Back-end:

package info.smartkit.shiny.guide.configs;

import java.io.IOException;

import org.apache.commons.lang.StringUtils;

import org.apache.logging.log4j.LogManager;

import org.apache.logging.log4j.Logger;

import org.springframework.context.ApplicationContextInitializer;

import org.springframework.context.ConfigurableApplicationContext;

import org.springframework.core.env.ConfigurableEnvironment;

import org.springframework.core.io.support.ResourcePropertySource;

/\*\*

\* Register this with the DispatcherServlet in a ServletInitializer class like:

\* dispatcherServlet.setContextInitializers(new PropertiesInitializer());

\*/

public class PropertiesInitializer implements ApplicationContextInitializer<ConfigurableApplicationContext> {

private static final Logger LOG = LogManager.getLogger(PropertiesInitializer.class);

/\*\*

\* Runs as appInitializer so properties are wired before spring beans

\*/

@Override

public void initialize(ConfigurableApplicationContext applicationContext) {

ConfigurableEnvironment env = applicationContext.getEnvironment();

String[] activeProfiles = getActiveProfiles(env);

for (String profileName : activeProfiles) {

LOG.info("Loading properties for Spring Active Profile: {}", profileName);

try {

ResourcePropertySource propertySource =

new ResourcePropertySource(profileName + "EnvProperties", "classpath:application-" + profileName

+ ".properties");

env.getPropertySources().addLast(propertySource);

LOG.debug("propertySource:" + propertySource.toString());

// Work-flow setting initialization here.

} catch (IOException e) {

LOG.error("ERROR during environment properties setup - TRYING TO LOAD: " + profileName, e);

// Okay to silently fail here, as we might have profiles that do

// not have properties files (like dev1, dev2, etc)

}

}

}

/\*\*

\* Returns either the ActiveProfiles, or if empty, then the DefaultProfiles from Spring

\*/

protected String[] getActiveProfiles(ConfigurableEnvironment env) {

String[] activeProfiles = env.getActiveProfiles();

if (activeProfiles.length > 0) {

LOG.info("Using registered Spring Active Profiles: {}", StringUtils.join(activeProfiles, ", "));

return activeProfiles;

}

String[] defaultProfiles = env.getDefaultProfiles();

LOG.info("No Active Profiles found, using Spring Default Profiles: {}", StringUtils.join(defaultProfiles, ", "));

return defaultProfiles;

}

}

package info.smartkit.shiny.guide.configs;

import java.io.IOException;

import javax.servlet.Filter;

import javax.servlet.FilterChain;

import javax.servlet.FilterConfig;

import javax.servlet.ServletException;

import javax.servlet.ServletRequest;

import javax.servlet.ServletResponse;

import javax.servlet.http.HttpServletResponse;

import org.springframework.stereotype.Component;

/\*\*

\* @author yangboz

\* @see https://spring.io/guides/gs/rest-service-cors/

\*/

@Component

public class SimpleCORSFilter implements Filter {

public void doFilter(ServletRequest req, ServletResponse res, FilterChain chain) throws IOException, ServletException {

HttpServletResponse response = (HttpServletResponse) res;

response.setHeader("Access-Control-Allow-Origin", "\*");

response.setHeader("Access-Control-Allow-Methods", "POST, GET, OPTIONS, PUT, DELETE, PATCH");

response.setHeader("Access-Control-Max-Age", "3600");

response.setHeader("Access-Control-Allow-Headers", "Origin, X-Requested-With, Content-Type, Accept");

chain.doFilter(req, res);

}

public void init(FilterConfig filterConfig) {}

public void destroy() {}

}package info.smartkit.shiny.guide.configs;

import org.springframework.context.annotation.Configuration;

import org.springframework.web.servlet.config.annotation.ResourceHandlerRegistry;

import org.springframework.web.servlet.config.annotation.WebMvcConfigurerAdapter;

@Configuration

public class StaticResourceConfiguration extends WebMvcConfigurerAdapter {

private static final String[] CLASSPATH\_RESOURCE\_LOCATIONS = {

"classpath:/META-INF/resources/", "classpath:/resources/",

"classpath:/static/", "classpath:/public/"};

/\*\*

\* Add our static resources folder mapping.

\*/

// @Override

// public void addResourceHandlers(ResourceHandlerRegistry registry) {

// //

// registry.addResourceHandler("/\*\*").addResourceLocations(

// CLASSPATH\_RESOURCE\_LOCATIONS);

// //

// registry.addResourceHandler("/uploads/\*\*").addResourceLocations(

// "classpath:/uploads/");

// // Activiti repository resources(diagram picture,process BPM files).

// // registry.addResourceHandler("/repository/\*\*").addResourceLocations("classpath:/repository/");

// // Jasper report

// registry.addResourceHandler("/static/\*\*").addResourceLocations(

// "classpath:/static/");

// // registry.addResourceHandler("/reports/\*\*").addResourceLocations("classpath:/reports/");

// //

// super.addResourceHandlers(registry);

// }

@Override

public void addResourceHandlers(ResourceHandlerRegistry registry) {

registry.addResourceHandler("swagger-ui.html")

.addResourceLocations("classpath:/META-INF/resources/");

registry.addResourceHandler("/webjars/\*\*")

.addResourceLocations("classpath:/META-INF/resources/webjars/");

registry.addResourceHandler("/uploads/\*\*").addResourceLocations(

"classpath:/uploads/");

}

}

package info.smartkit.shiny.guide.configs;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.web.bind.annotation.RequestMethod;

import springfox.documentation.builders.PathSelectors;

import springfox.documentation.builders.RequestHandlerSelectors;

import springfox.documentation.builders.ResponseMessageBuilder;

import springfox.documentation.schema.ModelRef;

import springfox.documentation.service.ApiInfo;

import springfox.documentation.spi.DocumentationType;

import springfox.documentation.spring.web.plugins.Docket;

import springfox.documentation.swagger2.annotations.EnableSwagger2;

import java.util.ArrayList;

/\*\*

\* The Class SwaggerConfig.

\*/

@Configuration

@EnableSwagger2

public class SwaggerConfig {

@Bean

public Docket api() {

return new Docket(DocumentationType.SWAGGER\_2)

.select()

.apis(RequestHandlerSelectors.basePackage("info.smartkit.shiny.guide.controllers"))

// .paths(PathSelectors.ant("/shiny.guide"))

.build()

.apiInfo(apiInfo());

// .useDefaultResponseMessages(false)

// .globalResponseMessage(RequestMethod.GET,

// new ArrayList(new ResponseMessageBuilder()

// .code(500)

// .message("500 message")

// .responseModel(new ModelRef("Error"))

// .build(),

// new ResponseMessageBuilder()

// .code(403)

// .message("Forbidden!")

// .build()));

}

private ApiInfo apiInfo() {

ApiInfo apiInfo = new ApiInfo(

"TD.SMARTKIT.INFO REST API",

"SMARTKIT.INFO description of API.",

"0.0.1",

"Terms of service",

"smartkit@msn.com",

"License of API",

"API license URL");

return apiInfo;

}

}

package info.smartkit.shiny.guide.controllers;

import info.smartkit.shiny.guide.dao.ConsultInfoDao;

import info.smartkit.shiny.guide.dao.UserInfoDao;

import info.smartkit.shiny.guide.dto.JsonObject;

import info.smartkit.shiny.guide.vo.ConsultInfo;

import info.smartkit.shiny.guide.vo.UserInfo;

import io.swagger.annotations.ApiOperation;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.\*;

import javax.validation.Valid;

import javax.ws.rs.core.MediaType;

/\*\*

\* Created by smartkit on 27/02/2017.

\*/

@RestController

// @see: http://spring.io/guides/gs/reactor-thumbnailer/

@RequestMapping(value = "/consult")

public class ConsultInfoController {

@Autowired

private ConsultInfoDao DAO;

/\*\*

\* Posting anew ConsultInfo

\*

\* @param DAO

\* @return Response a string describing if the user info is successfully created or not.

\*/

@RequestMapping(method = RequestMethod.POST, consumes = MediaType.APPLICATION\_JSON)

@ApiOperation(httpMethod = "POST", value = "Response a string describing if the user info is successfully created or not.")

public JsonObject create(@RequestBody @Valid ConsultInfo info) {

return new JsonObject(DAO.save(info));

}

@RequestMapping(value = "/{id}", method = RequestMethod.GET)

@ApiOperation(httpMethod = "GET", value = "Response a string describing if the user info id is successfully get or not.")

public JsonObject get(@PathVariable("id") long id) {

return new JsonObject(this.DAO.findOne(id));

}

@RequestMapping(value = "/{id}", method = RequestMethod.DELETE)

@ApiOperation(httpMethod = "DELETE", value = "Response a string describing if the user info is successfully delete or not.")

public JsonObject delete(@PathVariable("id") long id) {

try {

this.DAO.delete(id);

} catch (Exception e) {

return new JsonObject(e.toString());

}

return new JsonObject(new ResponseEntity<Boolean>(Boolean.TRUE, HttpStatus.OK));

}

@RequestMapping(method = RequestMethod.GET)

@ApiOperation(httpMethod = "GET", value = "Response a list describing all of user info that is successfully get or not.")

public JsonObject list() {

// return new JsonObject(this.\_alarmInfoDao.findAll());

return new JsonObject(this.DAO.findAll());

}

@RequestMapping(value = "/{id}/iid/{iid}/pid/{pid}", method = RequestMethod.PUT)//

@ApiOperation(httpMethod = "PUT", value = "Response a string describing if the consult info is successfully patched or not.")

public JsonObject patch(@PathVariable("id") long id, @PathVariable("iid") int iid, @PathVariable("pid") int pid) {

ConsultInfo find = this.DAO.findOne(id);

find.setIid(iid);

find.setPid(pid);

return new JsonObject(this.DAO.save(find));

}

}package info.smartkit.shiny.guide.controllers;

import com.javacodegeeks.drools.model.Customer;

import com.javacodegeeks.drools.model.Product;

import com.mysql.jdbc.jdbc2.optional.MysqlDataSource;

import info.smartkit.shiny.guide.dao.\*;

import info.smartkit.shiny.guide.dto.ConsultInfoFull;

import info.smartkit.shiny.guide.service.DiagnosisService;

import info.smartkit.shiny.guide.settings.DataBaseSettings;

import info.smartkit.shiny.guide.utils.MahoutUtils;

import info.smartkit.shiny.guide.vo.\*;

import io.swagger.annotations.ApiOperation;

import io.swagger.annotations.ApiResponse;

import org.apache.logging.log4j.LogManager;

import org.apache.logging.log4j.Logger;

import org.apache.mahout.cf.taste.common.TasteException;

import org.apache.mahout.cf.taste.impl.model.jdbc.MySQLJDBCDataModel;

import org.apache.mahout.cf.taste.model.JDBCDataModel;

import org.apache.mahout.cf.taste.recommender.RecommendedItem;

import org.drools.template.jdbc.ResultSetGenerator;

import org.kie.api.io.ResourceType;

import org.kie.internal.KnowledgeBase;

import org.kie.internal.KnowledgeBaseFactory;

import org.kie.internal.builder.DecisionTableConfiguration;

import org.kie.internal.builder.DecisionTableInputType;

import org.kie.internal.builder.KnowledgeBuilder;

import org.kie.internal.builder.KnowledgeBuilderFactory;

import org.kie.internal.io.ResourceFactory;

import org.kie.internal.runtime.StatelessKnowledgeSession;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.RestController;

import java.io.FileInputStream;

import java.io.IOException;

import java.util.List;

/\*\*

\* Created by smartkit on 01/03/2017.

\*/

@RestController

// @see: http://spring.io/guides/gs/reactor-thumbnailer/

@RequestMapping(value = "/diagnosis")

public class DiagnosisController {

@Autowired

private UserInfoDao userInfoDao;

@Autowired

private MPrescriptionDao mPrescriptionDao;

@Autowired

private EInstructionDao eInstructionDao;

@Autowired

private ConsultInfoDao consultInfoDao;

@Autowired

private ItemDetailDao itemDetailDao;

@Autowired DiagnosisService diagnosisService;

private static Logger LOG = LogManager.getLogger(DiagnosisController.class);

private static StatelessKnowledgeSession session;

@RequestMapping(value = "user/infer/{id}/{order}", method = RequestMethod.GET)

@ApiOperation(httpMethod = "GET", value = "Response a consult info fully describing if the user's diagnosis is successfully inference or not.")

public ConsultEinstrMpers inferByFacts(@PathVariable("id") long id,@PathVariable("order") int order) throws Exception {

// long uiid = userInfoDao.findOne(id).getItemId();

UserInfo userInfo = userInfoDao.findOne(id);

LOG.info("find one user info:"+userInfo.toString());

return diagnosisService.inferByFacts(userInfo.getItemId(),order);

}

@RequestMapping(value = "user/{id}", method = RequestMethod.GET)

@ApiOperation(httpMethod = "GET", value = "Response a consult info fully describing if the user's diagnosis is successfully get or not.")

public ConsultInfoFull getByUser(@PathVariable("id") long id) throws Exception {

try{

long findUserId = userInfoDao.findOne(id).getId();

}catch (NullPointerException ex){

}

/\*

//http://docs.jboss.org/drools/release/6.0.1.Final/drools-docs/html\_single/index.html#d0e4969

With Rule Templates the data is separated from the rule and there are no restrictions on which part of the rule is data-driven.

So whilst you can do everything you could do in decision tables you can also do the following:

1.store your data in a database (or any other format)

2.conditionally generate rules based on the values in the data

3.use data for any part of your rules (e.g. condition operator, class name, property name)

4.run different templates over the same data

\*/

//maintain rules in a database table, rule templates:

// Get results from your DB query...

// resultSet = preparedStmt.executeQuery();

// // Generate the DRL...

// ResultSetGenerator resultSetGenerator = new ResultSetGenerator();

// String drl = resultSetGenerator.compile(resultSet,

// new FileInputStream("path/to/template.drt"));

//

KnowledgeBase knowledgeBase = createKnowledgeBaseFromSpreadsheet();

session = knowledgeBase.newStatelessKnowledgeSession();

//

Customer customer = Customer.newCustomer();

ItemDetail findOne = itemDetailDao.findOne(id);

customer.addItems(findOne.getProducts());

//hard-code for testing.

customer.setUltimate(findOne.getZzzd());

customer.setPathology(findOne.getBlzd());

customer.setEndoscope(findOne.getNjzd());

LOG.info("new customer:"+customer.toString());

session.execute(customer);

//

//

ConsultInfoFull consultInfoFull = null;

try {

long consultId = customer.getConsult();

LOG.info("consultId:"+consultId);

ConsultInfo find = consultInfoDao.findOne(consultId);

//FIXME:join table to select.

long instructionId = find.getIid();

EInstruction eInstruction = eInstructionDao.findOne(instructionId);

long prescriptionId = find.getPid();

MPrescription mPrescription = mPrescriptionDao.findOne(prescriptionId);

consultInfoFull = new ConsultInfoFull(eInstruction, mPrescription);

}catch (NullPointerException ex){

////out of inference,turn to recommendation

}

return consultInfoFull;

}

private static KnowledgeBase createKnowledgeBaseFromSpreadsheet()

throws Exception {

DecisionTableConfiguration dtconf = KnowledgeBuilderFactory

.newDecisionTableConfiguration();

dtconf.setInputType(DecisionTableInputType.XLS);

KnowledgeBuilder knowledgeBuilder = KnowledgeBuilderFactory

.newKnowledgeBuilder();

knowledgeBuilder.add(ResourceFactory

.newClassPathResource("td\_user\_item.xls"),

ResourceType.DTABLE, dtconf);

if (knowledgeBuilder.hasErrors()) {

throw new RuntimeException(knowledgeBuilder.getErrors().toString());

}

KnowledgeBase knowledgeBase = KnowledgeBaseFactory.newKnowledgeBase();

knowledgeBase.addKnowledgePackages(knowledgeBuilder

.getKnowledgePackages());

return knowledgeBase;

}

}

package info.smartkit.shiny.guide.controllers;

/\*\*

\* Created by yangboz on 23/02/2017.

\*/

import info.smartkit.shiny.guide.dao.EInstructionDao;

import info.smartkit.shiny.guide.dao.UserInfoDao;

import info.smartkit.shiny.guide.dto.JsonObject;

import info.smartkit.shiny.guide.vo.EInstruction;

import info.smartkit.shiny.guide.vo.UserInfo;

import io.swagger.annotations.ApiOperation;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.\*;

import javax.validation.Valid;

import javax.ws.rs.core.MediaType;

/\*\*

\* Created by smartkit on 20/02/2017.

\*/

@RestController

// @see: http://spring.io/guides/gs/reactor-thumbnailer/

@RequestMapping(value = "/instruction")

public class EInstructionController {

@Autowired

private EInstructionDao DAO;

/\*\*

\* Posting anew EInstruction

\*

\* @param EInstruction

\* @return Response a string describing if the user info is successfully created or not.

\*/

@RequestMapping(method = RequestMethod.POST, consumes = MediaType.APPLICATION\_JSON)

@ApiOperation(httpMethod = "POST", value = "Response a string describing if the EInstruction is successfully created or not.")

public JsonObject create(@RequestBody @Valid EInstruction info) {

return new JsonObject(DAO.save(info));

}

@RequestMapping(value = "/{id}", method = RequestMethod.GET)

@ApiOperation(httpMethod = "GET", value = "Response a string describing if the EInstruction id is successfully get or not.")

public JsonObject get(@PathVariable("id") long id) {

return new JsonObject(this.DAO.findOne(id));

}

@RequestMapping(value = "/{id}", method = RequestMethod.DELETE)

@ApiOperation(httpMethod = "DELETE", value = "Response a string describing if the EInstruction is successfully delete or not.")

public JsonObject delete(@PathVariable("id") long id) {

try {

this.DAO.delete(id);

} catch (Exception e) {

return new JsonObject(e.toString());

}

return new JsonObject(new ResponseEntity<Boolean>(Boolean.TRUE, HttpStatus.OK));

}

@RequestMapping(method = RequestMethod.GET)

@ApiOperation(httpMethod = "GET", value = "Response a list describing all of EInstruction that is successfully get or not.")

public JsonObject list() {

// return new JsonObject(this.\_alarmInfoDao.findAll());

return new JsonObject(this.DAO.findAll());

}

// @RequestMapping(value = "/{id}/status/{value}", method = RequestMethod.PUT)//

// @ApiOperation(httpMethod = "PUT", value = "Response a string describing if the alarm status is successfully patched or not.")

// public JsonObject patch(@PathVariable("id") long id, @PathVariable("value") int value) {

// UserInfo find = this.userInfoDao.findOne(id);

// find.setStatus(value);

// return new JsonObject(this.userInfoDao.save(find));

// }

}

package info.smartkit.shiny.guide.controllers;

import info.smartkit.shiny.guide.dao.ItemDetailDao;

import info.smartkit.shiny.guide.dao.ItemInfoDao;

import info.smartkit.shiny.guide.dto.JsonObject;

import info.smartkit.shiny.guide.vo.ItemDetail;

import info.smartkit.shiny.guide.vo.ItemInfo;

import io.swagger.annotations.ApiOperation;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.\*;

import javax.validation.Valid;

import javax.ws.rs.core.MediaType;

/\*\*

\* Created by smartkit on 28/02/2017.

\*/

@RestController

// @see: http://spring.io/guides/gs/reactor-thumbnailer/

@RequestMapping(value = "/item/detail")

public class ItemDetailController {

@Autowired

private ItemDetailDao DAO;

/\*\*

\* Posting anew UserInfo

\*

\* @param DAO

\* @return Response a string describing if the user info is successfully created or not.

\*/

@RequestMapping(method = RequestMethod.POST, consumes = MediaType.APPLICATION\_JSON)

@ApiOperation(httpMethod = "POST", value = "Response a string describing if the user info is successfully created or not.")

public JsonObject create(@RequestBody @Valid ItemDetail detail) {

return new JsonObject(DAO.save(detail));

}

@RequestMapping(value = "/{id}", method = RequestMethod.GET)

@ApiOperation(httpMethod = "GET", value = "Response a string describing if the user info id is successfully get or not.")

public JsonObject get(@PathVariable("id") long id) {

return new JsonObject(this.DAO.findOne(id));

}

@RequestMapping(value = "/{id}", method = RequestMethod.DELETE)

@ApiOperation(httpMethod = "DELETE", value = "Response a string describing if the user info is successfully delete or not.")

public JsonObject delete(@PathVariable("id") long id) {

try {

this.DAO.delete(id);

} catch (Exception e) {

return new JsonObject(e.toString());

}

return new JsonObject(new ResponseEntity<Boolean>(Boolean.TRUE, HttpStatus.OK));

}

@RequestMapping(method = RequestMethod.GET)

@ApiOperation(httpMethod = "GET", value = "Response a list describing all of user info that is successfully get or not.")

public JsonObject list() {

return new JsonObject(this.DAO.findAll());

}

@RequestMapping(value = "/{id}", method = RequestMethod.PUT)//

@ApiOperation(httpMethod = "PUT", value = "Response a string describing if the item detail is successfully patched or not.")

public JsonObject patch(@PathVariable("id") long id,@RequestBody @Valid ItemDetail detail) {

ItemDetail find = this.DAO.findOne(id);

//only for lab update.

find.setLabL(detail.getLabL());

find.setLabA(detail.getLabA());

find.setLabB(detail.getLabB());

return new JsonObject(this.DAO.save(find));

}

}

package info.smartkit.shiny.guide.controllers;

import info.smartkit.shiny.guide.dao.ItemInfoDao;

import info.smartkit.shiny.guide.dao.UserInfoDao;

import info.smartkit.shiny.guide.dto.JsonObject;

import info.smartkit.shiny.guide.vo.ItemInfo;

import info.smartkit.shiny.guide.vo.UserInfo;

import io.swagger.annotations.ApiOperation;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.\*;

import javax.validation.Valid;

import javax.ws.rs.core.MediaType;

/\*\*

\* Created by yangboz on 25/02/2017.

\*/

@RestController

// @see: http://spring.io/guides/gs/reactor-thumbnailer/

@RequestMapping(value = "/item/info")

public class ItemInfoController {

@Autowired

private ItemInfoDao DAO;

/\*\*

\* Posting anew UserInfo

\*

\* @param DAO

\* @return Response a string describing if the user info is successfully created or not.

\*/

@RequestMapping(method = RequestMethod.POST, consumes = MediaType.APPLICATION\_JSON)

@ApiOperation(httpMethod = "POST", value = "Response a string describing if the user info is successfully created or not.")

public JsonObject create(@RequestBody @Valid ItemInfo info) {

return new JsonObject(DAO.save(info));

}

@RequestMapping(value = "/{id}", method = RequestMethod.GET)

@ApiOperation(httpMethod = "GET", value = "Response a string describing if the user info id is successfully get or not.")

public JsonObject get(@PathVariable("id") long id) {

return new JsonObject(this.DAO.findOne(id));

}

@RequestMapping(value = "/{id}", method = RequestMethod.DELETE)

@ApiOperation(httpMethod = "DELETE", value = "Response a string describing if the user info is successfully delete or not.")

public JsonObject delete(@PathVariable("id") long id) {

try {

this.DAO.delete(id);

} catch (Exception e) {

return new JsonObject(e.toString());

}

return new JsonObject(new ResponseEntity<Boolean>(Boolean.TRUE, HttpStatus.OK));

}

@RequestMapping(method = RequestMethod.GET)

@ApiOperation(httpMethod = "GET", value = "Response a list describing all of user info that is successfully get or not.")

public JsonObject list() {

return new JsonObject(this.DAO.findAll());

}

@RequestMapping(value = "/{id}/detail/{value}", method = RequestMethod.PUT)//

@ApiOperation(httpMethod = "PUT", value = "Response a string describing if the item detailID is successfully patched or not.")

public JsonObject patch(@PathVariable("id") long id, @PathVariable("value") int value) {

ItemInfo find = this.DAO.findOne(id);

find.setDetailId(value);

return new JsonObject(this.DAO.save(find));

}

}package info.smartkit.shiny.guide.controllers;

/\*\*

\* Created by yangboz on 23/02/2017.

\*/

import info.smartkit.shiny.guide.dao.MPrescriptionDao;

import info.smartkit.shiny.guide.dto.JsonObject;

import info.smartkit.shiny.guide.vo.MPrescription;

import io.swagger.annotations.ApiOperation;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.\*;

import javax.validation.Valid;

import javax.ws.rs.core.MediaType;

@RestController

// @see: http://spring.io/guides/gs/reactor-thumbnailer/

@RequestMapping(value = "/prescription")

public class MPrescriptionController {

@Autowired

private MPrescriptionDao DAO;

/\*\*

\* Posting anew MPrescription

\*

\* @param MPrescription

\* @return Response a string describing if the user info is successfully created or not.

\*/

@RequestMapping(method = RequestMethod.POST, consumes = MediaType.APPLICATION\_JSON)

@ApiOperation(httpMethod = "POST", value = "Response a string describing if the MPrescription is successfully created or not.")

public JsonObject create(@RequestBody @Valid MPrescription info) {

return new JsonObject(DAO.save(info));

}

@RequestMapping(value = "/{id}", method = RequestMethod.GET)

@ApiOperation(httpMethod = "GET", value = "Response a string describing if the MPrescription id is successfully get or not.")

public JsonObject get(@PathVariable("id") long id) {

return new JsonObject(this.DAO.findOne(id));

}

@RequestMapping(value = "/{id}", method = RequestMethod.DELETE)

@ApiOperation(httpMethod = "DELETE", value = "Response a string describing if the MPrescription is successfully delete or not.")

public JsonObject delete(@PathVariable("id") long id) {

try {

this.DAO.delete(id);

} catch (Exception e) {

return new JsonObject(e.toString());

}

return new JsonObject(new ResponseEntity<Boolean>(Boolean.TRUE, HttpStatus.OK));

}

@RequestMapping(method = RequestMethod.GET)

@ApiOperation(httpMethod = "GET", value = "Response a list describing all of MPrescription that is successfully get or not.")

public JsonObject list() {

// return new JsonObject(this.\_alarmInfoDao.findAll());

return new JsonObject(this.DAO.findAll());

}

// @RequestMapping(value = "/{id}/status/{value}", method = RequestMethod.PUT)//

// @ApiOperation(httpMethod = "PUT", value = "Response a string describing if the alarm status is successfully patched or not.")

