**Occupation Master Controller**

package com.maan.crm.controller;

import java.util.Collections;

import java.util.List;

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

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

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

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

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

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

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

import com.maan.crm.document.req.OccupationMasterGetReq;

import com.maan.crm.req.OccupationMasterGetAllReq;

import com.maan.crm.req.OccupationSaveReq;

import com.maan.crm.res.CommonCrmRes;

import com.maan.crm.res.DropDownRes;

import com.maan.crm.res.OccupationDropDownRes;

import com.maan.crm.res.SuccessRes;

import com.maan.crm.service.OccupationMasterService;

import com.maan.crm.service.PrintReqService;

import com.maan.crm.util.error.Error;

@RestController

@RequestMapping("/master")

public class OccupationMasterController {

@Autowired

private OccupationMasterService occupationService;

@Autowired

private PrintReqService reqPrinter;

// Occupation Drop Down Type

@GetMapping("/dropdown/occupation")

public ResponseEntity<CommonCrmRes> getOccupationDropdown() {

CommonCrmRes data = new CommonCrmRes();

// Save

List<DropDownRes> res = occupationService.getOccupationDropdown();

data.setCommonResponse(res);

data.setIsError(false);

data.setErrorMessage(Collections.emptyList());

data.setMessage("Success");

if (res != null) {

return new ResponseEntity<CommonCrmRes>(data, HttpStatus.CREATED);

} else {

return new ResponseEntity<>(null, HttpStatus.BAD\_REQUEST);

}

}

// Get All

@PostMapping("/getalloccupation")

public ResponseEntity<CommonCrmRes> getallOccupationDropdown(@RequestBody OccupationMasterGetAllReq req) {

reqPrinter.reqPrint(req);

CommonCrmRes data = new CommonCrmRes();

//Get All

List<OccupationDropDownRes> res = occupationService.getallOccupationDropdown(req);

data.setCommonResponse(res);

data.setIsError(false);

data.setErrorMessage(Collections.emptyList());

data.setMessage("Success");

if (res != null) {

return new ResponseEntity<CommonCrmRes>(data, HttpStatus.CREATED);

} else {

return new ResponseEntity<>(null, HttpStatus.BAD\_REQUEST);

}

}

// Occupation Get By Occupation Id

@PostMapping("/getoccupationbyid")

public ResponseEntity<CommonCrmRes> getOccupationId(@RequestBody OccupationMasterGetReq req ) {

CommonCrmRes data = new CommonCrmRes();

OccupationDropDownRes res = occupationService.getOccupationId(req);

data.setCommonResponse(res);

data.setIsError(false);

data.setErrorMessage(Collections.emptyList());

data.setMessage("Success");

if (res != null) {

return new ResponseEntity<CommonCrmRes>(data, HttpStatus.CREATED);

} else {

return new ResponseEntity<>(null, HttpStatus.BAD\_REQUEST);

}

}

// Insert

@PostMapping("/saveoccupation")

public ResponseEntity<CommonCrmRes> saveOccupation(@RequestBody OccupationSaveReq req) {

reqPrinter.reqPrint(req);

CommonCrmRes data = new CommonCrmRes();

List<Error> validation = occupationService.validateOccupation(req);

//// validation

if (validation != null && validation.size() != 0) {

data.setCommonResponse(null);

data.setIsError(true);

data.setErrorMessage(validation);

data.setMessage("Failed");

return new ResponseEntity<CommonCrmRes>(data, HttpStatus.OK);

} else {

/////// save

SuccessRes res = occupationService.saveCrmCityMaster(req);

data.setCommonResponse(res);

data.setIsError(false);

data.setErrorMessage(Collections.emptyList());

data.setMessage("Success");

if (res != null) {

return new ResponseEntity<CommonCrmRes>(data, HttpStatus.CREATED);

} else {

return new ResponseEntity<>(null, HttpStatus.BAD\_REQUEST);

}

}

}

}

**Service Impl**

package com.maan.crm.service.impl;

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.Calendar;

import java.util.Date;

import java.util.GregorianCalendar;

import java.util.List;

import javax.persistence.EntityManager;

import javax.persistence.PersistenceContext;

import javax.persistence.TypedQuery;

import javax.persistence.criteria.CriteriaBuilder;

import javax.persistence.criteria.CriteriaQuery;

import javax.persistence.criteria.Order;

import javax.persistence.criteria.Root;

import javax.persistence.criteria.Subquery;

import org.apache.commons.lang3.StringUtils;

import org.apache.logging.log4j.LogManager;

import org.apache.logging.log4j.Logger;

import org.modelmapper.ModelMapper;

