LoginController.java

| package com.i2f.training.controller.login;  import com.i2f.training.common.ResultMessage;  import com.i2f.training.model.base.User;  import com.i2f.training.model.vo.TypeVo;  import com.i2f.training.service.login.LoginService;  import org.springframework.web.bind.annotation.PostMapping;  import org.springframework.web.bind.annotation.RequestBody;  import org.springframework.web.bind.annotation.RestController;  import javax.annotation.Resource;  /\*\*  \* @package: com.i2f.training.controller.login  \* @author: 第零组---FengBoTong  \* @date: 2020/3/9 10:44  \* @description: 登录  \*/  @RestController  public class LoginController {  @Resource  LoginService loginService;  @PostMapping("login")  public ResultMessage login(@RequestBody User user){  return loginService.login(user.getMobile(),user.getPassword());  }  @PostMapping("setFingerprintType")  public ResultMessage setFingerprintType(@RequestBody TypeVo typeVo){  return loginService.setFingerprintType(typeVo.getType());  }  @PostMapping("setGestureType")  public ResultMessage setGestureType(@RequestBody TypeVo typeVo){  return loginService.setGestureType(typeVo.getType());  }  } |
| --- |

LoginService.java

| package com.i2f.training.service.login;  import com.i2f.training.common.ResultMessage;  import com.i2f.training.model.vo.LoginVo;  /\*\*  \* @package: com.i2f.training.service.login  \* @author: 第零组---FengBoTong  \* @date: 2020/3/9 9:30  \* @description: 登录  \*/  public interface LoginService {  ResultMessage<LoginVo> login(String mobile,String password);  ResultMessage setGestureType(String type);  ResultMessage setFingerprintType(String type);  } |
| --- |

LoginServiceImpl.java

| package com.i2f.training.service.login;  import com.i2f.training.common.Constants;  import com.i2f.training.common.ResultMessage;  import com.i2f.training.common.exception.TrainingException;  import com.i2f.training.common.util.TokenUtils;  import com.i2f.training.mapper.UserMapper;  import com.i2f.training.model.base.User;  import com.i2f.training.model.example.UserExample;  import com.i2f.training.model.vo.LoginVo;  import org.apache.commons.lang3.StringUtils;  import org.springframework.stereotype.Service;  import org.springframework.web.context.request.RequestContextHolder;  import org.springframework.web.context.request.ServletRequestAttributes;  import javax.annotation.Resource;  import javax.servlet.http.HttpServletRequest;  import java.util.List;  import java.util.UUID;  /\*\*  \* @package: com.i2f.training.serviceImpl.login  \* @author: 第零组---FengBoTong  \* @date: 2020/3/9 9:31  \* @description: 登录  \*/  @Service  public class LoginServiceImpl implements LoginService {  @Resource  UserMapper userMapper;  @Override  public ResultMessage<LoginVo> login(String mobile, String password) {  UserExample userExample = new UserExample();  userExample.createCriteria().andMobileEqualTo(mobile);  List<User> users = userMapper.selectByExample(userExample);  //todo 判断是否存在该用户  if (users.isEmpty()) {  throw new TrainingException(Constants.Code.USER\_NOT\_FOUND,Constants.Meaning.USER\_NOT\_FOUND);  }  //todo 判断该用户密码是否正确  User user = users.get(0);  if (!StringUtils.equals(user.getPassword(),password)) {  throw new TrainingException(Constants.Code.PASS\_NOT\_EXIST,Constants.Meaning.PASS\_NOT\_EXIST);  }  //todo 生成token  String token = UUID.randomUUID().toString().replaceAll("-", "");  TokenUtils.createToken(token,user.getId());  //todo 为VO中参数设值  LoginVo loginVo = new LoginVo();  loginVo.setId(user.getId());  loginVo.setMobile(mobile);  loginVo.setStatus(user.getStatus());  loginVo.setToken(token);  loginVo.setFingerprint(user.getFingerprint());  loginVo.setGesturePassword(user.getGesturePassword());  loginVo.setGesture(user.getGesture());  return new ResultMessage<>(Constants.Code.SUCCESS,Constants.Meaning.SUCCESS,loginVo);  }  @Override  public ResultMessage setGestureType(String type) {  ServletRequestAttributes servletRequestAttributes = (ServletRequestAttributes) RequestContextHolder.currentRequestAttributes();  HttpServletRequest request = servletRequestAttributes.getRequest();  String userId = (String) request.getAttribute("userId");  User user = userMapper.selectByPrimaryKey(userId);  user.setGesture(type);  int i = userMapper.updateByPrimaryKeySelective(user);  return new ResultMessage(Constants.Code.SUCCESS,Constants.Meaning.SUCCESS,"1");  }  @Override  public ResultMessage setFingerprintType(String type) {  ServletRequestAttributes servletRequestAttributes = (ServletRequestAttributes) RequestContextHolder.currentRequestAttributes();  HttpServletRequest request = servletRequestAttributes.getRequest();  String userId = (String) request.getAttribute("userId");  User user = userMapper.selectByPrimaryKey(userId);  user.setFingerprint(type);  int i = userMapper.updateByPrimaryKeySelective(user);  return new ResultMessage(Constants.Code.SUCCESS,Constants.Meaning.SUCCESS,"1");  }  } |
| --- |

