NAMA : JAMALUDIN

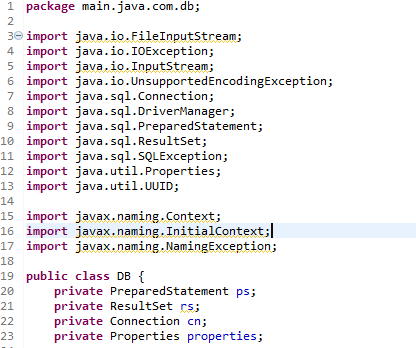
ALUR PROGRAM SPRING BOOT---QUADRANT API---API CALLER

1. Membuat sebuah aplikasi spring, masuk terlebih ke websitenya spring.io lalu masukkan semua nama, depedency lalu genereate.

Pada aplikasi api-quadrant kita melibatkan database, pada package main.java.com.db terdapat class yang menjelaskan hubungan dengan database.

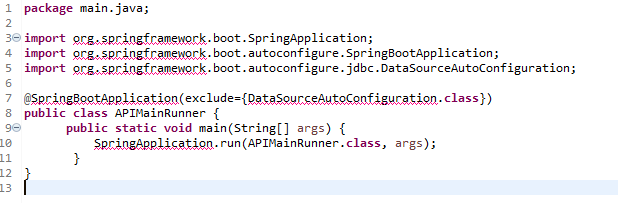
Dengan mengimport libarary yang di butuhkan untuk koneksi ke database bisa dilihat di bawah ini, ada drivermanager untuk mengatur drivernya , connection dll.

1. Database



1. Pada saat membuka generate tadi maka spring sudah menyediakan class yang isinya seperti di bawah ini, pada class ini pula nantinya aplikasi ini run.

Classs ini terdapat di package main.java terdapat annotasi @SpringBootApplication yang artinya untuk merunning si spring ini, annotasi nya harus disertai dengan library dari spring.



Library yang dibutuhkan untuk memanggil annotasi terletak pada file pom.xml, dalam pom.xml bukan hanya annotasi melainkan semua depedency yang berhubungan dengan program yang dibuat, library web untuk membuat aplikasi berbasis web, ada juga mysql-connector yang berhubungan dengan database, jpa berhubungan dengan pengolahan database dll. Semua terletak di pom.xml dan maven bertugas untuk mendownload semua depedency yang dibutuhkan.

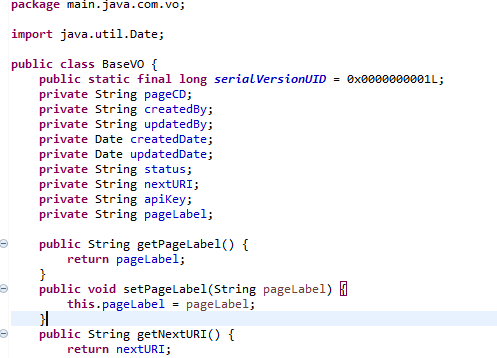
Berikut depedency yang dibutuhkan dalam aplikasi “quadrant-api”



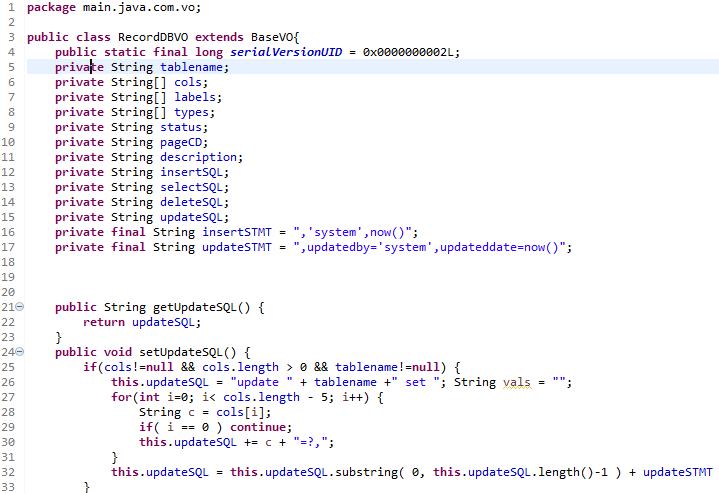


1. Aplikasi ini berbasis web dan menggunakan pola MVC (model, view, controller), dalam package main.java.com.vo dibuat model dengan type data string dan juga getter/setter yang nantinya akan di panggil ke controller.