// public JsonObject patch(@PathVariable("id") long id, @PathVariable("value") int value) {

// UserInfo find = this.userInfoDao.findOne(id);

// find.setStatus(value);

// return new JsonObject(this.userInfoDao.save(find));

// }

}

package info.smartkit.shiny.guide.controllers;

import com.mysql.jdbc.jdbc2.optional.MysqlDataSource;

import info.smartkit.shiny.guide.dao.UserInfoDao;

import info.smartkit.shiny.guide.dto.JsonObject;

import info.smartkit.shiny.guide.settings.DataBaseSettings;

import info.smartkit.shiny.guide.settings.UploadSettings;

import info.smartkit.shiny.guide.utils.MahoutUtils;

import info.smartkit.shiny.guide.vo.UserInfo;

import io.swagger.annotations.ApiOperation;

import org.apache.logging.log4j.LogManager;

import org.apache.logging.log4j.Logger;

import org.apache.mahout.cf.taste.common.TasteException;

import org.apache.mahout.cf.taste.impl.model.file.FileDataModel;

import org.apache.mahout.cf.taste.impl.model.jdbc.MySQLJDBCDataModel;

import org.apache.mahout.cf.taste.model.DataModel;

import org.apache.mahout.cf.taste.model.JDBCDataModel;

import org.apache.mahout.cf.taste.recommender.RecommendedItem;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.\*;

import javax.validation.Valid;

import javax.ws.rs.core.MediaType;

import java.io.File;

import java.io.IOException;

import java.util.List;

import java.util.Map;

/\*\*

\* Created by smartkit on 27/02/2017.

\*/

@RestController

// @see: http://spring.io/guides/gs/reactor-thumbnailer/

@RequestMapping(value = "/recommend")

public class RecommendController {

@Autowired

private UserInfoDao userInfoDao;

@Autowired

private DataBaseSettings dataBaseSettings;

private static Logger LOG = LogManager.getLogger(RecommendController.class);

@RequestMapping(value = "user/{id}", method = RequestMethod.GET)

@ApiOperation(httpMethod = "GET", value = "Response a recommend list describing if the user/item based recommend is successfully get or not.")

public List<RecommendedItem> getByUser(@PathVariable("id") long id) throws TasteException, IOException {

//

List<RecommendedItem> recommendations = null;

MahoutUtils.userCF(this.getJDBCDataModel());

return recommendations;

}

@RequestMapping(value = "item/{id}", method = RequestMethod.GET)

@ApiOperation(httpMethod = "GET", value = "Response a recommend list describing if the user/item based recommend is successfully get or not.")

public List<RecommendedItem> getByItem(@PathVariable("id") long id) throws IOException, TasteException {

List<RecommendedItem> recommendations = null;

MahoutUtils.itemCF(this.getJDBCDataModel());

return recommendations;

}

private JDBCDataModel getJDBCDataModel(){

//Creating data model

MysqlDataSource dataSource = new MysqlDataSource();

dataSource.setServerName(dataBaseSettings.getServer());

dataSource.setUser(dataBaseSettings.getUsername());

dataSource.setPassword(dataBaseSettings.getPassword());

dataSource.setDatabaseName(dataBaseSettings.getDatabase());

JDBCDataModel datamodel = new MySQLJDBCDataModel(dataSource,

"user\_info", "id", "item\_id", "consult\_id", null);

return datamodel;

}

}package info.smartkit.shiny.guide.controllers;

import java.io.BufferedOutputStream;

import java.io.File;

import java.io.FileOutputStream;

import java.io.IOException;

import java.text.SimpleDateFormat;

import java.util.\*;

import javax.validation.Valid;

import javax.validation.constraints.NotNull;

import javax.ws.rs.core.MediaType;

import com.blogspot.na5cent.exom.ExOM;

import info.smartkit.shiny.guide.dao.ItemDetailDao;

import info.smartkit.shiny.guide.dao.ItemInfoDao;

import info.smartkit.shiny.guide.dao.UserInfoDao;

import info.smartkit.shiny.guide.dao.UserItemDetailDao;

import info.smartkit.shiny.guide.settings.UploadSettings;

import info.smartkit.shiny.guide.utils.MahoutUtils;

import info.smartkit.shiny.guide.vo.ItemDetail;

import info.smartkit.shiny.guide.vo.ItemInfo;

import info.smartkit.shiny.guide.vo.UserInfo;

import info.smartkit.shiny.guide.vo.UserItemDetail;

import io.swagger.annotations.ApiOperation;

import org.apache.commons.io.FilenameUtils;

import org.apache.logging.log4j.LogManager;

import org.apache.logging.log4j.Logger;

import org.apache.mahout.cf.taste.common.TasteException;

import org.apache.mahout.cf.taste.impl.model.file.FileDataModel;

import org.apache.mahout.cf.taste.model.DataModel;

import org.apache.mahout.cf.taste.recommender.RecommendedItem;

import org.hibernate.validator.constraints.NotBlank;

import org.im4java.core.ConvertCmd;

import org.im4java.core.IM4JavaException;

import org.im4java.core.IMOperation;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.context.annotation.Bean;

import org.springframework.web.bind.annotation.\*;

import org.springframework.web.multipart.MultipartFile;

import info.smartkit.shiny.guide.dto.JsonObject;

import info.smartkit.shiny.guide.utils.FileUtil;

/\*\*

\* The Class UploadController.

\*/

@RestController

// @see: http://spring.io/guides/gs/reactor-thumbnailer/

@RequestMapping(value = "/upload")

public class UploadController {

//

private static Logger LOG = LogManager.getLogger(UploadController.class);

@Autowired

private ItemDetailDao itemDetailDao;

@Autowired

private ItemInfoDao itemInfoDao;

@Autowired

private UploadSettings uploadSettings;

@Autowired

private UserItemDetailDao userItemDetailDao;

@Autowired

private UserInfoDao userInfoDao;

//@SEE:http://www.ibm.com/developerworks/cn/java/j-lo-mahout/index.html

// @see: https://spring.io/guides/gs/uploading-files/

@RequestMapping(method = RequestMethod.POST, value = "/recommend", consumes = MediaType.MULTIPART\_FORM\_DATA)

@ApiOperation(value = "Response a list of recommend of TD' picture is successfully uploaded or not.")

// @ApiImplicitParams({@ApiImplicitParam(name="Authorization", value="Authorization DESCRIPTION")})

public

@ResponseBody

List<RecommendedItem> TDFileUpload(

// @RequestParam(value = "name", required = false, defaultValue =

// "default\_input\_image\_file\_name") String name,

// @RequestParam(value = "owner", required = false, defaultValue =

// "default\_intellif\_corp") String owner,

@RequestPart(value = "file") @Valid @NotNull @NotBlank MultipartFile file) throws IOException, TasteException {

// @Validated MultipartFileWrapper file, BindingResult result, Principal

// principal){

String fileName = null;

List<RecommendedItem> recommendations = null;

if (!file.isEmpty()) {

Map<String, String> \_imageMagickOutput = this.fileOperation(file, "tmahout");

// Image resize operation.

fileName = uploadSettings.getTmahout() + \_imageMagickOutput.get(ImageSize.ori.toString());

//Creating data model

DataModel datamodel = new FileDataModel(new File(fileName)); //data

long startTime = System.currentTimeMillis();

MahoutUtils.userCF(datamodel);

long stopTime = System.currentTimeMillis();

System.out.println("Mahout Took: " + (stopTime - startTime) + " millis");

} else {

LOG.error("You failed to upload " + file.getName() + " because the file was empty.");

}

return recommendations;

}

// @see: https://spring.io/guides/gs/uploading-files/

@RequestMapping(method = RequestMethod.POST, value = "/tcsv/{mrow}", consumes = MediaType.MULTIPART\_FORM\_DATA)

@ApiOperation(value = "Response a list of recommend of TD' picture is successfully uploaded or not.")

// @ApiImplicitParams({@ApiImplicitParam(name="Authorization", value="Authorization DESCRIPTION")})

public

@ResponseBody

JsonObject TCsvFileUpload(

// @RequestParam(value = "owner", required = false, defaultValue =

// "default\_intellif\_corp") String owner,

@RequestPart(value = "file") @Valid @NotNull @NotBlank MultipartFile file,

@PathVariable("mrow") Boolean mrow) throws Throwable {

// @Validated MultipartFileWrapper file, BindingResult result, Principal

// principal){

long startTime = System.currentTimeMillis();

long lastID = -1;

String fileName = null;

UserItemDetail lastSingleItem = null;

UserItemDetail lastItem = null;

List<UserItemDetail> lastItems = new ArrayList<>();

if (!file.isEmpty()) {

Map<String, String> \_imageMagickOutput = this.fileOperation(file, "tcsv");

// Image resize operation.

fileName = FileUtil.getUploads("tcsv") + \_imageMagickOutput.get(ImageSize.ori.toString());

List<UserItemDetail> items = ExOM.mapFromExcel(new File(fileName))

.toObjectOf(UserItemDetail.class)

.map();

//

if(mrow) {

for (UserItemDetail item : items) {

//Save to database return last id;

lastItem = userItemDetailDao.save(item);

lastID = lastItem.getId();

LOG.info("Saved ItemDetail.id--> {}", lastID);

lastItems.add(lastItem);

//Save Item

ItemDetail anewItemDetail = new ItemDetail(item.getNjzd(),item.getBlzd(),item.getZzzd(),item.getHoubao(),

item.getFuni(),item.getRunzao(),item.getBotai(),item.getPangshou(),item.getChihen(),item.getDianci(),

item.getLiewen(),item.getYuban(),item.getTaizhi(),item.getShexing(),item.getShese(),item.getTaise(),

item.getJieguo(),item.getRgbR(),item.getRgbG(),item.getRgbB(),item.getHsvH(),

item.getHsvS(),item.getHsvV(),item.getLabL(),item.getLabA(),item.getLabB(),item.getQ1\_2(),item.getQ1\_3r(),

item.getQ1\_3y(),item.getQ2\_1());

ItemDetail anewItemDetailSaved = itemDetailDao.save(anewItemDetail);

LOG.info("Saved anewItemDetail: "+anewItemDetailSaved.toString());

//Save ItemInfo,update item detail, item info, relationship.

ItemInfo anewItemInfo = new ItemInfo(null,null,(int)anewItemDetailSaved.getId());

ItemInfo anewItemInfoSaved = itemInfoDao.save(anewItemInfo);

LOG.info("Saved anewItemInfo: "+anewItemInfoSaved.toString());

//Save UserInfo

UserInfo anewUserInfo = new UserInfo();

anewUserInfo.setName(lastItem.getXingming());

//FIXME:0 value issue.

anewUserInfo.setAge(lastItem.getNianning());

anewUserInfo.setGender(lastItem.getXingbie());

//update item info, user info relationship.

anewUserInfo.setItemId(anewItemInfoSaved.getId());

UserInfo anewUserInfoSaved = userInfoDao.save(anewUserInfo);

LOG.info("Saved anewUserInfo: "+anewUserInfoSaved.toString());

}

}else {

lastSingleItem = items.get(items.size() - 1);

LOG.info("Mapped last ItemDetail-->" + lastSingleItem.toString());

//Save to database return last id;

lastItem = userItemDetailDao.save(lastSingleItem);

lastItems.add(lastItem);

//Update item detail, item info, relationship.

}

LOG.info("Saved lastItems--> " + lastItems.toString());

} else {

LOG.error("You failed to upload " + file.getName() + " because the file was empty.");

}

return new JsonObject(lastItems);

}

// @see: https://spring.io/guides/gs/uploading-files/

@RequestMapping(method = RequestMethod.POST, value = "/timage", consumes = MediaType.MULTIPART\_FORM\_DATA)

@ApiOperation(value = "Response a string describing Tongue' picture is successfully uploaded or not.")

// @ApiImplicitParams({@ApiImplicitParam(name="Authorization", value="Authorization DESCRIPTION")})

public

@ResponseBody

JsonObject TImageFileUpload(

// @RequestParam(value = "name", required = false, defaultValue =

// "default\_input\_image\_file\_name") String name,

// @RequestParam(value = "owner", required = false, defaultValue =

// "default\_intellif\_corp") String owner,

@RequestPart(value = "file") @Valid @NotNull @NotBlank MultipartFile file) {

// @Validated MultipartFileWrapper file, BindingResult result, Principal

// principal){

ItemInfo saved = null;

String fileName = null;

if (!file.isEmpty()) {

// ImageMagick convert options; @see:

// http://paxcel.net/blog/java-thumbnail-generator-imagescalar-vs-imagemagic/

Map<String, String> \_imageMagickOutput = this.fileOperation(file, "timage");

// Save to database.

try {

// Image resize operation.

fileName = \_imageMagickOutput.get(ImageSize.ori.toString());

LOG.info("ImageMagick output success: " + fileName);

String imageUrl = uploadSettings.getTimage() + fileName;

//Save to database as an item info

ItemInfo itemInfo = new ItemInfo(fileName, imageUrl, -1);

saved = itemInfoDao.save(itemInfo);

LOG.info("Saved ItemInfo--> " + saved.toString());

} catch (Exception ex) {

LOG.error(ex.toString());

}

} else {

LOG.error("You failed to upload " + file.getName() + " because the file was empty.");

}

return new JsonObject(saved);

}

//

@SuppressWarnings("unused")

private String thumbnailImage(int width, int height, String source)

throws IOException, InterruptedException, IM4JavaException {

//

String small4dbBase = FilenameUtils.getBaseName(source) + "\_" + String.valueOf(width) + "x"

+ String.valueOf(height) + "." + FilenameUtils.getExtension(source);

String small4db = FileUtil.getUploads("timage") + small4dbBase;

String small = getClassPath() + small4db;

// @see:

// http://paxcel.net/blog/java-thumbnail-generator-imagescalar-vs-imagemagic/

ConvertCmd cmd = new ConvertCmd();

// cmd.setSearchPath("");

File thumbnailFile = new File(small);

if (!thumbnailFile.exists()) {

IMOperation op = new IMOperation();

op.addImage(source);

op.thumbnail(width);

op.addImage(small);

cmd.run(op);

LOG.info("ImageMagick success result:" + small);

}

return small4dbBase;

}

private Map<String, String> fileOperation(MultipartFile file, String context) {

Map<String, String> \_imageMagickOutput = new HashMap<String, String>();

String dbFileName = null;

String fullFileName = null;

try {

byte[] bytes = file.getBytes();

String fileExt = FilenameUtils.getExtension(file.getOriginalFilename());

String fileNameAppendix

// = "temp" + "." + fileExt;

= new SimpleDateFormat("yyyy-MM-dd-HH-mm-ss-SSS").format(new Date()) + "." + fileExt;

dbFileName = FileUtil.getUploads(context) + fileNameAppendix;

fullFileName = dbFileName;

BufferedOutputStream stream = new BufferedOutputStream(new FileOutputStream(new File(fullFileName)));

stream.write(bytes);

stream.close();

// System.out.println("Upload file success." + fullFileName);

LOG.info("Upload file success." + fullFileName);

// ImageMagick convert options; @see:

// http://paxcel.net/blog/java-thumbnail-generator-imagescalar-vs-imagemagic/

\_imageMagickOutput.put(ImageSize.ori.toString(), fileNameAppendix);

// \_imageMagickOutput.put(ImageSize.sml.toString(),

// thumbnailImage(150, 150, fullFileName));

// \_imageMagickOutput.put(ImageSize.ico.toString(),

// thumbnailImage(32, 32, fullFileName));

return \_imageMagickOutput;

} catch (Exception e) {

// System.out.println("You failed to convert " + fullFileName + " =>

// " + e.toString());

LOG.error("You failed to convert " + fullFileName + " => " + e.toString());

}

return \_imageMagickOutput;

}

// @Autowired

// private FolderSetting folderSetting;

public String getClassPath() {

String classPath = this.getClass().getResource("/").getPath();

return classPath;

}

// Enum for image size.

enum ImageSize {

ori, sml, ico

}

}package info.smartkit.shiny.guide.controllers;

import info.smartkit.shiny.guide.dao.UserInfoDao;

import info.smartkit.shiny.guide.dto.JsonObject;

import info.smartkit.shiny.guide.vo.UserInfo;

import io.swagger.annotations.ApiOperation;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.\*;

import javax.validation.Valid;

import javax.ws.rs.core.MediaType;

import java.util.List;

/\*\*

\* Created by smartkit on 20/02/2017.

\*/

@RestController

// @see: http://spring.io/guides/gs/reactor-thumbnailer/

@RequestMapping(value = "/user")

public class UserInfoController {

@Autowired

private UserInfoDao userInfoDao;

/\*\*

\* Posting anew UserInfo

\*

\* @param userInfoDao

\* @return Response a string describing if the user info is successfully created or not.

\*/

@RequestMapping(method = RequestMethod.POST, consumes = MediaType.APPLICATION\_JSON)

@ApiOperation(httpMethod = "POST", value = "Response a string describing if the user info is successfully created or not.")

public JsonObject create(@RequestBody @Valid UserInfo userInfo) {

return new JsonObject(userInfoDao.save(userInfo));

}

@RequestMapping(value = "/{id}", method = RequestMethod.GET)

@ApiOperation(httpMethod = "GET", value = "Response a string describing if the user info id is successfully get or not.")

public JsonObject get(@PathVariable("id") long id) {

return new JsonObject(this.userInfoDao.findOne(id));

}

@RequestMapping(value = "/{id}", method = RequestMethod.DELETE)

@ApiOperation(httpMethod = "DELETE", value = "Response a string describing if the user info is successfully delete or not.")

public JsonObject delete(@PathVariable("id") long id) {

try {

this.userInfoDao.delete(id);

} catch (Exception e) {

return new JsonObject(e.toString());

}

return new JsonObject(new ResponseEntity<Boolean>(Boolean.TRUE, HttpStatus.OK));

}

@RequestMapping(method = RequestMethod.GET)

@ApiOperation(httpMethod = "GET", value = "Response a list describing all of user info by consult ID that is successfully get or not.")

public JsonObject list() {

return new JsonObject(this.userInfoDao.findAll());

}

@RequestMapping(value = "c/{id}",method = RequestMethod.GET)

@ApiOperation(httpMethod = "GET", value = "Response a list describing all of user info that is successfully get or not.")

public JsonObject listByConsultId(@PathVariable("id") long cid) {

return new JsonObject(userInfoDao.findByConsultId(cid));

}

@RequestMapping(value = "/{id}/cid/{value}", method = RequestMethod.PUT)//

@ApiOperation(httpMethod = "PUT", value = "Response a string describing if the user consult ID is successfully patched or not.")

public JsonObject patch(@PathVariable("id") long id, @PathVariable("value") int value) {

UserInfo find = this.userInfoDao.findOne(id);

find.setConsultId(value);

return new JsonObject(this.userInfoDao.save(find));

}

}

package info.smartkit.shiny.guide.dao;

import info.smartkit.shiny.guide.vo.ConsultInfo;

import org.springframework.data.repository.CrudRepository;

/\*\*

\* Created by smartkit on 27/02/2017.

\*/

public interface ConsultInfoDao extends CrudRepository<ConsultInfo,Long>{

}

package info.smartkit.shiny.guide.dao;

import info.smartkit.shiny.guide.vo.EInstruction;

import org.springframework.data.repository.CrudRepository;

/\*\*

\* Created by yangboz on 23/02/2017.

\*/

public interface EInstructionDao extends CrudRepository<EInstruction, Long> {

}

package info.smartkit.shiny.guide.dao;

import info.smartkit.shiny.guide.vo.ItemDetail;

import org.springframework.data.repository.CrudRepository;

/\*\*

\* Created by yangboz on 25/02/2017.

\*/

public interface ItemDetailDao extends CrudRepository<ItemDetail, Long> {

}

package info.smartkit.shiny.guide.dao;

import info.smartkit.shiny.guide.vo.ItemInfo;

import org.springframework.data.repository.CrudRepository;

/\*\*

\* Created by yangboz on 25/02/2017.

\*/

public interface ItemInfoDao extends CrudRepository<ItemInfo, Long> {

}

package info.smartkit.shiny.guide.dao;

import info.smartkit.shiny.guide.vo.MPrescription;

import org.springframework.data.repository.CrudRepository;

/\*\*

\* Created by yangboz on 23/02/2017.

\*/

public interface MPrescriptionDao extends CrudRepository<MPrescription, Long> {

}

package info.smartkit.shiny.guide.dao;

import info.smartkit.shiny.guide.vo.UserInfo;

import org.springframework.data.repository.CrudRepository;

import java.util.List;

/\*\*

\* Created by smartkit on 20/02/2017.

\*/

public interface UserInfoDao extends CrudRepository<UserInfo, Long> {

List<UserInfo> findByConsultId(long cid);

}

package info.smartkit.shiny.guide.dao;

import info.smartkit.shiny.guide.vo.UserItemDetail;

import org.springframework.data.repository.CrudRepository;

/\*\*

\* Created by smartkit on 05/05/2017.

\*/

public interface UserItemDetailDao extends CrudRepository<UserItemDetail,Long> {

}

package info.smartkit.shiny.guide;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class DataServiceApplication {

public static void main(String[] args) {

SpringApplication.run(DataServiceApplication.class, args);

}

}

package info.smartkit.shiny.guide.dto;

/\*\*

\* Created by smartkit on 02/07/2017.

\*/

public class ConsultDecision {

private Boolean doAlert;

public ConsultDecision() {

this.doAlert = false;

}

public Boolean getDoAlert() {

return doAlert;

}

public void setDoAlert(Boolean doAlert) {

this.doAlert = doAlert;

}

}

package info.smartkit.shiny.guide.dto;

import info.smartkit.shiny.guide.vo.EInstruction;

import info.smartkit.shiny.guide.vo.MPrescription;

/\*\*

\* Created by smartkit on 02/03/2017.

\*/

public class ConsultInfoFull {

// Tongue Body 舌质

// Pink 淡红舌

// Slightly red 红舌

// Red 绛舌

// Purple 紫舌

// Pale or white 淡白舌

//

// Tongue's coating 舌苔

// No coating 无苔

// White 白苔

// Dirty white 白腻苔

// Slightly yellow 淡黄苔

// Yellow 黄苔

// Brown or black 灰黑苔

// Thick 厚苔

// Thin 薄苔

// Geographic 地图舌

//

// Cracks 裂纹

// Teethmarks 齿痕

// Spots and dots 点刺

// Veins 舌下络脉（这个我们没有采用）

public long getId() {

return id;

}

public void setId(long id) {

this.id = id;

}

public EInstruction geteInstruction() {

return eInstruction;

}

@Override

public String toString() {

return "ConsultInfoFull{" +

"id=" + id +

", eInstruction=" + eInstruction +

", mPrescription=" + mPrescription +

'}';

}

public ConsultInfoFull() {

}

public ConsultInfoFull(EInstruction eInstruction, MPrescription mPrescription) {

this.eInstruction = eInstruction;

this.mPrescription = mPrescription;

}

public ConsultInfoFull(long id, EInstruction eInstruction, MPrescription mPrescription) {

this.id = id;

this.eInstruction = eInstruction;

this.mPrescription = mPrescription;

}

public void seteInstruction(EInstruction eInstruction) {

this.eInstruction = eInstruction;

}

public MPrescription getmPrescription() {

return mPrescription;

}

public void setmPrescription(MPrescription mPrescription) {

this.mPrescription = mPrescription;

}

private long id;

private EInstruction eInstruction;

private MPrescription mPrescription;

}

package info.smartkit.shiny.guide.dto;

/\*\*

\* The Class JsonObject.

\*/

public class JsonObject {

public JsonObject(Object data) {

this.data = data;

}

// Setter,getters

private Object data;

public Object getData() {

return data;

}

public void setData(Object data) {

this.data = data;

}

}

package info.smartkit.shiny.guide.dto;

public class JsonString {

public JsonString(String body) {

this.body = body;

}

// Setter,getters

private String body;

public String getBody() {

return body;

}

public void setBody(String body) {

this.body = body;

}

}

package info.smartkit.shiny.guide.event;

/\*\*

\* Created by smartkit on 02/07/2017.

\*/

public class ConsultEvent implements Event{

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

private int id;

@Override public String toString() {

return "ConsultEvent{" + "id=" + id + ", taise='" + taise + '\'' + ", shese='" + shese + '\'' + ", runzao='" + runzao + '\'' + ", houbao='" + houbao + '\'' + ", chihen='" + chihen + '\'' + ", yuban='" + yuban + '\'' + ", liewen='" + liewen

+ '\'' + ", pangshou='" + pangshou + '}';

}

private String taise;

public String getShese() {

return shese;

}

public void setShese(String shese) {

this.shese = shese;

}

public String getRunzao() {

return runzao;

}

public void setRunzao(String runzao) {

this.runzao = runzao;

}

public String getChihen() {

return chihen;

}

public void setChihen(String chihen) {

this.chihen = chihen;

}

public String getYuban() {

return yuban;

}

public void setYuban(String yuban) {

this.yuban = yuban;

}

public String getLiewen() {

return liewen;

}

public void setLiewen(String liewen) {

this.liewen = liewen;

}

public String getPangshou() {

return pangshou;

}

public void setPangshou(String pangshou) {

this.pangshou = pangshou;

}

private String shese;

private String runzao;

private String houbao;

private String chihen;

private String yuban;

private String liewen;

private String pangshou;

public String getTaise() {

return taise;

}

public void setTaise(String taise) {

this.taise = taise;

}

public String getHoubao() {

return houbao;

}

public void setHoubao(String houbao) {

this.houbao = houbao;

}

}

package info.smartkit.shiny.guide.event;

/\*\*

\* Created by smartkit on 02/07/2017.

\*/

public interface Event {

}

package info.smartkit.shiny.guide.rule;

import java.util.HashMap;

import java.util.Map;

/\*\*

\* Created by smartkit on 02/07/2017.