FindMobileController.java

| package com.i2f.training.controller.findMoblie;  import com.i2f.training.common.ResultMessage;  import com.i2f.training.model.vo.MobileVo;  import com.i2f.training.service.findMobile.FindMobileService;  import org.springframework.stereotype.Controller;  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.ResponseBody;  import javax.annotation.Resource;  /\*\*  \* @author Yutao Ma  \* @date 2020/3/9  \*/  @Controller  public class FindMobileController {  @Resource  FindMobileService findMobileService;  @PostMapping("findMobile")  @ResponseBody  public ResultMessage findMobile(@RequestBody MobileVo mobile) {  return findMobileService.findMobile(mobile.getMobile());  }  } |
| --- |

FindMobileService.java

| package com.i2f.training.service.findMobile;  import com.i2f.training.common.ResultMessage;  /\*\*  \* @author Yutao Ma  \* @date 2020/3/9  \*/  public interface FindMobileService {  ResultMessage findMobile(String mobile);  } |
| --- |

FindMobileServiceImpl.java

| package com.i2f.training.service.findMobile;  import com.i2f.training.common.Constants;  import com.i2f.training.common.ResultMessage;  import com.i2f.training.mapper.UserMapper;  import com.i2f.training.model.base.User;  import com.i2f.training.model.example.UserExample;  import org.springframework.stereotype.Service;  import javax.annotation.Resource;  import java.util.List;  /\*\*  \* @author Yutao Ma  \* @date 2020/3/9  \*/  @Service  public class FindMobileServiceImpl implements FindMobileService {  @Resource  UserMapper userMapper;  @Override  public ResultMessage findMobile(String mobile) {  UserExample userExample = new UserExample();  userExample.createCriteria().andMobileEqualTo(mobile);  List<User> userList = userMapper.selectByExample(userExample);  if(userList.isEmpty()){  return new ResultMessage(Constants.Code.SUCCESS, Constants.Code.SUCCESS, "0");  }else {  return new ResultMessage("999999", "已存在该手机号", "1");  }  }  } |
| --- |

OpenAccountController.java

| package com.i2f.training.controller.openAccounts;  import com.i2f.training.common.ResultMessage; import com.i2f.training.model.vm.OpenAccountVm; import com.i2f.training.service.openAccounts.OpenAccountService; import org.springframework.web.bind.annotation.PostMapping; import org.springframework.web.bind.annotation.RequestBody; import org.springframework.web.bind.annotation.RestController;  import javax.annotation.Resource;  /\*\*  \* @author LiuDaKa  \* @create 2020-02-18 17:56  \*/ @RestController public class OpenAccountController {   @Resource  OpenAccountService openAccountService;   @PostMapping("openAccount")  public ResultMessage openAccountController(@RequestBody OpenAccountVm openAccountVm) {  return openAccountService.openAccount(openAccountVm.getRealName(), openAccountVm.getIdCard(),openAccountVm.getPassword(),openAccountVm.getConfirmPassword());  } } |
| --- |

OpenAccountService.java

| package com.i2f.training.service.openAccounts; //改过包名  import com.i2f.training.common.ResultMessage;  /\*\*  \* @author LiuDaKa  \* @create 2020-02-19 7:53  \*/  public interface OpenAccountService {  ResultMessage openAccount(String realName, String idCard, String password, String confirmPassword);   } |
| --- |

