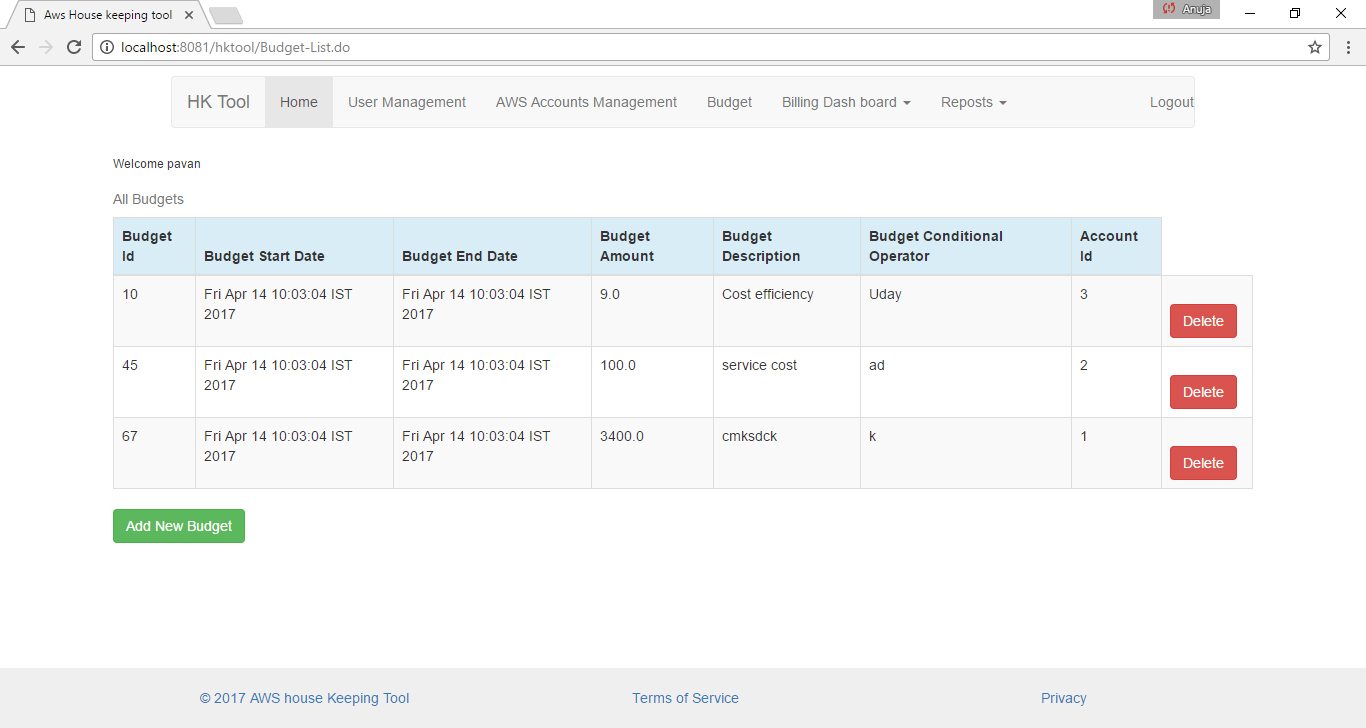
**-AWS Account management page**

****

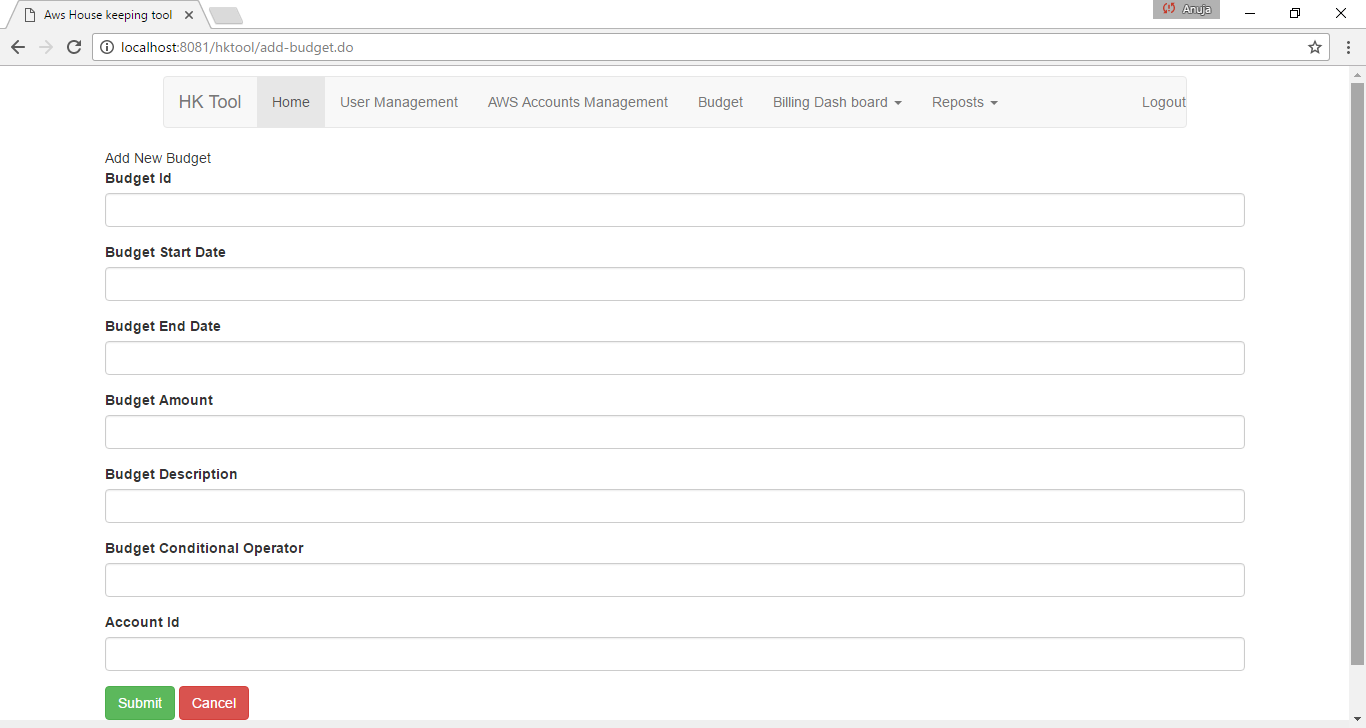
**-Add Account page**

****

**-Budget page**

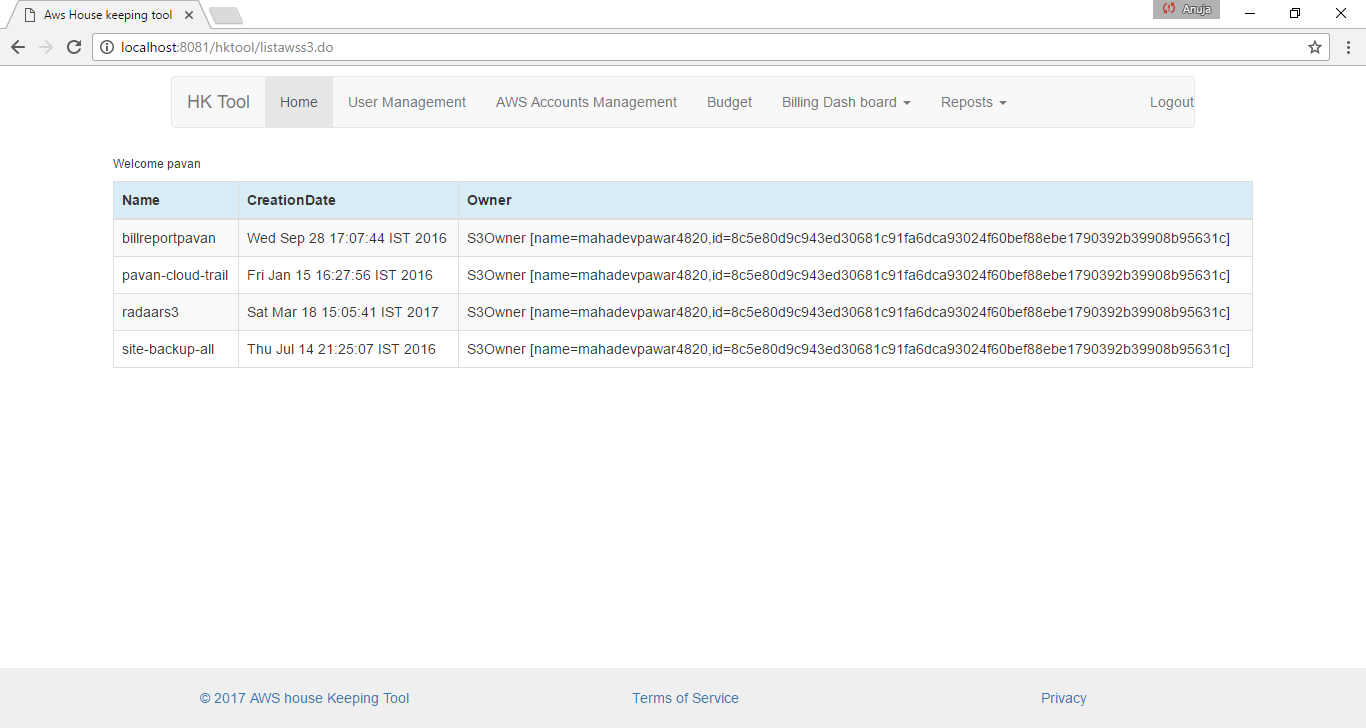
****

**-Add Budget page**

****

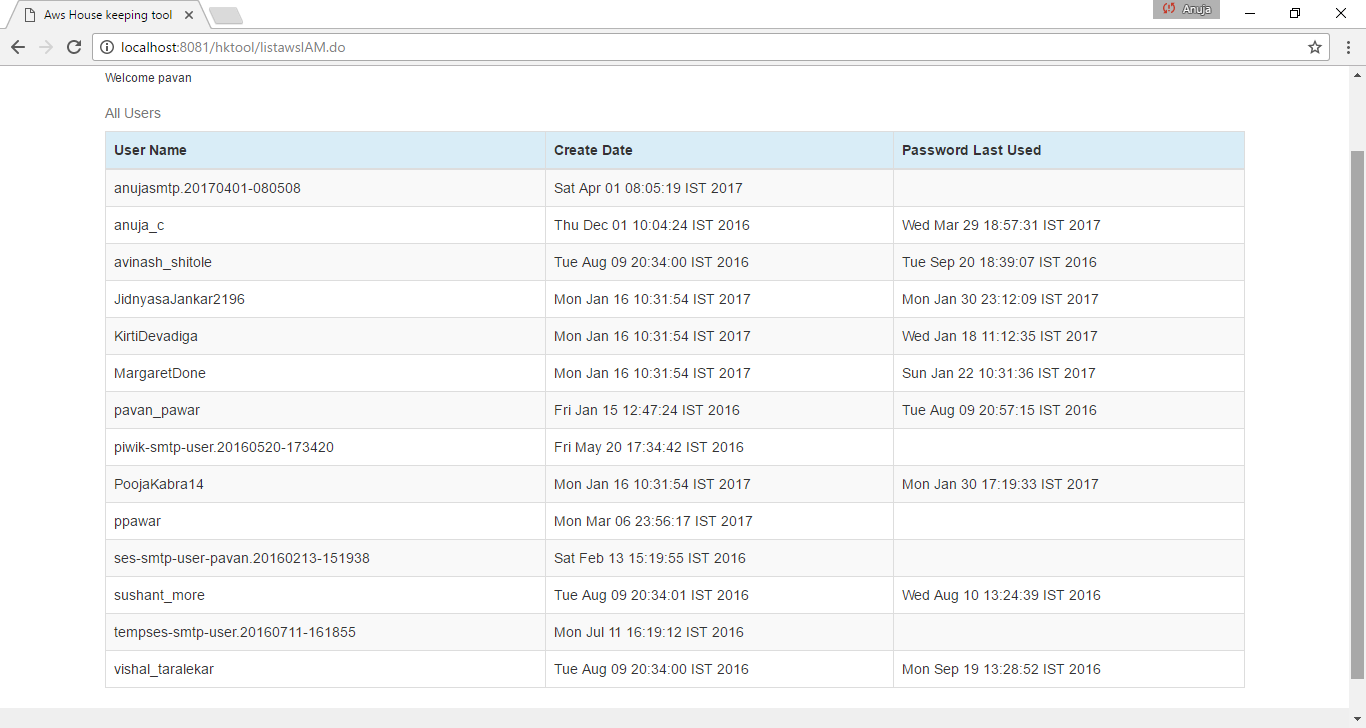
* + - * **Billing Dashboard**

**-Amazon Simple Storage Service Security**



-**User login**

**- IAM user**



**10.Publication Details**

**11.Industry Sponsorship Letters**

**12.References**

**References**

* Amazon Web Services

Available: [http://www.aws.amazon.com](http://www.aws.amazon.com/)

* White Papers - Security, logging

Available: [http://www.aws.amazonl.com/whitepapers](http://www.aws.amazonl.com/whitepapers/)

* Cloud Computing

Available: [http://www.cloudcomputing.com](http://www.cloudcomputing.com/)

* Sonarqube

Available: <https://www.sonarqube.org>

* Netflix-ICE tool

Available: <https://netflix-dives-into-aws-usage-monitoring-with-ice.org>

### [Maven – Download Apache Maven](https://maven.apache.org/download.cgi)

Available: <https://maven.apache.org/download.cgi>

**Bibliography**