\*/

public class Condition {

private String field;

private Object value;

private Condition.Operator operator;

public String getField() {

return field;

}

public void setField(String field) {

this.field = field;

}

public Object getValue() {

return value;

}

public void setValue(Object value) {

this.value = value;

}

public Condition.Operator getOperator() {

return operator;

}

public void setOperator(Condition.Operator operator) {

this.operator = operator;

}

public static enum Operator {

C\_SIMILAR\_TO("COLOR\_SIMILAR\_TO"),

S\_SIMILAR\_TO("STRING\_SIMILAR\_TO"),

NOT\_EQUAL\_TO("NOT\_EQUAL\_TO"),

EQUAL\_TO("EQUAL\_TO"),

GREATER\_THAN("GREATER\_THAN"),

LESS\_THAN("LESS\_THAN"),

GREATER\_THAN\_OR\_EQUAL\_TO("GREATER\_THAN\_OR\_EQUAL\_TO"),

LESS\_THAN\_OR\_EQUAL\_TO("LESS\_THAN\_OR\_EQUAL\_TO");

private final String value;

private static Map<String, Operator> constants = new HashMap<String, Operator>();

static {

for (Condition.Operator c : values()) {

constants.put(c.value, c);

}

}

private Operator(String value) {

this.value = value;

}

@Override

public String toString() {

return this.value;

}

public static Condition.Operator fromValue(String value) {

Condition.Operator constant = constants.get(value);

if (constant == null) {

throw new IllegalArgumentException(value);

} else {

return constant;

}

}

}

}

package info.smartkit.shiny.guide.rule;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

/\*\*

\* Created by smartkit on 02/07/2017.

\*/

public class Rule {

private List<Condition> conditions;

private Rule.eventType eventType;

public List<Condition> getConditions() {

return conditions;

}

public void setConditions(List<Condition> conditions) {

this.conditions = conditions;

}

public Rule.eventType getEventType() {

return eventType;

}

public void setEventType(Rule.eventType eventType) {

this.eventType = eventType;

}

@Override

public String toString(){

StringBuilder statementBuilder = new StringBuilder();

for (Condition condition : getConditions()) {

String operator = null;

switch (condition.getOperator()) {

case EQUAL\_TO:

operator = "==";

break;

case NOT\_EQUAL\_TO:

operator = "!=";

break;

case GREATER\_THAN:

operator = ">";

break;

case LESS\_THAN:

operator = "<";

break;

case GREATER\_THAN\_OR\_EQUAL\_TO:

operator = ">=";

break;

case LESS\_THAN\_OR\_EQUAL\_TO:

operator = "<=";

break;

case S\_SIMILAR\_TO:

operator = "==";//color similar/difference.

break;

case C\_SIMILAR\_TO:

operator = "==";//string fuzzy contain/equal

break;

}

statementBuilder.append(condition.getField()).append(" ").append(operator).append(" ");

if (condition.getValue() instanceof String) {

statementBuilder.append("'").append(condition.getValue()).append("'");

} else {

statementBuilder.append(condition.getValue());

}

statementBuilder.append(" && ");

}

String statement = statementBuilder.toString();

// remove trailing &&

return statement.substring(0, statement.length() - 4);

}

public static enum eventType {

ORDER("ORDER"),

INVOICE("INVOICE");

private final String value;

private static Map<String, Rule.eventType> constants = new HashMap<String, Rule.eventType>();

static {

for (Rule.eventType c: values()) {

constants.put(c.value, c);

}

}

private eventType(String value) {

this.value = value;

}

public static Rule.eventType fromValue(String value) {

Rule.eventType constant = constants.get(value);

if (constant == null) {

throw new IllegalArgumentException(value);

} else {

return constant;

}

}

}

}

package info.smartkit.shiny.guide.service;

import info.smartkit.shiny.guide.dto.ConsultInfoFull;

import info.smartkit.shiny.guide.vo.ConsultEinstrMpers;

/\*\*

\* Created by smartkit on 03/07/2017.

\*/

public interface ConsultInfoService {

ConsultEinstrMpers findConsultEinstrMpers(long cid);

}

package info.smartkit.shiny.guide.service;

import info.smartkit.shiny.guide.dto.ConsultInfoFull;

import info.smartkit.shiny.guide.vo.ConsultEinstrMpers;

import info.smartkit.shiny.guide.vo.UserItemConsultInfoIdDetails;

import org.apache.logging.log4j.LogManager;

import org.apache.logging.log4j.Logger;

import org.springframework.stereotype.Service;

import javax.persistence.EntityManager;

import javax.persistence.PersistenceContext;

import java.util.List;

/\*\*

\* Created by smartkit on 03/07/2017.

\*/

@Service

public class ConsultInfoServiceImpl implements ConsultInfoService {

@PersistenceContext

private EntityManager em;

private static Logger LOG = LogManager.getLogger(ConsultInfoServiceImpl.class);

@Override public ConsultEinstrMpers findConsultEinstrMpers(long cid) {

String nativeQuery = "select consult\_info.id as id,mprescription.id as mpid, einstruction.id as eiid , consult\_info.id as hsvs,consult\_info.id as rgbs,consult\_info.id as labs,"

+ "einstruction.name as einame, einstruction.content as eicontent,mprescription.name as mpname,"

+ "mprescription.content as mpcontent from consult\_info "

+ "left join mprescription on consult\_info.pid = mprescription.id "

+ "left join einstruction on consult\_info.iid = einstruction.id where consult\_info.id =" +cid;

//

ConsultEinstrMpers consultEinstrMpers= (ConsultEinstrMpers)this.em.createNativeQuery(nativeQuery,ConsultEinstrMpers.class).getResultList().get(0);

LOG.info("native query consultEinstrMpers:"+consultEinstrMpers);

return consultEinstrMpers;

}

}

package info.smartkit.shiny.guide.service;

import info.smartkit.shiny.guide.vo.ConsultEinstrMpers;

import info.smartkit.shiny.guide.vo.UserItemConsultInfoIdDetails;

import java.util.List;

/\*\*

\* Created by smartkit on 02/07/2017.

\*/

public interface DiagnosisService {

ConsultEinstrMpers inferByFacts(long uiId,int order) throws Exception;

List<UserItemConsultInfoIdDetails> findUserItemConsultInfoDetail()throws Exception;

}

package info.smartkit.shiny.guide.service;

import info.smartkit.shiny.guide.dao.ConsultInfoDao;

import info.smartkit.shiny.guide.dao.ItemDetailDao;

import info.smartkit.shiny.guide.dao.ItemInfoDao;

import info.smartkit.shiny.guide.dto.ConsultDecision;

import info.smartkit.shiny.guide.utils.ColorUtil;

import info.smartkit.shiny.guide.utils.FuzzyStringUtil;

import info.smartkit.shiny.guide.vo.ConsultEinstrMpers;

import info.smartkit.shiny.guide.vo.UserItemConsultInfoIdDetails;

import info.smartkit.shiny.guide.event.ConsultEvent;

import info.smartkit.shiny.guide.rule.Condition;

import info.smartkit.shiny.guide.rule.Rule;

import info.smartkit.shiny.guide.utils.DroolsUtil;

import info.smartkit.shiny.guide.vo.ItemDetail;

import info.smartkit.shiny.guide.vo.ItemInfo;

import org.apache.commons.imaging.color.ColorCieLab;

import org.apache.commons.imaging.color.ColorHsv;

import org.apache.commons.lang.ArrayUtils;

import org.apache.logging.log4j.LogManager;

import org.apache.logging.log4j.Logger;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import javax.persistence.EntityManager;

import javax.persistence.PersistenceContext;

import java.awt.\*;

import java.util.Arrays;

import java.util.Collections;

import java.util.List;

/\*\*

\* Created by smartkit on 02/07/2017.

\*/

@Service

public class DiagnosisServiceImpl implements DiagnosisService {

@Autowired ItemInfoDao itemInfoDao;

@Autowired ItemDetailDao itemDetailDao;

@Autowired ConsultInfoDao consultInfoDao;

@Autowired ConsultInfoService consultInfoService;

@PersistenceContext

private EntityManager em;

private static Logger LOG = LogManager.getLogger(DiagnosisServiceImpl.class);

// @Query("select user\_info.id,name,age, item\_info.id, consult\_info.id, mprescription.id, einstruction.id "

// + "from user\_info left join item\_info on user\_info.item\_id = item\_info.id "

// + "left join item\_detail on item\_info.detail\_id = item\_detail.id "

// + "left join consult\_info on user\_info.consult\_id = consult\_info.id "

// + "left join mprescription on consult\_info.pid = mprescription.id "

// + "left join einstruction on consult\_info.iid = einstruction.id "

// + "where user\_info.consult\_id <> -1")

@Override public List<UserItemConsultInfoIdDetails> findUserItemConsultInfoDetail(){

String nativeQuery = "select user\_info.id, item\_info.id as iid, "

+ "item\_detail.id as idid, item\_detail.yuban as yuban,item\_detail.shese as shese,item\_detail.runzao as runzao,item\_detail.pangshou as pangshou,"

+ "item\_detail.houbao as houbao,item\_detail.liewen as liewen,item\_detail.taise as taise,item\_detail.chihen as chihen,"

+ "item\_detail.rgbR as rgbR,item\_detail.rgbG as rgbG,item\_detail.rgbB as rgbB,"

+ "item\_detail.hsvH as hsvH,item\_detail.hsvS as hsvS,item\_detail.hsvV as hsvV,"

+ "item\_detail.labL as labL,item\_detail.labA as labA,item\_detail.labB as labB,"

+ "consult\_info.id as cid, mprescription.id as mpid, einstruction.id as eiid "

+ "from user\_info left join item\_info on user\_info.item\_id = item\_info.id "

+ "left join item\_detail on item\_info.detail\_id = item\_detail.id "

+ "left join consult\_info on user\_info.consult\_id = consult\_info.id "

+ "left join mprescription on consult\_info.pid = mprescription.id "

+ "left join einstruction on consult\_info.iid = einstruction.id "

+ "where user\_info.consult\_id <> -1";

//

List<UserItemConsultInfoIdDetails> userItemConsultInfoDetailIds = this.em.createNativeQuery(nativeQuery,UserItemConsultInfoIdDetails.class).getResultList();

LOG.info("userItemConsultInfoDetails:"+userItemConsultInfoDetailIds);

return userItemConsultInfoDetailIds;

}

//

@Override public ConsultEinstrMpers inferByFacts(long uiId,int order) throws Exception {

ConsultEinstrMpers consultEinstrMpers = null;

long inferedConsultId;

//inference item

ItemInfo itemInfo = itemInfoDao.findOne(uiId);

long idid = itemInfo.getDetailId();

LOG.info("user\_item\_detail\_id:" + idid);

ItemDetail itemDetail = itemDetailDao.findOne(idid);

//FOR LOOP by facts/memory works:

List<UserItemConsultInfoIdDetails> userItemConsultInfoDetails = findUserItemConsultInfoDetail();

if(order>0)//currently,reverse

{

Collections.reverse(userItemConsultInfoDetails);

}

for(UserItemConsultInfoIdDetails userItemConsultInfoIdDetails : userItemConsultInfoDetails) {

// Create an event that will be tested against the rule. In reality, the event would be read from some external store.

ConsultEvent consultEvent = new ConsultEvent();

//

String houbaoF = FuzzyStringUtil.getHoubaoF(userItemConsultInfoIdDetails.getHoubao());

consultEvent.setHoubao(houbaoF);

//

String taiseF = FuzzyStringUtil.getTaiseF(userItemConsultInfoIdDetails.getTaise());

consultEvent.setTaise(taiseF);

//

String chihenF = FuzzyStringUtil.getChihenF(userItemConsultInfoIdDetails.getChihen());

consultEvent.setChihen(chihenF);

//

String liewenF = FuzzyStringUtil.getLiewenF(userItemConsultInfoIdDetails.getLiewen());

consultEvent.setLiewen(liewenF);

//

String pangshouF = FuzzyStringUtil.getPangshouF(userItemConsultInfoIdDetails.getPangshou());

consultEvent.setPangshou(pangshouF);

//

String runzaoF = FuzzyStringUtil.getRunzaoF(userItemConsultInfoIdDetails.getRunzao());

consultEvent.setRunzao(runzaoF);

//

String sheseF = FuzzyStringUtil.getSheseF(userItemConsultInfoIdDetails.getShese());

consultEvent.setShese(sheseF);

//

String yubanF = FuzzyStringUtil.getYubanF(userItemConsultInfoIdDetails.getYuban());

consultEvent.setYuban(yubanF);

//Rule

Rule similarVariableValuesFuzzyEqualRule = new Rule();

//Conditions

//RGB operator

Condition similarVariableCondition\_rgb = new Condition();

similarVariableCondition\_rgb.setField("rgb");

// similarVariableCondition\_rgb.setOperator(Condition.Operator.C\_SIMILAR\_TO);

similarVariableCondition\_rgb.setValue(itemDetail.getRgb());

//HSV

Condition similarVariableCondition\_hsv = new Condition();

similarVariableCondition\_hsv.setField("hsv");

// similarVariableCondition\_hsv.setOperator(Condition.Operator.C\_SIMILAR\_TO);

similarVariableCondition\_hsv.setValue(itemDetail.getHsv());

//Lab

Condition similarVariableCondition\_lbl = new Condition();

similarVariableCondition\_lbl.setField("lab");

// similarVariableCondition\_lbl.setOperator(Condition.Operator.C\_SIMILAR\_TO);

similarVariableCondition\_lbl.setValue(itemDetail.getLab());

//shese

Condition similarVariableCondition\_shese = new Condition();

similarVariableCondition\_shese.setField("shese");

similarVariableCondition\_shese.setOperator(Condition.Operator.S\_SIMILAR\_TO);

similarVariableCondition\_shese.setValue(itemDetail.getSheseF());

//taise

Condition similarVariableCondition\_taise = new Condition();

similarVariableCondition\_taise.setField("taise");

similarVariableCondition\_taise.setOperator(Condition.Operator.S\_SIMILAR\_TO);

similarVariableCondition\_taise.setValue(itemDetail.getTaiseF());

//houbao

Condition similarVariableCondition\_houbao = new Condition();

similarVariableCondition\_houbao.setField("houbao");

similarVariableCondition\_houbao.setOperator(Condition.Operator.S\_SIMILAR\_TO);

similarVariableCondition\_houbao.setValue(itemDetail.getHoubaoF());

//runzao

Condition similarVariableCondition\_runzao = new Condition();

similarVariableCondition\_runzao.setField("runzao");

similarVariableCondition\_runzao.setOperator(Condition.Operator.S\_SIMILAR\_TO);

similarVariableCondition\_runzao.setValue(itemDetail.getRunzaoF());

//pangshou

Condition similarVariableCondition\_pangshou = new Condition();

similarVariableCondition\_pangshou.setField("pangshou");

similarVariableCondition\_pangshou.setOperator(Condition.Operator.S\_SIMILAR\_TO);

similarVariableCondition\_pangshou.setValue(itemDetail.getPangshouF());

//chihen

Condition similarVariableCondition\_chihen = new Condition();

similarVariableCondition\_chihen.setField("chihen");

similarVariableCondition\_chihen.setOperator(Condition.Operator.S\_SIMILAR\_TO);

similarVariableCondition\_chihen.setValue(itemDetail.getChihenF());

//liewen

Condition similarVariableCondition\_liewen = new Condition();

similarVariableCondition\_liewen.setField("liewen");

similarVariableCondition\_liewen.setOperator(Condition.Operator.S\_SIMILAR\_TO);

similarVariableCondition\_liewen.setValue(itemDetail.getLiewenF());

//yuban

Condition similarVariableCondition\_yuban = new Condition();

similarVariableCondition\_yuban.setField("yuban");

similarVariableCondition\_yuban.setOperator(Condition.Operator.S\_SIMILAR\_TO);

similarVariableCondition\_yuban.setValue(itemDetail.getYubanF());

// In reality, you would have multiple rules for different types of events.

// The eventType property would be used to find rules relevant to the event

similarVariableValuesFuzzyEqualRule.setEventType(Rule.eventType.ORDER);

similarVariableValuesFuzzyEqualRule.setConditions(

Arrays.asList(similarVariableCondition\_taise, similarVariableCondition\_houbao, similarVariableCondition\_chihen, similarVariableCondition\_liewen, similarVariableCondition\_pangshou, similarVariableCondition\_runzao,

similarVariableCondition\_shese, similarVariableCondition\_yuban));

String drl = DroolsUtil.applyRuleTemplate(consultEvent, similarVariableValuesFuzzyEqualRule);

ConsultDecision consultDecision = DroolsUtil.evaluate(drl, consultEvent);

//

if(consultDecision.getDoAlert()) {

inferedConsultId = userItemConsultInfoIdDetails.getCid();

LOG.info("inference by facts, and consult decision result,id:" +inferedConsultId);

consultEinstrMpers = consultInfoService.findConsultEinstrMpers(inferedConsultId);

//use ColorSimilarity as operator.

Color colorRgb = ColorUtil.getColorRgb(userItemConsultInfoIdDetails.getRgbR(),userItemConsultInfoIdDetails.getRgbG(),userItemConsultInfoIdDetails.getRgbB());

ColorHsv colorHsv = ColorUtil.getColorHsv(userItemConsultInfoIdDetails.getHsvH(),userItemConsultInfoIdDetails.getHsvS(),userItemConsultInfoIdDetails.getHsvV());

ColorCieLab colorLab = ColorUtil.getColorLab(userItemConsultInfoIdDetails.getLabL(),userItemConsultInfoIdDetails.getLabA(),userItemConsultInfoIdDetails.getLabB());

double rgbS = ColorUtil.similarity(colorRgb,itemDetail.getRgb());

double hsvS = ColorUtil.similarity(colorHsv,itemDetail.getHsv());

double labS = ColorUtil.similarity(colorLab,itemDetail.getLab());

LOG.info("rgbS:"+rgbS+",hsvS:"+hsvS+",labS:"+labS);

consultEinstrMpers.setRgbS(rgbS);

consultEinstrMpers.setHsvS(hsvS);

consultEinstrMpers.setLabS(labS);

LOG.info("inference by facts, and consult decision result,detail:" +consultEinstrMpers.toString());

break;

}

}

if(consultEinstrMpers==null){

//TODO:mahout recommend by similarity calculation.

LOG.warn("TODO:mahout recommend by similarity calculation.");

}

//

return consultEinstrMpers;

}

}

package info.smartkit.shiny.guide.settings;

import org.springframework.boot.context.properties.ConfigurationProperties;

import org.springframework.stereotype.Component;

/\*\*

\* Created by smartkit on 27/02/2017.

\*/

@ConfigurationProperties(prefix = "spring.datasource")

@Component

public class DataBaseSettings {

public String getServer() {

return server;

}

public void setServer(String server) {

this.server = server;

}

@Override

public String toString() {

return "DataBaseSettings{" +

"server='" + server + '\'' +

", username='" + username + '\'' +

", password='" + password + '\'' +

", database='" + database + '\'' +

'}';

}

public DataBaseSettings(String server, String username, String password, String database) {

this.server = server;

this.username = username;

this.password = password;

this.database = database;

}

private String server;

private String username;

private String password;

private String database;

public DataBaseSettings() {

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getDatabase() {

return database;

}

public void setDatabase(String database) {

this.database = database;

}

}

package info.smartkit.shiny.guide.settings;

import org.springframework.boot.context.properties.ConfigurationProperties;

import org.springframework.context.annotation.Bean;

import org.springframework.stereotype.Component;

/\*\*

\* Created by yangboz on 25/02/2017.

\*/

@ConfigurationProperties(prefix = "uploads")

@Component

public class UploadSettings {

// uploads.host = http://118.190.3.169/td

// uploads.timage = ${uploads.host}/uploads/timage/

// uploads.tcsv = ${uploads.host}/uploads/tcsv/

private String host;

private String timage;

private String tcsv;

private String tmahout;

public UploadSettings(String host, String timage, String tcsv, String tmahout) {

this.host = host;

this.timage = timage;

this.tcsv = tcsv;

this.tmahout = tmahout;

}

public UploadSettings() {

}

@Override

public String toString() {

return "UploadSettings{" +

"host='" + host + '\'' +

", timage='" + timage + '\'' +

", tcsv='" + tcsv + '\'' +

", tmahout='" + tmahout + '\'' +

'}';

}

public String getTmahout() {

return tmahout;

}

public void setTmahout(String tmahout) {

this.tmahout = tmahout;

}

public String getHost() {

return host;

}

public void setHost(String host) {

this.host = host;

}

public String getTimage() {

return timage;

}

public void setTimage(String timage) {

this.timage = timage;

}

public String getTcsv() {

return tcsv;

}

public void setTcsv(String tcsv) {

this.tcsv = tcsv;

}

}

package info.smartkit.shiny.guide.utils;

import org.apache.commons.imaging.color.ColorCieLab;

import org.apache.commons.imaging.color.ColorHsv;

import org.apache.logging.log4j.LogManager;

import org.apache.logging.log4j.Logger;

import java.awt.\*;

/\*\*

\* Created by smartkit on 02/07/2017.

\* @see: https://stackoverflow.com/questions/9018016/how-to-compare-two-colors-for-similarity-difference

\*/

public class ColorUtil {

private static Logger LOG = LogManager.getLogger(ColorUtil.class);

public static Color getColorRgb(String r,String g,String b){

int rInt = (int) Double.parseDouble(r);

int gInt = (int) Double.parseDouble(g);

int bInt = (int) Double.parseDouble(b);

return new Color(rInt,gInt,bInt);

}

//

public static ColorHsv getColorHsv(String h,String s,String v){

double hDou = Double.parseDouble(h);

double sDou = Double.parseDouble(s);

double vDou = Double.parseDouble(v);

return new ColorHsv(hDou,sDou,vDou);

}

//

public static ColorCieLab getColorLab(String l,String a,String b){

double lDou = Double.parseDouble(l);

double aDou = Double.parseDouble(a);

double bDou = Double.parseDouble(b);

return new ColorCieLab(lDou,aDou,bDou);

}

/\*

\*F.E.

\*

\* Point1 has R1 G1 B1

\* Point2 has R2 G2 B2

\*Distance between colors is

\*

\* d=sqrt((r2-r1)^2+(g2-g1)^2+(b2-b1)^2)

\*Percentage is 1-differ

\*

\* p=d/sqrt((255)^2+(255)^2+(255)^2)

\*/

public static double similarity(Color c1,Color c2){

LOG.info("C1,"+c1.toString()+",C2:"+c2.toString());

//

double r2\_r1 = distance(c2.getRed(),c1.getRed());

double g2\_g1 = distance(c2.getGreen(),c1.getGreen());

double b2\_b1 = distance(c2.getBlue(),c1.getBlue());

double distance = Math.sqrt(r2\_r1+g2\_g1+b2\_b1);

double avgRgb = Math.sqrt(Math.pow(255,2)+Math.pow(255,2)+Math.pow(255,2));

double differ = distance/avgRgb;

LOG.info("differ,"+differ+"distance:"+distance+",avgRgb:"+avgRgb);

return 1.00-differ;

}

/\*

\* F.E.

\* avghue = (color1.hue + color2.hue)/2

\* Distance between colors is

\* distance = abs(color1.hue-avghue)

\* Percentage is 1-differ

\*

\*/

public static double similarity(ColorHsv c1,ColorHsv c2){

LOG.info("Ch1,"+c1.toString()+",Ch2:"+c2.toString());

//

double avgHue = (c1.H + c2.H)/2;

double distance = Math.abs(c1.H-avgHue);

double differ = distance/avgHue;

LOG.info("differ:"+differ+"distance:"+distance+",avgHue:"+avgHue);

return 1.00-differ;

}

/\*

\*@see: http://colormine.org/delta-e-calculator

\* var differences = Distance(a.L, b.L) + Distance(a.A, b.A) + Distance(a.B, b.B);

\* return Math.Sqrt(differences);

\*/

public static double similarity(ColorCieLab c1,ColorCieLab c2){

LOG.info("C1,"+c1.toString()+",C2:"+c2.toString());

//

double l2\_l1 = distance(c2.L,c1.L);

double a2\_a1 = distance(c2.a,c1.a);

double b2\_b1 = distance(c2.b,c1.b);

double differences = l2\_l1 + a2\_a1 + b2\_b1;

double differ = Math.sqrt(differences);

LOG.info("differ,"+differ+"differences:"+differences);

return 1.00-differ;

}

private static double distance(double a, double b)

{

double d = Math.abs(a-b);

return Math.pow(d,2);

}

}

package info.smartkit.shiny.guide.utils;

import info.smartkit.shiny.guide.dto.ConsultDecision;

import info.smartkit.shiny.guide.event.Event;

import info.smartkit.shiny.guide.rule.Rule;

import org.drools.template.ObjectDataCompiler;

import org.kie.api.KieServices;

import org.kie.api.builder.KieFileSystem;

import org.kie.api.runtime.KieContainer;

import org.kie.api.runtime.StatelessKieSession;

import java.util.Arrays;

import java.util.HashMap;

import java.util.Map;

/\*\*

\* Created by smartkit on 02/07/2017.

\* @see http://oncodesign.io/2015/08/10/dynamically-create-rules-using-drools--rule-templates/

\*/

public class DroolsUtil {

static public ConsultDecision evaluate(String drl, Event event) throws Exception {

KieServices kieServices = KieServices.Factory.get();

KieFileSystem kieFileSystem = kieServices.newKieFileSystem();

kieFileSystem.write("src/main/resources/rule.drl", drl);

kieServices.newKieBuilder(kieFileSystem).buildAll();

KieContainer kieContainer = kieServices.newKieContainer(kieServices.getRepository().getDefaultReleaseId());

StatelessKieSession statelessKieSession = kieContainer.getKieBase().newStatelessKieSession();

ConsultDecision consultDecision = new ConsultDecision();

statelessKieSession.getGlobals().set("consultDecision", consultDecision);

statelessKieSession.execute(event);

return consultDecision;

}

static public String applyRuleTemplate(Event event, Rule rule) throws Exception {

Map<String, Object> data = new HashMap<String, Object>();

ObjectDataCompiler objectDataCompiler = new ObjectDataCompiler();

data.put("rule", rule);

data.put("eventType", event.getClass().getName());

return objectDataCompiler.compile(Arrays.asList(data), Thread.currentThread().getContextClassLoader().getResourceAsStream("rule-template.drl"));

}

}

package info.smartkit.shiny.guide.utils;

public class FileUtil {

public static String getWorkingDir() {

String workingDir = System.getProperty("user.dir");

return workingDir;

}

public static String getUploads(String context) {

// if(!new File("/uploads/").exists()) new File("/uploads/").mkdirs();

// return "/uploads/";

return getWorkingDir() + "/target/classes/uploads/" + context;

}

}

package info.smartkit.shiny.guide.utils;

import info.smartkit.shiny.guide.vo.ItemDetail;

/\*\*

\* Created by smartkit on 03/07/2017.