* **Class BaseVO**



Dalam package main.java.com.vo, Pada saat perintah select, update, delete dan insert di jalankan class ini akan menjalankan



1. **a. SelectSVC**

|  |
| --- |
| **package** main.java.com.services; // berada dalam package services  **import** main.java.com.bo.GateBO; //import class gateBO  **import** main.java.com.vo.RecordVO; // import class RecordVO  **public** **class** SelectSVC **extends** SERVICES **implements** SERVICESINTF { //nama class dan inheritance ke class service    **private** **final** String svcCD = "SVCSEL";  **private** SERVICES nextSVC = **new** DeleteSVC(); //membuat methode  @Override //  **public** RecordVO runService(RecordVO vo) {  System.***out***.println("Before Command called: " + **this**.getSvcCD() + " vs " + vo.getSvcCD());  //membuat sebuah kondisi  **if**(vo != **null** && vo.getSvcCD() != **null** && **this**.getSvcCD().equalsIgnoreCase( vo.getSvcCD() ) ) {  System.***out***.println("Command called: " + vo.getSvcCD());  GateBO bo = **new** GateBO(); **return** bo.select(vo);  } **else** **if**( **this**.getNextSVC() != **null** ) {  **return** **this**.getNextSVC().runService(vo);  }  **return** **null**;  }  @Override  **public** SERVICES setNextService(SERVICES svc) {  **return** **this**.nextSVC = svc;  }  **public** SERVICES getNextSVC() {  **return** nextSVC;  }  **public** **void** setNextSVC(SERVICES nextSVC) {  **this**.nextSVC = nextSVC;  }  **public** String getSvcCD() {  **return** svcCD;  }  } |

**b. InsertSVC**

|  |
| --- |
| **package** main.java.com.services;  **import** main.java.com.bo.GateBO;  **import** main.java.com.vo.RecordVO;  **public** **class** InsertSVC **extends** SERVICES **implements** SERVICESINTF {  **private** **final** String svcCD = "SVCINS";  **private** SERVICES nextSVC = **new** UpdateSVC();  @Override  **public** RecordVO runService(RecordVO vo) {  System.***out***.println("Before Command called: " + **this**.getSvcCD() + " vs " + vo.getSvcCD());  **if**(vo != **null** && vo.getSvcCD() != **null** && **this**.getSvcCD().equalsIgnoreCase( vo.getSvcCD() ) ) {  System.***out***.println("Command called: " + vo.getSvcCD());  GateBO bo = **new** GateBO(); **return** bo.addRecord(vo);  } **else** **if**( **this**.getNextSVC() != **null** ) {  **return** **this**.getNextSVC().runService(vo);  }  **return** **null**;  }  @Override  **public** SERVICES setNextService(SERVICES svc) {  **return** **this**.nextSVC = svc;  }  **public** SERVICES getNextSVC() {  **return** nextSVC;  }  **public** **void** setNextSVC(SERVICES nextSVC) {  **this**.nextSVC = nextSVC;  }  **public** String getSvcCD() {  **return** svcCD;  }      } |

