the UNIVERSITY of GREENWICH

Project 2.23

Electric Power Consume Management

Supervisor: Vo Thanh Duy

Student ID: 000901372

Full name: Nguyen Tien Khoa

Abstract

This project is completed as a result to prove that I earn knowledge from study program at the University of Greenwich. In this project, I will analysis, design, develop and maintain a system, which use to manage consume electric number.

This report contains all of project documents, that I make during build this project. Such as technology analysis, class diagrams, work flow.

Acknowledgment

I would like to say thanks so much to PhD. Vo Thanh Duy who is supervisor of this project. He provides useful guide and tutorial to help me done almost of project requirement. Moreover he also encourage and give a lot of worth advice to help me building this project.

Contents

A. Introduction

Vietnamese Electric Power Company has some problem to collect data from consume monitor device. So, they need a program to collect data information from those devices quicker and cheaper. The program should support to get data about electric consume on monitor device though internet. Moreover, It can help customers to be aware about electric bill. Therefore, company can track consume electric of their customer then give good advices.

Because of using internet to collect data, the system must provide sever. And all devices must to have internet connection. But, device works 24/7 to monitor electric consuming, internet connection would be interrupted and device would be down sometimes by some reason. So, server should have a strategy to deal with offline device.

Surely, monitor devices are not perfect. They may be offline, conflict data, dirty data, etc. So, server should be save these problems to write report. Report will be read by administrators, the electric engineers. Then, they can track to solve problems faster and better. Therefore, the new system will satisfy customer and improve productivity and quality.

In short, the new system manage monitor device to collect data from customer then print it to report. It helps Vietnamese electric company manage and support electric consuming customers.

Open point

Customers regard to their money, product and service. Example, they need to know when electric bill become expensive, why electric lost, etc. So, system should have SMS, email or something to notify customer about their information.

Customers should able to use the new system to interact and discuss about problems and needs.

B. Analysis

Base on requirement of this project and my experiences in Java, Java programming language is used to develop. Focus on available tools, low price and fast development, this project includes 3 modules to contain web service, device simulator and share project.

Web service, this is the main project.

Device simulator, this is project to simulate real device. Actually, I don't have any real device, this simulator would support to make system be visible.

Share project, this contain model classes, constant is used by both service and device.

I. Resources analysis

1. Human resources:

This project is a personal coursework, therefore I plan to develop by one Java developer.

2. Time resources:

Deadline is November 24, 2016

3. Tools resources:

In the first product, this application will be used for education purpose therefore I will use all the application or other related to freeware.

Java programming language is used to develop, because there are a lot free Java frameworks and libraries.

MySQL Database has community version is free to download.

Linux OS work faster with Java, and it is free. I developed this product on Linux OS environment, Ubuntu 16.04 LTE base on Debian and Fedora 24 base on Red Hat.

Eclipse neon is IDE used to develop this product. Of course, it is totally free.

All frameworks, libraries and software are used to developed this project belong to Apache Software Foundation, Red Hat Community, Software in the Public Interest Inc, Pivotal Software, Oracle Corporation, The Eclipse Foundation and PrimeTek Informatics.

II. Requirement analysis

Program must use java open source technique for keeping low price.

Program must delivery within 3 months with main functions.

Program must use an incremental development model for integrating other requirements.

Program must allow device to register into system automatically.

Program must be limited access, Administrator only on this version.

Program must allow admin to modify customer data.

Program must have a report reviewer, such as dashboard.

Program must have calculator to built electric bill on demand.

Program should validate device information to track problems on devices.

Program would allow admin to modify consume policy data.

Program would allow administrator edit some properties on device.

Program should allow registering new user account.

Program should update device on database when device reconnect.

Program should poll devices every 10 minutes, in this version.

Device Simulator should able to simulate a device error.

Device Simulator should able to be polled by server.

Device Simulator should able to send data to server.

Device Simulator should able to store data in internal memory, such as files.

III. Technology analysis

1. **Spring Framework:** System must have server to provide service. With the main target is faster and cheaper, using open source library to build the service. Base on those target, Spring framework seem to be the good choice with wide technique support and easy to use. On top, it is a famous Java framework is used by many java projects. Spring framework has a lot of libraries with strong support other technology. In this project, Spring framework support Spring Security, Spring web, integrate with Hibernate, MySQL and JMS ActiveMQ.
2. **Runnable JAR:** In this article, monitor device is not developed and built so that a virtual device run on JVM is using to simulate real device. I considered to use Spring Shell, but It is a bit heavy with this requirement and I have no experience. So Spring Shell will be expensive. Runnable JAR is the most light weight technique to build device simulator.
3. **Java Message Service (JMS) and Active MQ:** In this project, deeply knowledge about Network is not require. For connection between device and server, JMS is good technique. ActiveMQ service is build to store, forward and broadcast to device and receive message from devices. Device and server just connect to ActiveMQ for communication through IP Address, port and protocol of ActiveMQ server broker. Therefore, it takes less time in researching new technology. It cost less time, mean it is cheaper. So, ActiveMQ is used to communicate between service and devices. Moreover, Spring framework has good support to JMS and ActiveMQ.
4. **JavaScript Object Notation (JSON):** There is no important personal information is used. So, JSON is consider to used to convert Java Object to Text for transfer in communication. The reason are easy to use, easy to read, fast speed, light weight. Jackson JSON is used instead of Google GSON because of speed and feature powerful.
5. **Java Sevlet Facet (JSF) and Primefaces:** It is famous of powerful and less Javascript technology. For user interface, Javascript is difficult to Java developer like me. Primefaces framework is supported to integrate with Spring framework by Spring webflow library. So, with a bit Javascript knowledge and time to research spring webflow and Primefaces, JSF is considered to be best choice to build UI.

   By the way, JSF is MVC component-oriented and spring web is MVC event-oriented. In this project JSF is use to make up UI, and their component is spring bean, Spring framework manage all component instead of JSF.

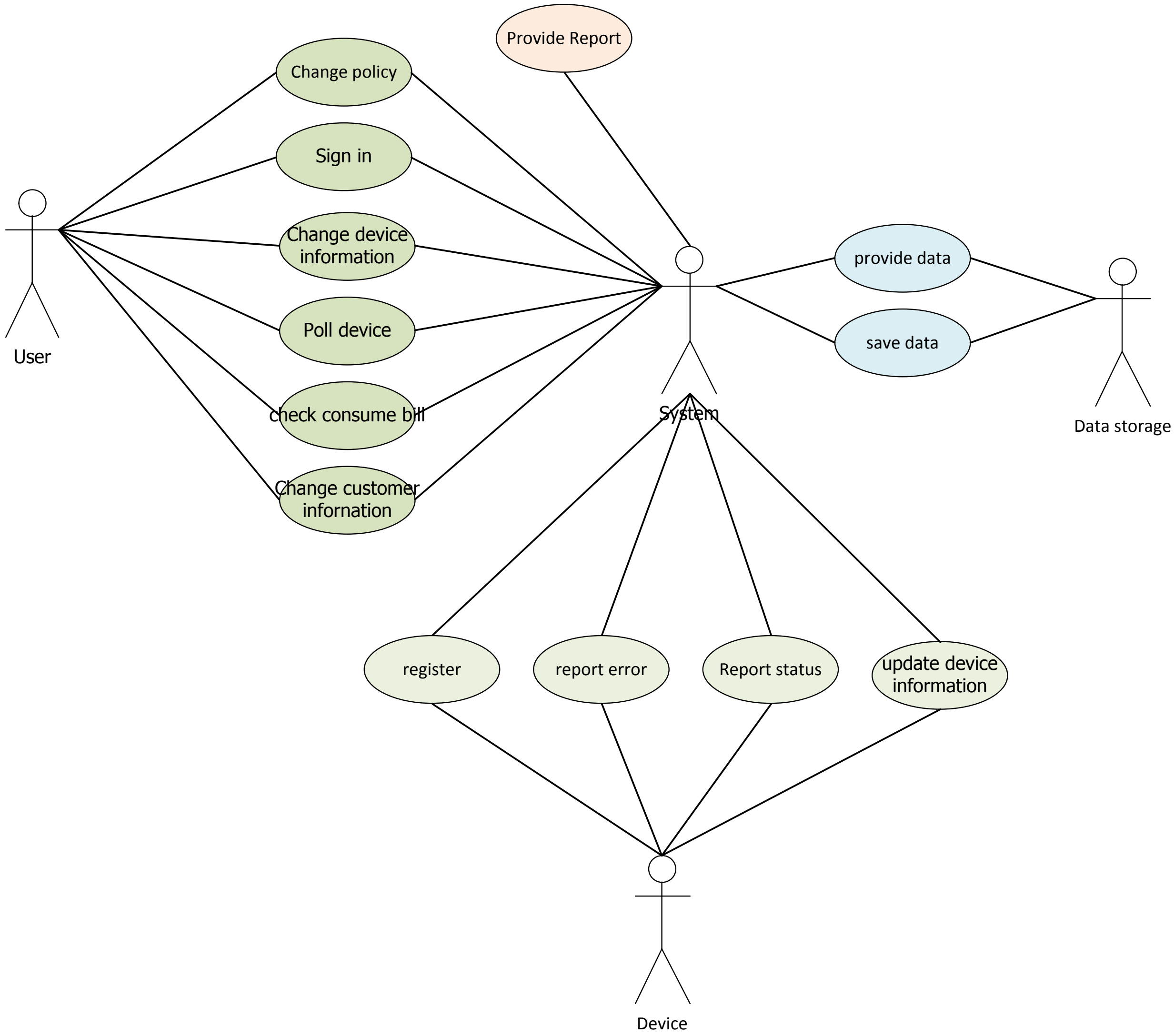
   The benefit of using spring bean to manage is the strength of Dependency Injection, @Inject or @Autowired annotation. This support help spring bypass arguments to a component easily. By Using this, I can manage components and services better.

6. **MySQL:** System have Database server to store device information and customer information. MySQL is used to build a database server. In future, with a huge database, the service should separate into 2 server, one other will use No-SQL data like mongoDB for lighter and faster query device information. Because the security and restrains on device information is not import.
7. **Hibernate:** It is a famous framework support Java Persistence Access(JPA). There is no reason to avoid Hibernate, this easy to use and powerful. But, I consider to remove it when separate data into SQL Database and No SQL database. In this case, Spring Database is good choice to support JPA because It supported both JPA and MongoDB.
8. **Spring webflow and JSF:** As I mentioned in JSF, Spring webflow is the solution to integrate JSF with Spring web MVC. Spring web MVC and JSF are 2 difference framework use Model View Controller (MVC) Architecture, so that they need a midleware to work together. Moreover, Spring webflow also support to control life cycle and navigation such as define variable and view, follow step and check state. In this version, I just use Spring webflow to integrate Spring web MVC and JSF, especially Primefaces.
9. **Spring Security:** Spring security support protect server from unexpected access. Encryption is support by Spring security by many algorithm such as MD5 on 32 characters and Bcrypt on 64 characters. Of course, It supports access with JPA database.
10. **Apache Maven:** A build tool from Apache, it must work with other project come from Apache Software Foundation. Apache has a repository, which provide many java libraries. Other build is strong and famous as Maven is Gradle, but I am familiar with Maven. Therefore Maven is the final choice to build this project.
11. **Apache Tomcat:** A web server from Apache. It has small plug-ins on apache maven which can run instantly when compile and package progress finish. Because of using maven, tomcat is using to deploy web service, a web server come from Apache Software Foundation.
12. **Java Development Kit 1.8:** supported for using functional method for easier understand code and lesser aware of how the code work. In JDK 1.8, Lambda Expression is also supported. Therefore, code is shorter and easier to read and understand. Coders and readers don't need to aware how code work, they just need to know what code do.

Someone talk that Java8 is harder to debug. In my opinion, it not problem when debug mode become more complex because I don't care about how it work.

C. Design

I. Use case Diagram



User can access in to system through authentication system. After that, user with role "ADMIN" can change device information, change customer information, change policy setting, poll device and check consume bill.

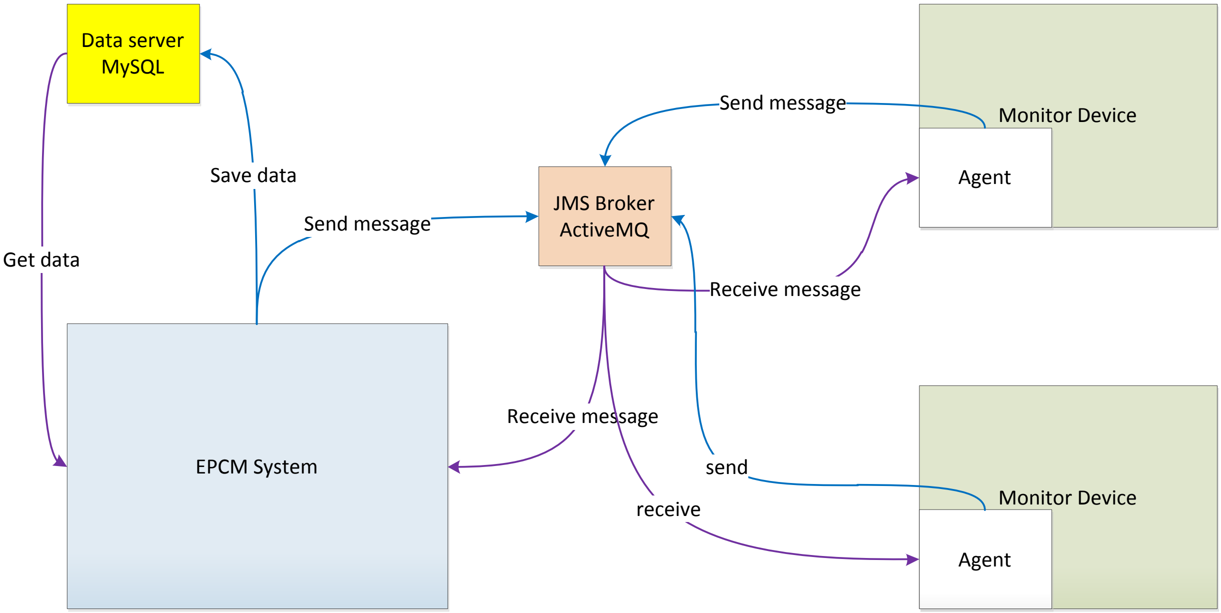
Data storage is use to store information of devices, settings and customer.