\*/

public class FuzzyStringUtil {

static public String getPangshouF(String pangshou){

if(pangshou==null) return null;

String fuzzyStr = ItemDetail.Epangshou.PANG.value;

if(pangshou.contains(ItemDetail.Epangshou.SHOU.value)){

fuzzyStr = ItemDetail.Epangshou.SHOU.value;

}

if(pangshou.contains(ItemDetail.Epangshou.SHIZHONG.value)){

fuzzyStr = ItemDetail.Epangshou.SHIZHONG.value;

}

return fuzzyStr;

}

static public String getChihenF(String chihen){

if(chihen==null) return null;

String fuzzyStr = chihen.contains(ItemDetail.Echihen.YES.value)? ItemDetail.Echihen.YES.value: ItemDetail.Echihen.NO.value;

return fuzzyStr;

}

static public String getLiewenF(String liewen){

if(liewen==null) return null;

String fuzzyStr = liewen.contains(ItemDetail.Eliewen.YES.value)? ItemDetail.Eliewen.YES.value: ItemDetail.Eliewen.NO.value;

return fuzzyStr;

}

static public String getYubanF(String yuban){

if(yuban==null) return null;

String fuzzyStr = yuban.contains(ItemDetail.Eyuban.YES.value)? ItemDetail.Eyuban.YES.value: ItemDetail.Eyuban.NO.value;

return fuzzyStr;

}

static public String getHoubaoF(String houbao){

if(houbao==null) return null;

String fuzzyStr = houbao.contains(ItemDetail.Ehoubao.HOU.value)? ItemDetail.Ehoubao.HOU.value: ItemDetail.Ehoubao.BAO.value;

return fuzzyStr;

}

static public String getRunzaoF(String runzao){

if(runzao==null) return null;

String fuzzyStr = ItemDetail.Erunzao.RUN.value;

if(runzao.contains(ItemDetail.Erunzao.ZAO.value)){

fuzzyStr = ItemDetail.Erunzao.ZAO.value;

}

if(runzao.contains(ItemDetail.Erunzao.ZHONG.value)){

fuzzyStr = ItemDetail.Erunzao.ZHONG.value;

}

return fuzzyStr;

}

static public String getSheseF(String shese){

if(shese==null) return null;

String fuzzyStr = ItemDetail.Eshese.HONG.value;

if(shese.contains(ItemDetail.Eshese.DANHONG.value)){

fuzzyStr = ItemDetail.Eshese.DANHONG.value;

}

if(shese.contains(ItemDetail.Eshese.DANZI.value)){

fuzzyStr = ItemDetail.Eshese.DANZI.value;

}

if(shese.contains(ItemDetail.Eshese.ANHONG.value)){

fuzzyStr = ItemDetail.Eshese.ANHONG.value;

}

if(shese.contains(ItemDetail.Eshese.JIANG.value)){

fuzzyStr = ItemDetail.Eshese.JIANG.value;

}

if(shese.contains(ItemDetail.Eshese.ZI.value)){

fuzzyStr = ItemDetail.Eshese.ZI.value;

}

if(shese.contains(ItemDetail.Eshese.DAN.value)){

fuzzyStr = ItemDetail.Eshese.DAN.value;

}

return fuzzyStr;

}

static public String getTaiseF(String taise){

if(taise==null) return null;

String fuzzyStr = ItemDetail.Etaise.BAI.value;

if (taise==null) {

taise = "";

}

if(taise.contains(ItemDetail.Etaise.HUANGBAI.value)){

fuzzyStr = ItemDetail.Etaise.HUANGBAI.value;

}

if(taise.contains(ItemDetail.Etaise.HUIHEI.value)){

fuzzyStr = ItemDetail.Etaise.HUANGBAI.value;

}

if(taise.contains(ItemDetail.Etaise.HUANG.value)){

fuzzyStr = ItemDetail.Etaise.HUANG.value;

}

return fuzzyStr;

}

}

package info.smartkit.shiny.guide.utils;

import org.apache.mahout.cf.taste.common.TasteException;

import org.apache.mahout.cf.taste.eval.RecommenderBuilder;

import org.apache.mahout.cf.taste.impl.common.LongPrimitiveIterator;

import org.apache.mahout.cf.taste.impl.recommender.svd.ALSWRFactorizer;

import org.apache.mahout.cf.taste.model.DataModel;

import org.apache.mahout.cf.taste.neighborhood.UserNeighborhood;

import org.apache.mahout.cf.taste.similarity.ItemSimilarity;

import org.apache.mahout.cf.taste.similarity.UserSimilarity;

import java.util.List;

/\*\*

\* Created by yangboz on 23/02/2017.

\*

\* @see http://blog.fens.me/mahout-recommendation-api/

\*/

public class MahoutUtils {

final static int NEIGHBORHOOD\_NUM = 2;

final static int RECOMMENDER\_NUM = 3;

public static void userCF(DataModel dataModel) throws TasteException {

UserSimilarity userSimilarity = RecommendFactory.userSimilarity(RecommendFactory.SIMILARITY.EUCLIDEAN, dataModel);

UserNeighborhood userNeighborhood = RecommendFactory.userNeighborhood(RecommendFactory.NEIGHBORHOOD.NEAREST, userSimilarity, dataModel, NEIGHBORHOOD\_NUM);

RecommenderBuilder recommenderBuilder = RecommendFactory.userRecommender(userSimilarity, userNeighborhood, true);

RecommendFactory.evaluate(RecommendFactory.EVALUATOR.AVERAGE\_ABSOLUTE\_DIFFERENCE, recommenderBuilder, null, dataModel, 0.7);

RecommendFactory.statsEvaluator(recommenderBuilder, null, dataModel, 2);

LongPrimitiveIterator iter = dataModel.getUserIDs();

while (iter.hasNext()) {

long uid = iter.nextLong();

List list = recommenderBuilder.buildRecommender(dataModel).recommend(uid, RECOMMENDER\_NUM);

RecommendFactory.showItems(uid, list, true);

}

}

public static void itemCF(DataModel dataModel) throws TasteException {

ItemSimilarity itemSimilarity = RecommendFactory.itemSimilarity(RecommendFactory.SIMILARITY.EUCLIDEAN, dataModel);

RecommenderBuilder recommenderBuilder = RecommendFactory.itemRecommender(itemSimilarity, true);

RecommendFactory.evaluate(RecommendFactory.EVALUATOR.AVERAGE\_ABSOLUTE\_DIFFERENCE, recommenderBuilder, null, dataModel, 0.7);

RecommendFactory.statsEvaluator(recommenderBuilder, null, dataModel, 2);

LongPrimitiveIterator iter = dataModel.getUserIDs();

while (iter.hasNext()) {

long uid = iter.nextLong();

List list = recommenderBuilder.buildRecommender(dataModel).recommend(uid, RECOMMENDER\_NUM);

RecommendFactory.showItems(uid, list, true);

}

}

// public static void slopeOne(DataModel dataModel) throws TasteException {

// RecommenderBuilder recommenderBuilder = RecommendFactory.slopeOneRecommender();

//

// RecommendFactory.evaluate(RecommendFactory.EVALUATOR.AVERAGE\_ABSOLUTE\_DIFFERENCE, recommenderBuilder, null, dataModel, 0.7);

// RecommendFactory.statsEvaluator(recommenderBuilder, null, dataModel, 2);

//

// LongPrimitiveIterator iter = dataModel.getUserIDs();

// while (iter.hasNext()) {

// long uid = iter.nextLong();

// List list = recommenderBuilder.buildRecommender(dataModel).recommend(uid, RECOMMENDER\_NUM);

// RecommendFactory.showItems(uid, list, true);

// }

// }

// public static void itemKNN(DataModel dataModel) throws TasteException {

// ItemSimilarity itemSimilarity = RecommendFactory.itemSimilarity(RecommendFactory.SIMILARITY.EUCLIDEAN, dataModel);

// RecommenderBuilder recommenderBuilder = RecommendFactory.itemKNNRecommender(itemSimilarity, new NonNegativeQuadraticOptimizer(), 10);

//

// RecommendFactory.evaluate(RecommendFactory.EVALUATOR.AVERAGE\_ABSOLUTE\_DIFFERENCE, recommenderBuilder, null, dataModel, 0.7);

// RecommendFactory.statsEvaluator(recommenderBuilder, null, dataModel, 2);

//

// LongPrimitiveIterator iter = dataModel.getUserIDs();

// while (iter.hasNext()) {

// long uid = iter.nextLong();

// List list = recommenderBuilder.buildRecommender(dataModel).recommend(uid, RECOMMENDER\_NUM);

// RecommendFactory.showItems(uid, list, true);

// }

// }

public static void svd(DataModel dataModel) throws TasteException {

RecommenderBuilder recommenderBuilder = RecommendFactory.svdRecommender(new ALSWRFactorizer(dataModel, 10, 0.05, 10));

RecommendFactory.evaluate(RecommendFactory.EVALUATOR.AVERAGE\_ABSOLUTE\_DIFFERENCE, recommenderBuilder, null, dataModel, 0.7);

RecommendFactory.statsEvaluator(recommenderBuilder, null, dataModel, 2);

LongPrimitiveIterator iter = dataModel.getUserIDs();

while (iter.hasNext()) {

long uid = iter.nextLong();

List list = recommenderBuilder.buildRecommender(dataModel).recommend(uid, RECOMMENDER\_NUM);

RecommendFactory.showItems(uid, list, true);

}

}

// public static void treeCluster(DataModel dataModel) throws TasteException {

// UserSimilarity userSimilarity = RecommendFactory.userSimilarity(RecommendFactory.SIMILARITY.LOGLIKELIHOOD, dataModel);

// ClusterSimilarity clusterSimilarity = RecommendFactory.clusterSimilarity(RecommendFactory.SIMILARITY.FARTHEST\_NEIGHBOR\_CLUSTER, userSimilarity);

// RecommenderBuilder recommenderBuilder = RecommendFactory.treeClusterRecommender(clusterSimilarity, 10);

//

// RecommendFactory.evaluate(RecommendFactory.EVALUATOR.AVERAGE\_ABSOLUTE\_DIFFERENCE, recommenderBuilder, null, dataModel, 0.7);

// RecommendFactory.statsEvaluator(recommenderBuilder, null, dataModel, 2);

//

// LongPrimitiveIterator iter = dataModel.getUserIDs();

// while (iter.hasNext()) {

// long uid = iter.nextLong();

// List list = recommenderBuilder.buildRecommender(dataModel).recommend(uid, RECOMMENDER\_NUM);

// RecommendFactory.showItems(uid, list, true);

// }

// }

}

package info.smartkit.shiny.guide.utils;

import java.io.IOException;

import java.util.List;

import info.smartkit.shiny.guide.utils.RecommendFactory;

import org.apache.mahout.cf.taste.common.TasteException;

import org.apache.mahout.cf.taste.eval.RecommenderBuilder;

//import org.apache.mahout.cf.taste.impl.recommender.ClusterSimilarity;

//import org.apache.mahout.cf.taste.impl.recommender.knn.NonNegativeQuadraticOptimizer;

import org.apache.mahout.cf.taste.impl.recommender.svd.ALSWRFactorizer;

import org.apache.mahout.cf.taste.model.DataModel;

import org.apache.mahout.cf.taste.neighborhood.UserNeighborhood;

import org.apache.mahout.cf.taste.recommender.RecommendedItem;

import org.apache.mahout.cf.taste.similarity.ItemSimilarity;

import org.apache.mahout.cf.taste.similarity.UserSimilarity;

//@SEE https://github.com/bsspirit/maven\_mahout\_template/tree/mahout-0.8/datafile

public class RecommenderEvaluator {

final static int NEIGHBORHOOD\_NUM = 2;

final static int RECOMMENDER\_NUM = 3;

public static void main(String[] args) throws TasteException, IOException {

String file = "datafile/job/pv.csv";

DataModel dataModel = RecommendFactory.buildDataModelNoPref(file);

userLoglikelihood(dataModel);

userCityBlock(dataModel);

userTanimoto(dataModel);

itemLoglikelihood(dataModel);

itemCityBlock(dataModel);

itemTanimoto(dataModel);

// slopeOne(dataModel);

}

public static RecommenderBuilder userLoglikelihood(DataModel dataModel) throws TasteException, IOException {

System.out.println("userLoglikelihood");

UserSimilarity userSimilarity = RecommendFactory.userSimilarity(RecommendFactory.SIMILARITY.LOGLIKELIHOOD, dataModel);

UserNeighborhood userNeighborhood = RecommendFactory.userNeighborhood(RecommendFactory.NEIGHBORHOOD.NEAREST, userSimilarity, dataModel, NEIGHBORHOOD\_NUM);

RecommenderBuilder recommenderBuilder = RecommendFactory.userRecommender(userSimilarity, userNeighborhood, false);

RecommendFactory.evaluate(RecommendFactory.EVALUATOR.AVERAGE\_ABSOLUTE\_DIFFERENCE, recommenderBuilder, null, dataModel, 0.7);

RecommendFactory.statsEvaluator(recommenderBuilder, null, dataModel, 2);

return recommenderBuilder;

}

public static RecommenderBuilder userCityBlock(DataModel dataModel) throws TasteException, IOException {

System.out.println("userCityBlock");

UserSimilarity userSimilarity = RecommendFactory.userSimilarity(RecommendFactory.SIMILARITY.CITYBLOCK, dataModel);

UserNeighborhood userNeighborhood = RecommendFactory.userNeighborhood(RecommendFactory.NEIGHBORHOOD.NEAREST, userSimilarity, dataModel, NEIGHBORHOOD\_NUM);

RecommenderBuilder recommenderBuilder = RecommendFactory.userRecommender(userSimilarity, userNeighborhood, false);

RecommendFactory.evaluate(RecommendFactory.EVALUATOR.AVERAGE\_ABSOLUTE\_DIFFERENCE, recommenderBuilder, null, dataModel, 0.7);

RecommendFactory.statsEvaluator(recommenderBuilder, null, dataModel, 2);

return recommenderBuilder;

}

public static RecommenderBuilder userTanimoto(DataModel dataModel) throws TasteException, IOException {

System.out.println("userTanimoto");

UserSimilarity userSimilarity = RecommendFactory.userSimilarity(RecommendFactory.SIMILARITY.TANIMOTO, dataModel);

UserNeighborhood userNeighborhood = RecommendFactory.userNeighborhood(RecommendFactory.NEIGHBORHOOD.NEAREST, userSimilarity, dataModel, NEIGHBORHOOD\_NUM);

RecommenderBuilder recommenderBuilder = RecommendFactory.userRecommender(userSimilarity, userNeighborhood, false);

RecommendFactory.evaluate(RecommendFactory.EVALUATOR.AVERAGE\_ABSOLUTE\_DIFFERENCE, recommenderBuilder, null, dataModel, 0.7);

RecommendFactory.statsEvaluator(recommenderBuilder, null, dataModel, 2);

return recommenderBuilder;

}

public static RecommenderBuilder itemLoglikelihood(DataModel dataModel) throws TasteException, IOException {

System.out.println("itemLoglikelihood");

ItemSimilarity itemSimilarity = RecommendFactory.itemSimilarity(RecommendFactory.SIMILARITY.LOGLIKELIHOOD, dataModel);

RecommenderBuilder recommenderBuilder = RecommendFactory.itemRecommender(itemSimilarity, false);

RecommendFactory.evaluate(RecommendFactory.EVALUATOR.AVERAGE\_ABSOLUTE\_DIFFERENCE, recommenderBuilder, null, dataModel, 0.7);

RecommendFactory.statsEvaluator(recommenderBuilder, null, dataModel, 2);

return recommenderBuilder;

}

public static RecommenderBuilder itemCityBlock(DataModel dataModel) throws TasteException, IOException {

System.out.println("itemCityBlock");

ItemSimilarity itemSimilarity = RecommendFactory.itemSimilarity(RecommendFactory.SIMILARITY.CITYBLOCK, dataModel);

RecommenderBuilder recommenderBuilder = RecommendFactory.itemRecommender(itemSimilarity, false);

RecommendFactory.evaluate(RecommendFactory.EVALUATOR.AVERAGE\_ABSOLUTE\_DIFFERENCE, recommenderBuilder, null, dataModel, 0.7);

RecommendFactory.statsEvaluator(recommenderBuilder, null, dataModel, 2);

return recommenderBuilder;

}

public static RecommenderBuilder itemTanimoto(DataModel dataModel) throws TasteException, IOException {

System.out.println("itemTanimoto");

ItemSimilarity itemSimilarity = RecommendFactory.itemSimilarity(RecommendFactory.SIMILARITY.TANIMOTO, dataModel);

RecommenderBuilder recommenderBuilder = RecommendFactory.itemRecommender(itemSimilarity, false);

RecommendFactory.evaluate(RecommendFactory.EVALUATOR.AVERAGE\_ABSOLUTE\_DIFFERENCE, recommenderBuilder, null, dataModel, 0.7);

RecommendFactory.statsEvaluator(recommenderBuilder, null, dataModel, 2);

return recommenderBuilder;

}

// public static RecommenderBuilder slopeOne(DataModel dataModel) throws TasteException, IOException {

// System.out.println("slopeOne");

// RecommenderBuilder recommenderBuilder = RecommendFactory.slopeOneRecommender();

//

// RecommendFactory.evaluate(RecommendFactory.EVALUATOR.AVERAGE\_ABSOLUTE\_DIFFERENCE, recommenderBuilder, null, dataModel, 0.7);

// RecommendFactory.statsEvaluator(recommenderBuilder, null, dataModel, 2);

// return recommenderBuilder;

// }

// public static RecommenderBuilder knnLoglikelihood(DataModel dataModel) throws TasteException, IOException {

// System.out.println("knnLoglikelihood");

// ItemSimilarity itemSimilarity = RecommendFactory.itemSimilarity(RecommendFactory.SIMILARITY.LOGLIKELIHOOD, dataModel);

// RecommenderBuilder recommenderBuilder = RecommendFactory.itemKNNRecommender(itemSimilarity, new NonNegativeQuadraticOptimizer(), 10);

//

// RecommendFactory.evaluate(RecommendFactory.EVALUATOR.AVERAGE\_ABSOLUTE\_DIFFERENCE, recommenderBuilder, null, dataModel, 0.7);

// RecommendFactory.statsEvaluator(recommenderBuilder, null, dataModel, 2);

//

// return recommenderBuilder;

// }

//

// public static RecommenderBuilder knnTanimoto(DataModel dataModel) throws TasteException, IOException {

// System.out.println("knnTanimoto");

// ItemSimilarity itemSimilarity = RecommendFactory.itemSimilarity(RecommendFactory.SIMILARITY.TANIMOTO, dataModel);

// RecommenderBuilder recommenderBuilder = RecommendFactory.itemKNNRecommender(itemSimilarity, new NonNegativeQuadraticOptimizer(), 10);

//

// RecommendFactory.evaluate(RecommendFactory.EVALUATOR.AVERAGE\_ABSOLUTE\_DIFFERENCE, recommenderBuilder, null, dataModel, 0.7);

// RecommendFactory.statsEvaluator(recommenderBuilder, null, dataModel, 2);

//

// return recommenderBuilder;

// }

//

// public static RecommenderBuilder knnCityBlock(DataModel dataModel) throws TasteException, IOException {

// System.out.println("knnCityBlock");

// ItemSimilarity itemSimilarity = RecommendFactory.itemSimilarity(RecommendFactory.SIMILARITY.CITYBLOCK, dataModel);

// RecommenderBuilder recommenderBuilder = RecommendFactory.itemKNNRecommender(itemSimilarity, new NonNegativeQuadraticOptimizer(), 10);

//

// RecommendFactory.evaluate(RecommendFactory.EVALUATOR.AVERAGE\_ABSOLUTE\_DIFFERENCE, recommenderBuilder, null, dataModel, 0.7);

// RecommendFactory.statsEvaluator(recommenderBuilder, null, dataModel, 2);

//

// return recommenderBuilder;

// }

public static RecommenderBuilder svd(DataModel dataModel) throws TasteException {

System.out.println("svd");

RecommenderBuilder recommenderBuilder = RecommendFactory.svdRecommender(new ALSWRFactorizer(dataModel, 5, 0.05, 10));

RecommendFactory.evaluate(RecommendFactory.EVALUATOR.AVERAGE\_ABSOLUTE\_DIFFERENCE, recommenderBuilder, null, dataModel, 0.7);

RecommendFactory.statsEvaluator(recommenderBuilder, null, dataModel, 2);

return recommenderBuilder;

}

// public static RecommenderBuilder treeClusterLoglikelihood(DataModel dataModel) throws TasteException {

// System.out.println("treeClusterLoglikelihood");

// UserSimilarity userSimilarity = RecommendFactory.userSimilarity(RecommendFactory.SIMILARITY.LOGLIKELIHOOD, dataModel);

// ClusterSimilarity clusterSimilarity = RecommendFactory.clusterSimilarity(RecommendFactory.SIMILARITY.FARTHEST\_NEIGHBOR\_CLUSTER, userSimilarity);

// RecommenderBuilder recommenderBuilder = RecommendFactory.treeClusterRecommender(clusterSimilarity, 3);

//

// RecommendFactory.evaluate(RecommendFactory.EVALUATOR.AVERAGE\_ABSOLUTE\_DIFFERENCE, recommenderBuilder, null, dataModel, 0.7);

// RecommendFactory.statsEvaluator(recommenderBuilder, null, dataModel, 2);

//

// return recommenderBuilder;

// }

}package info.smartkit.shiny.guide.utils;

import java.io.File;

import java.io.IOException;

import java.util.List;

import org.apache.commons.digester.parser.GenericParser;

import org.apache.mahout.cf.taste.common.TasteException;

import org.apache.mahout.cf.taste.eval.DataModelBuilder;

import org.apache.mahout.cf.taste.eval.IRStatistics;

import org.apache.mahout.cf.taste.eval.RecommenderBuilder;

import org.apache.mahout.cf.taste.eval.RecommenderEvaluator;

import org.apache.mahout.cf.taste.eval.RecommenderIRStatsEvaluator;

import org.apache.mahout.cf.taste.impl.common.FastByIDMap;

import org.apache.mahout.cf.taste.impl.eval.AverageAbsoluteDifferenceRecommenderEvaluator;

import org.apache.mahout.cf.taste.impl.eval.GenericRecommenderIRStatsEvaluator;

import org.apache.mahout.cf.taste.impl.eval.RMSRecommenderEvaluator;

import org.apache.mahout.cf.taste.impl.model.GenericBooleanPrefDataModel;

import org.apache.mahout.cf.taste.impl.model.file.FileDataModel;

import org.apache.mahout.cf.taste.impl.neighborhood.NearestNUserNeighborhood;

import org.apache.mahout.cf.taste.impl.neighborhood.ThresholdUserNeighborhood;

//import org.apache.mahout.cf.taste.impl.recommender.ClusterSimilarity;

//import org.apache.mahout.cf.taste.impl.recommender.FarthestNeighborClusterSimilarity;

import org.apache.mahout.cf.taste.impl.recommender.GenericBooleanPrefItemBasedRecommender;

import org.apache.mahout.cf.taste.impl.recommender.GenericBooleanPrefUserBasedRecommender;

import org.apache.mahout.cf.taste.impl.recommender.GenericItemBasedRecommender;

import org.apache.mahout.cf.taste.impl.recommender.GenericUserBasedRecommender;

//import org.apache.mahout.cf.taste.impl.recommender.NearestNeighborClusterSimilarity;

//import org.apache.mahout.cf.taste.impl.recommender.TreeClusteringRecommender;

//import org.apache.mahout.cf.taste.impl.recommender.knn.KnnItemBasedRecommender;

//import org.apache.mahout.cf.taste.impl.recommender.knn.Optimizer;

//import org.apache.mahout.cf.taste.impl.recommender.slopeone.SlopeOneRecommender;

import org.apache.mahout.cf.taste.impl.recommender.svd.Factorizer;

import org.apache.mahout.cf.taste.impl.recommender.svd.SVDRecommender;

import org.apache.mahout.cf.taste.impl.similarity.CityBlockSimilarity;

import org.apache.mahout.cf.taste.impl.similarity.EuclideanDistanceSimilarity;

import org.apache.mahout.cf.taste.impl.similarity.GenericUserSimilarity;

import org.apache.mahout.cf.taste.impl.similarity.LogLikelihoodSimilarity;

import org.apache.mahout.cf.taste.impl.similarity.PearsonCorrelationSimilarity;

import org.apache.mahout.cf.taste.impl.similarity.SpearmanCorrelationSimilarity;

import org.apache.mahout.cf.taste.impl.similarity.TanimotoCoefficientSimilarity;

import org.apache.mahout.cf.taste.impl.similarity.UncenteredCosineSimilarity;

import org.apache.mahout.cf.taste.model.DataModel;

import org.apache.mahout.cf.taste.model.PreferenceArray;

import org.apache.mahout.cf.taste.neighborhood.UserNeighborhood;

import org.apache.mahout.cf.taste.recommender.RecommendedItem;

import org.apache.mahout.cf.taste.recommender.Recommender;

import org.apache.mahout.cf.taste.similarity.ItemSimilarity;

import org.apache.mahout.cf.taste.similarity.UserSimilarity;

/\*\*

\* @author bsspirit@gmail.com

\* @link http://blog.fens.me/mahout-recommendation-api/

\*/

public final class RecommendFactory {

/\*\*

\* build Data model from file

\*/

public static DataModel buildDataModel(String file) throws TasteException, IOException {

return new FileDataModel(new File(file));

}

public static DataModel buildDataModelNoPref(String file) throws TasteException, IOException {

return new GenericBooleanPrefDataModel(GenericBooleanPrefDataModel.toDataMap(new FileDataModel(new File(file))));

}

public static DataModelBuilder buildDataModelNoPrefBuilder() {

return new DataModelBuilder() {

@Override

public DataModel buildDataModel(FastByIDMap<PreferenceArray> trainingData) {

return new GenericBooleanPrefDataModel(GenericBooleanPrefDataModel.toDataMap(trainingData));

}

};

}

public static UserSimilarity userSimilarity(SIMILARITY type, DataModel m) throws TasteException {

switch (type) {

case PEARSON:

return new PearsonCorrelationSimilarity(m);

case COSINE:

return new UncenteredCosineSimilarity(m);

case TANIMOTO:

return new TanimotoCoefficientSimilarity(m);

case LOGLIKELIHOOD:

return new LogLikelihoodSimilarity(m);

case SPEARMAN:

return new SpearmanCorrelationSimilarity(m);

case CITYBLOCK:

return new CityBlockSimilarity(m);

case EUCLIDEAN:

default:

return new EuclideanDistanceSimilarity(m);