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import com.google.gson.Gson;

import com.maan.crm.bean.OccupationMaster;

import com.maan.crm.document.req.OccupationMasterGetReq;

import com.maan.crm.repository.OccupationMasterRepository;

import com.maan.crm.req.OccupationMasterGetAllReq;

import com.maan.crm.req.OccupationSaveReq;

import com.maan.crm.res.DropDownRes;

import com.maan.crm.res.OccupationDropDownRes;

import com.maan.crm.res.SuccessRes;

import com.maan.crm.service.OccupationMasterService;

import com.maan.crm.util.error.Error;

@Service

@Transactional

public class OccupationMasterServiceImpl implements OccupationMasterService{

@Autowired

private OccupationMasterRepository repo;

@PersistenceContext

private EntityManager em;

Gson json = new Gson();

private Logger log = LogManager.getLogger(OccupationMasterServiceImpl.class);

// Occupation Drop Down

@Override

public List<DropDownRes> getOccupationDropdown() {

List<DropDownRes> resList = new ArrayList<DropDownRes>();

try {

Date today = new Date();

Calendar cal = new GregorianCalendar();

cal.setTime(today);

cal.set(Calendar.HOUR\_OF\_DAY, 23);

cal.set(Calendar.MINUTE, 1);

today = cal.getTime();

// Criteria

CriteriaBuilder cb = em.getCriteriaBuilder();

CriteriaQuery<OccupationMaster> query = cb.createQuery(OccupationMaster.class);

List<OccupationMaster> list = new ArrayList<OccupationMaster>();

// Find All

Root<OccupationMaster> c = query.from(OccupationMaster.class);

// Select

query.select(c );

// Amend Id Max Filter

Subquery<Long> amendId = query.subquery(Long.class);

Root<OccupationMaster> ocpm1 = amendId.from(OccupationMaster.class);

amendId.select(cb.max(ocpm1.get("amendId")));

javax.persistence.criteria.Predicate a1 = cb.equal(c.get("occupationId"),ocpm1.get("occupationId") );

amendId.where(a1);

// Order By

List<Order> orderList = new ArrayList<Order>();

orderList.add(cb.asc(c.get("occupationName")));

// Where

javax.persistence.criteria.Predicate n1 = cb.equal(c.get("status"), "Y");

javax.persistence.criteria.Predicate n2 = cb.lessThanOrEqualTo(c.get("effectiveDate"), today);

javax.persistence.criteria.Predicate n3 = cb.equal(c.get("amendId"),amendId) ;

query.where(n1,n2,n3).orderBy(orderList);

// Get Result

TypedQuery<OccupationMaster> result = em.createQuery(query);

list = result.getResultList();

for(OccupationMaster data : list ) {

// Response

DropDownRes res = new DropDownRes();

res.setCode(data.getOccupationId().toString());

res.setCodeDesc(data.getOccupationName());

resList.add(res);

}

} catch (Exception e) {

e.printStackTrace();

log.info("Exception is ---> " + e.getMessage());

return null;

}

return resList;

}

// Occupation Get All

@Override

public List<OccupationDropDownRes> getallOccupationDropdown(OccupationMasterGetAllReq req) {

List<OccupationDropDownRes> resList = new ArrayList<OccupationDropDownRes>();

ModelMapper mapper = new ModelMapper();

SimpleDateFormat dbf = new SimpleDateFormat("yyyy-MM-dd");

SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");

try {

// Limit Offset

int limit = StringUtils.isBlank(req.getLimit())?0:Integer.valueOf(req.getLimit());

int offset = StringUtils.isBlank(req.getOffset())?100:Integer.valueOf(req.getOffset());

// Criteria

CriteriaBuilder cb = em.getCriteriaBuilder();

CriteriaQuery<OccupationMaster> query = cb.createQuery(OccupationMaster.class);

List<OccupationMaster> list = new ArrayList<OccupationMaster>();

// Find All

Root<OccupationMaster> c = query.from(OccupationMaster.class);

// Select

query.select(c );

// Effective Date Max Filter

Subquery<Long> effectiveDate = query.subquery(Long.class);

Root<OccupationMaster> ocpm1 = effectiveDate.from(OccupationMaster.class);

effectiveDate.select(cb.max(ocpm1.get("effectiveDate")));

javax.persistence.criteria.Predicate a1 = cb.equal(c.get("occupationId"),ocpm1.get("occupationId") );

effectiveDate.where(a1);

// Amend Id Max Filter

Subquery<Long> amendId = query.subquery(Long.class);

Root<OccupationMaster> ocpm2 = amendId.from(OccupationMaster.class);

amendId.select(cb.max(ocpm2.get("amendId")));

javax.persistence.criteria.Predicate a2 = cb.equal(c.get("occupationId"),ocpm2.get("occupationId") );

javax.persistence.criteria.Predicate a3 = cb.equal(c.get("effectiveDate"),ocpm2.get("effectiveDate") );

amendId.where(a2,a3);