System process request from users and devices. Then, the result is store in data storage.

System collects and calculates information of device for providing chart report to user.

II. Workflow diagrams

1. Data flow:



- Data is store in data server MySQL, EPCM System access and save data to data server.

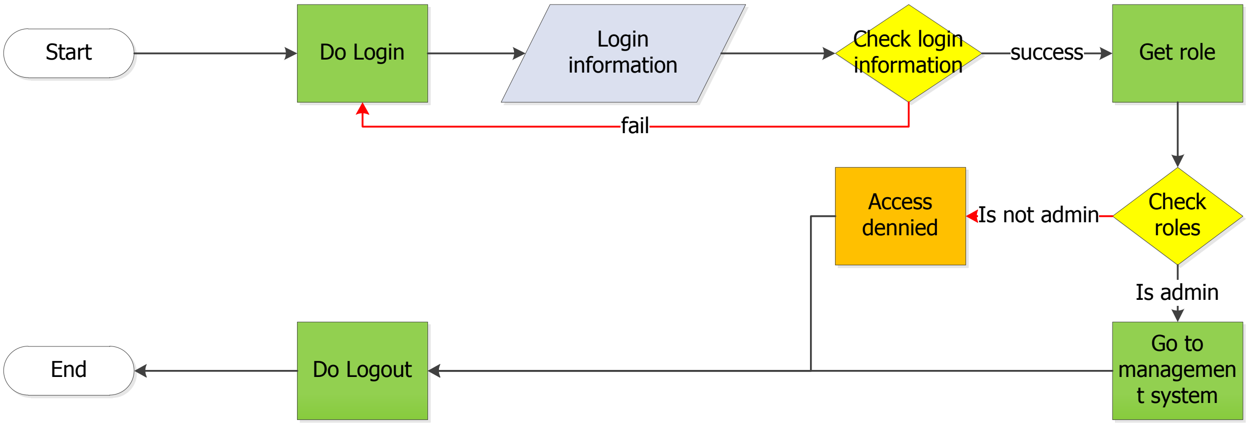
- EPCM System sends message to Device through JMS broker. JMS broker is ActiveMQ server, store and forward message from system to device. Actually, System send a broadcast message to broker with MAC address of device contained in message header. Then, message is received by device has MAC address same as MAC Address in message header.

- Agent is software run on device. It helps device send and receive message.

- Monitor device use agent for send request to System. Message is receive by ActiveMQ server then forward to system. In case, device response a request, message is sent to a temporary destination create by requester.

For more information, please read Appendix A session, JMS Topic and Queue.

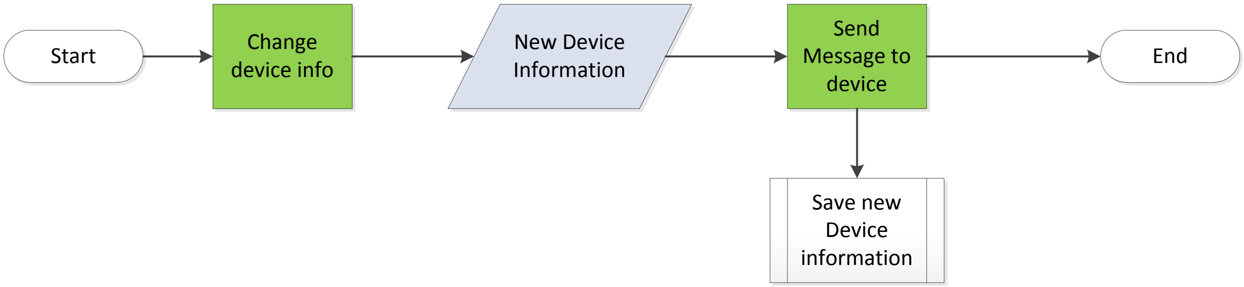
2. Authentication flow:



Workflow of user authentication and authorization. Step to get access to management system.

User login to system by providing login information. System check login information from request, if invalid information, system show error message. When information is corrected, system checks role, if user has not role admin, system show access denied message. When use has admin permission, it go to management page. User can log out, to go to login page.

3. Change device data flow:

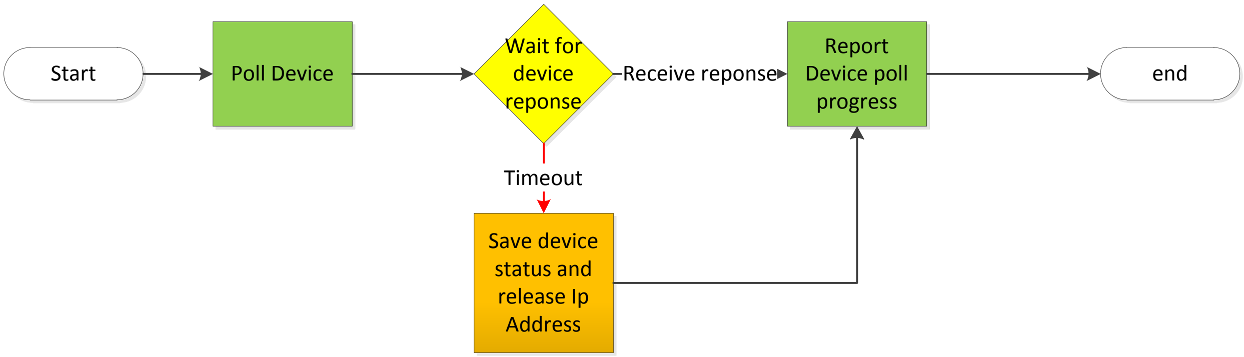


Workflow of Changing Device Information by system.

User change device on system and do save new device information. New device information will be saved into database. Then a message is send to device for requesting update new device info. In case device is offline, message is stayed on broker until device get connection and get the request message. For dirty prevent, ActiveMQ messages have "time to live" attribute to detect out of date message.

In case, User remove device from manager list, device just been removed from manager list. There is no request to device after device is remove from the list. When have request contains device info from device, device is registered into system again. Receive data flow diagram will discuss about process message from devices.

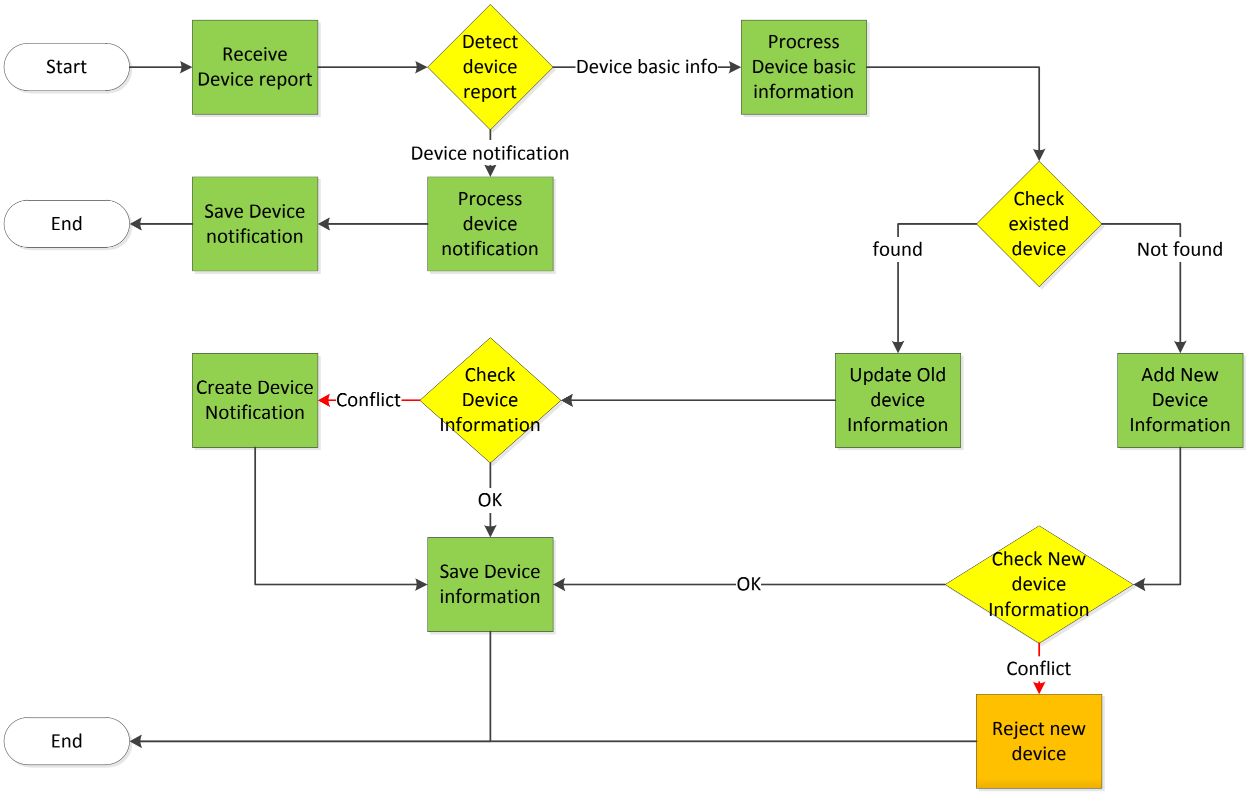
4. Poll device data flow:


Start
Poll Device
Wait for device reponse
Receive reponse
Report Device poll progress
end
Timeout
Save device status and release Ip Address

Workflow of poll device data. Poll device is request device report it status and send it data to EPCM System.

When user do poll devices on management list. Message is sent to broadcast and wait for response from receiver. Each message has time to live. After 10 second, if there is no device receives and responses, system save polled devices is down with status offline and IP address is release, e.g. "0.0.0.0" . If message response in 10 second, system will show polling progress for this devices is success. Devices will send its information device after that. Receive data flow diagram will discuss about process message from devices.

5. Receive device data flow:



Workflow of process when system receive data from devices.

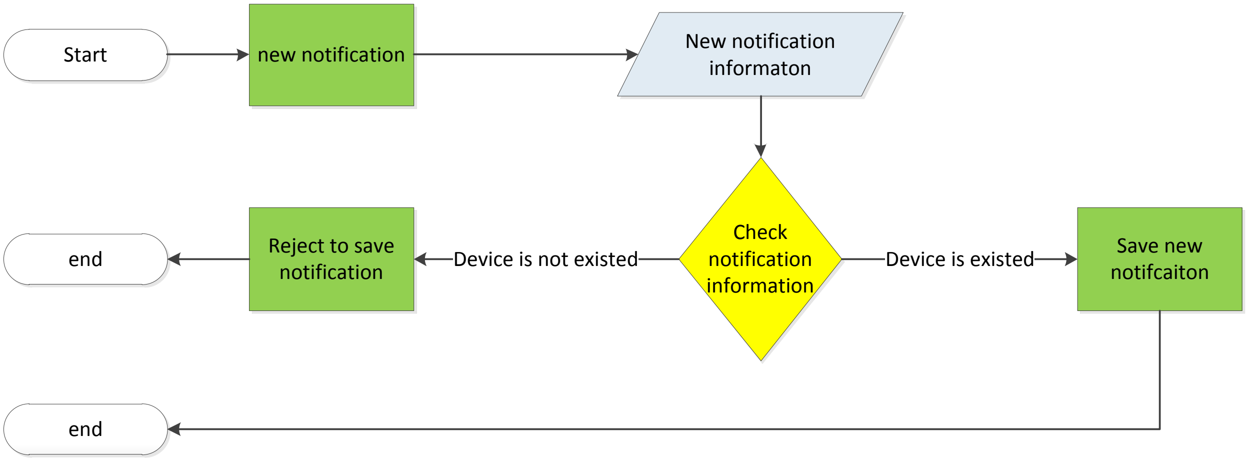
When message is receive on system, system will detect type of data.

In case, message is a device basic information, system tried to find confliction with old data such as, consume number is lower, device's IP address conflict with other one. If there is conflictions, System create notification for those devices have conflictions and save into database. After that device is save into database with as a new device or a old one base on MAC address.

In case, message is a device notification, a new device notification is create and save into database.

Open point, Notification should check existed of device or device will send a basic info message before send device notification for ensure that notification is searchable by device.

6. Process device notification data flow:

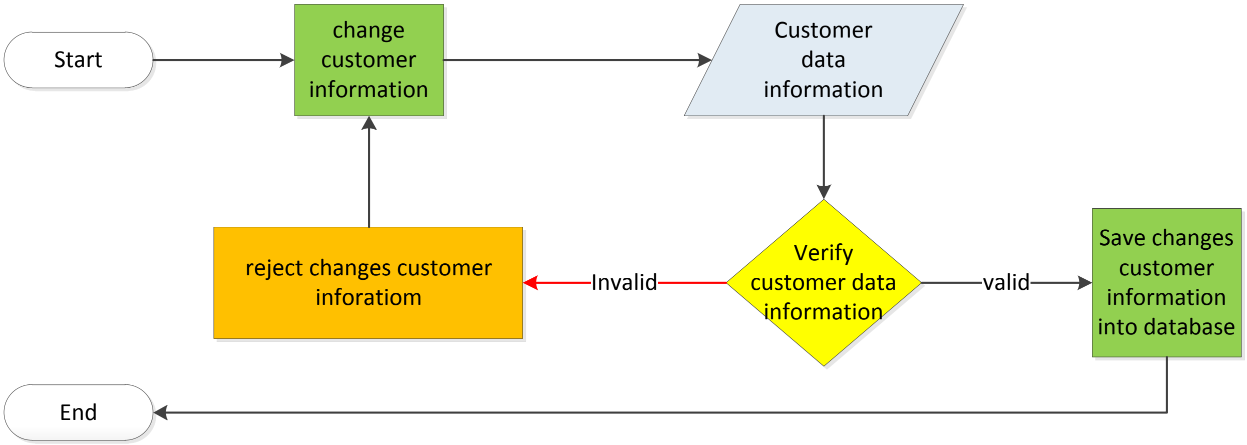


Device notification is created when data information of devices meet problems, e.g. IP address confliction, device information is illogical, unexpected device turn off. System will create device notifications to track those events. Other words, this feature look like logging, a kind of device diary.