}

}

public static ItemSimilarity itemSimilarity(SIMILARITY type, DataModel m) throws TasteException {

switch (type) {

case PEARSON:

return new PearsonCorrelationSimilarity(m);

case COSINE:

return new UncenteredCosineSimilarity(m);

case TANIMOTO:

return new TanimotoCoefficientSimilarity(m);

case LOGLIKELIHOOD:

return new LogLikelihoodSimilarity(m);

case CITYBLOCK:

return new CityBlockSimilarity(m);

case EUCLIDEAN:

default:

return new EuclideanDistanceSimilarity(m);

}

}

// public static ClusterSimilarity clusterSimilarity(SIMILARITY type, UserSimilarity us) throws TasteException {

// switch (type) {

// case NEAREST\_NEIGHBOR\_CLUSTER:

// return new NearestNeighborClusterSimilarity(us);

// case FARTHEST\_NEIGHBOR\_CLUSTER:

// default:

// return new FarthestNeighborClusterSimilarity(us);

// }

// }

public static UserNeighborhood userNeighborhood(NEIGHBORHOOD type, UserSimilarity s, DataModel m, double num) throws TasteException {

switch (type) {

case NEAREST:

return new NearestNUserNeighborhood((int) num, s, m);

case THRESHOLD:

default:

return new ThresholdUserNeighborhood(num, s, m);

}

}

public static RecommenderBuilder userRecommender(final UserSimilarity us, final UserNeighborhood un, boolean pref) throws TasteException {

return pref ? new RecommenderBuilder() {

@Override

public Recommender buildRecommender(DataModel model) throws TasteException {

return new GenericUserBasedRecommender(model, un, us);

}

} : new RecommenderBuilder() {

@Override

public Recommender buildRecommender(DataModel model) throws TasteException {

return new GenericBooleanPrefUserBasedRecommender(model, un, us);

}

};

}

public static RecommenderBuilder itemRecommender(final ItemSimilarity is, boolean pref) throws TasteException {

return pref ? new RecommenderBuilder() {

@Override

public Recommender buildRecommender(DataModel model) throws TasteException {

return new GenericItemBasedRecommender(model, is);

}

} : new RecommenderBuilder() {

@Override

public Recommender buildRecommender(DataModel model) throws TasteException {

return new GenericBooleanPrefItemBasedRecommender(model, is);

}

};

}

// public static RecommenderBuilder slopeOneRecommender() throws TasteException {

// return new RecommenderBuilder() {

// @Override

// public Recommender buildRecommender(DataModel dataModel) throws TasteException {

// return new SlopeOneRecommender(dataModel);

// }

//

// };

// }

//

// public static RecommenderBuilder itemKNNRecommender(final ItemSimilarity is, final Optimizer op, final int n) throws TasteException {

// return new RecommenderBuilder() {

// @Override

// public Recommender buildRecommender(DataModel dataModel) throws TasteException {

// return new KnnItemBasedRecommender(dataModel, is, op, n);

// }

// };

// }

public static RecommenderBuilder svdRecommender(final Factorizer factorizer) throws TasteException {

return new RecommenderBuilder() {

@Override

public Recommender buildRecommender(DataModel dataModel) throws TasteException {

return new SVDRecommender(dataModel, factorizer);

}

};

}

// public static RecommenderBuilder treeClusterRecommender(final ClusterSimilarity cs, final int n) throws TasteException {

// return new RecommenderBuilder() {

// @Override

// public Recommender buildRecommender(DataModel dataModel) throws TasteException {

// return new TreeClusteringRecommender(dataModel, cs, n);

// }

// };

// }

public static void showItems(long uid, List<RecommendedItem> recommendations, boolean skip) {

if (!skip || recommendations.size() > 0) {

System.out.printf("uid:%s,", uid);

for (RecommendedItem recommendation : recommendations) {

System.out.printf("(%s,%f)", recommendation.getItemID(), recommendation.getValue());

}

System.out.println();

}

}

public static RecommenderEvaluator buildEvaluator(EVALUATOR type) {

switch (type) {

case RMS:

return new RMSRecommenderEvaluator();

case AVERAGE\_ABSOLUTE\_DIFFERENCE:

default:

return new AverageAbsoluteDifferenceRecommenderEvaluator();

}

}

public static void evaluate(EVALUATOR type, RecommenderBuilder rb, DataModelBuilder mb, DataModel dm, double trainPt) throws TasteException {

System.out.printf("%s Evaluater Score:%s\n", type.toString(), buildEvaluator(type).evaluate(rb, mb, dm, trainPt, 1.0));

}

public static void evaluate(RecommenderEvaluator re, RecommenderBuilder rb, DataModelBuilder mb, DataModel dm, double trainPt) throws TasteException {

System.out.printf("Evaluater Score:%s\n", re.evaluate(rb, mb, dm, trainPt, 1.0));

}

/\*\*

\* statsEvaluator

\*/

public static void statsEvaluator(RecommenderBuilder rb, DataModelBuilder mb, DataModel m, int topn) throws TasteException {

RecommenderIRStatsEvaluator evaluator = new GenericRecommenderIRStatsEvaluator();

IRStatistics stats = evaluator.evaluate(rb, mb, m, null, topn, GenericRecommenderIRStatsEvaluator.CHOOSE\_THRESHOLD, 1.0);

// System.out.printf("Recommender IR Evaluator: %s\n", stats);

System.out.printf("Recommender IR Evaluator: [Precision:%s,Recall:%s]\n", stats.getPrecision(), stats.getRecall());

}

/\*\*

\* similarity

\*/

public enum SIMILARITY {

PEARSON, EUCLIDEAN, COSINE, TANIMOTO, LOGLIKELIHOOD, SPEARMAN, CITYBLOCK, FARTHEST\_NEIGHBOR\_CLUSTER, NEAREST\_NEIGHBOR\_CLUSTER

}

/\*\*

\* neighborhood

\*/

public enum NEIGHBORHOOD {

NEAREST, THRESHOLD

}

/\*\*

\* recommendation

\*/

public enum RECOMMENDER {

USER, ITEM

}

/\*\*

\* evaluator

\*/

public enum EVALUATOR {

AVERAGE\_ABSOLUTE\_DIFFERENCE, RMS

}

}package info.smartkit.shiny.guide.utils;

import java.net.InetAddress;

import java.net.UnknownHostException;

/\*\*

\* The Class ServerUtil.

\*/

public class ServerUtil {

public static InetAddress getInetAddress() {

InetAddress inetAddr = null;

try {

inetAddr = InetAddress.getLocalHost();

// hostname = inetAddr.getHostName();

System.out.println("Your current Internet address : " + inetAddr.toString());

} catch (UnknownHostException e) {

e.printStackTrace();

}

return inetAddr;

}

}

package info.smartkit.shiny.guide.utils;

/\* ====================================================================

Licensed to the Apache Software Foundation (ASF) under one or more

contributor license agreements. See the NOTICE file distributed with

this work for additional information regarding copyright ownership.

The ASF licenses this file to You under the Apache License, Version 2.0

(the "License"); you may not use this file except in compliance with

the License. You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

==================================================================== \*/

import java.io.BufferedWriter;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.FileWriter;

import java.io.FilenameFilter;

import java.io.IOException;

import java.util.ArrayList;

import org.apache.poi.openxml4j.exceptions.InvalidFormatException;

import org.apache.poi.ss.usermodel.Cell;

// import org.apache.poi.ss.usermodel.CellType;

import org.apache.poi.ss.usermodel.DataFormatter;

import org.apache.poi.ss.usermodel.FormulaEvaluator;

import org.apache.poi.ss.usermodel.Row;

import org.apache.poi.ss.usermodel.Sheet;

import org.apache.poi.ss.usermodel.Workbook;

import org.apache.poi.ss.usermodel.WorkbookFactory;

/\*\*

\* Demonstrates <em>one</em> way to convert an Excel spreadsheet into a CSV

\* file. This class makes the following assumptions;

\* <list>

\* <li>1. Where the Excel workbook contains more that one worksheet, then a single

\* CSV file will contain the data from all of the worksheets.</li>

\* <li>2. The data matrix contained in the CSV file will be square. This means that

\* the number of fields in each record of the CSV file will match the number

\* of cells in the longest row found in the Excel workbook. Any short records

\* will be 'padded' with empty fields - an empty field is represented in the

\* the CSV file in this way - ,,.</li>

\* <li>3. Empty fields will represent missing cells.</li>

\* <li>4. A record consisting of empty fields will be used to represent an empty row

\* in the Excel workbook.</li>

\* </list>

\* Therefore, if the worksheet looked like this;

\*

\* <pre>

\* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\* | | | | | |

\* | A | B | C | D | E |

\* \_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|

\* | | | | | |

\* 1 | 1 | 2 | 3 | 4 | 5 |

\* \_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|

\* | | | | | |

\* 2 | | | | | |

\* \_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|

\* | | | | | |

\* 3 | | A | | B | |

\* \_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|

\* | | | | | |

\* 4 | | | | | Z |

\* \_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|

\* | | | | | |

\* 5 | 1,400 | | 250 | | |

\* \_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|

\*

\* </pre>

\*

\* Then, the resulting CSV file will contain the following lines (records);

\* <pre>

\* 1,2,3,4,5

\* ,,,,

\* ,A,,B,

\* ,,,,Z

\* "1,400",,250,,

\* </pre><p>

\* Typically, the comma is used to separate each of the fields that, together,

\* constitute a single record or line within the CSV file. This is not however

\* a hard and fast rule and so this class allows the user to determine which

\* character is used as the field separator and assumes the comma if none other

\* is specified.

\* </p><p>

\* If a field contains the separator then it will be escaped. If the file should

\* obey Excel's CSV formatting rules, then the field will be surrounded with

\* speech marks whilst if it should obey UNIX conventions, each occurrence of

\* the separator will be preceded by the backslash character.

\* </p><p>

\* If a field contains an end of line (EOL) character then it too will be

\* escaped. If the file should obey Excel's CSV formatting rules then the field

\* will again be surrounded by speech marks. On the other hand, if the file

\* should follow UNIX conventions then a single backslash will precede the

\* EOL character. There is no single applicable standard for UNIX and some

\* appications replace the CR with \r and the LF with \n but this class will

\* not do so.

\* </p><p>

\* If the field contains double quotes then that character will be escaped. It

\* seems as though UNIX does not define a standard for this whilst Excel does.

\* Should the CSV file have to obey Excel's formmating rules then the speech

\* mark character will be escaped with a second set of speech marks. Finally, an

\* enclosing set of speah marks will also surround the entire field. Thus, if

\* the following line of text appeared in a cell - "Hello" he said - it would

\* look like this when converted into a field within a CSV file - """Hello"" he

\* said".

\* </p><p>

\* Finally, it is worth noting that talk of CSV 'standards' is really slightly

\* missleading as there is no such thing. It may well be that the code in this

\* class has to be modified to produce files to suit a specific application

\* or requirement.

\* </p>

\* @author Mark B

\* @version 1.00 9th April 2010

\* 1.10 13th April 2010 - Added support for processing all Excel

\* workbooks in a folder along with the ability

\* to specify a field separator character.

\* 2.00 14th April 2010 - Added support for embedded characters; the

\* field separator, EOL and double quotes or

\* speech marks. In addition, gave the client

\* the ability to select how these are handled,

\* either obeying Excel's or UNIX formatting

\* conventions.

\*/

public class ToCSV {

private Workbook workbook = null;

private ArrayList<ArrayList<String>> csvData = null;

private int maxRowWidth = 0;

private int formattingConvention = 0;

private DataFormatter formatter = null;

private FormulaEvaluator evaluator = null;

private String separator = null;

private static final String CSV\_FILE\_EXTENSION = ".csv";

private static final String DEFAULT\_SEPARATOR = ",";

/\*\*

\* Identifies that the CSV file should obey Excel's formatting conventions

\* with regard to escaping certain embedded characters - the field separator,

\* speech mark and end of line (EOL) character

\*/

public static final int EXCEL\_STYLE\_ESCAPING = 0;

/\*\*

\* Identifies that the CSV file should obey UNIX formatting conventions

\* with regard to escaping certain embedded characters - the field separator

\* and end of line (EOL) character

\*/

public static final int UNIX\_STYLE\_ESCAPING = 1;

/\*\*

\* Process the contents of a folder, convert the contents of each Excel

\* workbook into CSV format and save the resulting file to the specified

\* folder using the same name as the original workbook with the .xls or

\* .xlsx extension replaced by .csv. This method will ensure that the

\* CSV file created contains the comma field separator and that embedded

\* characters such as the field separator, the EOL and double quotes are

\* escaped in accordance with Excel's convention.

\*

\* @param strSource An instance of the String class that encapsulates the

\* name of and path to either a folder containing those Excel

\* workbook(s) or the name of and path to an individual Excel workbook

\* that is/are to be converted.

\* @param strDestination An instance of the String class encapsulating the

\* name of and path to a folder that will contain the resulting CSV

\* files.

\* @throws java.io.FileNotFoundException Thrown if any file cannot be located

\* on the filesystem during processing.

\* @throws java.io.IOException Thrown if the filesystem encounters any

\* problems during processing.

\* @throws java.lang.IllegalArgumentException Thrown if the values passed

\* to the strSource parameter refers to a file or folder that does not

\* exist or if the value passed to the strDestination paramater refers

\* to a folder that does not exist or simply does not refer to a

\* folder.

\* @throws org.apache.poi.openxml4j.exceptions.InvalidFormatException Thrown

\* if the xml markup encountered whilst parsing a SpreadsheetML

\* file (.xlsx) is invalid.

\*/

public void convertExcelToCSV(String strSource, String strDestination)

throws FileNotFoundException, IOException,

IllegalArgumentException, InvalidFormatException {

// Simply chain the call to the overloaded convertExcelToCSV(String,

// String, String, int) method, pass the default separator and ensure

// that certain embedded characters are escaped in accordance with

// Excel's formatting conventions

this.convertExcelToCSV(strSource, strDestination,

ToCSV.DEFAULT\_SEPARATOR, ToCSV.EXCEL\_STYLE\_ESCAPING);

}

/\*\*

\* Process the contents of a folder, convert the contents of each Excel

\* workbook into CSV format and save the resulting file to the specified

\* folder using the same name as the original workbook with the .xls or

\* .xlsx extension replaced by .csv. This method allows the client to

\* define the field separator but will ensure that embedded characters such

\* as the field separator, the EOL and double quotes are escaped in

\* accordance with Excel's convention.

\*

\* @param strSource An instance of the String class that encapsulates the

\* name of and path to either a folder containing those Excel

\* workbook(s) or the name of and path to an individual Excel workbook

\* that is/are to be converted.

\* @param strDestination An instance of the String class encapsulating the

\* name of and path to a folder that will contain the resulting CSV

\* files.

\* @param separator An instance of the String class that encapsulates the

\* character or characters the client wishes to use as the field

\* separator.

\* @throws java.io.FileNotFoundException Thrown if any file cannot be located

\* on the filesystem during processing.

\* @throws java.io.IOException Thrown if the filesystem encounters any

\* problems during processing.

\* @throws java.lang.IllegalArgumentException Thrown if the values passed

\* to the strSource parameter refers to a file or folder that does not

\* exist or if the value passed to the strDestination paramater refers

\* to a folder that does not exist or simply does not refer to a

\* folder.

\* @throws org.apache.poi.openxml4j.exceptions.InvalidFormatException Thrown

\* if the xml markup encounetered whilst parsing a SpreadsheetML

\* file (.xlsx) is invalid.

\*/

public void convertExcelToCSV(String strSource, String strDestination,

String separator)

throws FileNotFoundException, IOException,

IllegalArgumentException, InvalidFormatException {

// Simply chain the call to the overloaded convertExcelToCSV(String,

// String, String, int) method and ensure that certain embedded

// characters are escaped in accordance with Excel's formatting

// conventions

this.convertExcelToCSV(strSource, strDestination,

separator, ToCSV.EXCEL\_STYLE\_ESCAPING);

}

/\*\*

\* Process the contents of a folder, convert the contents of each Excel

\* workbook into CSV format and save the resulting file to the specified

\* folder using the same name as the original workbook with the .xls or

\* .xlsx extension replaced by .csv

\*

\* @param strSource An instance of the String class that encapsulates the

\* name of and path to either a folder containing those Excel

\* workbook(s) or the name of and path to an individual Excel workbook

\* that is/are to be converted.

\* @param strDestination An instance of the String class encapsulating the name

\* of and path to a folder that will contain the resulting CSV files.

\* @param formattingConvention A primitive int whose value will determine

\* whether certain embedded characters should be escaped in accordance

\* with Excel's or UNIX formatting conventions. Two constants are

\* defined to support this option; ToCSV.EXCEL\_STYLE\_ESCAPING and

\* ToCSV.UNIX\_STYLE\_ESCAPING

\* @param separator An instance of the String class encapsulating the

\* characters or characters that should be used to separate items

\* on a line within the CSV file.

\* @throws java.io.FileNotFoundException Thrown if any file cannot be located

\* on the filesystem during processing.

\* @throws java.io.IOException Thrown if the filesystem encounters any

\* problems during processing.

\* @throws java.lang.IllegalArgumentException Thrown if the values passed

\* to the strSource parameter refers to a file or folder that does not

\* exist, if the value passed to the strDestination paramater refers

\* to a folder that does not exist, if the value passed to the

\* strDestination parameter does not refer to a folder or if the

\* value passed to the formattingConvention parameter is other than

\* one of the values defined by the constants ToCSV.EXCEL\_STYLE\_ESCAPING

\* and ToCSV.UNIX\_STYLE\_ESCAPING.

\* @throws org.apache.poi.openxml4j.exceptions.InvalidFormatException Thrown

\* if the xml markup encounetered whilst parsing a SpreadsheetML

\* file (.xlsx) is invalid.

\*/

public void convertExcelToCSV(String strSource, String strDestination,

String separator, int formattingConvention)

throws FileNotFoundException, IOException,

IllegalArgumentException, InvalidFormatException {

File source = new File(strSource);

File destination = new File(strDestination);

File[] filesList = null;

String destinationFilename = null;

// Check that the source file/folder exists.

if(!source.exists()) {

throw new IllegalArgumentException("The source for the Excel " +

"file(s) cannot be found.");

}

// Ensure thaat the folder the user has chosen to save the CSV files

// away into firstly exists and secondly is a folder rather than, for

// instance, a data file.

if(!destination.exists()) {

throw new IllegalArgumentException("The folder/directory for the " +

"converted CSV file(s) does not exist.");

}

if(!destination.isDirectory()) {

throw new IllegalArgumentException("The destination for the CSV " +

"file(s) is not a directory/folder.");

}

// Ensure the value passed to the formattingConvention parameter is

// within range.

if(formattingConvention != ToCSV.EXCEL\_STYLE\_ESCAPING &&

formattingConvention != ToCSV.UNIX\_STYLE\_ESCAPING) {

throw new IllegalArgumentException("The value passed to the " +

"formattingConvention parameter is out of range.");

}

// Copy the spearator character and formatting convention into local

// variables for use in other methods.

this.separator = separator;

this.formattingConvention = formattingConvention;

// Check to see if the sourceFolder variable holds a reference to

// a file or a folder full of files.

if(source.isDirectory()) {

// Get a list of all of the Excel spreadsheet files (workbooks) in

// the source folder/directory

filesList = source.listFiles(new ExcelFilenameFilter());

}

else {

// Assume that it must be a file handle - although there are other

// options the code should perhaps check - and store the reference

// into the filesList variable.

filesList = new File[]{source};

}

// Step through each of the files in the source folder and for each

// open the workbook, convert it's contents to CSV format and then

// save the resulting file away into the folder specified by the

// contents of the destination variable. Note that the name of the

// csv file will be created by taking the name of the Excel file,

// removing the extension and replacing it with .csv. Note that there

// is one drawback with this approach; if the folder holding the files

// contains two workbooks whose names match but one is a binary file

// (.xls) and the other a SpreadsheetML file (.xlsx), then the names

// for both CSV files will be identical and one CSV file will,

// therefore, over-write the other.

if (filesList != null) {

for(File excelFile : filesList) {

// Open the workbook

this.openWorkbook(excelFile);

// Convert it's contents into a CSV file

this.convertToCSV();

// Build the name of the csv folder from that of the Excel workbook.

// Simply replace the .xls or .xlsx file extension with .csv

destinationFilename = excelFile.getName();

destinationFilename = destinationFilename.substring(

0, destinationFilename.lastIndexOf(".")) +

ToCSV.CSV\_FILE\_EXTENSION;

// Save the CSV file away using the newly constricted file name

// and to the specified directory.

this.saveCSVFile(new File(destination, destinationFilename));

}

}

}

/\*\*

\* Open an Excel workbook ready for conversion.

\*

\* @param file An instance of the File class that encapsulates a handle

\* to a valid Excel workbook. Note that the workbook can be in

\* either binary (.xls) or SpreadsheetML (.xlsx) format.

\* @throws java.io.FileNotFoundException Thrown if the file cannot be located.

\* @throws java.io.IOException Thrown if a problem occurs in the file system.

\* @throws org.apache.poi.openxml4j.exceptions.InvalidFormatException Thrown

\* if invalid xml is found whilst parsing an input SpreadsheetML

\* file.

\*/

private void openWorkbook(File file) throws FileNotFoundException,

IOException, InvalidFormatException {

FileInputStream fis = null;

try {

System.out.println("Opening workbook [" + file.getName() + "]");

fis = new FileInputStream(file);

// Open the workbook and then create the FormulaEvaluator and

// DataFormatter instances that will be needed to, respectively,

// force evaluation of forumlae found in cells and create a

// formatted String encapsulating the cells contents.

this.workbook = WorkbookFactory.create(fis);

this.evaluator = this.workbook.getCreationHelper().createFormulaEvaluator();

this.formatter = new DataFormatter(true);

}

finally {

if(fis != null) {

fis.close();

}

}

}

/\*\*

\* Called to convert the contents of the currently opened workbook into

\* a CSV file.

\*/

private void convertToCSV() {

Sheet sheet = null;

Row row = null;

int lastRowNum = 0;

this.csvData = new ArrayList<ArrayList<String>>();

System.out.println("Converting files contents to CSV format.");

// Discover how many sheets there are in the workbook....

int numSheets = this.workbook.getNumberOfSheets();

// and then iterate through them.

for(int i = 0; i < numSheets; i++) {

// Get a reference to a sheet and check to see if it contains

// any rows.

sheet = this.workbook.getSheetAt(i);

if(sheet.getPhysicalNumberOfRows() > 0) {

// Note down the index number of the bottom-most row and

// then iterate through all of the rows on the sheet starting

// from the very first row - number 1 - even if it is missing.

// Recover a reference to the row and then call another method

// which will strip the data from the cells and build lines

// for inclusion in the resylting CSV file.

lastRowNum = sheet.getLastRowNum();

for(int j = 0; j <= lastRowNum; j++) {

row = sheet.getRow(j);

this.rowToCSV(row);

}

}

}

}

/\*\*

\* Called to actually save the data recovered from the Excel workbook

\* as a CSV file.

\*

\* @param file An instance of the File class that encapsulates a handle

\* referring to the CSV file.

\* @throws java.io.FileNotFoundException Thrown if the file cannot be found.

\* @throws java.io.IOException Thrown to indicate and error occurred in the

\* underylying file system.

\*/

private void saveCSVFile(File file)

throws FileNotFoundException, IOException {

FileWriter fw = null;

BufferedWriter bw = null;

ArrayList<String> line = null;

StringBuffer buffer = null;

String csvLineElement = null;

try {

System.out.println("Saving the CSV file [" + file.getName() + "]");

// Open a writer onto the CSV file.

fw = new FileWriter(file);

bw = new BufferedWriter(fw);

// Step through the elements of the ArrayList that was used to hold

// all of the data recovered from the Excel workbooks' sheets, rows

// and cells.

for(int i = 0; i < this.csvData.size(); i++) {

buffer = new StringBuffer();

// Get an element from the ArrayList that contains the data for

// the workbook. This element will itself be an ArrayList

// containing Strings and each String will hold the data recovered

// from a single cell. The for() loop is used to recover elements

// from this 'row' ArrayList one at a time and to write the Strings

// away to a StringBuffer thus assembling a single line for inclusion

// in the CSV file. If a row was empty or if it was short, then

// the ArrayList that contains it's data will also be shorter than

// some of the others. Therefore, it is necessary to check within

// the for loop to ensure that the ArrayList contains data to be

// processed. If it does, then an element will be recovered and

// appended to the StringBuffer.

line = this.csvData.get(i);

for(int j = 0; j < this.maxRowWidth; j++) {

if(line.size() > j) {

csvLineElement = line.get(j);

if(csvLineElement != null) {

buffer.append(this.escapeEmbeddedCharacters(

csvLineElement));

}

}

if(j < (this.maxRowWidth - 1)) {

buffer.append(this.separator);

}

}

// Once the line is built, write it away to the CSV file.

bw.write(buffer.toString().trim());

// Condition the inclusion of new line characters so as to

// avoid an additional, superfluous, new line at the end of

// the file.

if(i < (this.csvData.size() - 1)) {

bw.newLine();

}

}

}

finally {

if(bw != null) {

bw.flush();

bw.close();

}

}

}

/\*\*

\* Called to convert a row of cells into a line of data that can later be

\* output to the CSV file.

\*

\* @param row An instance of either the HSSFRow or XSSFRow classes that

\* encapsulates information about a row of cells recovered from

\* an Excel workbook.

\*/

private void rowToCSV(Row row) {

Cell cell = null;

int lastCellNum = 0;

ArrayList<String> csvLine = new ArrayList<String>();

// Check to ensure that a row was recovered from the sheet as it is

// possible that one or more rows between other populated rows could be

// missing - blank. If the row does contain cells then...

if(row != null) {

// Get the index for the right most cell on the row and then

// step along the row from left to right recovering the contents

// of each cell, converting that into a formatted String and

// then storing the String into the csvLine ArrayList.

lastCellNum = row.getLastCellNum();

for(int i = 0; i <= lastCellNum; i++) {

cell = row.getCell(i);

if(cell == null) {

csvLine.add("");

}

else {

if(cell.getCellType() != Cell.CELL\_TYPE\_FORMULA) {

csvLine.add(this.formatter.formatCellValue(cell));

}

else {

csvLine.add(this.formatter.formatCellValue(cell, this.evaluator));

}

}

}

// Make a note of the index number of the right most cell. This value

// will later be used to ensure that the matrix of data in the CSV file

// is square.

if(lastCellNum > this.maxRowWidth) {

this.maxRowWidth = lastCellNum;

}

}

this.csvData.add(csvLine);

}

/\*\*

\* Checks to see whether the field - which consists of the formatted

\* contents of an Excel worksheet cell encapsulated within a String - contains

\* any embedded characters that must be escaped. The method is able to

\* comply with either Excel's or UNIX formatting conventions in the

\* following manner;

\*

\* With regard to UNIX conventions, if the field contains any embedded

\* field separator or EOL characters they will each be escaped by prefixing

\* a leading backspace character. These are the only changes that have yet

\* emerged following some research as being required.