OpenAccountServiceImpl.java

| package com.i2f.training.service.openAccounts;  import com.i2f.training.client.CheckIdCardAndNameClient; import com.i2f.training.client.OpenAccountClient; import com.i2f.training.common.Constants; import com.i2f.training.common.ResultMessage; import com.i2f.training.common.exception.TrainingException; import com.i2f.training.mapper.OpenAccountSerialMapper; import com.i2f.training.mapper.UserMapper; import com.i2f.training.model.base.OpenAccountSerial; import com.i2f.training.model.base.User; import lombok.extern.slf4j.Slf4j; import org.apache.commons.lang3.StringUtils; import org.springframework.stereotype.Service; import org.springframework.transaction.annotation.Transactional; import org.springframework.util.DigestUtils; import org.springframework.web.context.request.RequestContextHolder; import org.springframework.web.context.request.ServletRequestAttributes; import javax.annotation.Resource; import javax.servlet.http.HttpServletRequest; import java.util.Date; import java.util.Map; import java.util.UUID;  /\*\*  \* @author LiuDaKa  \* @create 2020-02-19 10:31  \*/ @Slf4j @Service public class OpenAccountServiceImpl implements OpenAccountService {  @Resource  CheckIdCardAndNameClient checkIdCardAndNameClient;  @Resource  OpenAccountClient openAccountClient;  @Resource  OpenAccountSerialMapper openAccountSerialMapper;  @Resource  UserMapper sUserMapper;  @Transactional(rollbackFor = Exception.class)  @Override  public ResultMessage openAccount(String realName, String idCard, String confirmPassword, String password) {  //获取userid  ServletRequestAttributes servletRequestAttributes = (ServletRequestAttributes) RequestContextHolder.currentRequestAttributes();  HttpServletRequest request = servletRequestAttributes.getRequest();  String userId = (String) request.getAttribute("userId");  User user = sUserMapper.selectByPrimaryKey(userId);  //查询用户密码和电话号码  //String userPassword = user.getPassword();  String userMobile = user.getMobile();  //判断两次输入的密码  if (!StringUtils.equals(password,confirmPassword)){  throw new TrainingException(Constants.Code.PASSWORD\_ARE\_INCONSISTENT,Constants.Meaning.PASSWORD\_ARE\_INCONSISTENT);  }    Map<String, String> stringMap1 = null;  //获取serialid  String serialId = UUID.randomUUID().toString().replaceAll("-", "");   stringMap1 = checkIdCardAndNameClient.openAccountClient(serialId, realName, idCard,userMobile);  //验证签名  String status1 = stringMap1.get("status");   //0继续往下执行  if (StringUtils.equals(status1,"1")) {  throw new TrainingException(Constants.Code.ID\_CARD\_NOT\_MATCHING,Constants.Meaning.ID\_CARD\_NOT\_MATCHING);  }  if (StringUtils.equals(status1,"2")) {  throw new TrainingException(Constants.Code.NAME\_NOT\_MATCHING,Constants.Meaning.NAME\_NOT\_MATCHING);  }  //异常请求  if (StringUtils.equals(status1,"3")) {  throw new TrainingException(Constants.Code.OPENACCOUNT\_SIGN\_VER\_FAIL,Constants.Meaning.OPENACCOUNT\_SIGN\_VER\_FAIL);  }  //用户已经开过户  if (StringUtils.equals(status1,"4")) {  throw new TrainingException(Constants.Code.USER\_EXIST,Constants.Meaning.USER\_EXIST);  }  OpenAccountSerial openAccountSerial1 = new OpenAccountSerial();  openAccountSerial1.setId(serialId);  openAccountSerial1.setIdCard(idCard);  openAccountSerial1.setRealName(realName);  openAccountSerial1.setStatus("1");  openAccountSerial1.setUserId(userId);  openAccountSerial1.setTime(new Date());  openAccountSerial1.setSystem("公安系统");  openAccountSerialMapper.insertSelective(openAccountSerial1);   Map<String, String> stringMap = null;  String serialId2 = UUID.randomUUID().toString().replaceAll("-", "");   stringMap = openAccountClient.openAccountClient(serialId2,realName,userMobile,idCard,userId, password, confirmPassword);  String status = stringMap.get("status");  String accountNum = stringMap.get("accountNum");  String timeStampbank1 = stringMap.get("timeStampbank1");  String autographbank1 = stringMap.get("autographbank1");   //核心系统交互成功用户表插入数据  User suser = sUserMapper.selectByPrimaryKey(userId);  suser.setStatus(status);  suser.setEAccountNum(accountNum);  sUserMapper.updateByPrimaryKeySelective(suser);    //核心交互成功向开户流水表插入记录  OpenAccountSerial openAccountSerial = new OpenAccountSerial();  openAccountSerial.setId(serialId);  openAccountSerial.setUserId(userId);  openAccountSerial.setIdCard(idCard);  openAccountSerial.setRealName(realName);  openAccountSerial.setStatus("1");  openAccountSerial.setTime(new Date());  openAccountSerial.setSystem("互联网核心系统");  //openAccountSerialMapper.insertSelective(openAccountSerial);  return new ResultMessage(Constants.Code.SUCCESS, Constants.Meaning.SUCCESS, "1");   } } |
| --- |