Open point, System should able to request device send is logging file. The logging file must contain more details of device activities.

By the way, device notification is separated into 4 kinds base on severity of notification. Please see Data design for more information.

7. Process customer information data flow:

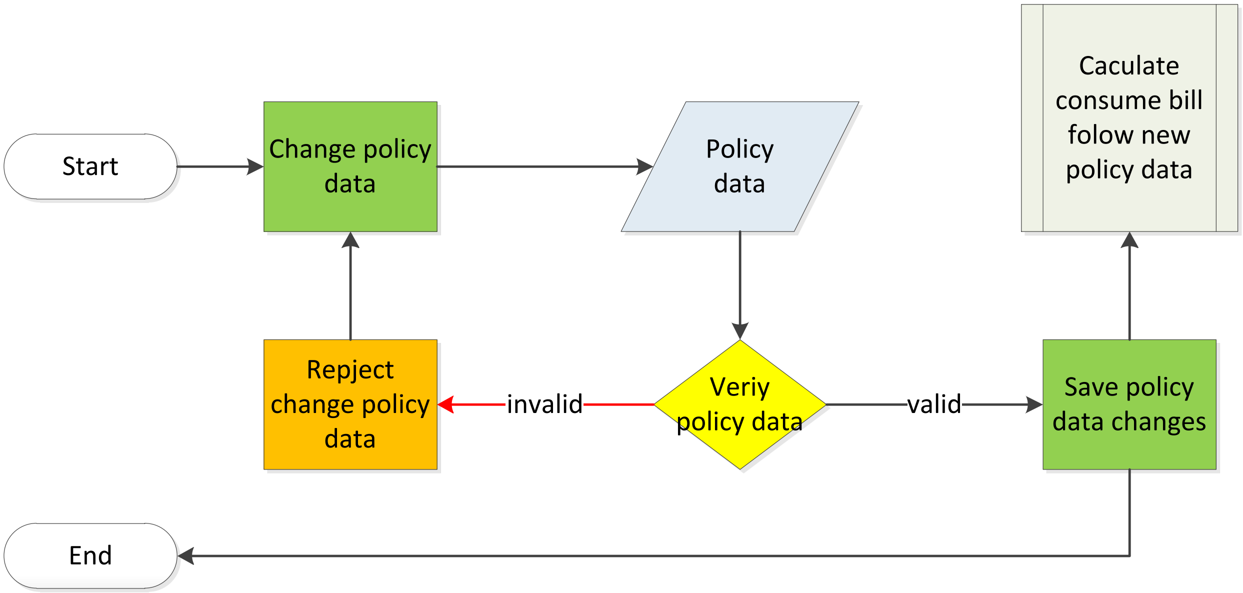

Start
change customer information
Customer data information
Verify customer data information
Invalid
valid
reject changes customer inforatiom
Save changes customer information into database
End

Customer is created when has a real customer sign up contact to use electric of company. Customer information will store in database and can only be access by administrators of electric power company.

When customer request or other reason, customer information is changed and saved into data. Before save customer information changes, system will verify that customer information for preventing error in system. If customer information changes are invalid with system standard format, there is no changes is saved into database.

Customer information changes should be create, update and remove customer information in database server. In case customer is remove from database, all other data relate to removed customer will be cleared.

8. Process consume policy data flow:



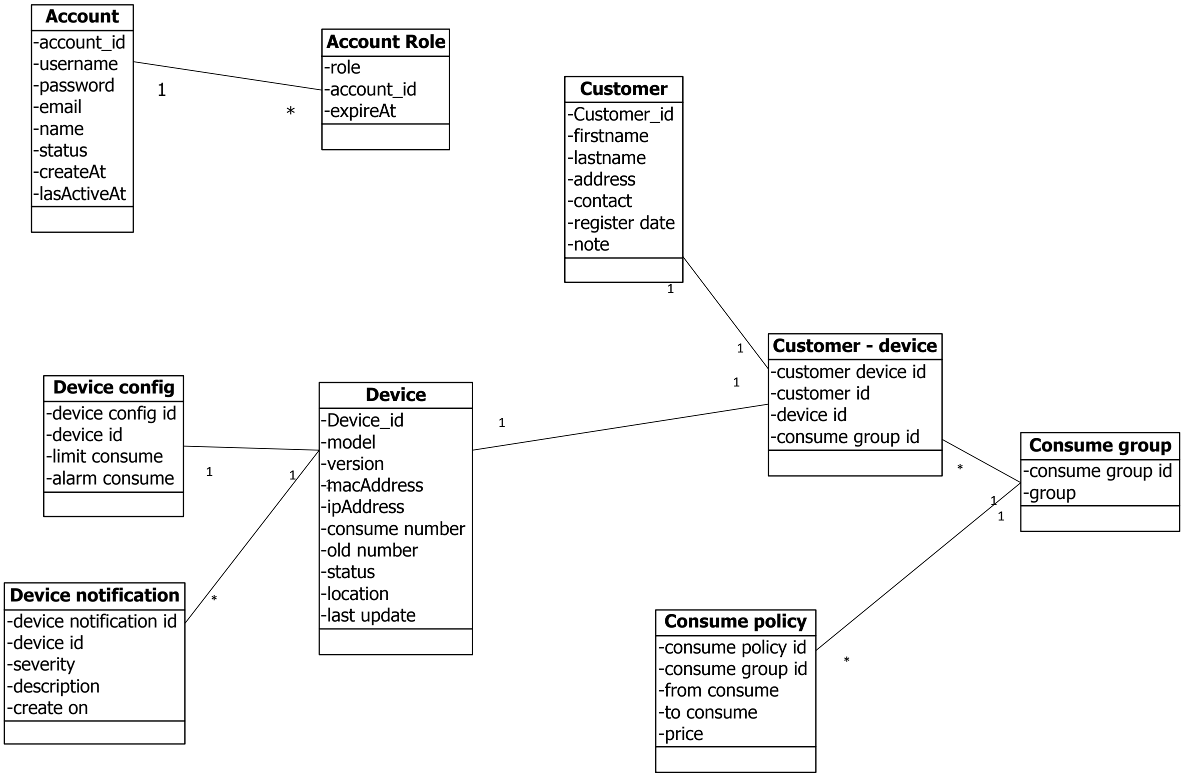
Administrator is possible to change policy configuration, which used to calculate electric consume bill. Therefore, after policy data is changed successfully, all bill must be re-calculate base on new policy.

Base on commercial of this function, it should be carefully verify before do changes saved into database. For example, confirm message is show before system do save into database and process update all consume bill.

Open point, system should verify by enter administrator password before do continue process.

III. Entity Class Diagrams

Entity classes are define base on database entities. They have all attributes of entities and mapped correctly.



There are 9 entity classes: account, account role, customer, customer device, device, device notification, device configuration, consume group and consume policy.

Entities

Account

Account entity presents user account, which user use to access system.

Attribute	Type	Description
Account_id	INT	Primary key and Auto increment. Identification of account
Username	TEXT	Max 50 characters, Unique. Username used to enter login form
Password	TEXT	64 characters, Bcrypt hashing encryption. Password used to enter login form
Email	TEXT	Max 50 characters, format email ,e.g. sample@yahoo.com.vn. Email use to contact and enter login form
Name	TEXT	Max 50 characters, display account name, optional. Name use to display on User Interface. if name is not defined. username is used to display.
Status	TEXT	Max 10 characters, status of account, e.g. inactive, active, banned
Create At	TIMESTAMP	Time of account creation
LastActiveAt	TIMESTAMP	Time of successful last log in

Primary key: Account_id

Account table has relationship many to many with Account Role table. It means one user can have many roles and one role can be assigned to many user.

Account Role

Account entity present role of account, used to authorize and grant permissions to user.

Attribute	Type	Description
Role	TEXT	Primary key, max 10 characters. Role name of account, e.g. admin, user
Account_id	INT	Primary key, Foreign key references to account_id of Account table Identification of account, which is assigned to this role.
Expired At	Time	Time when this role is expired

Primary key: Combine key by role and account Id

Account Role table has relationship many to many with Account table. It means one user can have many roles and one role can be assigned to many user.

Customer

Customer entity present customer, which store customer information in system.

Attribute	**Type**	**Description**
Customer_id	INT	Primary key, auto increment. Identification of customer.
Firstname	TEXT	Max 50 characters First name of customer.
Lastname	TEXT	Max 50 characters Last name of customer.
Address	TEXT	Max 250 characters. Location of customer house. Usually, it is same as device location.
Contact	TEXT	Max 250 characters. Contact of customer. e.g. email, phone number, fax or house address
Register date	TIMESTAMP	Time, when customer registers to use electric of company.
Note	TEXT	Max 250 characters. Note of customer. something are special of customer, system need to save for better productivity.

Primary key: customer_id

Customer table has one-to-one relation with customer-device table. It means one customer has one Customer-Device. In other word, one customer can used one device at a time.

Customer Device

Customer device present the which customer is assigned to use the device in which group. In other word, it stores data of usage device information. For example, customer A is use device TheWatcher380 to consume electric for house daily activities.

Attribute	**Type**	**Description**
CustomerDevice_id	INT	Primary key, auto increment. Identification of customer device
Customer_id	INT	Foreign key references to customer_id in Customer Table
Device_id	INT	Foreign key references to device_id in Device Table
ConsumeGroup_id	INT	Foreign key references to consumeGroup_id in Consume Group Table

Primary key: CustomerDevice_id

Customer device has one-to-one relationship with Customer Table. It means one customer has no or one customer device.

Customer device has one-to-one relationship with Device Table. It means one device has no or one customer device.

Customer device has many-to-one relationship with Consume group table. It means one consume group has no, one or more customer device.

Device

Device entity present device, which store device information.

Attribute	**Type**	**Description**
Device_id	INT	Primary key, auto increment Identification of device
Model	TEXT	Max 20 characters. Model of device. e.g. Aru7342, TheWatcher380, E103
Version	TEXT	Max 10 characters. Version of firmware run device. e.g. 1.0.Beta
Mac Address	TEXT	18 characters. Physical Address of device. e.g. de:1f:50:9f:e4:30, de:1f:50:9f:e4:31
IP Address	TEXT	Max 16 characters. Internet Protocol Address of device. e.g. 192.74.22.10, 192.74.22.11
ConsumeNumber	NUMBER	Greater or equal 0. Current consume number of device.
OldNumber	NUMBER	Old consume of the last bill payment.
Status	BIT	0 or 1. Status of device. 0 means 'ON' and 1 means 'OFF'
Location	TEXT	Max 250 characters. Location of device. e.g. 'Line 1742. Quang Trung Software City'
LastUpdate	TIMESTAMP	Last time this device is update information to server.

Primary key: device_id

Device table has one-to-one relation with device configuration table. It means one device has no or one device configuration.

Device table has one-to-many relation with device notification table. It means one device has no, one or many device notifications.

Device table has one-to-one relation with customer device. It mean one device has one or no customer device.

Device Notification

Device Notification entity present notification of device, which stores information about notification.

Attribute	**Type**	**Description**
deviceNotification_id	INT	Primary key, auto increment Identification of device notification
Device_id	INT	Foreign key references device_id in device table
Severity	TEXT	Max 10 characters Severity of Notification. e.g. ERROR, WARN, INFO, NOTICE
Description	TEXT	Max 250 characters Description of notification used to discuss about notification
Create On	TIMESTAMP	Time of Notification creation.

Primary key: DeviceNotification_id

Device notification table has many-to-one relation with device table. It means one device has no, one or many device notifications.

Device Configuration

Device Configuration entity present configuration of device, which use to trigger event on device. This table is in development, not release in this feature.

Attribute	**Type**	**Description**
DeviceConfig_id	INT	Primary key, auto increment Identification of Device configuration
Device_id	INT	Foreign key references device_id in device table
LimitedConsume	NUMBER	Limited Consume Number. Trigger when consume number reaches a number
AlarmConsume	NUMBER	Alarm Consume Number. Trigger when consume number reaches a number

Primary key: DeviceConfig_id

Device configuration table has many-to-one relationship with device table. It means one device has no or one or many device configuration.

Consume Group

Consume group entity present type of customer, it use to apply on customer device for detect what kind or purpose of consuming.

Attribute	Type	Description
ComsumeGroup_id	INT	Primary key, auto increment Identification of consume group
Name	TEXT	Max 50 characters. Display name of group on User Interface.

Primary key: ConsumeGroup_id

Consume Group Table has one-to-many with Consume policy table. It means one consume group has many consume policies.

Consume Policy

Consume policy entity present policy to apply in calculating electric consume bill.

Attribute	Type	Description
ConsumePolicy_id	INT	Primary key, auto increment Identification of consume policy.
consumeGroup_id	INT	Foreign key references consumeGroup_id in consume group table
FromConsume	NUMBER	Greater than 0 and lesser than to consume Trigger number to begin applying policy
ToConsume	NUMBER	Greater than from consume Trigger number to end applying policy

Primary key: ConsumePolicy_id

Consume policy has many-to-one relationship with consume group table. It mean one consume group has no, one or many consume policies.