\*

\* Excel has other embedded character escaping requirements, some that emerged

\* from empirical testing, other through research. Firstly, with regards to

\* any embedded speech marks ("), each occurrence should be escaped with

\* another speech mark and the whole field then surrounded with speech marks.

\* Thus if a field holds <em>"Hello" he said</em> then it should be modified

\* to appear as <em>"""Hello"" he said"</em>. Furthermore, if the field

\* contains either embedded separator or EOL characters, it should also

\* be surrounded with speech marks. As a result <em>1,400</em> would become

\* <em>"1,400"</em> assuming that the comma is the required field separator.

\* This has one consequence in, if a field contains embedded speech marks

\* and embedded separator characters, checks for both are not required as the

\* additional set of speech marks that should be placed around ay field

\* containing embedded speech marks will also account for the embedded

\* separator.

\*

\* It is worth making one further note with regard to embedded EOL

\* characters. If the data in a worksheet is exported as a CSV file using

\* Excel itself, then the field will be surounded with speech marks. If the

\* resulting CSV file is then re-imports into another worksheet, the EOL

\* character will result in the original simgle field occupying more than

\* one cell. This same 'feature' is replicated in this classes behaviour.

\*

\* @param field An instance of the String class encapsulating the formatted

\* contents of a cell on an Excel worksheet.

\* @return A String that encapsulates the formatted contents of that

\* Excel worksheet cell but with any embedded separator, EOL or

\* speech mark characters correctly escaped.

\*/

private String escapeEmbeddedCharacters(String field) {

StringBuffer buffer = null;

// If the fields contents should be formatted to confrom with Excel's

// convention....

if(this.formattingConvention == ToCSV.EXCEL\_STYLE\_ESCAPING) {

// Firstly, check if there are any speech marks (") in the field;

// each occurrence must be escaped with another set of spech marks

// and then the entire field should be enclosed within another

// set of speech marks. Thus, "Yes" he said would become

// """Yes"" he said"

if(field.contains("\"")) {

buffer = new StringBuffer(field.replaceAll("\"", "\\\"\\\""));

buffer.insert(0, "\"");

buffer.append("\"");

}

else {

// If the field contains either embedded separator or EOL

// characters, then escape the whole field by surrounding it

// with speech marks.

buffer = new StringBuffer(field);

if((buffer.indexOf(this.separator)) > -1 ||

(buffer.indexOf("\n")) > -1) {

buffer.insert(0, "\"");

buffer.append("\"");

}

}

return(buffer.toString().trim());

}

// The only other formatting convention this class obeys is the UNIX one

// where any occurrence of the field separator or EOL character will

// be escaped by preceding it with a backslash.

else {

if(field.contains(this.separator)) {

field = field.replaceAll(this.separator, ("\\\\" + this.separator));

}

if(field.contains("\n")) {

field = field.replaceAll("\n", "\\\\\n");

}

return(field);

}

}

/\*\*

\* The main() method contains code that demonstrates how to use the class.

\*

\* @param args An array containing zero, one or more elements all of type

\* String. Each element will encapsulate an argument specified by the

\* user when running the program from the command prompt.

\*/

public static void main(String[] args) {

// Check the number of arguments passed to the main method. There

// must be two, three or four; the name of and path to either the folder

// containing the Excel files or an individual Excel workbook that is/are

// to be converted, the name of and path to the folder to which the CSV

// files should be written, - optionally - the separator character

// that should be used to separate individual items (fields) on the

// lines (records) of the CSV file and - again optionally - an integer

// that idicates whether the CSV file ought to obey Excel's or UNIX

// convnetions with regard to formatting fields that contain embedded

// separator, Speech mark or EOL character(s).

//

// Note that the names of the CSV files will be derived from those

// of the Excel file(s). Put simply the .xls or .xlsx extension will be

// replaced with .csv. Therefore, if the source folder contains files

// with matching names but different extensions - Test.xls and Test.xlsx

// for example - then the CSV file generated from one will overwrite

// that generated from the other.

ToCSV converter = null;

boolean converted = true;

long startTime = System.currentTimeMillis();

try {

converter = new ToCSV();

if(args.length == 2) {

// Just the Source File/Folder and Destination Folder were

// passed to the main method.

converter.convertExcelToCSV(args[0], args[1]);

}

else if(args.length == 3){

// The Source File/Folder, Destination Folder and Separator

// were passed to the main method.

converter.convertExcelToCSV(args[0], args[1], args[2]);

}

else if(args.length == 4) {

// The Source File/Folder, Destination Folder, Separator and

// Formatting Convnetion were passed to the main method.

converter.convertExcelToCSV(args[0], args[1],

args[2], Integer.parseInt(args[3]));

}

else {

// None or more than four parameters were passed so display

//a Usage message.

System.out.println("Usage: java ToCSV [Source File/Folder] " +

"[Destination Folder] [Separator] [Formatting Convention]\n" +

"\tSource File/Folder\tThis argument should contain the name of and\n" +

"\t\t\t\tpath to either a single Excel workbook or a\n" +

"\t\t\t\tfolder containing one or more Excel workbooks.\n" +

"\tDestination Folder\tThe name of and path to the folder that the\n" +

"\t\t\t\tCSV files should be written out into. The\n" +

"\t\t\t\tfolder must exist before running the ToCSV\n" +

"\t\t\t\tcode as it will not check for or create it.\n" +

"\tSeparator\t\tOptional. The character or characters that\n" +

"\t\t\t\tshould be used to separate fields in the CSV\n" +

"\t\t\t\trecord. If no value is passed then the comma\n" +

"\t\t\t\twill be assumed.\n" +

"\tFormatting Convention\tOptional. This argument can take one of two\n" +

"\t\t\t\tvalues. Passing 0 (zero) will result in a CSV\n" +

"\t\t\t\tfile that obeys Excel's formatting conventions\n" +

"\t\t\t\twhilst passing 1 (one) will result in a file\n" +

"\t\t\t\tthat obeys UNIX formatting conventions. If no\n" +

"\t\t\t\tvalue is passed, then the CSV file produced\n" +

"\t\t\t\twill obey Excel's formatting conventions.");

converted = false;

}

}

// It is not wise to have such a wide catch clause - Exception is very

// close to being at the top of the inheritance hierarchy - though it

// will suffice for this example as it is really not possible to recover

// easilly from an exceptional set of circumstances at this point in the

// program. It should however, ideally be replaced with one or more

// catch clauses optimised to handle more specific problems.

catch(Exception ex) {

System.out.println("Caught an: " + ex.getClass().getName());

System.out.println("Message: " + ex.getMessage());

System.out.println("Stacktrace follows:.....");

ex.printStackTrace(System.out);

converted = false;

}

if (converted) {

System.out.println("Conversion took " +

(int)((System.currentTimeMillis() - startTime)/1000) + " seconds");

}

}

/\*\*

\* An instance of this class can be used to control the files returned

\* be a call to the listFiles() method when made on an instance of the

\* File class and that object refers to a folder/directory

\*/

class ExcelFilenameFilter implements FilenameFilter {

/\*\*

\* Determine those files that will be returned by a call to the

\* listFiles() method. In this case, the name of the file must end with

\* either of the following two extension; '.xls' or '.xlsx'. For the

\* future, it is very possible to parameterise this and allow the

\* containing class to pass, for example, an array of Strings to this

\* class on instantiation. Each element in that array could encapsulate

\* a valid file extension - '.xls', '.xlsx', '.xlt', '.xlst', etc. These

\* could then be used to control which files were returned by the call

\* to the listFiles() method.

\*

\* @param file An instance of the File class that encapsulates a handle

\* referring to the folder/directory that contains the file.

\* @param name An instance of the String class that encapsulates the

\* name of the file.

\* @return A boolean value that indicates whether the file should be

\* included in the array retirned by the call to the listFiles()

\* method. In this case true will be returned if the name of the

\* file ends with either '.xls' or '.xlsx' and false will be

\* returned in all other instances.

\*/

@Override

public boolean accept(File file, String name) {

return(name.endsWith(".xls") || name.endsWith(".xlsx"));

}

}

}

package info.smartkit.shiny.guide.vo;

import javax.persistence.\*;

/\*\*

\* Created by smartkit on 03/07/2017.

\*/

@Entity

public class ConsultEinstrMpers {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private long id;

public ConsultEinstrMpers() {

}

public long getEiid() {

return eiid;

}

public void setEiid(long eiid) {

this.eiid = eiid;

}

public long getMpid() {

return mpid;

}

public void setMpid(long mpid) {

this.mpid = mpid;

}

public String getEiname() {

return einame;

}

public void setEiname(String einame) {

this.einame = einame;

}

public String getEicontent() {

return eicontent;

}

public void setEicontent(String eicontent) {

this.eicontent = eicontent;

}

public String getMpname() {

return mpname;

}

public void setMpname(String mpname) {

this.mpname = mpname;

}

public String getMpcontent() {

return mpcontent;

}

public void setMpcontent(String mpcontent) {

this.mpcontent = mpcontent;

}

private long eiid;

private long mpid;

private String einame;

private String eicontent;

private String mpname;

private String mpcontent;

@Override public String toString() {

return "ConsultEinstrMpers{" + "id=" + id + ", eiid=" + eiid + ", mpid=" + mpid + ", einame='" + einame + '\'' + ", eicontent='" + eicontent + '\'' + ", mpname='" + mpname + '\'' + ", mpcontent='" + mpcontent + '\'' + ", rgbS=" + rgbS

+ ", hsvS=" + hsvS + '}';

}

@Transient

public double getRgbS() {

return rgbS;

}

public void setRgbS(double rgbS) {

this.rgbS = rgbS;

}

@Transient

public double getHsvS() {

return hsvS;

}

public void setHsvS(double hsvS) {

this.hsvS = hsvS;

}

private double rgbS=0.00;//similarity of colorRgb.

private double hsvS=0.00;//similarity of colorHsv.

@Transient

public double getLabS() {

return labS;

}

public void setLabS(double labS) {

this.labS = labS;

}

private double labS=0.00;//similarity of colorLab.

}

package info.smartkit.shiny.guide.vo;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import java.io.Serializable;

/\*\*

\* Created by smartkit on 27/02/2017.

\*/

@Entity

public class ConsultInfo implements Serializable {

private

static final long serialVersionUID = -8473186697076849607L;

@Override

public String toString() {

return "ConsultInfo{" +

"id=" + id +

", iid=" + iid +

", pid=" + pid +

'}';

}

public ConsultInfo() {

}

public ConsultInfo(long iid, long pid) {

this.iid = iid;

this.pid = pid;

}

public long getId() {

return id;

}

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private long id;

public long getIid() {

return iid;

}

public void setIid(long iid) {

this.iid = iid;

}

public long getPid() {

return pid;

}

public void setPid(long pid) {

this.pid = pid;

}

private long iid;//Expert's instruction.

private long pid;//Medical prescription.

}

package info.smartkit.shiny.guide.vo;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import java.io.Serializable;

/\*\*

\* Created by yangboz on 23/02/2017.

\* Expert's instruction.

\*/

@Entity

public class EInstruction implements Serializable {

private

static final long serialVersionUID = -8473186697276849607L;

public long getId() {

return id;

}

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private long id;

private String name;

private String content;

public EInstruction() {

}

public EInstruction(String name, String content) {

this.name = name;

this.content = content;

}

public String getContent() {

return content;

}

public void setContent(String content) {

this.content = content;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

@Override

public String toString() {

return "EInstruction{" +

"id=" + id +

", name='" + name + '\'' +

", content=" + content +

'}';

}

}

package info.smartkit.shiny.guide.vo;

import com.blogspot.na5cent.exom.annotation.Column;

import com.fasterxml.jackson.annotation.JsonInclude;

import com.fasterxml.jackson.databind.annotation.JsonSerialize;

import com.javacodegeeks.drools.model.Product;

import info.smartkit.shiny.guide.utils.ColorUtil;

import info.smartkit.shiny.guide.utils.FuzzyStringUtil;

import org.apache.commons.imaging.color.ColorCieLab;

import org.apache.commons.imaging.color.ColorHsv;

import javax.persistence.\*;

import java.awt.\*;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Created by yangboz on 23/02/2017.

\*/

@Entity

public class ItemDetail {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private long id;

public String getNjzd() {

return njzd;

}

public void setNjzd(String njzd) {

this.njzd = njzd;

}

public String getBlzd() {

return blzd;

}

public void setBlzd(String blzd) {

this.blzd = blzd;

}

public String getZzzd() {

return zzzd;

}

public void setZzzd(String zzzd) {

this.zzzd = zzzd;

}

@Column(name = "内镜诊断") //must same excel header (first row)

private String njzd;

@Column(name = "病理诊断") //must same excel header (first row)

private String blzd;

public ItemDetail(String njzd, String blzd, String zzzd, String houbao, String funi, String runzao, String botai, String pangshou, String chihen, String dianci, String liewen, String yuban, String taizhi, String shexing, String shese, String taise, String jieguo, String rgbR, String rgbG, String rgbB, String hsvH, String hsvS, String hsvV, String labL, String labA, String labB, String q1\_2, String q1\_3r, String q1\_3y, String q2\_1) {

this.njzd = njzd;

this.blzd = blzd;

this.zzzd = zzzd;

this.houbao = houbao;

this.funi = funi;

this.runzao = runzao;

this.botai = botai;

this.pangshou = pangshou;

this.chihen = chihen;

this.dianci = dianci;

this.liewen = liewen;

this.yuban = yuban;

this.taizhi = taizhi;

this.shexing = shexing;

this.shese = shese;

this.taise = taise;

this.jieguo = jieguo;

this.rgbR = rgbR;

this.rgbG = rgbG;

this.rgbB = rgbB;

this.hsvH = hsvH;

this.hsvS = hsvS;

this.hsvV = hsvV;

this.labL = labL;

this.labA = labA;

this.labB = labB;

this.q1\_2 = q1\_2;

this.q1\_3r = q1\_3r;

this.q1\_3y = q1\_3y;

this.q2\_1 = q2\_1;

}

@Column(name = "最终诊断") //must same excel header (first row)

private String zzzd;

@Column(name = "厚薄") //must same excel header (first row)

private String houbao;

@Column(name = "腐腻")

private String funi;

@Column(name = "润燥")

private String runzao;

@Column(name = "剥苔")

private String botai;

@Column(name = "胖瘦")

private String pangshou;

@Column(name = "齿痕")

private String chihen;

@Column(name = "点刺")

private String dianci;

@Column(name = "裂纹")

private String liewen;

@Column(name = "瘀斑")

private String yuban;

@Column(name = "苔质")

private String taizhi;

@Column(name = "舌形")

private String shexing;

@Column(name = "整体舌色")

private String shese;

@Column(name = "整体苔色")

private String taise;

@Column(name = "结果")

private String jieguo;

@Column(name = "RGB\_R")

private String rgbR;

@Column(name = "RGB\_R")

private String rgbG;

@Column(name = "RGB\_B")

private String rgbB;

@Column(name = "HSV\_H")

private String hsvH;

@Column(name = "HSV\_S")

private String hsvS;

@Column(name = "HSV\_V")

private String hsvV;

public String getLabL() {

return labL;

}

public void setLabL(String labL) {

this.labL = labL;

}

public String getLabA() {

return labA;

}

public void setLabA(String labA) {

this.labA = labA;

}

public String getLabB() {

return labB;

}

public void setLabB(String labB) {

this.labB = labB;

}

@Column(name = "Lab\_L")

private String labL;

@Column(name = "Lab\_A")

private String labA;

@Column(name = "Lab\_B")

private String labB;

//

@Column(name = "q1\_2")

private String q1\_2;

@Column(name = "q1\_3r")

private String q1\_3r;

@Column(name = "q1\_3y")

private String q1\_3y;

@Column(name = "q2\_1")

private String q2\_1;

public void setProducts(List<Product> products) {

this.products = products;

}

@JsonInclude(JsonInclude.Include.NON\_NULL)

@Transient

//convert to products

private List<Product> products;

@Override

public String toString() {

return "ItemDetail{" +

"id=" + id +

", njzd='" + njzd + '\'' +

", blzd='" + blzd + '\'' +

", zzzd='" + zzzd + '\'' +

", houbao='" + houbao + '\'' +

", funi='" + funi + '\'' +

", runzao='" + runzao + '\'' +

", botai='" + botai + '\'' +

", pangshou='" + pangshou + '\'' +

", chihen='" + chihen + '\'' +

", dianci='" + dianci + '\'' +

", liewen='" + liewen + '\'' +

", yuban='" + yuban + '\'' +

", taizhi='" + taizhi + '\'' +

", shexing='" + shexing + '\'' +

", shese='" + shese + '\'' +

", taise='" + taise + '\'' +

", jieguo='" + jieguo + '\'' +

", rgbR='" + rgbR + '\'' +

", rgbG='" + rgbG + '\'' +

", rgbB='" + rgbB + '\'' +

", hsvH='" + hsvH + '\'' +

", hsvS='" + hsvS + '\'' +

", hsvV='" + hsvV + '\'' +

", labelL='" + labL + '\'' +

", labelA='" + labA + '\'' +

", labelB='" + labB + '\'' +

", q1\_2='" + q1\_2 + '\'' +

", q1\_3r='" + q1\_3r + '\'' +

", q1\_3y='" + q1\_3y + '\'' +

", q2\_1='" + q2\_1 + '\'' +

", products=" + products +

'}';

}

public ItemDetail() {

}

public long getId() {

return id;

}

public String getQ1\_2() {

return q1\_2;

}

public void setQ1\_2(String q1\_2) {

this.q1\_2 = q1\_2;

}

public String getQ1\_3r() {

return q1\_3r;

}

public void setQ1\_3r(String q1\_3r) {

this.q1\_3r = q1\_3r;

}

public String getQ1\_3y() {

return q1\_3y;

}

public void setQ1\_3y(String q1\_3y) {

this.q1\_3y = q1\_3y;

}

public String getQ2\_1() {

return q2\_1;

}

public void setQ2\_1(String q2\_1) {

this.q2\_1 = q2\_1;

}

public String getHoubao() {

return houbao;

}

public void setHoubao(String houbao) {

this.houbao = houbao;

}

public String getFuni() {

return funi;

}

public void setFuni(String funi) {

this.funi = funi;

}

public String getRunzao() {

return runzao;

}

public void setRunzao(String runzao) {

this.runzao = runzao;

}

public String getBotai() {

return botai;

}

public void setBotai(String botai) {

this.botai = botai;

}

public String getPangshou() {

return pangshou;

}

public void setPangshou(String pangshou) {

this.pangshou = pangshou;

}

public String getChihen() {

return chihen;

}

public void setChihen(String chihen) {

this.chihen = chihen;

}

public String getDianci() {

return dianci;

}

public void setDianci(String dianci) {

this.dianci = dianci;

}

public String getLiewen() {

return liewen;

}

public void setLiewen(String liewen) {

this.liewen = liewen;

}

public String getYuban() {

return yuban;

}

public void setYuban(String yuban) {

this.yuban = yuban;

}

public String getTaizhi() {

return taizhi;

}

public void setTaizhi(String taizhi) {

this.taizhi = taizhi;

}

public String getShexing() {

return shexing;

}

public void setShexing(String shexing) {

this.shexing = shexing;

}

public String getShese() {

return shese;

}

public void setShese(String shese) {

this.shese = shese;

}

public String getTaise() {

return taise;

}

public void setTaise(String taise) {

this.taise = taise;

}

public String getJieguo() {

return jieguo;

}

public void setJieguo(String jieguo) {

this.jieguo = jieguo;

}

public String getRgbR() {

return rgbR;

}

public void setRgbR(String rgbR) {

this.rgbR = rgbR;

}

public String getRgbG() {

return rgbG;

}

public void setRgbG(String rgbG) {

this.rgbG = rgbG;

}

public String getRgbB() {

return rgbB;

}

public void setRgbB(String rgbB) {

this.rgbB = rgbB;

}

public String getHsvH() {

return hsvH;

}

public void setHsvH(String hsvH) {

this.hsvH = hsvH;

}

public String getHsvS() {

return hsvS;

}

public void setHsvS(String hsvS) {

this.hsvS = hsvS;

}

public String getHsvV() {

return hsvV;

}

public void setHsvV(String hsvV) {

this.hsvV = hsvV;

}

public List<Product> getProducts(){

List<Product> products = new ArrayList<Product>();

products.add(new Product(this.getJieguo(),1));

products.add(new Product(this.getNjzd(),35));

products.add(new Product(this.getBlzd(),65));

products.add(new Product(this.getZzzd(),85));

//

return products;

}

public Color getRgb() {

int r=0,g=0,b=0;

if(rgbR!=null){

r = (int)Double.parseDouble(rgbR);

}

if(rgbG!=null) {

g = (int) Double.parseDouble(rgbG);

}

if(rgbB!=null) {

b = (int) Double.parseDouble(rgbB);

}

return new Color(r,g,b);

}

public ColorHsv getHsv() {

double h=0,s=0,v=0;

if(hsvH!=null) {

h = Double.parseDouble(hsvH);

}

if(hsvS!=null) {

s = Double.parseDouble(hsvS);

}

if(hsvV!=null) {

v = Double.parseDouble(hsvV);

}

return new ColorHsv(h,s,v);

}

public ColorCieLab getLab(){

double l=0,a=0,b=0;

if(labA!=null) {

l = Double.parseDouble(labL);

}

if(labA!=null) {

a = Double.parseDouble(labA);

}

if(labB!=null) {

b = Double.parseDouble(labB);

}

return new ColorCieLab(l,a,b);

}

//Fuzzy values.

public String getPangshouF(){

return FuzzyStringUtil.getPangshouF(pangshou);

}

public String getChihenF(){

return FuzzyStringUtil.getChihenF(chihen);

}

public String getLiewenF(){

return FuzzyStringUtil.getLiewenF(liewen);

}

public String getYubanF(){

return FuzzyStringUtil.getYubanF(yuban);

}

public String getHoubaoF(){

return FuzzyStringUtil.getHoubaoF(houbao);

}

public String getRunzaoF(){

return FuzzyStringUtil.getRunzaoF(runzao);

}

public String getSheseF(){

return FuzzyStringUtil.getSheseF(shese);

}

public String getTaiseF(){

return FuzzyStringUtil.getTaiseF(taise);

}

public enum Epangshou {

PANG("胖"),SHOU("瘦"),SHIZHONG("适中");

public String value;

Epangshou(String value) {

this.value = value;

}

}

public enum Echihen {

YES("有"),NO("无");

public String value;

Echihen(String value) {

this.value = value;

}

}

public enum Edianci {

YES("有"),NO("无");

private String value;

Edianci(String value) {

this.value = value;

}

}

public enum Eliewen {

YES("有"),NO("无");

public String value;

Eliewen(String value) {

this.value = value;

}

}

public enum Eyuban {

YES("有"),NO("无");

public String value;

Eyuban(String value) {

this.value = value;

}

}

public enum Ehoubao {

HOU("厚"),BAO("薄");

public String value;

Ehoubao(String value) {

this.value = value;

}

}

public enum Erunzao {

RUN("润"),ZAO("燥"),ZHONG("中");

public String value;

Erunzao(String value) {

this.value = value;

}

}

public enum Eshese {

DAN("淡"),DANHONG("淡红"),HONG("红"),ANHONG("暗红"),JIANG("绛"),DANZI("淡紫"),ZI("紫");

public String value;

Eshese(String value) {

this.value = value;

}

}

public enum Etaise {

BAI("白"),HUANGBAI("灰白"),HUANG("黄"),HUIHEI("灰黑");

public String value;

Etaise(String value) {

this.value = value;

}

}

}

package info.smartkit.shiny.guide.vo;

import javax.persistence.\*;

import java.io.Serializable;

import java.util.Date;

/\*\*

\* Created by yangboz on 23/02/2017.

\*/

@Entity

public class ItemInfo implements Serializable {

public long getId() {

return id;

}

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private long id;

private String name;//"齿痕淡红舌薄白苔"

private String timage;//舌像

private int detailId = -1;

public ItemInfo(String name, String timage, int detailId) {

this.name = name;

this.timage = timage;

this.detailId = detailId;

}

public ItemInfo() {

}

public ItemInfo(String name) {

this.name = name;

}

public String getTimage() {

return timage;

}

public void setTimage(String timage) {

this.timage = timage;

}

public int getDetailId() {

return detailId;

}

public void setDetailId(int detailId) {

this.detailId = detailId;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

@Override

public String toString() {

return "ItemInfo{" +

"id=" + id +

", name='" + name + '\'' +

", timage='" + timage + '\'' +

", detailId=" + detailId +

'}';

}

}

package info.smartkit.shiny.guide.vo;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import java.io.Serializable;

/\*\*

\* Created by yangboz on 23/02/2017.

\* Medical prescription.

\*/

@Entity

public class MPrescription implements Serializable {

private

static final long serialVersionUID = -8473185797276849677L;

public long getId() {

return id;

}

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private long id;

private String name;

private String content;

public MPrescription() {

}

public MPrescription(String name, String content) {

this.name = name;

this.content = content;

}

public String getContent() {

return content;

}

public void setContent(String content) {

this.content = content;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

@Override

public String toString() {

return "MPrescription{" +

"id=" + id +

", name='" + name + '\'' +

", content=" + content +

'}';

}

}

package info.smartkit.shiny.guide.vo;

import java.io.Serializable;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Entity;

import java.io.Serializable;

/\*\*

\* Created by smartkit on 20/02/2017.