// Order By

List<Order> orderList = new ArrayList<Order>();

orderList.add(cb.asc(c.get("occupationName")));

// Where

javax.persistence.criteria.Predicate n1 = cb.equal(c.get("status"), "Y");

javax.persistence.criteria.Predicate n2 = cb.equal(c.get("effectiveDate"), effectiveDate);

javax.persistence.criteria.Predicate n3 = cb.equal(c.get("amendId"),amendId) ;

query.where(n1,n2,n3).orderBy(orderList);

// Get Result

TypedQuery<OccupationMaster> result = em.createQuery(query);

result.setFirstResult(limit \* offset);

result.setMaxResults(offset);

list = result.getResultList();

for (OccupationMaster data : list) {

OccupationDropDownRes res = new OccupationDropDownRes();

res = mapper.map(data, OccupationDropDownRes.class);

res.setEntryDate(sdf.format(data.getEntryDate()));

res.setEffectiveDate(sdf.format(data.getEffectiveDate()));

resList.add(res);

}

} catch (Exception e) {

e.printStackTrace();

log.info("Exception is ---> " + e.getMessage());

return null;

}

return resList;

}

// Occupation Get By Occupation Id

@Override

public OccupationDropDownRes getOccupationId(OccupationMasterGetReq req) {

OccupationDropDownRes res = new OccupationDropDownRes();

ModelMapper mapper = new ModelMapper();

SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");

try {

// Criteria

CriteriaBuilder cb = em.getCriteriaBuilder();

CriteriaQuery<OccupationMaster> query = cb.createQuery(OccupationMaster.class);

List<OccupationMaster> list = new ArrayList<OccupationMaster>();

// Find All

Root<OccupationMaster> c = query.from(OccupationMaster.class);

// Select

query.select(c );

// Effective Date Max Filter

Subquery<Long> effectiveDate = query.subquery(Long.class);

Root<OccupationMaster> ocpm1 = effectiveDate.from(OccupationMaster.class);

effectiveDate.select(cb.max(ocpm1.get("effectiveDate")));

javax.persistence.criteria.Predicate a1 = cb.equal(c.get("occupationId"),ocpm1.get("occupationId") );

effectiveDate.where(a1);

// Amend Id Max Filter

Subquery<Long> amendId = query.subquery(Long.class);

Root<OccupationMaster> ocpm2 = amendId.from(OccupationMaster.class);

amendId.select(cb.max(ocpm2.get("amendId")));

javax.persistence.criteria.Predicate a2 = cb.equal(c.get("occupationId"),ocpm2.get("occupationId") );

javax.persistence.criteria.Predicate a3 = cb.equal(c.get("effectiveDate"),ocpm2.get("effectiveDate") );

amendId.where(a2,a3);

// Order By

List<Order> orderList = new ArrayList<Order>();

orderList.add(cb.asc(c.get("occupationName")));

// Where

javax.persistence.criteria.Predicate n1 = cb.equal(c.get("status"), "Y");

javax.persistence.criteria.Predicate n2 = cb.equal(c.get("effectiveDate"), effectiveDate);

javax.persistence.criteria.Predicate n3 = cb.equal(c.get("amendId"),amendId) ;

javax.persistence.criteria.Predicate n4 = cb.equal(c.get("occupationId"),req.getOccupationId()) ;

query.where(n1,n2,n3,n4).orderBy(orderList);

// Get Result

TypedQuery<OccupationMaster> result = em.createQuery(query);

list = result.getResultList();

res = mapper.map(list.get(0) , OccupationDropDownRes.class);

res.setEntryDate(sdf.format(list.get(0).getEntryDate()));

res.setEffectiveDate(sdf.format(list.get(0).getEffectiveDate()));

} catch (Exception e) {

e.printStackTrace();

log.info("Exception is ---> " + e.getMessage());

return null;

}

return res;

}

/// Validation

@Override

public List<Error> validateOccupation(OccupationSaveReq req) {

List<Error> errors = new ArrayList<Error>();

SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");

try {

if (req.getOccupationName() == null || StringUtils.isBlank(req.getOccupationName())) {

errors.add(new Error("01", "Occupation Name", "Please Enter Occupation Name"));

} else if (req.getOccupationName().length() > 50) {

errors.add(new Error("01", "Occupation Name", "Please Enter Occupation Name within 50 Characters"));

}

if (req.getStatus() == null || StringUtils.isBlank(req.getStatus())) {

errors.add(new Error("02", "Status", "Please Enter Status"));

}

else if (req.getStatus().length() > 1) {

errors.add(new Error("02", "Status", "Please Enter Status within 1 Character"));