D. Implement

I. Environment

1. Maven

After analysis, Program use Apache Maven for build tool. Therefore, settings of framework library is configured in pom.xml file.

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
	<modelVersion>4.0.0</modelVersion>
	<groupId>com.ntk.epcm</groupId>
	<artifactId>root</artifactId>
	<version>0.1</version>
	<packaging>pom</packaging>
	<properties>
		<epcm.version>0.1</epcm.version>
		<failOnMissingWebXml>false</failOnMissingWebXml>
		<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
		<jdk.version>1.8</jdk.version>
		<java.compiler.version>3.5.1</java.compiler.version>
		<java.jar.version>3.0.2</java.jar.version>
		<servlet.version>2.5</servlet.version>
		<jsf.version>2.2.13</jsf.version>
		<jstl.version>1.2</jstl.version>
		<spring.version>4.3.1.RELEASE</spring.version>
		<spring.security.version>4.1.3.RELEASE</spring.security.version>
		<spring.webflow.version>2.4.4.RELEASE</spring.webflow.version>
		<mysql.connector.version>5.1.39</mysql.connector.version>
		<hibernate.version>5.2.1.Final</hibernate.version>
		<activemq.version>5.14.1</activemq.version>
		<jackson.version>2.8.4</jackson.version>
		<slf4j.version>1.7.21</slf4j.version>
		<logback.version>1.1.7</logback.version>
		<junit.version>4.12</junit.version>
		<mockito.version>1.9.5</mockito.version>
		<primefaces.version>6.0</primefaces.version>
		<atmosphere.version>2.4.7</atmosphere.version>
		<commons-fileupload.version>1.2.1</commons-fileupload.version>
		<commons-io.version>1.4</commons-io.version>
	</properties>
	<modules>
		<module>epcm-share</module>
		<module>epcm-web</module>
		<module>epcm-simulator</module>
	</modules>
</project>
```

pom.xml file set all version libraries that tell maven to download from repository on internet. For example, when maven execute with install target, Spring framework 4.3.1.RELEASE jar and related libraries is downloaded and imported into project.

Above pom.xml file is root configuration, it does not clearly set which libraries is downloaded or which repositories stored these libraries. All details of configurations for each module is declared in pom.xml file of its.

In detailed:

Module epcm-share configuration is defined in <root>/epcm-share/pom.xml file.

Module epcm-web configuration is defined in <root>/epcm-web/pom.xml file.

Module epcm-simulator configuration is defined in <root>/epcm-simulator/pom.xml file.

2. Spring web MVC and JSF

In this project, I combine two of web MVC architecture to build web service.

Spring web MVC is base on Spring framework, so it support all other projects base on spring framework, which used in this project. But Spring is strong in back-end development, and web user interface is developed by JavaScript.

Primefaces base on JSF is very strong in user interface development. There are many components and AJAX supported with a bit JavaScript. And It use Java Bean to instant components in project.

The reason of avoiding JavaScript is hard to debug, hard to understand and hard to read. Although JavaScript has a huge libraries supported, it is a complex program language.

With supported JSF of Spring framework on Spring webflow project, and Spring Expression Language, Primefaces can be integrated in this project for building user interface.

In nut shell, back-end components are managed by Spring framework. And user interface is use JSF and Primefaces components.

3. How to combine Spring and JSF

In base, JSF interacts with Faces Bean by id. For example, FacesConverter with id deviceConverter will called by "converter='deviceConverter'" in JSF.

In this project, Components is managed by Spring Beans. Therefore, FacesConverter is not used to bind UI components with Managed Beans. Instead, Spring Bean is instant and bind to UI component by Spring EL.

For example, UI components is bind to back-end component "bind='#{deviceConverter}'", device Converter is a Spring Bean instead of Faces Bean.

4. Benefits of combine Spring and JSF

The first, Spring bean support Dependency Injection, which is easier to manage and create an instant bean with many parameters. Data Access Object is injected into beans by using @Inject or @Autowired annotation.

For example, Device DAO is injected into Device Service Component Bean with @Inject before declare variable

@Inject IDeviceDAO;

Spring will scan for finding instance of Device DAO, then inject it into DeviceService, which is created by Spring Context.

The second, JSF support UI Components interact with back-end beans instead of HTTP Request and HTTP Response. In other words, JSF covered HTTP Request and HTTP Request, so it become invisible with code reader. Therefore UI components interact with Spring look shorter and understandable when read code.

Moreover, Bean can response to AJAX request by FacetContext.getContextInstance() and RequestContext.getCurrentInstance().

The third, Primefaces strongly support Web socket in Primefaces Push project base on Atmosphere. Primefaces Push will use to develop live action function, such as live chat, live UI update.

The final, Spring webflow grant developers more control of webflow. Navigation, evaluation are covered by Spring webflow. For example, parameters can be set value before load view, decision to render view by checking conditions.

5. Difficult of combine Spring and JSF

Complex configuration: because of Spring web MVC base on request handler and JSF base on component handler so that it cannot avoid conflict and confuse of configuration progress. Developer have to configure Spring context for Spring framework, Spring Security Facet Tag library, Spring Security Filter, Primefaces Filter, especially Faces Servlet and Spring MVC servlet.

In this project Facets servlet is mapped to root and spring servlet is mapped to <url>/app/

Webflow configuration: because of awareness of webflow control, developer have to configure spring webflow. Webflow is managed by spring bean by scan and register all sub webflow configuration into main flow registry. When an HTTP request to get a URL, flow registry will scan and check to figure out the sub flow to be processed. Then figure out view to be rendered.