**c. DeleteSVC**

|  |
| --- |
| **package** main.java.com.services;  **import** main.java.com.bo.GateBO;  **import** main.java.com.vo.RecordVO;  **public** **class** DeleteSVC **extends** SERVICES **implements** SERVICESINTF {  **private** **final** String svcCD = "SVCDEL";  **private** SERVICES nextSVC = **null**;  @Override  **public** RecordVO runService(RecordVO vo) {  System.***out***.println("Before Command called: " + **this**.getSvcCD() + " vs " + vo.getSvcCD());  **if**(vo != **null** && vo.getSvcCD() != **null** && **this**.getSvcCD().equalsIgnoreCase( vo.getSvcCD() ) ) {  System.***out***.println("Command called: " + vo.getSvcCD());  GateBO bo = **new** GateBO(); **return** bo.delete(vo);  } **else** **if**( **this**.getNextSVC() != **null** ) {  **return** **this**.getNextSVC().runService(vo);  }  **return** **null**;  }  @Override  **public** SERVICES setNextService(SERVICES svc) {  **return** **this**.nextSVC = svc;  }  **public** SERVICES getNextSVC() {  **return** nextSVC;  }  **public** **void** setNextSVC(SERVICES nextSVC) {  **this**.nextSVC = nextSVC;  }  **public** String getSvcCD() {  **return** svcCD;  }      } |

d. UpdateSVC

|  |
| --- |
| **package** main.java.com.services; //package  //import class dari package lain  **import** main.java.com.bo.GateBO;  **import** main.java.com.vo.RecordVO;  **//class updateSVC inheritance class service**  **public** **class** UpdateSVC **extends** SERVICES **implements** SERVICESINTF {  **private** **final** String svcCD = "SVCUPD";  **private** SERVICES nextSVC = **new** SelectSVC();  @Override // untuk melakukan inject instance dari suatu bean ke objek yang memiliki dependency  **public** RecordVO runService(RecordVO vo) {  System.***out***.println("Before Command called: " + **this**.getSvcCD() + " vs " + vo.getSvcCD());  **if**(vo != **null** && vo.getSvcCD() != **null** && **this**.getSvcCD().equalsIgnoreCase( vo.getSvcCD() ) ) {  System.***out***.println("Command called: " + vo.getSvcCD());  GateBO bo = **new** GateBO(); **return** bo.update(vo);  } **else** **if**( **this**.getNextSVC() != **null** ) {  **return** **this**.getNextSVC().runService(vo);  }  **return** **null**;  }  @Override  **public** SERVICES setNextService(SERVICES svc) {  **return** **this**.nextSVC = svc;  }  **public** SERVICES getNextSVC() {  **return** nextSVC;  }  **public** **void** setNextSVC(SERVICES nextSVC) {  **this**.nextSVC = nextSVC;  }  **public** String getSvcCD() {  **return** svcCD;  }    } |

e. Controller

pada class controller ini, kegiatan crud saling terhubung, controller mengatur alur request dan respone

|  |
| --- |
| **package** main.java.com.controller; // package controller  //library yang sudah didownload di pom.xml melalui maven  **import** org.springframework.beans.factory.annotation.Autowired;  **import** org.springframework.http.HttpStatus;  **import** org.springframework.http.MediaType;  **import** org.springframework.http.ResponseEntity;  **import** org.springframework.web.bind.annotation.PathVariable;  **import** org.springframework.web.bind.annotation.RequestBody;  **import** org.springframework.web.bind.annotation.RequestMapping;  **import** org.springframework.web.bind.annotation.RequestMethod;  **import** org.springframework.web.bind.annotation.RestController;  **import** com.fasterxml.jackson.annotation.JsonIgnoreProperties;  **import** org.springframework.web.bind.annotation.ResponseBody;  **import** org.springframework.web.bind.annotation.RestController;  //import class dari package  **import** main.java.com.services.APIServicesIntf;  **import** main.java.com.services.SERVICES;  **import** main.java.com.vo.RecordVO;  **import** main.java.com.vo.BaseVO;  **import** com.fasterxml.jackson.annotation.JsonIgnoreProperties;  // agar spring container dapat membuat definisi bean dari suatu kelas dan menunjukkan kelas tersebut adalah kelas rest controller  @RestController  @JsonIgnoreProperties(ignoreUnknown = **true**)  **public** **class** GateController {  **private** **static** **final** String ***api\_public\_key*** = "quadrant\_api\_key";    @Autowired // Anotasi yang digunakan untuk melakukan inject instance dari suatu bean ke objek yang memiliki dependency  APIServicesIntf service;    //anotasi memetakan permintaan web ke metode di kelas penanganan permintaan dengan tanda tangan metode yang fleksibel.  @RequestMapping(method = RequestMethod.POST, value = "/api/coded", produces = org.springframework.http.MediaType.APPLICATION\_JSON\_VALUE)    //annotasi untuk merespone permintaan yang dihubungkan dengan recordVO  **public** @ResponseBody RecordVO getProduct(@RequestBody RecordVO vo) {    //sebuah kondisi yang mendapatkan keyapi jika sesaui maka akan login jika tidak akan gagal(failed for api key wrong)  **if**(vo!=**null** && vo.getApiKey() != **null** && vo.getApiKey().trim().equalsIgnoreCase(***api\_public\_key***)) {  System.***out***.println("PAGECD = " + vo.getPageCD() + " - SVCCODE = " + vo.getSvcCD() );  **return** service.runService(vo);  }  RecordVO r = **new** RecordVO(); r.setStatus("FAILED\_FOR\_API\_KEY\_WRONG");  **return** r;  }    } |