RegisterController.java

| package com.i2f.training.controller.register;  import com.i2f.training.common.ResultMessage; import com.i2f.training.model.vm.RegisterVm; import com.i2f.training.service.register.RegisterService; import org.springframework.stereotype.Controller; import org.springframework.web.bind.annotation.PostMapping; import org.springframework.web.bind.annotation.RequestBody; import org.springframework.web.bind.annotation.RestController;  import javax.annotation.Resource;  /\*\*  \* @author xwen  \* date 2020-03-09  \*/ @RestController public class RegisterController {  @Resource  RegisterService registerService;  @PostMapping("register")  public ResultMessage registerController(@RequestBody RegisterVm registerVm){  return registerService.register(registerVm.getMobile(),registerVm.getConfirmPassword(),registerVm.getPassword(),registerVm.getVerificationCode());  } } |
| --- |

RegisterService.java

| package com.i2f.training.service.register;  import com.i2f.training.common.ResultMessage; import com.i2f.training.model.vo.Status;  /\*\*  \* @author xwen  \* date 2020-03-09  \*/  public interface RegisterService {  /\*\*  \* 注册方法定义  \*/  ResultMessage register(String mobile, String password,String confirmPassword, String verificationCode) ; } |
| --- |

RegisterServiceImpl.java

| package com.i2f.training.service.register;  import com.i2f.training.common.Constants; import com.i2f.training.common.ResultMessage; import com.i2f.training.common.exception.TrainingException; import com.i2f.training.common.util.TokenUtils; import com.i2f.training.mapper.UserMapper; import com.i2f.training.model.base.User; import com.i2f.training.model.example.UserExample; import com.i2f.training.model.vo.Status; import lombok.extern.slf4j.Slf4j; import org.apache.commons.lang3.StringUtils; import org.springframework.stereotype.Service;  import javax.annotation.Resource; import java.util.Date; import java.util.List; import java.util.UUID;  /\*\*  \* @author xwen  \* date 2020-03-09  \*/ @Service @Slf4j public class RegisterServiceImpl implements RegisterService {  @Resource  UserMapper userMapper;   @Override  public ResultMessage register(String mobile, String password,String confirmPassword, String verificationCode) {  UserExample userExample = new UserExample();  // TODO: 判断用户输入的验证码是否正确  String verificationCodeByMobile = TokenUtils.getVerificationCodeByMobile(mobile);  //todo:删除验证码  TokenUtils.deleteVerificationCodeByMobile(mobile);  if(verificationCodeByMobile.isEmpty()){  // TODO: 2020/3/9 如果验证码为空则抛出验证码发送失败异常  throw new TrainingException(Constants.Code.VERIFICATION\_SEND\_ERROR,Constants.Meaning.VERIFICATION\_SEND\_ERROR);  }  if(!StringUtils.equals(verificationCode,verificationCodeByMobile)){  //todo：抛出验证码错误状态  throw new TrainingException(Constants.Code.VERIFICATION\_CODE\_ERROR,Constants.Meaning.VERIFICATION\_CODE\_ERROR);  };  // TODO: 2020/3/9 重复核验两次输入的密码是否一致防止中途篡改  if(!StringUtils.equals(password,confirmPassword)){  throw new TrainingException(Constants.Code.PASSWORD\_ARE\_INCONSISTENT,Constants.Meaning.PASSWORD\_ARE\_INCONSISTENT);  }  //todo:判断用户是否注册，如果注册了则不许用户注册  userExample.createCriteria().andMobileEqualTo(mobile);  List<User> users = userMapper.selectByExample(userExample);  if(!users.isEmpty()){  throw new TrainingException(Constants.Code.HVAE\_REGISTER,Constants.Meaning.HVAE\_REGISTER);  }   // TODO: 2020/3/9 终于可以注册了，在user表中插入相应数据  User user = new User();  user.setMobile(mobile);  user.setCreateTime(new Date());  user.setPassword(password);  user.setStatus("0");  user.setId(UUID.randomUUID().toString().replaceAll("-", ""));  userMapper.insertSelective(user);  Status status = new Status();  status.setStatus("1");  return new ResultMessage(Constants.Code.SUCCESS,Constants.Meaning.REGISTER\_SUCCESS,status);  } } |
| --- |