In conclusion, combination of Spring framework and JSF combine two powers together. "Greater power, more responsibility" this compile bring strength server and beautiful user interface. But, it need more resources to run Spring framework and JSF.

6. Tomcat server

Apache software foundation provides web server Tomcat plug-ins for Maven. All configuration of Tomcat server is default settings from Apache.

In this product, the purpose is Education, so tomcat setting should be default for prevent error with causes of Tomcat Server configuration.

In future, Tomcat server will be re-configure for security reason.

7. ActiveMQ, JMS broker server

Apache provides ActiveMQ project to support JMS. Broker can be setup and run by spring framework, or run independently on JVM. In this project, broker server run independently on JVM. So, before web server start, JMS broker have be started first.

Same as Tomcat server, configuration of ActiveMQ server is default and will be change in future for security.

8. Spring security

Spring security is configured to protect unexpected access into management system. In this project, spring security is configured to grant only administrator permission to access management system. Other user account is permit to access management system.

II. Code Design

1. Epcm-share module:

Share module stores model classes, constants and transfer data object.

9 model classes present for 9 entities class. They used to process data from database, and convert into transfer object for attract with message through network.

Constants classes use to define all constants is use in project.

Transfer data objects use to convert object into JSON with enough information. In this project transfer objects just use to attract more information into base data for transfer data.

In future, transfer data will be scale for removing unnecessary information. Therefore message through network become lighter.

2. Epcm-simulator module:

Simulator module use to build simulator device for showing result of functions.

There are 5 parts in this module

Main executor use to build JAR Runnable. When build jar, maven configure point to main file for executing program. Main will run services on simulator such as listener, data processor. When simulator is run, it send a message to web for report active status. And when shutting down, it also send message to web for report inactive status

Listener registers a consumer on JMS broker for catching message sent from service on EpcmServiceTopic. When listener receive a message on broker, with attracted head with Mac address same as Mac address of simulator, message is processed by a specified request processor.

Request Processor is managed by a registry. When message is receive registry find out correct request processor base on request type of message. Then, Data is de-attracted from message and is processed by specified data processor.

Data Processor is managed by a registry. When data is de-attracted from message, registry find out correct data processor base on data type for processing data. After data is processed, it will be saved into data file such as basicinfo.json

Data Generator is used to access data file for getting data. It works like a small data access on device.

3. Epcm-web module:

Data access objects used to accessing data on MySQL data server. They use hibernate for processing. This part is separated by implement DAO interfaces, for changing data server. When changing data server, developer only need to implement those interface for process data on new data server. It means system will be able to support many data servers such as MySQL and mongoDB.

Data services used for managing data on system. They extend Observable for handle data changes.

Manage Beans used for interacting with UI components. They support UI components to process request, and handle data changes by observer design. Manage bean observer data change on data service by implement Observer interface.

Face Converters Beans used to converts object into text for display on UI, and convert text into objects for processing data. They are bind to UI component by declare “ converter=’#{converterBean}’”.

Face Validator Beans are custom validators, support to validate data form input UI components. They are bind to UI component by declare “validator=’#{validatorBean}’”

JMS Tasks implement Runnable interface for running by Spring Task Executor. They used to supported multitasking. For example when system polled 7 devices, there are 7 tasks is created to send request to devices.

JMS Callbacks for handle result when JSM task completed. They process result and report it to UI

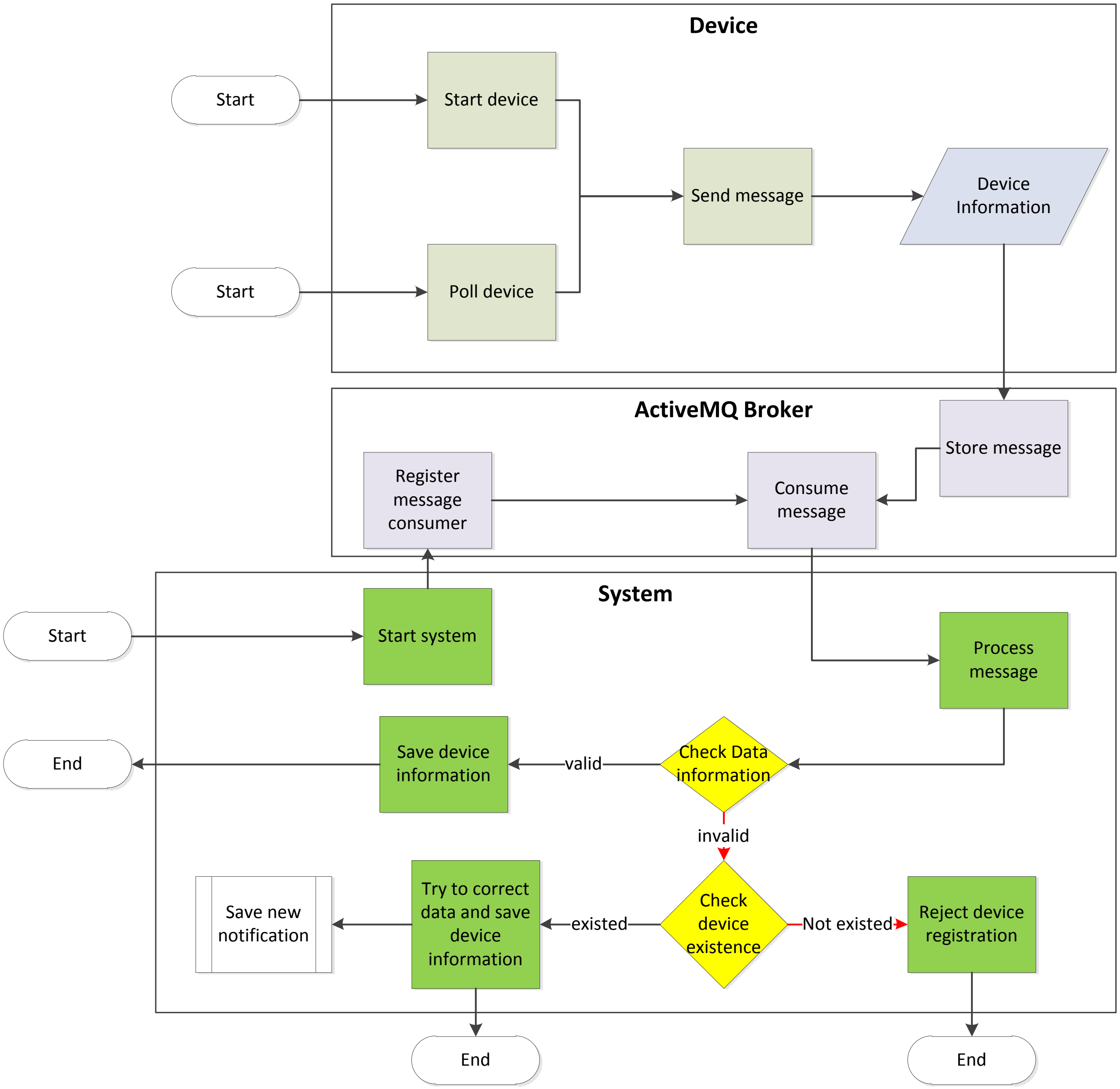
Data processor, as same as data processor on simulator, process data come from devices and save into database. Its designed for addable more processors when new data type is implemented.

III. Function implement:

1. Device registers into system

For ensuring that device is limit by access devices, system is not allow to enter dummy data into system. Therefore, almost device information is got from device

Work flow:



Before, system must started and device can connect to service. When device start, device sends a JMS message to DeviceServiceQueue on JMS broker.

After that, system receives this message on DeviceServiceQueue. The message will be processed.

In case, device information has problem, system will reject to register.

2. Device information is updated by device

Device data should be updated when device started or polled.

Work flow: see device registers into system work flow

Before, system must started and device can connect to service. When Device started or polled, it sends a message contain data information to DeviceServiceQueue on JMS Broker.

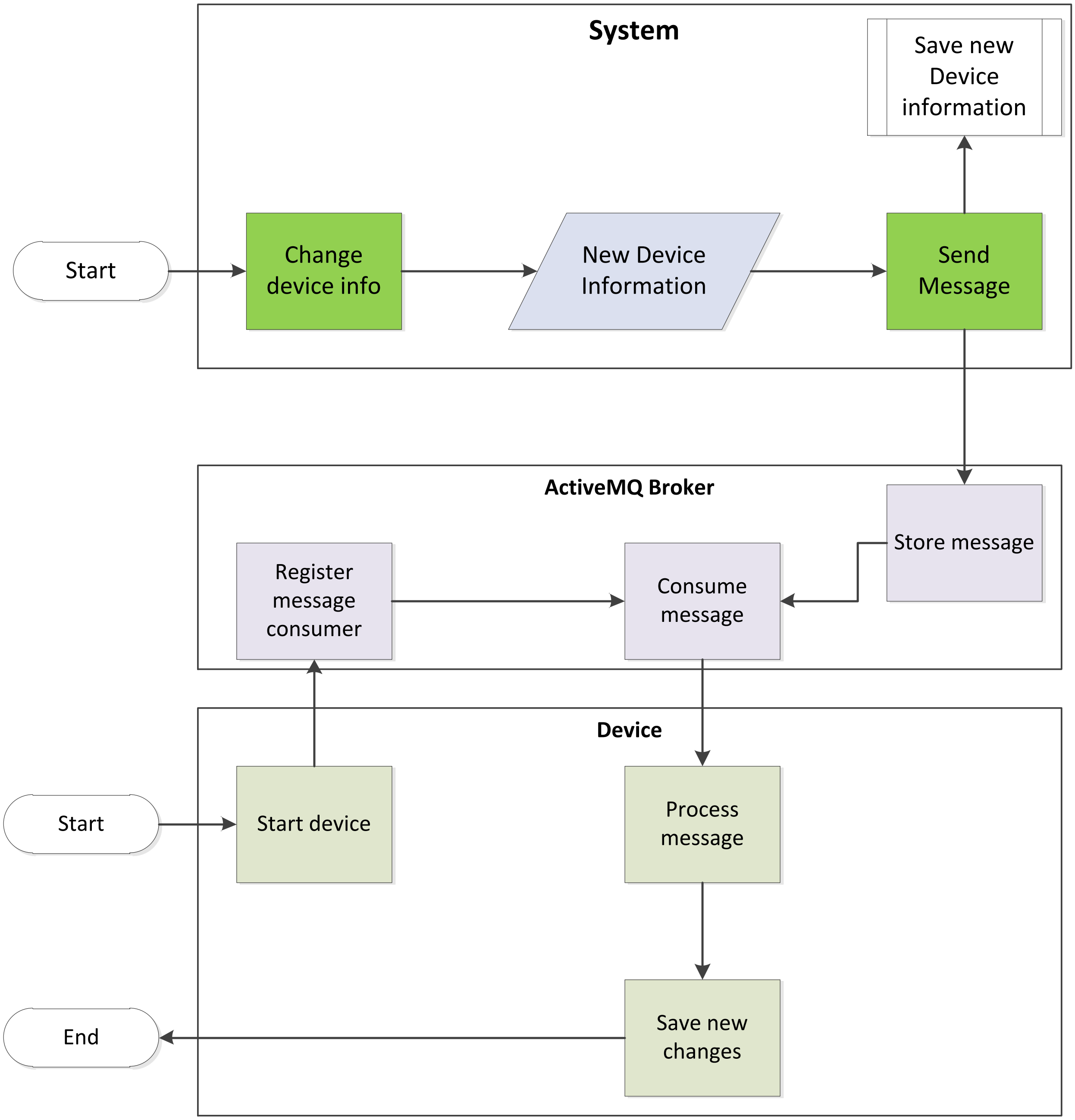
After that, system received message on DeviceServiceQueue. The message will be processed.

In case, device information has problem, System will try to correct device information and add notification for notify manager about this problems

3. Device information is updated by server

System is allow administrator to modify some device information such as device's location.