f. classBO

|  |
| --- |
| **package** main.java.com.bo; //berada di package bo  **import** main.java.com.vo.RecordVO; //import class recordVO  **public** **class** GateBO **implements** BaseIntf { //class GateBO yang menimplementasikan sebuah class interface yaitu BaseIntf  @Override  **public** RecordVO addRecord(RecordVO vo) {  BaseIntf bo = **new** BaseBO();  **return** bo.addRecord(vo);  }  @Override  **public** RecordVO update(RecordVO vo) {  BaseIntf bo = **new** BaseBO();  **return** bo.update(vo);  }  @Override  **public** RecordVO select(RecordVO vo) {  BaseIntf bo = **new** BaseBO();  **return** bo.select(vo);  }  @Override  **public** RecordVO delete(RecordVO vo) {  BaseIntf bo = **new** BaseBO();  **return** bo.delete(vo);  }  } |

Interfcae BaseIntf

|  |
| --- |
| **package** main.java.com.bo;  **import** java.sql.PreparedStatement;  **import** main.java.com.vo.RecordVO;  **public** **interface** BaseIntf {  **public** RecordVO addRecord(RecordVO vo);  **public** RecordVO update(RecordVO vo);  **public** RecordVO select(RecordVO vo);  **public** RecordVO delete(RecordVO vo);  } |

g. Interface BaseSvc

|  |
| --- |
| **package** main.java.com.services; //berada di package services  **import** main.java.com.vo.RecordVO; //import class recordVO  //sebuah interface yang menampilkan pengolahan insert, update, select dan delete yang berhubungan dengan frontend.  **public** **interface** BaseSvc {  **public** RecordVO addRecord(RecordVO vo);  **public** RecordVO update(RecordVO vo);  **public** RecordVO select(RecordVO vo);  **public** RecordVO delete(RecordVO vo);  } |

g. class APIServices

|  |
| --- |
| **package** main.java.com.services; // berada di package servcie  **import** main.java.com.vo.RecordVO; //import class recordVO  **import** org.springframework.stereotype.Service; //import library service  @Service // Anotasi yang digunakan agar Spring container dapat membuat definisi bean dari suatu kelas dan menunjukkan kelas tersebut adalah kelas service  //  **public** **class** APIServices **extends** SERVICES **implements** APIServicesIntf{  **public** RecordVO runService(RecordVO vo) {  System.***out***.println("APIServices.runService() is called. ");  **return** **new** InsertSVC().runService(vo);  }  } |

h. Interface APIServicesIntf

|  |
| --- |
| **package** main.java.com.services; // berada di package services  **import** main.java.com.vo.RecordVO; //import class RecordVO  //Interface untuk menjalanka  **public** **interface** APIServicesIntf {  **public** RecordVO runService(RecordVO vo);  } |