\*/

@Entity

public class UserInfo implements Serializable {

private static final long serialVersionUID = -8473185797276849607L;

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private long id;

private String name;

private int gender = 1;//1:man,0/2:women

private int age;

public long getConsultId() {

return consultId;

}

public void setConsultId(long consultId) {

this.consultId = consultId;

}

@Override

public String toString() {

return "UserInfo{" +

"id=" + id +

", name='" + name + '\'' +

", gender=" + gender +

", age=" + age +

", consultId=" + consultId +

", itemId=" + itemId +

'}';

}

public UserInfo(String name, int gender, int age, long consultId, long itemId) {

this.name = name;

this.gender = gender;

this.age = age;

this.consultId = consultId;

this.itemId = itemId;

}

private long consultId = -1;//consulting ID;

public long getItemId() {

return itemId;

}

public void setItemId(long itemId) {

this.itemId = itemId;

}

private long itemId = -1;

public UserInfo() {

}

public long getId() {

return id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getGender() {

return gender;

}

public void setGender(int gender) {

this.gender = gender;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

}

package info.smartkit.shiny.guide.vo;

import org.apache.commons.imaging.color.ColorHsv;

import org.springframework.data.annotation.Transient;

import javax.persistence.\*;

import java.awt.\*;

/\*\*

\* Created by smartkit on 03/07/2017.

\*/

@Entity

public class UserItemConsultInfoIdDetails {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private long id;

public int getIid() {

return iid;

}

public void setIid(int iid) {

this.iid = iid;

}

public int getIdid() {

return idid;

}

public void setIdid(int idid) {

this.idid = idid;

}

public long getCid() {

return cid;

}

public void setCid(long cid) {

this.cid = cid;

}

public int getMpid() {

return mpid;

}

public void setMpid(int mpid) {

this.mpid = mpid;

}

public int getEiid() {

return eiid;

}

public void setEiid(int eiid) {

this.eiid = eiid;

}

private int iid;

private int idid;

private long cid;

private int mpid;

private int eiid;

public String getShese() {

return shese;

}

public void setShese(String shese) {

this.shese = shese;

}

public String getTaise() {

return taise;

}

public void setTaise(String taise) {

this.taise = taise;

}

public String getPangshou() {

return pangshou;

}

public void setPangshou(String pangshou) {

this.pangshou = pangshou;

}

public String getLiewen() {

return liewen;

}

public void setLiewen(String liewen) {

this.liewen = liewen;

}

public String getYuban() {

return yuban;

}

public void setYuban(String yuban) {

this.yuban = yuban;

}

public String getRunzao() {

return runzao;

}

public void setRunzao(String runzao) {

this.runzao = runzao;

}

public String getHoubao() {

return houbao;

}

public void setHoubao(String houbao) {

this.houbao = houbao;

}

//item details.

private String shese;

private String taise;

private String pangshou;

public String getChihen() {

return chihen;

}

public void setChihen(String chihen) {

this.chihen = chihen;

}

private String chihen;

private String liewen;

private String yuban;

private String runzao;

private String houbao;

private String rgbR;

private String rgbG;

private String rgbB;

private String hsvH;

private String hsvS;

public String getRgbR() {

return rgbR;

}

public void setRgbR(String rgbR) {

this.rgbR = rgbR;

}

public String getRgbG() {

return rgbG;

}

public void setRgbG(String rgbG) {

this.rgbG = rgbG;

}

public String getRgbB() {

return rgbB;

}

public void setRgbB(String rgbB) {

this.rgbB = rgbB;

}

public String getHsvH() {

return hsvH;

}

public void setHsvH(String hsvH) {

this.hsvH = hsvH;

}

public String getHsvS() {

return hsvS;

}

public void setHsvS(String hsvS) {

this.hsvS = hsvS;

}

public String getHsvV() {

return hsvV;

}

public void setHsvV(String hsvV) {

this.hsvV = hsvV;

}

private String hsvV;

@Override public String toString() {

return "UserItemConsultInfoIdDetails{" + "id=" + id + ", iid=" + iid + ", idid=" + idid + ", cid=" + cid + ", mpid=" + mpid + ", eiid=" + eiid + ", shese='" + shese + '\'' + ", taise='" + taise + '\'' + ", pangshou='" + pangshou + '\''

+ ", chihen='" + chihen + '\'' + ", liewen='" + liewen + '\'' + ", yuban='" + yuban + '\'' + ", runzao='" + runzao + '\'' + ", houbao='" + houbao + '\'' + ", rgbR='" + rgbR + '\'' + ", rgbG='" + rgbG + '\'' + ", rgbB='" + rgbB + '\''

+ ", hsvH='" + hsvH + '\'' + ", hsvS='" + hsvS + '\'' + ", hsvV='" + hsvV + '\'' + ", labL='" + labL + '\'' + ", labA='" + labA + '\'' + ", labB='" + labB + '\'' + '}';

}

public String getLabL() {

return labL;

}

public void setLabL(String labL) {

this.labL = labL;

}

public String getLabA() {

return labA;

}

public void setLabA(String labA) {

this.labA = labA;

}

public String getLabB() {

return labB;

}

public void setLabB(String labB) {

this.labB = labB;

}

private String labL;

private String labA;

private String labB;

}

package info.smartkit.shiny.guide.vo;

import com.blogspot.na5cent.exom.annotation.Column;

import com.fasterxml.jackson.annotation.JsonInclude;

import com.javacodegeeks.drools.model.Product;

import javax.persistence.\*;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Created by yangboz on 05/05/2017.

\*/

@Entity

public class UserItemDetail {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private long id;

//User part

@Column(name = "姓名") //must same excel header (first row)

private String xingming;

@Column(name = "性别") //must same excel header (first row)0,1,2

private int xingbie;

@Column(name = "出生日期") //must same excel header (first row)

private String chushengriqi;

@Column(name = "年龄") //must same excel header (first row)

private int nianning;

@Column(name = "年龄阶段") //must same excel header (first row)

private int nianningjieduan;

//Item part

public String getNjzd() {

return njzd;

}

public void setNjzd(String njzd) {

this.njzd = njzd;

}

public String getBlzd() {

return blzd;

}

public void setBlzd(String blzd) {

this.blzd = blzd;

}

public String getZzzd() {

return zzzd;

}

public void setZzzd(String zzzd) {

this.zzzd = zzzd;

}

@Column(name = "内镜诊断") //must same excel header (first row)

private String njzd;

@Column(name = "病理诊断") //must same excel header (first row)

private String blzd;

public String getXingming() {

return xingming;

}

public void setXingming(String xingming) {

this.xingming = xingming;

}

public int getXingbie() {

return xingbie%2;//0/2

}

public void setXingbie(int xingbie) {

this.xingbie = xingbie;

}

public String getChushengriqi() {

return chushengriqi;

}

public void setChushengriqi(String chushengriqi) {

this.chushengriqi = chushengriqi;

}

public int getNianning() {

return nianning;

}

public void setNianning(int nianning) {

this.nianning = nianning;

}

public int getNianningjieduan() {

return nianningjieduan;

}

public void setNianningjieduan(int nianningjieduan) {

this.nianningjieduan = nianningjieduan;

}

public UserItemDetail() {

}

public UserItemDetail(String xingming, int xingbie, String chushengriqi, int nianning, int nianningjieduan, String njzd, String blzd, String zzzd, String houbao, String funi, String runzao, String botai, String pangshou, String chihen, String dianci, String liewen, String yuban, String taizhi, String shexing, String shese, String taise, String jieguo, String rgbR, String rgbG, String rgbB, String hsvH, String hsvS, String hsvV, String labL, String labA, String labB, String q1\_2, String q1\_3r, String q1\_3y, String q2\_1) {

this.xingming = xingming;

this.xingbie = xingbie;

this.chushengriqi = chushengriqi;

this.nianning = nianning;

this.nianningjieduan = nianningjieduan;

this.njzd = njzd;

this.blzd = blzd;

this.zzzd = zzzd;

this.houbao = houbao;

this.funi = funi;

this.runzao = runzao;

this.botai = botai;

this.pangshou = pangshou;

this.chihen = chihen;

this.dianci = dianci;

this.liewen = liewen;

this.yuban = yuban;

this.taizhi = taizhi;

this.shexing = shexing;

this.shese = shese;

this.taise = taise;

this.jieguo = jieguo;

this.rgbR = rgbR;

this.rgbG = rgbG;

this.rgbB = rgbB;

this.hsvH = hsvH;

this.hsvS = hsvS;

this.hsvV = hsvV;

this.labL = labL;

this.labA = labA;

this.labB = labB;

this.q1\_2 = q1\_2;

this.q1\_3r = q1\_3r;

this.q1\_3y = q1\_3y;

this.q2\_1 = q2\_1;

}

@Column(name = "最终诊断") //must same excel header (first row)

private String zzzd;

@Column(name = "厚薄") //must same excel header (first row)

private String houbao;

@Column(name = "腐腻")

private String funi;

@Column(name = "润燥")

private String runzao;

@Column(name = "剥苔")

private String botai;

@Column(name = "胖瘦")

private String pangshou;

@Column(name = "齿痕")

private String chihen;

@Column(name = "点刺")

private String dianci;

@Column(name = "裂纹")

private String liewen;

@Column(name = "瘀斑")

private String yuban;

@Column(name = "苔质")

private String taizhi;

@Column(name = "舌形")

private String shexing;

@Column(name = "整体舌色")

private String shese;

@Column(name = "整体苔色")

private String taise;

@Column(name = "结果")

private String jieguo;

@Column(name = "RGB\_R")

private String rgbR;

@Column(name = "RGB\_R")

private String rgbG;

@Column(name = "RGB\_B")

private String rgbB;

@Column(name = "HSV\_H")

private String hsvH;

@Column(name = "HSV\_S")

private String hsvS;

@Column(name = "HSV\_V")

private String hsvV;

public String getLabL() {

return labL;

}

public void setLabL(String labL) {

this.labL = labL;

}

public String getLabA() {

return labA;

}

public void setLabA(String labA) {

this.labA = labA;

}

public String getLabB() {

return labB;

}

public void setLabB(String labB) {

this.labB = labB;

}

@Column(name = "Lab\_L")

private String labL;

@Column(name = "Lab\_A")

private String labA;

@Column(name = "Lab\_B")

private String labB;

//

@Column(name = "q1\_2")

private String q1\_2;

@Column(name = "q1\_3r")

private String q1\_3r;

@Column(name = "q1\_3y")

private String q1\_3y;

@Column(name = "q2\_1")

private String q2\_1;

public void setProducts(List<Product> products) {

this.products = products;

}

@JsonInclude(JsonInclude.Include.NON\_NULL)

@Transient

//convert to products

private List<Product> products;

public long getId() {

return id;

}

public String getQ1\_2() {

return q1\_2;

}

public void setQ1\_2(String q1\_2) {

this.q1\_2 = q1\_2;

}

public String getQ1\_3r() {

return q1\_3r;

}

public void setQ1\_3r(String q1\_3r) {

this.q1\_3r = q1\_3r;

}

public String getQ1\_3y() {

return q1\_3y;

}

public void setQ1\_3y(String q1\_3y) {

this.q1\_3y = q1\_3y;

}

public String getQ2\_1() {

return q2\_1;

}

public void setQ2\_1(String q2\_1) {

this.q2\_1 = q2\_1;

}

public String getHoubao() {

return houbao;

}

public void setHoubao(String houbao) {

this.houbao = houbao;

}

public String getFuni() {

return funi;

}

public void setFuni(String funi) {

this.funi = funi;

}

public String getRunzao() {

return runzao;

}

public void setRunzao(String runzao) {

this.runzao = runzao;

}

public String getBotai() {

return botai;

}

public void setBotai(String botai) {

this.botai = botai;

}

public String getPangshou() {

return pangshou;

}

public void setPangshou(String pangshou) {

this.pangshou = pangshou;

}

public String getChihen() {

return chihen;

}

public void setChihen(String chihen) {

this.chihen = chihen;

}

public String getDianci() {

return dianci;

}

public void setDianci(String dianci) {

this.dianci = dianci;

}

public String getLiewen() {

return liewen;

}

public void setLiewen(String liewen) {

this.liewen = liewen;

}

public String getYuban() {

return yuban;

}

public void setYuban(String yuban) {

this.yuban = yuban;

}

public String getTaizhi() {

return taizhi;

}

public void setTaizhi(String taizhi) {

this.taizhi = taizhi;

}

public String getShexing() {

return shexing;

}

public void setShexing(String shexing) {

this.shexing = shexing;

}

public String getShese() {

return shese;

}

public void setShese(String shese) {

this.shese = shese;

}

public String getTaise() {

return taise;

}

public void setTaise(String taise) {

this.taise = taise;

}

public String getJieguo() {

return jieguo;

}

public void setJieguo(String jieguo) {

this.jieguo = jieguo;

}

public String getRgbR() {

return rgbR;

}

public void setRgbR(String rgbR) {

this.rgbR = rgbR;

}

public String getRgbG() {

return rgbG;

}

public void setRgbG(String rgbG) {

this.rgbG = rgbG;

}

public String getRgbB() {

return rgbB;

}

public void setRgbB(String rgbB) {

this.rgbB = rgbB;

}

public String getHsvH() {

return hsvH;

}

public void setHsvH(String hsvH) {

this.hsvH = hsvH;

}

public String getHsvS() {

return hsvS;

}

public void setHsvS(String hsvS) {

this.hsvS = hsvS;

}

public String getHsvV() {

return hsvV;

}

public void setHsvV(String hsvV) {

this.hsvV = hsvV;

}

@Override public String toString() {

return "UserItemDetail{" + "id=" + id + ", xingming='" + xingming + '\'' + ", xingbie=" + xingbie + ", chushengriqi='" + chushengriqi + '\'' + ", nianning=" + nianning + ", nianningjieduan=" + nianningjieduan + ", njzd='" + njzd + '\'' + ", blzd='"

+ blzd + '\'' + ", zzzd='" + zzzd + '\'' + ", houbao='" + houbao + '\'' + ", funi='" + funi + '\'' + ", runzao='" + runzao + '\'' + ", botai='" + botai + '\'' + ", pangshou='" + pangshou + '\'' + ", chihen='" + chihen + '\'' + ", dianci='"

+ dianci + '\'' + ", liewen='" + liewen + '\'' + ", yuban='" + yuban + '\'' + ", taizhi='" + taizhi + '\'' + ", shexing='" + shexing + '\'' + ", shese='" + shese + '\'' + ", taise='" + taise + '\'' + ", jieguo='" + jieguo + '\'' + ", rgbR='"

+ rgbR + '\'' + ", rgbG='" + rgbG + '\'' + ", rgbB='" + rgbB + '\'' + ", hsvH='" + hsvH + '\'' + ", hsvS='" + hsvS + '\'' + ", hsvV='" + hsvV + '\'' + ", labL='" + labL + '\'' + ", labA='" + labA + '\'' + ", labB='" + labB + '\'' + ", q1\_2='"

+ q1\_2 + '\'' + ", q1\_3r='" + q1\_3r + '\'' + ", q1\_3y='" + q1\_3y + '\'' + ", q2\_1='" + q2\_1 + '\'' + ", products=" + products + '}';

}

public List<Product> getProducts(){

List<Product> products = new ArrayList<Product>();

products.add(new Product(this.getJieguo(),1));

products.add(new Product(this.getNjzd(),35));

products.add(new Product(this.getBlzd(),65));

products.add(new Product(this.getZzzd(),85));

//

return products;

}

}

Front-end:

// Ionic Starter App

// angular.module is a global place for creating, registering and retrieving Angular modules

// 'starter' is the name of this angular module example (also set in a <body> attribute in index.html)

// the 2nd parameter is an array of 'requires'

// 'starter.services' is found in services.js

// 'starter.controllers' is found in controllers.js

angular.module('app', ['ionic', 'app.controllers', 'app.routes', 'app.directives','ngFileUpload','ionic-zoom-view'])

.config(function($ionicConfigProvider, $sceDelegateProvider){

$sceDelegateProvider.resourceUrlWhitelist([ 'self','\*://www.youtube.com/\*\*', '\*://player.vimeo.com/video/\*\*']);

})

//ENV\_config

.constant('CONFIG\_ENV', {

//'api\_endpoint\_base': 'http://localhost:8080/',

'api\_endpoint\_base': DynamicEnvironment.get('api\_endpoint\_base'),

'api\_endpoint': DynamicEnvironment.get('api\_endpoint\_base'),

'api\_version': '0.0.1',

'pubnub\_key': 'ws://www.xyz.com:61614/stomp',

'pubnub\_secret': 'v11.stomp',

'debug': false

})

////$log configure

.config(['$logProvider', function($logProvider){

var isProduction = false;

$logProvider.debugEnabled(!isProduction);

//TODO:https://github.com/ThomasBurleson/angularjs-logDecorator

}])

.config(['$resourceProvider', function($resourceProvider) {

// Don't strip trailing slashes from calculated URLs

$resourceProvider.defaults.stripTrailingSlashes = false;

}])

.config(function($httpProvider) {

$httpProvider.interceptors.push('TrendicityInterceptor');

})

.run(function($ionicPlatform) {

$ionicPlatform.ready(function() {

// Hide the accessory bar by default (remove this to show the accessory bar above the keyboard

// for form inputs)

if (window.cordova && window.cordova.plugins && window.cordova.plugins.Keyboard) {

cordova.plugins.Keyboard.hideKeyboardAccessoryBar(true);

cordova.plugins.Keyboard.disableScroll(true);

}

if (window.StatusBar) {

// org.apache.cordova.statusbar required

StatusBar.styleDefault();

}

});

})

/\*

This directive is used to disable the "drag to open" functionality of the Side-Menu

when you are dragging a Slider component.

\*/

.directive('disableSideMenuDrag', ['$ionicSideMenuDelegate', '$rootScope', function($ionicSideMenuDelegate, $rootScope) {

return {

restrict: "A",

controller: ['$scope', '$element', '$attrs', function ($scope, $element, $attrs) {

function stopDrag(){

$ionicSideMenuDelegate.canDragContent(false);

}

function allowDrag(){

$ionicSideMenuDelegate.canDragContent(true);

}

$rootScope.$on('$ionicSlides.slideChangeEnd', allowDrag);

$element.on('touchstart', stopDrag);

$element.on('touchend', allowDrag);

$element.on('mousedown', stopDrag);

$element.on('mouseup', allowDrag);

}]

};

}])

.filter('percentage', ['$filter', function ($filter) {

return function (input, decimals) {

return $filter('number')(input \* 100, decimals) + '%';

};

}])

/\*

This directive is used to open regular and dynamic href links inside of inappbrowser.

\*/

.directive('hrefInappbrowser', function() {

return {

restrict: 'A',

replace: false,

transclude: false,

link: function(scope, element, attrs) {

var href = attrs['hrefInappbrowser'];

attrs.$observe('hrefInappbrowser', function(val){

href = val;

});

element.bind('click', function (event) {

window.open(href, '\_system', 'location=yes');

event.preventDefault();

event.stopPropagation();

});

}

};

});// angular.module('app.controllers', ['ngFileUpload','pubnub.angular.service'])

angular.module('app.controllers', ['app.services','ngFileUpload'])

.controller('MainCtrl', // The following is the constructor function for this page's controller. See https://docs.angularjs.org/guide/controller

// You can include any angular dependencies as parameters for this function

// TIP: Access Route Parameters for your page via $stateParams.parameterName

function ($rootScope,$scope, $stateParams,$ionicModal,$log,$ionicPopup) {

//badge number

$rootScope.numOfunConsulted = 0;

$log.info("MainCtrl...");

//root scope variables for modal.

$rootScope.consultingModal = null;

$rootScope.userInfos = [];

$rootScope.allInstructions = [];

$rootScope.allPescriptions = [];

$rootScope.consultEinstrMpres = {id:-1,eiid:-1,mpid:-1,einame:"",mpname:"",eicontent:"",mpcontent:""};

$rootScope.consultInfofull = {eInstruction:{},mPrescription:{}};

//selected

$rootScope.unconsultUserInfos = [];

$rootScope.selectedUserInfo = {};

$rootScope.selectedItemInfo = {};

$rootScope.selectedItemDetail = {};

$rootScope.selectedConsultInfo = {pid:-1,iid:-1};

$rootScope.selectedInstruction = null;

$rootScope.selectedPrescription = null;

//ConsultModal

$ionicModal.fromTemplateUrl('templates/modal-consulting.html', {

scope: $scope,

backdropClickToClose: false

}).then(function (modal) {

// console.log("modal-login.html init!!!");

$rootScope.consultingModal = modal;

//

});

$rootScope.consultingAutoModal = null;

$ionicModal.fromTemplateUrl('templates/modal-consulting-auto.html', {

scope: $scope,

animation: 'slide-in-up'

}).then(function(modal) {

$rootScope.consultingAutoModal = modal;

});

$rootScope.consultingStaticModal = null;

$ionicModal.fromTemplateUrl('templates/modal-consulting-static.html', {

scope: $scope,

animation: 'slide-in-up'

}).then(function(modal) {

$rootScope.consultingStaticModal = modal;

});

//NewInstructionModal

$rootScope.newInstructionModal = null;

$ionicModal.fromTemplateUrl('templates/modal-instruction-new.html', {

scope: $scope,

animation: 'slide-in-up'

}).then(function(modal) {

$rootScope.newInstructionModal = modal;

// console.log(" $rootScope.newInstructionModal:"+ $rootScope.newInstructionModal);

});

//PrescriptionModal

$rootScope.newPrescriptionModal = null;

$ionicModal.fromTemplateUrl('templates/modal-prescription-new.html', {

scope: $scope,

animation: 'slide-in-up'

}).then(function(modal) {

$rootScope.newPrescriptionModal = modal;

// console.log(" $rootScope.newPrescriptionModal:"+ $rootScope.newPrescriptionModal);

});

//PrintPreviewModal

$rootScope.newPrintPreviewModal = null;

$ionicModal.fromTemplateUrl('templates/modal-print-preview.html', {

scope: $scope,

animation: 'slide-in-up'

}).then(function(modal) {

$rootScope.newPrintPreviewModal = modal;

// console.log(" $rootScope.newPrescriptionModal:"+ $rootScope.newPrescriptionModal);

});

// A Simple alert

$rootScope.showAlert = function($msg) {

var alertPopup = $ionicPopup.alert({

title: $msg

// template: 'It might taste good'

});

alertPopup.then(function (res) {

// console.log('Thank you for not eating my delicious ice cream cone');

});

}

// A confirm dialog

$rootScope.showPrompt = function($title) {

var confirmPopup = $ionicPopup.confirm({

title: $title,

template: '立即问诊?',

buttons: [{ // Array[Object] (optional). Buttons to place in the popup footer.

text: '取消',

type: 'button-default',

onTap: function(e) {

// e.preventDefault() will stop the popup from closing when tapped.

// e.preventDefault();

}

}, {

text: '确定',

type: 'button-positive',

onTap: function(e) {

// Returning a value will cause the promise to resolve with the given value.

$rootScope.getDiagnosis();

//

return null;

}

}]

});

confirmPopup.then(function(res) {

if(res) {

// console.log('You are sure');

} else {

// console.log('You are not sure');

}

})

};

})

//

.controller('page3Ctrl', function ($rootScope,$scope,$stateParams,$ionicModal,CONFIG\_ENV,Upload,$ionicLoading,$log,

UpdateItemInfoService,UserInfoService,Enum,$timeout,

RecommendUserService,RecommendItemService,DiagnosisService) {

//

// //PubNub

// Pubnub.init({

// publishKey: CONFIG\_ENV.pubnub\_key,

// subscribeKey: CONFIG\_ENV.pubnub\_secret

// });

$log.info("CONFIG\_ENV:",CONFIG\_ENV);

//ng-model

$scope.userInfo = {id:Enum.getUUID(),name:"", gender:1,age:50, itemId: "",itemDetailId:""};

$log.info("default userInfo:",$scope.userInfo);

$scope.yearsR = "中年";

//age drag input

$scope.drag = function(value) {

$scope.years = Math.floor(value / 12);

if(value<=44){

$scope.yearsR = "青年";

}else if(value>=60){

$scope.yearsR = "老年";

}else {

$scope.yearsR = "中年";

}

};

//

//FileUploader,@see:https://github.com/danialfarid/ng-file-upload

// upload on file select or drop

$scope.uploadItemInfo = function (file) {

if(!file){

$log.error("None file selected.");

}

Upload.upload({

url: CONFIG\_ENV.api\_endpoint+'upload/timage',

data: {file: file}

}).then(function (resp) {

//

$log.info('Success ' + resp.config.data.file.name + ',uploaded. Response: ');

$scope.userInfo.itemId = resp.data.data.id;

}, function (resp) {

$log.error('Error status: ' + resp.status);

}, function (evt) {

// console.log('evt:'+evt);

var progressPercentage = parseInt(100.0 \* evt.loaded / evt.total);

//

$log.info('progress: ' + progressPercentage + '% ' + evt.config.data.file.name);

});

};

// upload on file select or drop

$scope.uploadItemDetail = function (file) {

if(!file){

$log.error("None file selected.");

}

Upload.upload({

url: CONFIG\_ENV.api\_endpoint+'upload/tcsv/false',

data: {file: file}

}).then(function (resp) {

//

$log.debug('Success ' + resp.config.data.file.name + ',uploaded. Response: ');

$log.info(resp.data.data);

$scope.userInfo.itemDetailId = resp.data.data[0].id;

//UPDATE

var updateItemInfo = new UpdateItemInfoService();

updateItemInfo.$update({"Id":$scope.userInfo.itemId,"dId":$scope.userInfo.itemDetailId},function (resp) {

$log.info("updateItemInfo() success, response:", resp);

}, function (resp) {

$log.error('Error status: ' + resp.status);

});

}, function (resp) {

$log.error('Error status: ' + resp.status);

}, function (evt) {

var progressPercentage = parseInt(100.0 \* evt.loaded / evt.total);

console.log('progress: ' + progressPercentage + '% ' + evt.config.data.file.name);

});

};

// upload on file select or drop

$scope.uploadUserItemDetail = function (file) {

if(!file){

$log.error("None file selected.");

}

Upload.upload({

url: CONFIG\_ENV.api\_endpoint+'upload/tcsv/true',

data: {file: file}

}).then(function (resp) {

//

$log.debug('Success ' + resp.config.data.file.name + ',uploaded. Response: ');

$log.info(resp.data.data);

// $scope.userInfo.itemDetailId = resp.data.data.id;

//

}, function (resp) {

$log.error('Error status: ' + resp.status);

}, function (evt) {

var progressPercentage = parseInt(100.0 \* evt.loaded / evt.total);

console.log('progress: ' + progressPercentage + '% ' + evt.config.data.file.name);

});

};

//Select bind

$scope.preferencesItemType = Enum.genderType;

$scope.prefGender = Enum.genderType[0];//Default male.