Work flow:



Before, system must started and device can connect to service. Administrator modify device information on server. Then, sever is send a message to EpcmServiceTopic on JMS broker, message is attracted with MAC address of device.

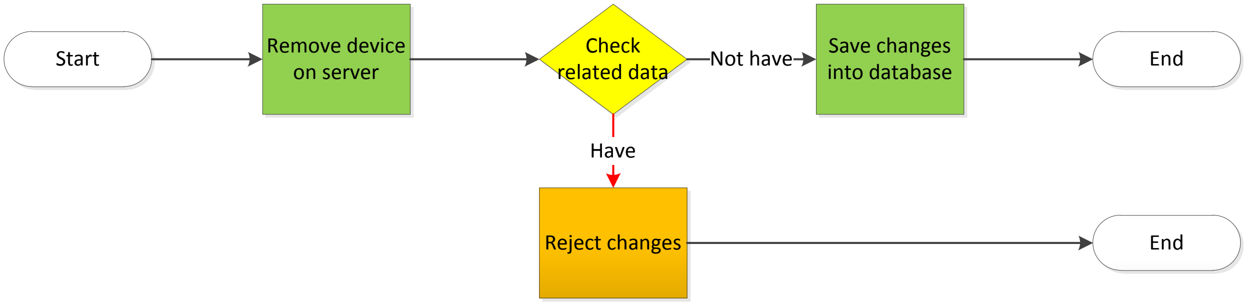
After that, device received the message from EpcmServiceTopic on JMS broker. Device processes message and save new changes into device's memory.

In case, device is offline message is stored on JMS broker service until device connect and get it.

4. Device is removed by server

This function only use to un-manage the devices, after remove device from server, all data of device on server will be lost and cannot be recover. The data on device will update to server like register new device. Device's data is not clear when device is removed on server

Work flow:



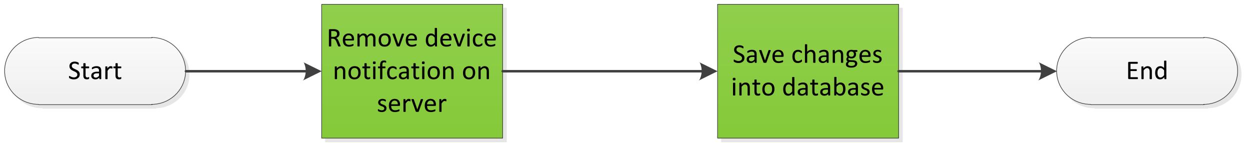
Administrator remove device on server, then device data on server is removed.

In case, device has related data, system reject to remove device.

5. Device Notification is removed on server

For cleaning data reason, and device is not an important data. So device notification is able to be removed from database.

Work flow:

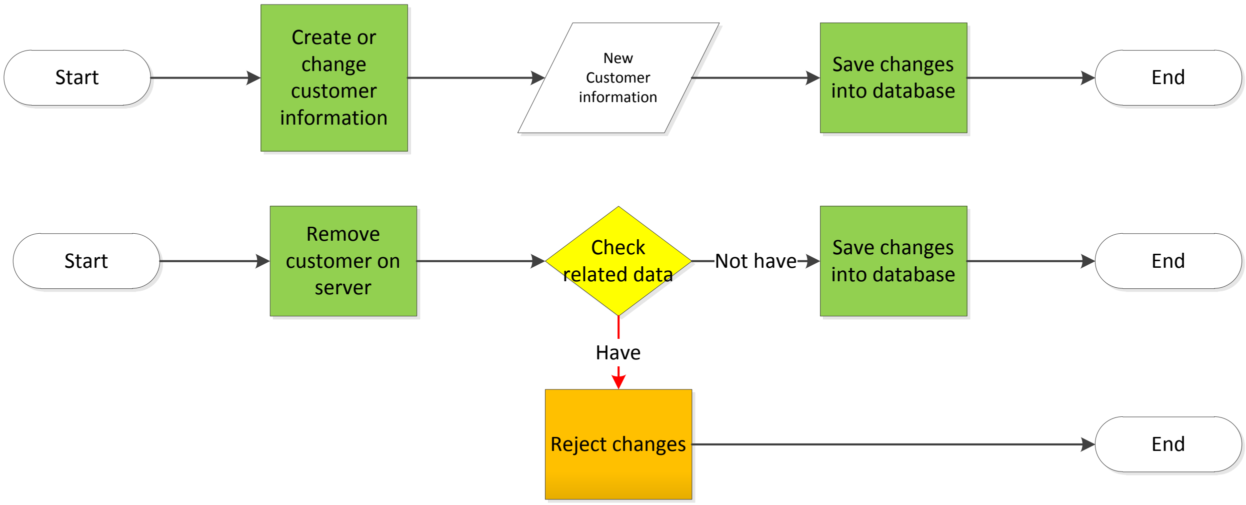

Start
Remove device notifcation on server
Save changes into database
End

When administrator remove device's notifications, the notifications are removed from database.

6. Administrator modify customer database

Administrator is allowed to modify customer information on database.

Work flow:


Start
Create or change customer information
New Customer information
Save changes into database
End
Start
Remove customer on server
Check related data
Not have
Save changes into database
End
Have
Reject changes
End

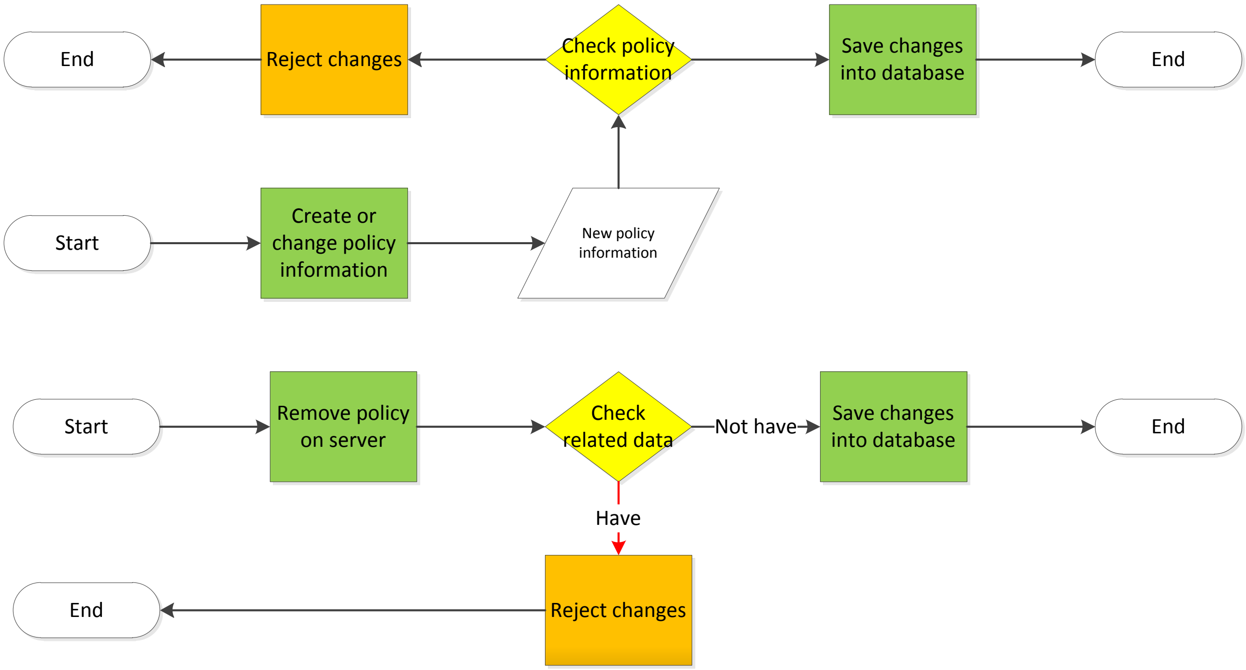
When customer create new customer, update, remove customer data, new customer information changes will be saved into database.

In case customer is removed on server, if customer has data relation on other table, system reject to remove this user.

7. Administrator modify policy database

Administrator is allowed to modify policy information data.

Work flow:



When administrator modify policy information data. System will validate information input.

If policy information is valid, system save it into database.

If policy information is invalid, system reject the changes.

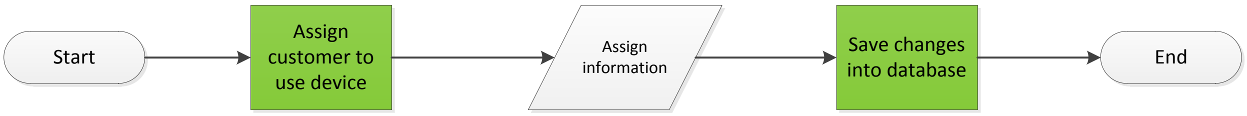
If group is not existed, new group is save into database.

In case policy is removed, all related data is checked. if it has related data, changes are rejected. If it has no related data, System remove it from database.

8. Administrator assigns customer to use device

Administrator is allowed to assign customer to use device. For example, Customer A is assigned to use Device AB123 in group Home Basic.

Work flow

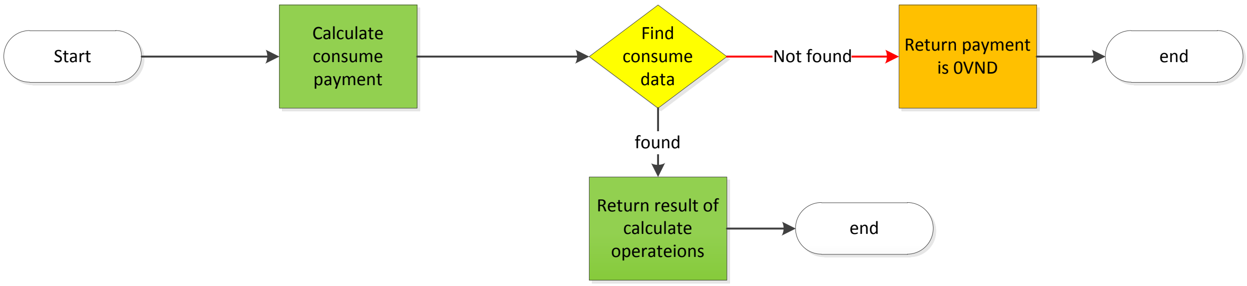


When administrator assign customer to user device, the customer device information is saved into database.

9. Calculate consume payment

Consume payment is calculate when a device is used to assign by a customer and have group.

work flow



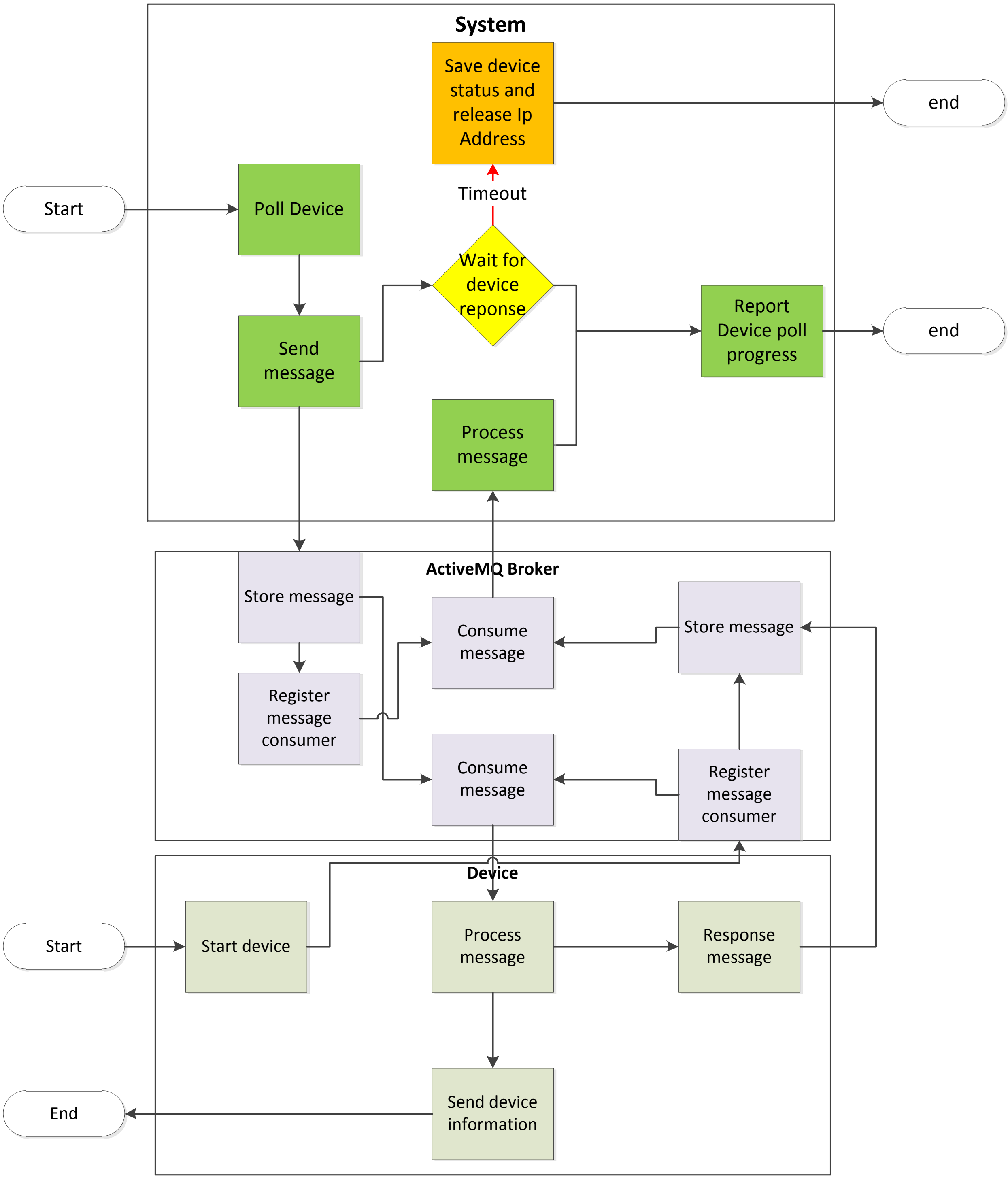
Consume payment is calculated on customer, device, group and policies data. If not enough data, calculate operation will return value is 0 VND.

In case, system found enough data to perform calculate operator, system will scan all policy and sum all to return final result.

10. Poll device

System allow administrator to make poll request devices to send data.

Work flow:



When system poll devices, message is sent to EpcmServiceTopic. The message is attracted MAC address of devices on header.

When device started a consumer is registered into ActiveMQ for receive message from with MAC address of device on EpcmServiceTopic. When receive message, device response a message to aware that device is active. After that, device send a data information message to DeviceServiceQueue as request.

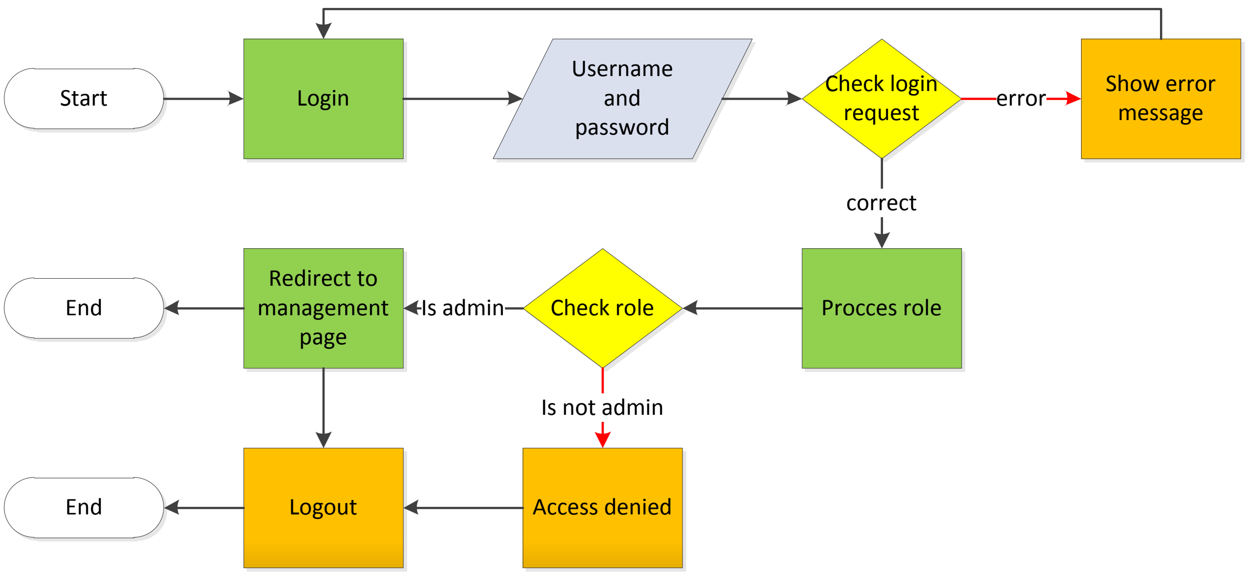
In case, there is no response from device, system will set device is inactive and release IP address.

In summary, device send first message for reporting that it is active. the second message is send with device information.

11. Authentication and authorization

To access management system, user have to login with account have admin role.

Work flow

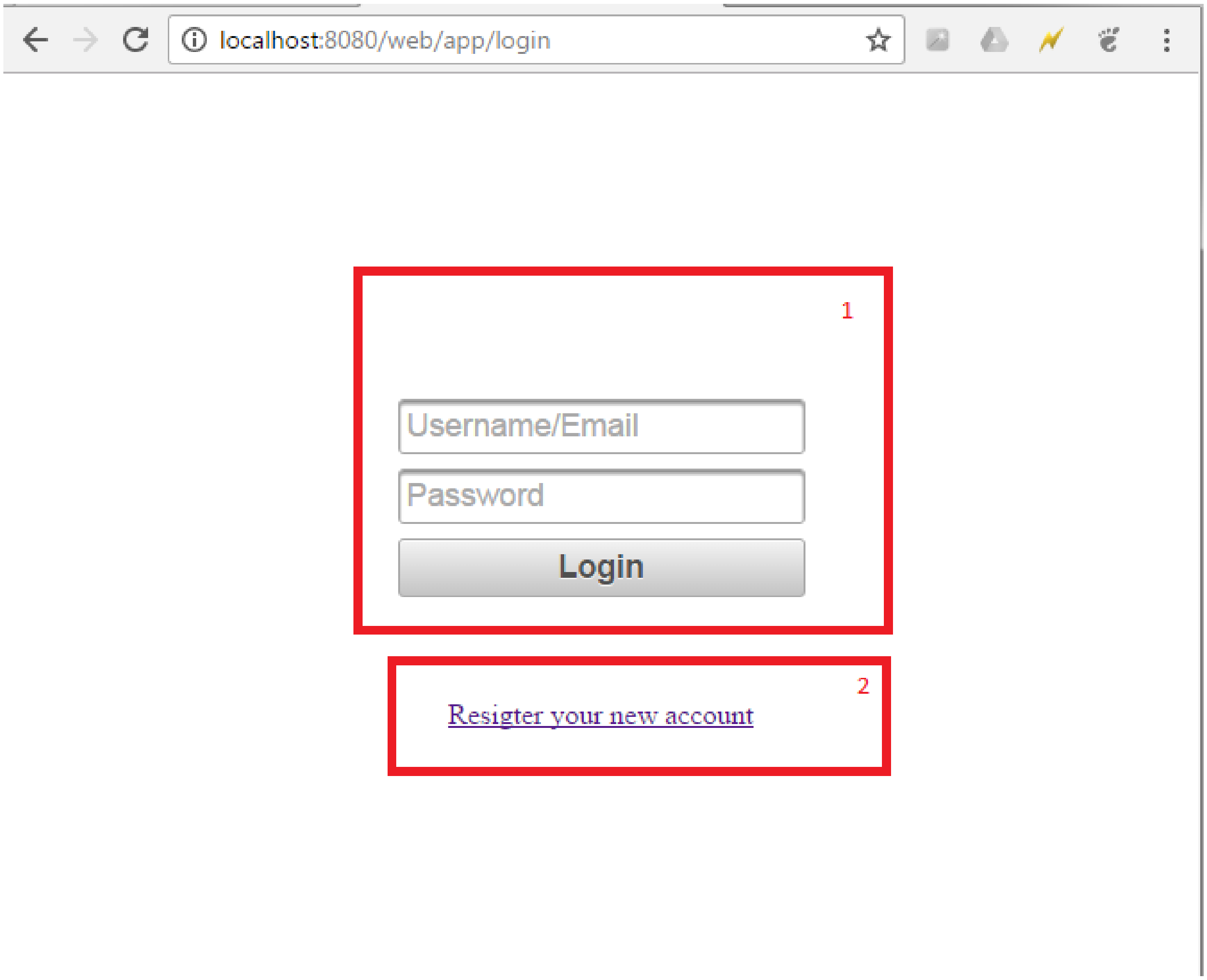


When user login into system, username and password is check, if username password is correct, system go to check role step. If an error happen, error message is showed on Login page.

After checked role, if user is admin, system redirect to management page. If user is not admin, It show access denied page and log out user.

IV. User Interface

1. Login page



1. Login form: with Username input and password input. There are a button below inputs.

2. Link to register page: redirect to register page after user click on.

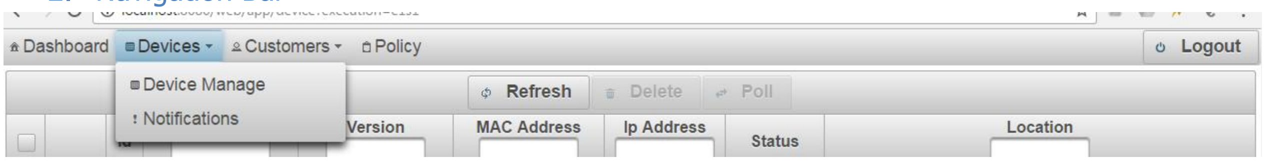
When login success, system navigate to dashboard page

When access denied, system navigate to 403 access denied page

When login fail, system show error message.

When form input is leaved empty, system show required message.

2. Navigation Bar



Navigation bar use to navigate between pages. There are:

Dashboard button: navigate to dashboard page.

Device menu with Device Manage and notification button.

Device Manager button: navigate to Device manage management page.

Notification button: navigate to Device notification management page.

Customer menu with Customer manage and assign device button.

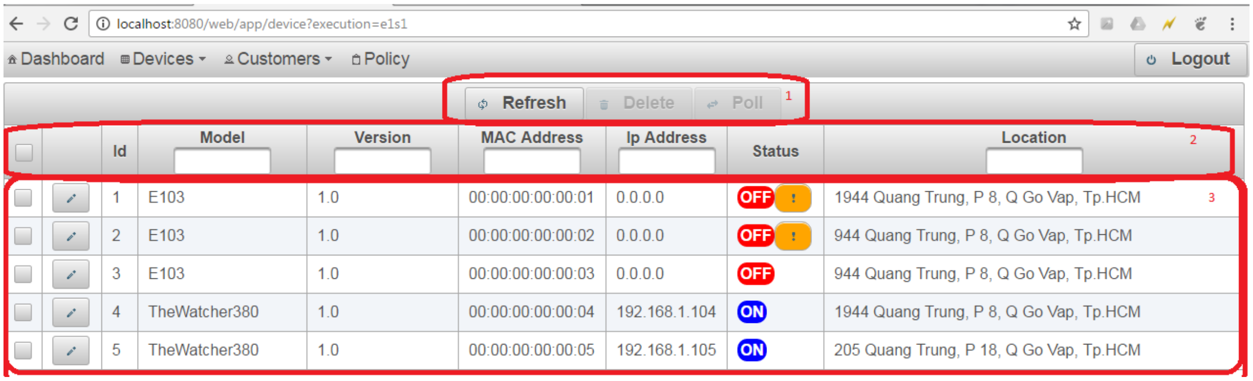
Customer manage button: navigate to customer management page.

Assign devices: navigate to assign device page.

Policy button: navigate to policy management page.

Logout button: send logout request. when user log out success, system navigate to login page.

3. Device manage page



□		Id	Model	Version	MAC Address	Ip Address	Status	Location
□		1	E103	1.0	00:00:00:00:00:01	0.0.0.0	OFF !	1944 Quang Trung, P 8, Q Go Vap, Tp.HCM
□		2	E103	1.0	00:00:00:00:00:02	0.0.0.0	OFF !	944 Quang Trung, P 8, Q Go Vap, Tp.HCM