} else if (!(req.getStatus().equals("Y") || req.getStatus().equals("N"))) {

errors.add(new Error("02", "Status", "Please Enter Status Properly Yes-Y or No-N"));

}

Calendar cal = Calendar.getInstance();

cal.add(Calendar.DATE, -1);

Date yesterday = cal.getTime();

Date a = sdf.parse(req.getEffectiveDate());

if (req.getEffectiveDate() == null || StringUtils.isBlank(req.getEffectiveDate().toString())) {

errors.add(new Error("03", "Effective Date", "Please Enter Effective Date"));

} else if (a.before(yesterday)) {

errors.add(new Error("03", "EffectiveDate", "Please Enter Future Date as EffectiveDate"));

} else if (!req.getEffectiveDate().matches("([0-9]{2})/([0-9]{2})/([0-9]{4})")) {

errors.add(new Error("03", "EffectiveDate",

"Effective Date format should be dd/MM/yyyy only allowed . Example :- 15/12/2022"));

}

if (req.getRemarks() == null || StringUtils.isBlank(req.getRemarks())) {

errors.add(new Error("04", "Remarks", "Please Enter Remarks"));

}

else if (req.getRemarks().length() > 50) {

errors.add(new Error("04", "Remarks", "Please Enter Remarks within 50 Characters"));

}

} catch (Exception e) {

e.printStackTrace();

log.info("Exception is --->" + e.getMessage());

return errors;

}

return errors;

}

/// Save

@Override

public SuccessRes saveCrmCityMaster(OccupationSaveReq req) {

SuccessRes res = new SuccessRes();

OccupationMaster saveData = new OccupationMaster();

SimpleDateFormat dbf = new SimpleDateFormat("yyyy-MM-dd");

SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");

ModelMapper mapper = new ModelMapper();

Integer occupationId = 0;

Integer amendId = 0;

try {

if (StringUtils.isBlank(req.getOccupationId())) {

// Save

// Criteria

CriteriaBuilder cb = em.getCriteriaBuilder();

CriteriaQuery<OccupationMaster> query = cb.createQuery(OccupationMaster.class);

// Find

Root<OccupationMaster> c = query.from(OccupationMaster.class);

// Select

query.select(c );

// Order By

List<Order> orderList = new ArrayList<Order>();

orderList.add(cb.desc(c.get("occupationId")));

query.orderBy(orderList);

// Get Result

TypedQuery<OccupationMaster> result = em.createQuery(query);

result.setFirstResult(0);

result.setMaxResults(1);

OccupationMaster data = new OccupationMaster();

data = result.getResultList().get(0);

occupationId = data.getOccupationId() + 1 ;

amendId = 0;

res.setResponse("Saved Successfully ");

} else {

// Update

occupationId = Integer.valueOf(req.getOccupationId());

List<OccupationMaster> findDatas = repo.findByOccupationIdAndEffectiveDateOrderByAmendIdDesc(occupationId ,sdf.parse(req.getEffectiveDate()) );

if (findDatas != null & findDatas.size()>0 ) {

amendId = findDatas.get(0).getAmendId() + 1 ;

} else {

amendId = 0 ;

}

res.setResponse("Updated Successfully ");

}

// Mapping

req.setEffectiveDate(dbf.format(sdf.parse(req.getEffectiveDate())));

saveData = mapper.map(req, OccupationMaster.class );

Date entryDate = new Date();

saveData.setEntryDate(entryDate);

saveData.setOccupationId(occupationId);

saveData.setAmendId(amendId);

repo.save(saveData);

log.info("Saved Details is ---> " + json.toJson(saveData));

} catch (Exception e) {

e.printStackTrace();

log.info("Exception is --->" + e.getMessage());

return null;

}

return res;

}

}

**Repository**

package com.maan.crm.repository;

import java.util.Date;

import java.util.List;

import java.util.Map;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.JpaSpecificationExecutor;

import org.springframework.stereotype.Repository;

import com.maan.crm.bean.OccupationMaster;

import com.maan.crm.bean.OccupationMasterId;

public interface OccupationMasterRepository extends JpaRepository<OccupationMaster, OccupationMasterId> , JpaSpecificationExecutor<OccupationMaster>{

// Occupation Drop Down

List<OccupationMaster> findByStatusOrderByOccupationNameAsc(String status);

// Occupation Get All

//List<OccupationMaster> findAllOrderByEffectiveDateDesc(String limit, String offset);

// Occupation Get By Id

OccupationMaster findByOccupationId(Integer occupationid);

List<OccupationMaster> findByOccupationIdAndEffectiveDateOrderByAmendIdDesc(Integer occupationId, Date parse);

}