$scope.setTypeSelected = function (type) {

$scope.prefGender = type;

$scope.userInfo.gender = type.data;

}

$scope.savedUserID = 0;

//CREATE,

$scope.createUserInfo = function () {

//

console.log($scope.userInfo);

var anewUserInfo = new UserInfoService();

anewUserInfo.age = $scope.userInfo.age;

anewUserInfo.name = $scope.userInfo.name;

anewUserInfo.gender = $scope.userInfo.gender;

anewUserInfo.itemId = $scope.userInfo.itemId;

//return $log.debug("createItem(),$scope.newItem:", anewItem);

//Save

anewUserInfo.$save(function (resp) {

$log.info("createUserInfo("+$scope.savedUserID+") success, response:", resp);

$scope.savedUserID = resp.data.id;

// $rootScope.showPrompt("采集成功!");

$rootScope.showAlert("采集成功!");

//Auto diagnosis testing

}, function (resp) {

$log.error('Error status: ' + resp.status);

});

}

$rootScope.getDiagnosis = function(){

DiagnosisService.get({id:$scope.savedUserID}, function (response) {

$log.debug("DiagnosisService.get("+$scope.savedUserID+") success!", response);

//

$rootScope.consultInfofull = response;

//

$rootScope.consultingAutoModal.show();

}, function (error) {

// failure handler

$log.error("DiagnosisService.get() failed:", JSON.stringify(error));

});

}

})

.controller('page4Ctrl', // The following is the constructor function for this page's controller. See https://docs.angularjs.org/guide/controller

// You can include any angular dependencies as parameters for this function

// TIP: Access Route Parameters for your page via $stateParams.parameterName

function ($rootScope,$scope, $stateParams,$log,Enum,ConsultUserInfoService,ItemInfoService

,UserInfoService,ItemDetailService,ConsultInfoService,InstructionService,PrescriptionService) {

$scope.userInfos = [

{

id:Enum.getUUID(),name:"", gender:1,age:50, itemId: "",itemDetailId:""

}

];

//Select binding

$scope.selectedUserInfo = $rootScope.userInfos[0];//Default 0ne.

//GET where none consulting.

$scope.loadUserInfos = function () {

ConsultUserInfoService.get({"id":-1}, function (response) {

$log.debug("ConsultUserInfoService.get() success!", response.data);

$scope.userInfos = response.data;

//

$rootScope.unconsultUserInfos = response.data;

$rootScope.numOfunConsulted = $rootScope.unconsultUserInfos.length;

}, function (error) {

// failure handler

$log.error("ConsultUserInfoService.get() failed:", JSON.stringify(error));

});

};

$scope.loadAllUserInfos = function () {

UserInfoService.get({}, function (response) {

$log.debug("UserInfoService.get() success!", response.data);

$scope.userInfos = response.data;

}, function (error) {

// failure handler

$log.error("UserInfoService.get() failed:", JSON.stringify(error));

});

};

$scope.loadConsultInfo = function () {

//drill down the item detail for select.

ConsultInfoService.get({id:$rootScope.selectedUserInfo.consultId}, function (response) {

$log.debug("ConsultInfoService.get("+$rootScope.selectedUserInfo.consultId+") success!", response.data);

$rootScope.selectedConsultInfo = response.data;

$log.debug(" $rootScope.selectedConsultInfo:", $rootScope.selectedConsultInfo);

//

$rootScope.setInstructionSelected();

$rootScope.setPrescrptionSelected();

//

}, function (error) {

// failure handler

$log.error("ConsultInfoService.get() failed:", JSON.stringify(error));

});

}

$scope.loadItemDetailOne = function (userInfo) {

//drill down the item detail for select.

ItemDetailService.get({id:$rootScope.selectedItemInfo.detailId}, function (response) {

$log.debug("ItemDetailService.get("+$rootScope.selectedItemInfo.detailId+") success!", response.data);

$rootScope.selectedItemDetail = response.data;

$log.debug(" $rootScope.selectedItemDetail:", $rootScope.selectedItemDetail);

//next

$scope.loadConsultInfo();

}, function (error) {

// failure handler

$log.error("ItemDetailService.get() failed:", JSON.stringify(error));

});

}

$scope.getUserItemInfo = function(userInfo){

//

$rootScope.selectedUserInfo = userInfo;

ItemInfoService.get({id:userInfo.itemId}, function (response) {

$log.debug("ItemInfoService.get("+userInfo.itemId+") success!", response.data);

$rootScope.selectedItemInfo = response.data;

//Select binding

$log.debug("$rootScope.selectedItemInfo:",$rootScope.selectedItemInfo);

//

$rootScope.consultingStaticModal.show();

//item related detail

$scope.loadItemDetailOne();

}, function (error) {

// failure handler

$log.error("ItemInfoService.get() failed:", JSON.stringify(error));

});

}

$scope.deleteUserInfo = function (userInfo) {

UserInfoService.delete({id:userInfo.id}, function (response) {

$log.info("UserInfoService.delete() success!", response.data);

//refresh

$scope.loadUserInfos();

}, function (error) {

// failure handler

$log.error("UserInfoService.get() failed:", JSON.stringify(error));

});

}

//default behaviors

console.log("loadUnConsultUsers...");

$scope.loadUserInfos();

})

.controller('ConsultingAutoCtrl', // The following is the constructor function for this page's controller. See https://docs.angularjs.org/guide/controller

// You can include any angular dependencies as parameters for this function

// TIP: Access Route Parameters for your page via $stateParams.parameterName

function ($rootScope,$scope, $stateParams,$ionicModal,$log,DiagnosisInferService,ConsultInfoService) {

$log.info("ConsultingAutoCtrl init...");

//by mahout recommend

$scope.getOtherConsultInfo = function ($order) {

$rootScope.getConsultEinstrMpres($order);

}

//CREATE,by drools infered.

$scope.createConsultInfoAuto = function () {

$log.info("$scope.createConsultInfoAuto called.");

$rootScope.selectedInstruction = {id:-1};

$rootScope.selectedInstruction.id = $rootScope.consultEinstrMpres.eiid;

$rootScope.selectedPrescription = {id:-1};

$rootScope.selectedPrescription.id = $rootScope.consultEinstrMpres.mpid;

$rootScope.createConsultInfo();

};

})

.controller('ConsultingCtrl', // The following is the constructor function for this page's controller. See https://docs.angularjs.org/guide/controller

// You can include any angular dependencies as parameters for this function

// TIP: Access Route Parameters for your page via $stateParams.parameterName

function ($rootScope,$scope, $stateParams,$log,$ionicModal,

UserInfoService,ItemInfoService,ItemDetailService,

InstructionService,PrescriptionService,

ConsultInfoService,UpdateUserInfoService,

ConsultUserInfoService,DiagnosisInferService) {

//

//Select binding

$scope.selectedUserInfo = null;

$scope.selectedItemInfo= null;

$scope.selectedInstruction = null;

$scope.selectedPrescription = null;

//GET where none consulting.

$scope.loadUserInfos = function () {

// UserInfoService.get({}, function (response) {

// $log.info("UserInfoService.get() success!", response.data);

// $rootScope.userInfos = response.data;

// //Select binding

// $scope.selectedUserInfo = $rootScope.userInfos[0];//Default 0ne.

// $log.debug("selectedUserInfo:",$scope.selectedUserInfo);

// }, function (error) {

// // failure handler

// $log.error("UserInfoService.get() failed:", JSON.stringify(error));

// });

//

ConsultUserInfoService.get({"id":-1}, function (response) {

$log.info("ConsultUserInfoService.get() success!", response.data);

$rootScope.userInfos = response.data;

//Select binding

$scope.selectedUserInfo = $rootScope.userInfos[0];//Default 0ne.

$log.debug("selectedUserInfo:",$scope.selectedUserInfo);

//default trigger.

// $scope.setUserInfoSelected($scope.selectedUserInfo);

}, function (error) {

// failure handler

$log.error("UserInfoService.get() failed:", JSON.stringify(error));

});

};

//GET

$scope.loadItemInfos = function () {

ItemInfoService.get({}, function (response) {

$log.debug("ItemInfoService.get() success!", response.data);

$rootScope.itemInfos = response.data;

//Select binding

// $scope.selectedItemInfo= $rootScope.itemInfos[0];//Default 0ne.

$log.debug("selectedItemInfo:",$scope.selectedItemInfo);

}, function (error) {

// failure handler

$log.error("ItemInfoService.get() failed:", JSON.stringify(error));

});

};

//GET

$rootScope.loadAllInstructions = function () {

InstructionService.get({}, function (response) {

$log.info("InstructionService.get() success!", response.data);

$rootScope.allInstructions = response.data;

//Select binding

$rootScope.selectedInstruction = $scope.selectedInstruction = $rootScope.allInstructions[0];//Default 0ne.

$log.debug("selectedInstruction:",$rootScope.selectedInstruction);

}, function (error) {

// failure handler

$log.error("InstructionService.get() failed:", JSON.stringify(error));

});

};

//GET

$rootScope.loadAllPrescriptions = function () {

PrescriptionService.get({}, function (response) {

$log.info("PrescriptionService.get() success!", response);

$rootScope.allPrescriptions = response.data;

//Select binding

$rootScope.selectedPrescription = $scope.selectedPrescription = $rootScope.allPescriptions[0];//Default 0ne.

$log.debug("selectedPrescription:",$rootScope.selectedPrescription);

}, function (error) {

// failure handler

$log.error("PrescriptionService.get() failed:", JSON.stringify(error));

});

}

$scope.loadUserAndItemInfos = function () {

console.log("loadUserAndItemInfos...");

$scope.loadUserInfos();//FIXME: load a sequence chain.

$scope.loadItemInfos();

}

$scope.loadInsAndPres = function () {

console.log("loadInsAndPres...");

$rootScope.loadAllInstructions();//FIXME: load a sequence chain.

$rootScope.loadAllPrescriptions();

}

//CREATE

$rootScope.createConsultInfo = function () {

//

var anewConsultInfo = new ConsultInfoService();

$log.info("$rootScope.selectedInstruction:",$rootScope.selectedInstruction);

$log.info("$rootScope.selectedPrescription:",$rootScope.selectedPrescription);

anewConsultInfo.iid = $rootScope.selectedInstruction.id;

anewConsultInfo.pid = $rootScope.selectedPrescription.id;

$log.info("anewConsultInfo:",anewConsultInfo);

//Save

anewConsultInfo.$save(function (resp) {

$log.info("CREATE ConsultInfoService() success, response:", resp);

var savedConsultId = resp.data.id;

//

$scope.updateUserInfo(savedConsultId);

}, function (resp) {

$log.error('Error status: ' + resp.status);

});

};

//UPDATE

$scope.updateUserInfo = function ($cid) {

var updateUserInfo = new UpdateUserInfoService();

updateUserInfo.$update({"Id":$rootScope.selectedUserInfo.id,"cId":$cid},function (resp) {

$log.info("updateUserInfo() success, response:", resp);

//alert success.

$rootScope.showAlert("答诊成功!");

//reload

// $scope.loadUserAndItemInfos();

//hide consulting modal.

$rootScope.consultingStaticModal.hide();

}, function (resp) {

$log.error('Error status: ' + resp.status);

});

};

//

$scope.loadItemDetailOne = function () {

$log.debug("SELECTED itemInfo's itemDetailId:",$scope.selectedItemInfo.detailId);

//drill down the item detail for select.

ItemDetailService.get({id:$scope.selectedItemInfo.detailId}, function (response) {

$log.debug("ItemDetailService.get(one) success!", response.data);

$scope.itemDetail = response.data;

}, function (error) {

// failure handler

$log.error("ItemInfoService.get() failed:", JSON.stringify(error));

});

}

//SELECT change

$rootScope.setUserInfoSelected = function ($selected) {

$scope.selectedUserInfo = $selected;//refresh.

$log.debug("SELECTED userInfo:",$scope.selectedUserInfo);

//drill down the item info for select.

ItemInfoService.get({id:$selected.itemId}, function (response) {

$log.debug("ItemInfoService.get(",$selected.itemId,") success!", response.data);

//

$scope.selectedItemInfo = response.data;

//

$scope.loadItemDetailOne();

}, function (error) {

// failure handler

$log.error("ItemInfoService.get() failed:", JSON.stringify(error));

});

}

$scope.updateInstructionSelected = function () {

//

$log.debug("SELECTED instruction:",$rootScope.selectedInstruction);

}

$scope.updatePrescriptionSelected = function () {

//

$log.debug("SELECTED prescription:",$rootScope.selectedPrescription);

//

}

//Manually set up selections

$rootScope.setInstructionSelected = function () {

if($rootScope.selectedConsultInfo) {//Only show diagnosised.

InstructionService.get({id: $rootScope.selectedConsultInfo.iid}, function (response) {

$log.debug("InstructionService.get(" + $rootScope.selectedConsultInfo.iid + ") success!", response.data);

$rootScope.selectedInstruction = response.data;

$log.debug(" $rootScope.selectedInstruction:", $rootScope.selectedInstruction);

}, function (error) {

// failure handler

$log.error("InstructionService.get() failed:", JSON.stringify(error));

});

}

}

$rootScope.setPrescrptionSelected = function () {

if($rootScope.selectedConsultInfo) {//Only show diagnosised.

PrescriptionService.get({id: $rootScope.selectedConsultInfo.pid}, function (response) {

$log.debug("PrescriptionService.get(" + $rootScope.selectedConsultInfo.pid + ") success!", response.data);

$rootScope.selectedPrescription = response.data;

$log.debug("$rootScope.selectedPrescription:", $rootScope.selectedPrescription);

}, function (error) {

// failure handler

$log.error("PrescriptionService.get() failed:", JSON.stringify(error));

});

}

}

$scope.printConsultInfo = function(){

$rootScope.newPrintPreviewModal.show();

}

//

$log.info("ConsultCtrl initialize...");

// $scope.loadUserAndItemInfos();

$scope.loadInsAndPres();

//

$rootScope.getConsultEinstrMpres = function ($order) {

console.log("$scope.getConsultEinstrMpres called,order:",$order);

$rootScope.consultingAutoModal.show();

$rootScope.consultEinstrMpres = {};//clear history data.

//

$uid = $rootScope.selectedUserInfo.id;

$log.info("$rootScope.selectedUserInfo.id:",$uid);

DiagnosisInferService.get({"id":$uid,"order":$order}, function (response) {

$log.info("DiagnosisInferService.get() success!", response);

$rootScope.consultEinstrMpres = response;

$log.debug("consultEinstrMpres :",$scope.consultEinstrMpres );

//default trigger.

}, function (error) {

// failure handler

$log.error("DiagnosisInferService.get() failed:", JSON.stringify(error));

});

}

})

.controller('NewInstructionCtrl', // The following is the constructor function for this page's controller. See https://docs.angularjs.org/guide/controller

// You can include any angular dependencies as parameters for this function

// TIP: Access Route Parameters for your page via $stateParams.parameterName

function ($rootScope,$scope, $stateParams,$ionicModal,InstructionService,$log) {

$scope.newInstruction = {name:null,content:null};

//CREATE,

$scope.createInstruction = function () {

//

var anewInstruction = new InstructionService();

anewInstruction.name = $scope.newInstruction.name;

anewInstruction.content = $scope.newInstruction.content;

//Save

anewInstruction.$save(function (resp) {

$log.info("createInstruction success, response:", resp);

$scope.savedUserID = resp.data.id;

//refresh.

$rootScope.loadAllInstructions();

}, function (resp) {

$log.error('Error status: ' + resp.status);

});

}

//DELETE,

$scope.deleteInstruction = function ($item) {

//

var deleteInstruction = new InstructionService();

//Delete

deleteInstruction.$delete({ id: $item.id },function (resp) {

$log.info("deleteInstruction $delete success, response:", resp);

//refresh.

$rootScope.loadAllInstructions();

}, function (resp) {

$log.error('Error status: ' + resp.status);

});

}

})

.controller('NewPrescriptionCtrl', // The following is the constructor function for this page's controller. See https://docs.angularjs.org/guide/controller

// You can include any angular dependencies as parameters for this function

// TIP: Access Route Parameters for your page via $stateParams.parameterName

function ($rootScope,$scope, $stateParams,$ionicModal,PrescriptionService,$log) {

$scope.newPrescription = {name:null,content:null};

//CREATE,

$scope.createPrescription = function () {

//

var anewPrescription = new PrescriptionService();

anewPrescription.name = $scope.newPrescription.name;

anewPrescription.content = $scope.newPrescription.content;

//Save

anewPrescription.$save(function (resp) {

$log.info("createPrescription success, response:", resp);

$scope.savedUserID = resp.data.id;

//refresh.

$rootScope.loadAllPrescriptions();

}, function (resp) {

$log.error('Error status: ' + resp.status);

});

}

//DELETE,

$scope.deletePrescription = function ($item) {

//

var deletePrescription = new PrescriptionService();

//Delete

deletePrescription.$delete({ id: $item.id },function (resp) {

$log.info("PrescriptionService $delete success, response:", resp);

//refresh.

$rootScope.loadAllPrescriptions();

}, function (resp) {

$log.error('Error status: ' + resp.status);

});

}

})

.controller('PrintPreviewCtrl', // The following is the constructor function for this page's controller. See https://docs.angularjs.org/guide/controller

// You can include any angular dependencies as parameters for this function

// TIP: Access Route Parameters for your page via $stateParams.parameterName

function ($rootScope,$scope, $stateParams,$ionicModal,UserInfoService,PrescriptionService,$log) {

//Display UserInfo,prescription,instruction ,and other information.

//

$scope.printNow = function (divID) {

//HIDE MODAL FIRST OFF.

$rootScope.newPrintPreviewModal.hide();

//Get the HTML of div

var divElements = document.getElementById(divID).innerHTML;

//Get the HTML of whole page

var oldPage = document.body.innerHTML;

//Reset the page's HTML with div's HTML only

document.body.innerHTML =

"<html><head><title></title></head><body>" +

divElements + "</body>";

//Print Page

window.print();

//Restore orignal HTML

document.body.innerHTML = oldPage;

}

})angular.module('app.directives', [])

.directive('fileButton', function() {

return {

link: function(scope, element, attributes) {

var el = angular.element(element)

var button = el.children()[0]

el.css({

position: 'relative',

overflow: 'hidden',

width: button.offsetWidth,

height: button.offsetHeight

})

var fileInput = angular.element('<input type="file" />')

fileInput.css({

position: 'absolute',

top: 0,

left: 0,

'z-index': '2',

width: '100%',

height: '100%',

opacity: '0',

cursor: 'pointer'

})

el.append(fileInput)

}

}

})

.directive('blankDirective', [function(){

}]);var DynamicEnvironment = DynamicEnvironment || {};

//Helper functions here.

/\*\*

\* You can have as many environments as you like in here

\* just make sure the host matches up to your hostname including port

\*/

var \_environment;

var \_environments = {

local: {

host: 'localhost',

config: {

/\*\*

\* Add any config properties you want in here for this environment

\*/

api\_endpoint\_base: 'http://localhost:8080/td/'

// api\_endpoint\_base: 'http://118.190.96.120:8070/td/'

,clientId:'clientApp'

,clientSecret:'1NDgzZGY1OWViOWRmNjI5ZT'

}

},

dev: {

host: '118.190.3.169',

config: {

/\*\*

\* Add any config properties you want in here for this environment

\*/

api\_endpoint\_base: 'http://118.190.3.169:8080/td/'

,clientId:'clientApp'

,clientSecret:'1NDgzZGY1OWViOWRmNjI5ZT'

}

},

test: {

host: '118.190.96.120',

config: {

/\*\*

\* Add any config properties you want in here for this environment

\*/

api\_endpoint\_base: 'http://118.190.96.120:8070/td/'

,clientId:'clientApp'

,clientSecret:'1NDgzZGY1OWViOWRmNjI5ZT'

}

}

};

\_getEnvironment = function () {

var protocol = location.protocol;

var slashes = protocol.concat("//");

// var host = slashes.concat(window.location.hostname);

var host = window.location.host;

console.log("host:"+host);

var hostname = window.location.hostname;

console.log("hostname:"+hostname);

if (\_environment) {

return \_environment;

}

for (var environment in \_environments) {

if (typeof \_environments[environment].host && \_environments[environment].host == hostname) {

\_environment = environment;

return \_environment;

}

}

return "local";//default

};

DynamicEnvironment.get = function (property) {

var result = \_environments[\_getEnvironment()].config[property];

//var result = \_environments["test"].config[property];

console.log("DynamicEnvironment.get():",result);

return result;

};

angular.module('app.routes', [])

.config(function($stateProvider, $urlRouterProvider) {

// Ionic uses AngularUI Router which uses the concept of states

// Learn more here: https://github.com/angular-ui/ui-router

// Set up the various states which the app can be in.

// Each state's controller can be found in controllers.js

$stateProvider

.state('tabsController.page3', {

url: '/page3',

views: {

'tab2': {

templateUrl: 'templates/page3.html',

controller: 'page3Ctrl'

}

}

})

.state('tabsController.page4', {

url: '/page4',

views: {

'tab3': {

templateUrl: 'templates/page4.html',

controller: 'page4Ctrl'

}

}

})

.state('tabsController', {

url: '/page1',

templateUrl: 'templates/tabsController.html',

abstract:true

})

$urlRouterProvider.otherwise('/page1/page3')

});angular.module('app.services', ['ngResource'])

//DiagnosisService

.factory('DiagnosisService', function ($resource,CONFIG\_ENV) {

var data = $resource(

CONFIG\_ENV.api\_endpoint + "diagnosis/user/:id"

)

return data;

})

//DiagnosisService

.factory('DiagnosisInferService', function ($resource,CONFIG\_ENV) {

var data = $resource(

CONFIG\_ENV.api\_endpoint + "diagnosis/user/infer/:id/:order"

)

return data;

})

//RecommendItemService

.factory('RecommendItemService', function ($resource,CONFIG\_ENV) {

var data = $resource(

CONFIG\_ENV.api\_endpoint + "recommend/item/:id"

)

return data;

})

//RecommendUserService

.factory('RecommendUserService', function ($resource,CONFIG\_ENV) {

var data = $resource(

CONFIG\_ENV.api\_endpoint + "recommend/user/:id"

)

return data;

})

//ItemDetailService

.factory('ItemDetailService', function ($resource,CONFIG\_ENV) {

var data = $resource(

CONFIG\_ENV.api\_endpoint + "item/detail/:id"

)

return data;

})

//ItemInfoService

.factory('ItemInfoService', function ($resource,CONFIG\_ENV) {

var data = $resource(

CONFIG\_ENV.api\_endpoint + "item/info/:id"

)

return data;

})

//UpdateItemInfoService

.factory('UpdateItemInfoService', function ($resource,CONFIG\_ENV) {

var data = $resource(CONFIG\_ENV.api\_endpoint + 'item/info/:Id/detail/:dId', {Id: "@Id",dId:"@dId"}, {

update: {method: 'PUT', params: {Id: "@Id",dId:"@dId"}}

});

return data;

})

//UserInfoService

.factory('UserInfoService', function ($resource,CONFIG\_ENV) {

var data = $resource(

CONFIG\_ENV.api\_endpoint + "user/:id"

)

return data;

})

//ConsultUserInfoService

.factory('ConsultUserInfoService', function ($resource,CONFIG\_ENV) {

var data = $resource(

CONFIG\_ENV.api\_endpoint + "user/c/:id"

)

return data;

})

//UpdateUserInfoService

.factory('UpdateUserInfoService', function ($resource,CONFIG\_ENV) {

var data = $resource(CONFIG\_ENV.api\_endpoint + 'user/:Id/cid/:cId', {Id: "@Id",cId:"@cId"}, {

update: {method: 'PUT', params: {Id: "@Id",cId:"@cId"}}

});

return data;

})

//PrescriptionService

.factory('PrescriptionService', function ($resource,CONFIG\_ENV) {

var data = $resource(

CONFIG\_ENV.api\_endpoint + "prescription/:id"

)

return data;

})

//InstructionService

.factory('InstructionService', function ($resource,CONFIG\_ENV) {

var data = $resource(

CONFIG\_ENV.api\_endpoint + "instruction/:id"

)

return data;

})

//ConsultInfoService

.factory('ConsultInfoService', function ($resource,CONFIG\_ENV) {

var data = $resource(

CONFIG\_ENV.api\_endpoint + "consult/:id"

)

return data;

})

///@see: http://forum.ionicframework.com/t/ionicloading-in-http-interceptor/4599/7

.factory('TrendicityInterceptor',

function ($injector, $q, $log) {

var hideLoadingModalIfNecessary = function () {

var $http = $http || $injector.get('$http');

if ($http.pendingRequests.length === 0) {

$injector.get('$ionicLoading').hide();

}

};

return {

request: function (config) {

$injector.get('$ionicLoading').show();

// Handle adding the access\_token or auth request.

return config;

},

requestError: function (rejection) {

hideLoadingModalIfNecessary();

return $q.reject(rejection);

},

response: function (response) {

hideLoadingModalIfNecessary();

return response;

},

responseError: function (rejection) {

hideLoadingModalIfNecessary();

//http status code check

$log.error("detected what appears to be an oAuth error...", rejection);

if (rejection.status == 400) {

rejection.status = 401; // Set the status to 401 so that angular-http-auth inteceptor will handle it

}

return $q.reject(rejection);

}

};

})

//@see http://stackoverflow.com/questions/16627860/angular-js-and-ng-swith-when-emulating-enum

.factory('Enum', [function () {

var service = {

//

genderType: [

//male:

{

name: "男",

data: "1"

},

//felmale:

{

name: "女",

data: "0"

}

]

, getUUID: function () {

// http://www.ietf.org/rfc/rfc4122.txt

var s = [];

var hexDigits = "0123456789abcdef";

for (var i = 0; i < 36; i++) {

s[i] = hexDigits.substr(Math.floor(Math.random() \* 0x10), 1);

}

s[14] = "4"; // bits 12-15 of the time\_hi\_and\_version field to 0010

s[19] = hexDigits.substr((s[19] & 0x3) | 0x8, 1); // bits 6-7 of the clock\_seq\_hi\_and\_reserved to 01

s[8] = s[13] = s[18] = s[23] = "-";

var uuid = s.join("");

return uuid;

}

, getTimestamp: function () {

var now = new Date;

var utc\_timestamp = Date.UTC(now.getUTCFullYear(), now.getUTCMonth(), now.getUTCDate(),

now.getUTCHours(), now.getUTCMinutes(), now.getUTCSeconds(), now.getUTCMilliseconds());

return utc\_timestamp;

}

};

return service;

}]);