□		3	E103	1.0	00:00:00:00:00:03	0.0.0.0	OFF	944 Quang Trung, P 8, Q Go Vap, Tp.HCM
□		4	TheWatcher380	1.0	00:00:00:00:00:04	192.168.1.104	ON	1944 Quang Trung, P 8, Q Go Vap, Tp.HCM
□		5	TheWatcher380	1.0	00:00:00:00:00:05	192.168.1.105	ON	205 Quang Trung, P 18, Q Go Vap, Tp.HCM

1. Toolbar with refresh, delete and poll button.

Refresh button: call reload data from database.

Delete button: call delete all selected devices in list.

Poll button: call poll request to all selected devices in list.

2. Column header: with field header and filter input.

Field header: show title of column.

filter input: input text to filter data in table.

3. Data table: with select checkboxes, edit button, data fields and notification button.

Checkboxes: check true or false to select devices in list .

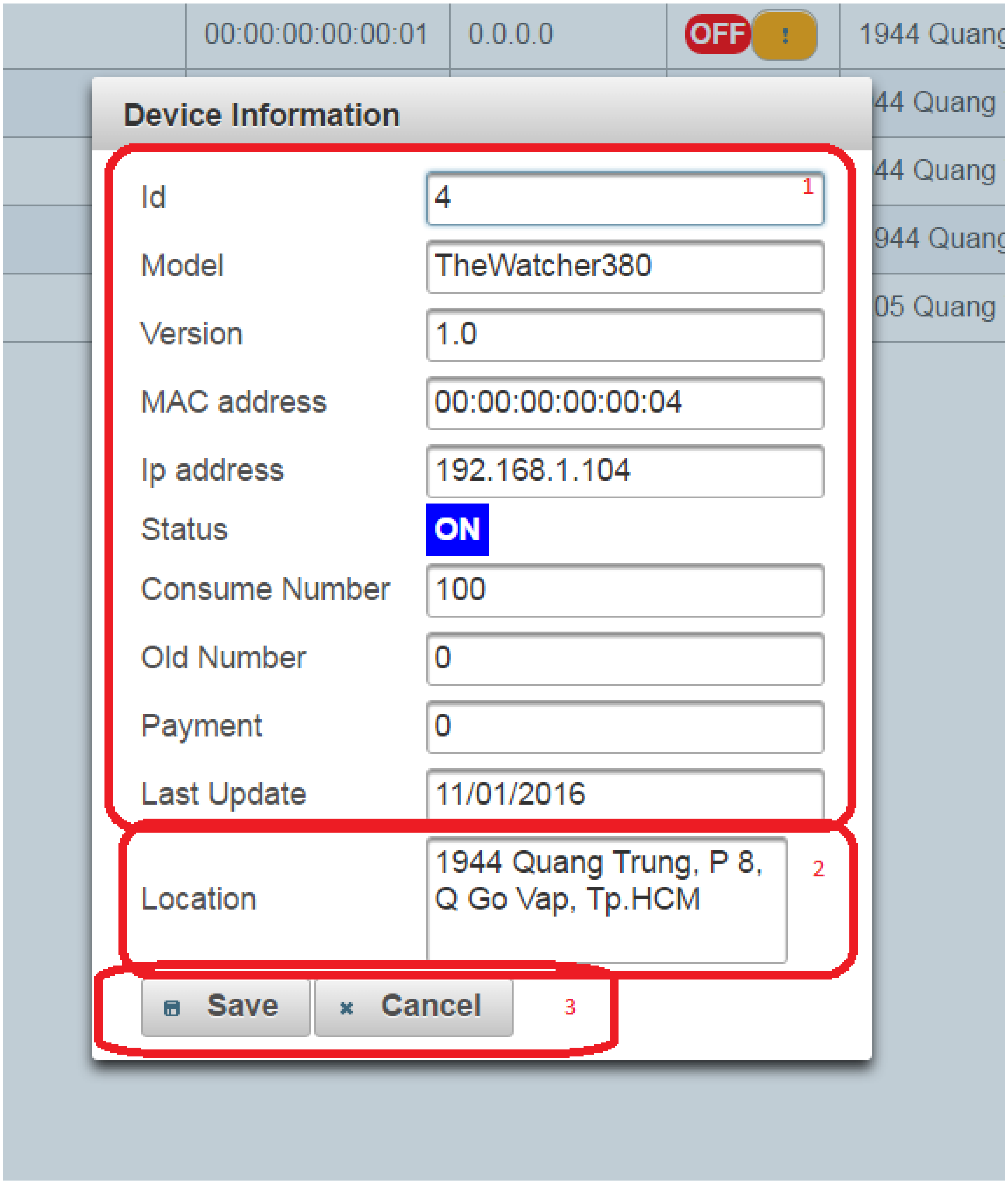
Edit button: Edit or view data of a specified device.

Data fields: display all data of devices such as id, model, version, MAC address, IP address, status, and location.

Notification button in status field: navigate to notification page with filter of a specified device.

4. Device information dialog

This dialog is showed when a device is selected to edit.



Device Information

Id	4
Model	TheWatcher380
Version	1.0
MAC address	00:00:00:00:00:04
Ip address	192.168.1.104
Status	ON
Consume Number	100
Old Number	0
Payment	0
Last Update	11/01/2016
Location	1944 Quang Trung, P 8, Q Go Vap, Tp.HCM

1. Data information of device: this group is data on device, so server cannot change value input of those fields. Id, Model, Version, MAC address, IP address status, consume number, old number, payment and last update. Especially, pay field is calculate when this dialog open.

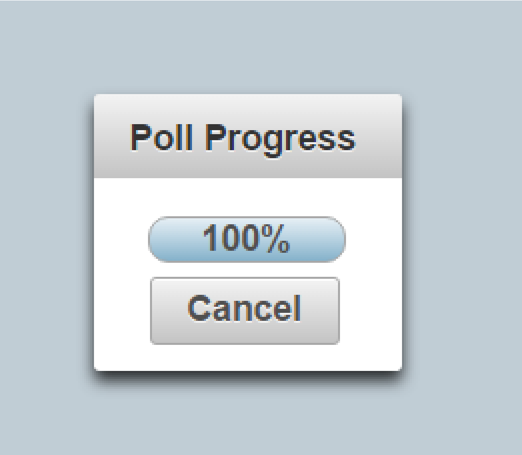
2. Editable information: this group is editable data, which will be changed when save button is clicked.

Save button: call save method to back-end bean. All of inputs are submit and process to save in database.

Cancel button: close dialog when click.

5. Poll progress dialog

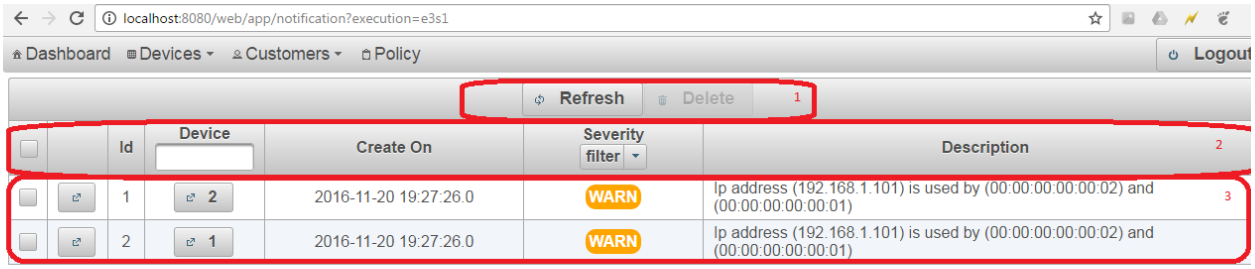
This dialog is showed when poll function is called.



The dialog contain a progress bar to show progress status and a button to cancel tasks.

Cancel Button: when clicked, call to back-end for cancel running tasks.

6. Device notification page



□		Id	Device	Create On	Severity	Description
□	⧉	1	2	2016-11-20 19:27:26.0	WARN	Ip address (192.168.1.101) is used by (00:00:00:00:00:02) and (00:00:00:00:00:01)
□	⧉	2	1	2016-11-20 19:27:26.0	WARN	Ip address (192.168.1.101) is used by (00:00:00:00:00:02) and (00:00:00:00:00:01)

1. Toolbar: with refresh button and delete button:

Refresh button: call request load notification information from database.

Delete button: call delete all selected notification in list.

2. Column header

Checkbox: selected all notifications in list.

Header: show column title

Input fields: filter list with value in column

Sort button: sort list follow selected column.

3. Notification Table List

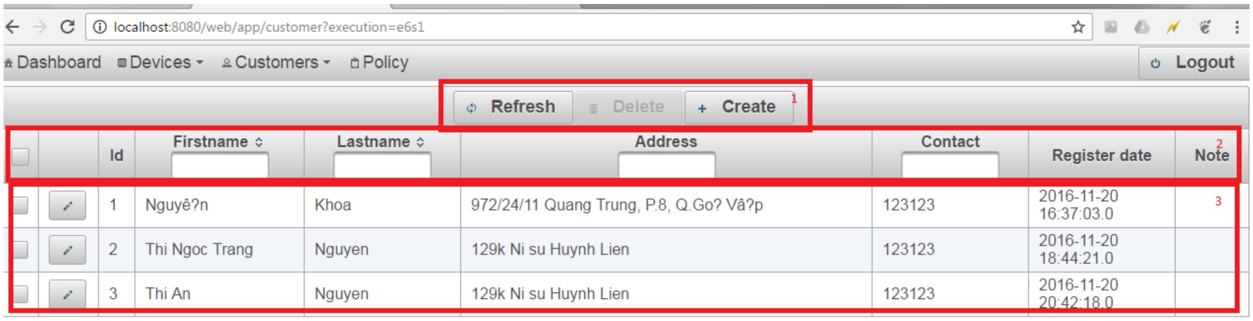
checkbox: select more one notification in list.

popup button: show Notification detail dialog.

Device button: show Device detail dialog.

Data fields: show data information of device's notification.

7. Customer page



□		Id	Firstname	Lastname	Address	Contact	Register date	Note
□	✎	1	Nguyê?n	Khoa	972/24/11 Quang Trung, P.8, Q.Go? Vâ?p	123123	2016-11-20 16:37:03.0	
□	✎	2	Thi Ngoc Trang	Nguyen	129k Ni su Huynh Lien	123123	2016-11-20 18:44:21.0	
□	✎	3	Thi An	Nguyen	129k Ni su Huynh Lien	123123	2016-11-20 20:42:18.0	

1. Toolbar:

Refresh button: call request reload data from database.

Delete button: delete all selected customer.

Create Button: show dialog to create new customer.

2. Column header:

Checkbox: select all customers in list.

header: display column title.

Filter input: filter customers follow text in filter inputs.

3. Data table:

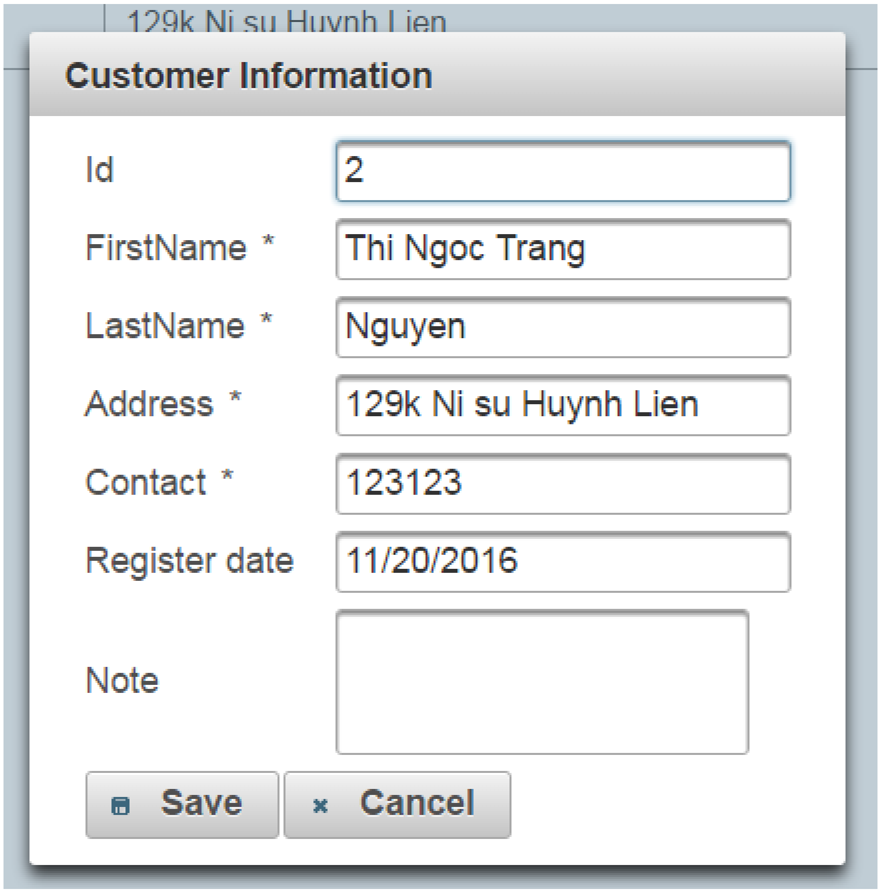
Checkbox: select one more customer.

Edit button: show customer information dialog with information of specified customer.

Data fields: show customer information.

8. Customer information dialog

This dialog is showed when create or modify customer information



129k Ni su Huynh Lien

Customer Information

Id	2
FirstName *	Thi Ngoc Trang
LastName *	Nguyen
Address *	129k Ni su Huynh Lien
Contact *	123123
Register date	11/20/2016
Note	

Customer information: Id, First name, last name, address, contact, register date and note.

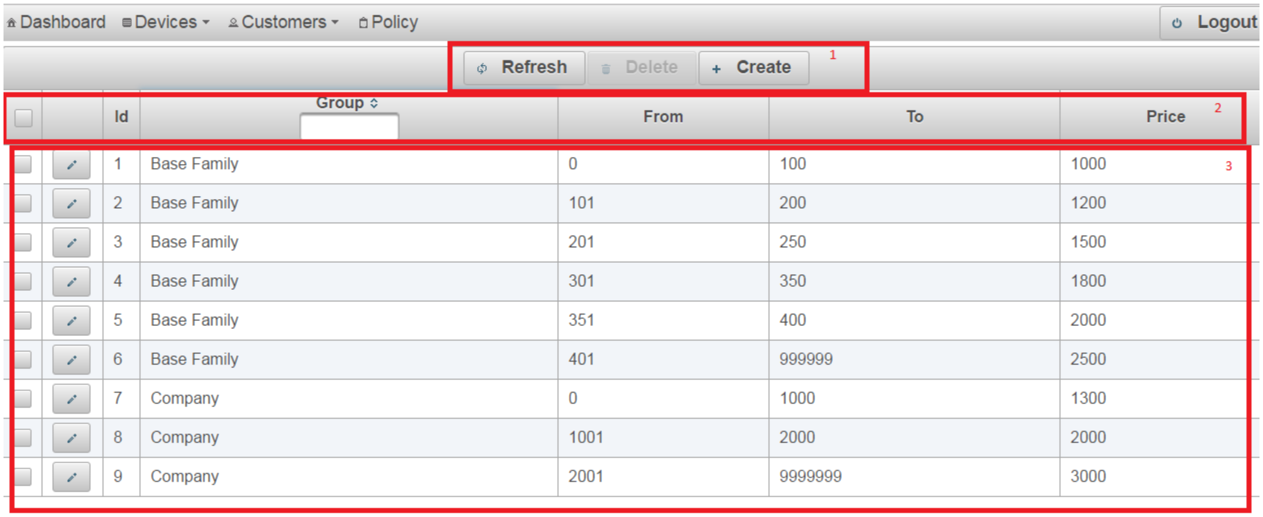
Id input is read-only.

First name, last name, address and contact fields are required validate checked.

Save button: Call save method to save those changes into database.

Cancel button: close this dialog when clicked.

9. Policy Page



□		Id	Group	From	To	Price
□	✎	1	Base Family	0	100	1000
□	✎	2	Base Family	101	200	1200
□	✎	3	Base Family	201	250	1500
□	✎	4	Base Family	301	350	1800
□	✎	5	Base Family	351	400	2000
□	✎	6	Base Family	401	999999	2500
□	✎	7	Company	0	1000	1300
□	✎	8	Company	1001	2000	2000
□	✎	9	Company	2001	9999999	3000

1.Toolbar

Refresh button: call request to back-end bean for loading data from database.

Delete button: call delete all selected policies.

Create button: show policy information creator dialog.

2. Column header

Checkbox: select all policies in list.

Filter input: filter policies in list by group filed.

Sort button: sort list by group.

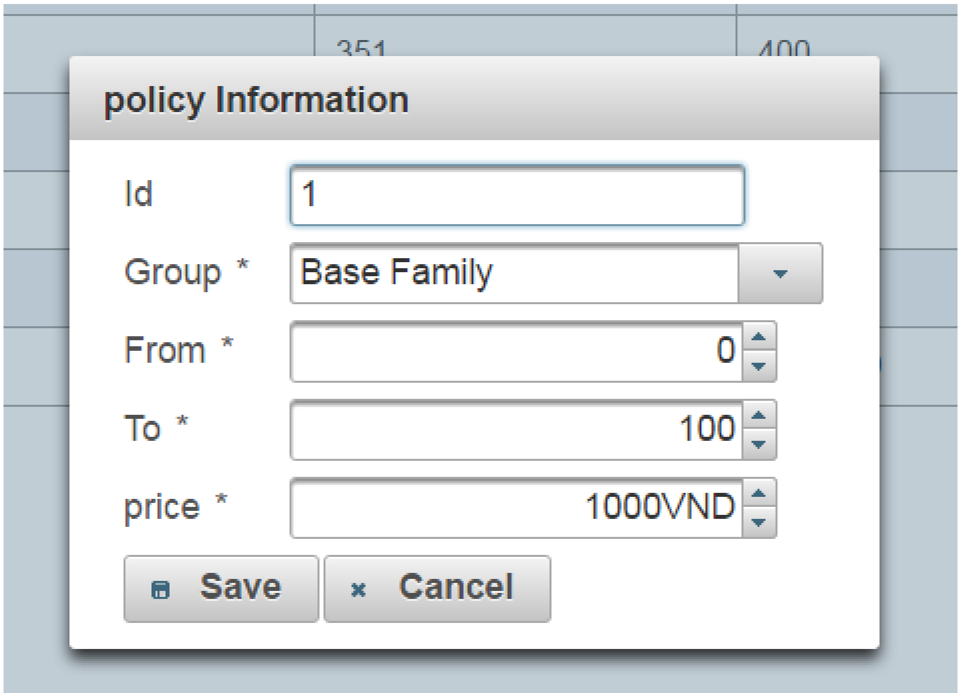
3. Data table:

Edit button: show policy information dialog with information of a specified policy.

Data fields: Id, Group, From, To, and price.

10. Policy information dialog

This dialog is show when create or edit a policy.



policy Information

Id 1

Group * Base Family

From * 0

To * 100

price * 1000VND

Id field is read-only.

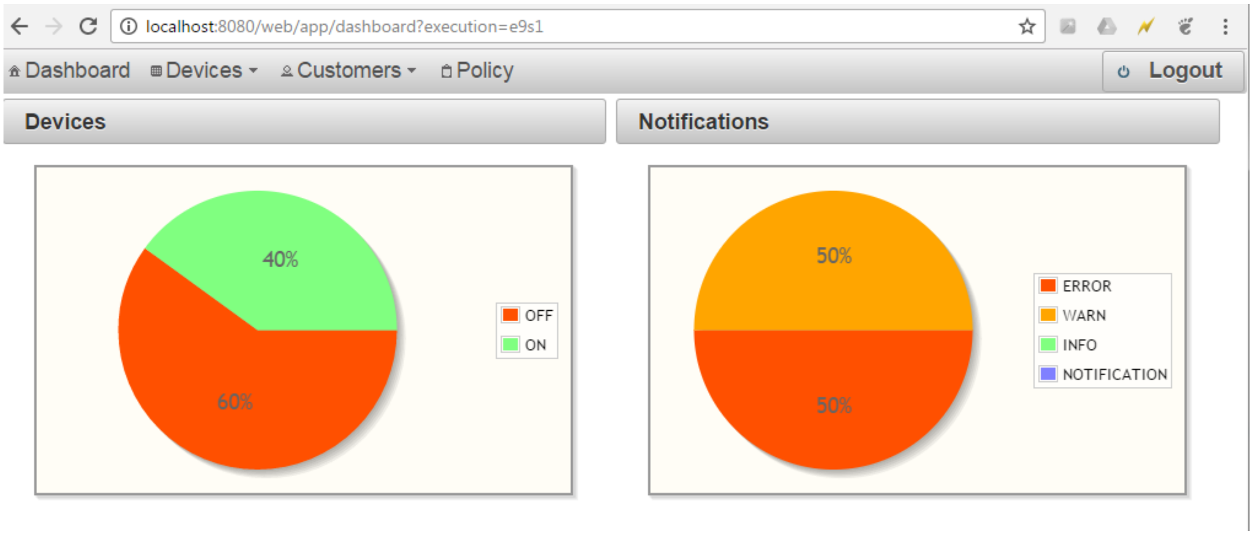
Group filed is support auto completed with drop down box type.

From, to and price is input support increase and decrease value by button.

Save button: call save request to save into database

Cancel button: close the dialog when click.

11. Dashboard Page



Dashboard Devices:

Summary and show how much devices are off and how much devices are on.

Dashboard Notification:

Summary and show how much notifications of each type.

E. Maintenance

Test plan

Condition	**Produce**	**Expected result**
Go to login page	Provide correct username and password of admin account	Go to management page
	Provide wrong password	Show error message
	Leave input blank	Show requirement error message
	Login with non admin account	Go to 403 access denied page
Login as administrator	Modify a device	Device information is modified
	Delete a device has relate data	Reject to delete device
	Delete a device has no relate data	Device is deleted
	Poll device	Device have to response its information
	Poll device and reach timeout	Device is set to off status and release IP address
	Start a device has in database	Update new device information
	Turn off device	Show device is OFF with IP address is released
	Start a new device	New device is add into device table
	Make device conflict IP address	Notification is created
	Make device update invalid information	Notification is created
	Create customer	New customer is created
	Modify customer	New customer information is update
	Create customer with blank inputs	Show required message
	Modify customer by blank inputs	Show required message

Condition	**Produce**	**Expected result**
	Delete customer has no related data	Customer is deleted
	Delete customer has related data	Reject to delete customer
	Create new policy	New policy is created
	Create new policy with invalid information	Reject to create new policy
	Update new policy	Policy information is changed
	Update new policy with invalid information	Reject to change to new device information
	Remove policy	Policy is remove
	Assign customer to use device	New assignment is created
	Change assign customer to other device	Change assignment
	Remove assignment	Assignment is removed
	View payment of assigned device	Show correct payment
	View payment of no assigned device	Show payment is 0
	Assign customer to use new device	Not defined

F. Evaluation

This project is solved main problem of Vietnamese electric power company. But, it is still a small project in education scope. Actually, EVN has developed a system to handle this problem from year of 2001. RF-Spider is researched in more than 10 years by EVN, but it is still no covered all there problem. There are about 35 percent of electric counter meter is replace with new one. EVN plan to deployed 100 percent of new electric counter meter in 2020. In my opinion, this project is a good try to solve problem of EVN, but it is not equal to RF-Spider of EVN.

If this project can be invested more, it may be replace EVN RF-Spider in future. Because of poor resources, and lack of experiences, the first version of this project is not met my requirement.

In future devices should be able to detect and help administrator track problem on devices.

Appendix A

Topics

In JMS a Topic implements *publish and subscribe* semantics. When you publish a message it goes to all the subscribers who are interested - so zero to many subscribers will receive a copy of the message. Only subscribers who had an active subscription at the time the broker receives the message will get a copy of the message.

Queues

A JMS Queue implements *load balancer* semantics. A single message will be received by exactly one consumer. If there are no consumers available at the time the message is sent it will be kept until a consumer is available that can process the message. If a consumer receives a message and does not acknowledge it before closing then the message will be redelivered to another consumer. A queue can have many consumers with messages *load balanced* across the available consumers.

So Queues implement a reliable load balancer in JMS.

Appendix B

Project installation guide

Download and install MySQL or XAMPP at https://www.apachefriends.org/download.html

Download and install ActiveMQ at http://activemq.apache.org/download.html

Download and install Maven at https://maven.apache.org/download.cgi

Run mySQL server and run <source code>/etc/database.sql

Run ActiveMQ service

There are two ways to deploy web on tomcat

Deploy war file: run tomcat server on XAMPP then deploy war file into webapp of tomcat

Deploy by maven tomcat plug in

Go to source code folder by command line, or terminal tool.

run command: mvn install

go to <source code>/epcm-web/

run command: mvn tomcat7:run

now service is available on localhost:8080/web/

Device simulator run:

attach simulator.zip and run run.bat file.

All project source code is available at **https://github.com/delvobmt/epcm**

References

Spring Framework Reference Documentation. 2016. *Spring Framework Reference Documentation*. [ONLINE] Available at: http://docs.spring.io/spring/docs/current/spring-framework-reference/htmlsingle/. [Accessed 22 November 2016].

Spring Web Flow Reference Guide. 2016. *Spring Web Flow Reference Guide*. [ONLINE] Available at: http://docs.spring.io/spring-webflow/docs/2.3.4.RELEASE/reference/html/. [Accessed 22 November 2016].

Spring Security Reference. 2016. *Spring Security Reference*. [ONLINE] Available at: http://docs.spring.io/spring-security/site/docs/4.2.0.RELEASE/reference/htmlsingle/. [Accessed 22 November 2016]

Apache ActiveMQ ™ -- Index . 2016. *Apache ActiveMQ ™ -- Index* . [ONLINE] Available at: http://activemq.apache.org/. [Accessed 22 November 2016].

. 2016. *Hibernate ORM documentation (5.2) - Hibernate ORM*. [ONLINE] Available at: http://hibernate.org/orm/documentation/5.2/. [Accessed 22 November 2016].

Çağatay Çivici. 2016. *Primefaces User Guide*. [ONLINE] Available at: http://www.primefaces.org/docs/guide/primefaces_user_guide_6_0.pdf. [Accessed 22 November 2016].

tutorialspoint.com. 2016. *MySQL Tutorial*. [ONLINE] Available at: https://www.tutorialspoint.com/mysql/. [Accessed 22 November 2016].

tutorialspoint.com. 2016. *Maven Tutorial*. [ONLINE] Available at: https://www.tutorialspoint.com/maven/. [Accessed 22 November 2016].

Stack Overflow. 2016. *Stack Overflow*. [ONLINE] Available at: http://stackoverflow.com/. [Accessed 22 November 2016].

CPCIT . 2016 [ONLINE] Available at: http://www.cpcit.vn/Detmeter.aspx?T=0&ID=26. [Accessed 22 November 2016].