TieCardController.java

| package com.i2f.training.controller.tieCards; //改过包名 import com.i2f.training.common.ResultMessage; import com.i2f.training.model.vm.CardVm; import com.i2f.training.model.vm.TieCardVm; import com.i2f.training.model.vo.OutLoginMsgVo; import com.i2f.training.service.tieCards.CheckCardService; import com.i2f.training.service.tieCards.TieCardService; import org.jetbrains.annotations.NotNull; import org.springframework.web.bind.annotation.PostMapping; import org.springframework.web.bind.annotation.RequestBody; import org.springframework.web.bind.annotation.RestController;  import javax.annotation.Resource;  /\*\*  \* @author  \* date 2020-02-19  \*/ @RestController public class TieCardController {  @Resource  TieCardService tieCardService;  @Resource  CheckCardService checkCardService;  @PostMapping("tiecard")  public ResultMessage<OutLoginMsgVo> login(@NotNull @RequestBody TieCardVm tieCardVm){  return tieCardService.tieCard(tieCardVm.getName(),tieCardVm.getIdCard(), tieCardVm.getBankCardNum(),tieCardVm.getMobile(),tieCardVm.getVerificationCode());  }  @PostMapping("check")  public ResultMessage check(@RequestBody CardVm cardVm){  return checkCardService.check(cardVm);  }  } |
| --- |

TieCardService.java

| package com.i2f.training.service.tieCards; //改过包名 import com.i2f.training.common.ResultMessage;  /\*\*  \* @author xwen  \* date 2020-02-19  \*/ //todo:定义绑卡的方法 public interface TieCardService {  ResultMessage tieCard(String name, String idCard, String bankCardNum, String mobile, String verificationCode);   } |
| --- |

TieCardServiceImpl.java

| package com.i2f.training.service.tieCards;  import com.i2f.training.client.TieCardClient; import com.i2f.training.common.Constants; import com.i2f.training.common.ResultMessage; import com.i2f.training.common.exception.TrainingException; import com.i2f.training.common.util.TokenUtils; import com.i2f.training.mapper.TieCardSerialMapper; import com.i2f.training.mapper.UserMapper; import com.i2f.training.model.base.TieCardSerial; import com.i2f.training.model.base.User; import lombok.extern.slf4j.Slf4j; import org.apache.commons.lang3.StringUtils; import org.springframework.stereotype.Service; import org.springframework.util.DigestUtils; import org.springframework.web.context.request.RequestContextHolder; import org.springframework.web.context.request.ServletRequestAttributes;  import javax.annotation.Resource; import javax.servlet.http.HttpServletRequest; import java.util.Date; import java.util.Map; import java.util.UUID;  /\*\*  \* @author date 2020-02-19  \*/ @Service @Slf4j public class TieCardServiceImpl implements TieCardService {  @Resource  UserMapper sUserMapper;  @Resource  TieCardSerialMapper tieCardSerialMapper;  @Resource  TieCardClient tieCardClient;   @Override  public ResultMessage tieCard(String name, String idCard, String bankCardNum, String mobile, String verificationCode) {  //todo：银行互联网核心系统根据虚拟银行系统请求报文的银行卡号查找银行卡信息，若存在该银行卡信息，则进行持卡人身份认证  TieCardSerial tieCardSerial = new TieCardSerial();  //todo 比对验证码  String verificationCodeByMobile = TokenUtils.getVerificationCodeByMobile(mobile);  if (!StringUtils.equals(verificationCodeByMobile, verificationCode)) {  throw new TrainingException(Constants.Code.VERIFICATION\_CODE\_ERROR, Constants.Meaning.VERIFICATION\_CODE\_ERROR);  }  //todo 删除验证码  TokenUtils.deleteVerificationCodeByMobile(mobile);   //todo:获取userid  ServletRequestAttributes servletRequestAttributes = (ServletRequestAttributes) RequestContextHolder.currentRequestAttributes();  HttpServletRequest request = servletRequestAttributes.getRequest();  String userId = (String) request.getAttribute("userId");  //userId = "047831";  User sUser = sUserMapper.selectByPrimaryKey(userId);  //todo:获取二类户id  String accountNum = sUser.getEAccountNum();  //todo:查看状态如果为0则用户还未开户，抛出信息  String status1 = sUser.getStatus();  if (StringUtils.equals(status1, "0")) {  throw new TrainingException(Constants.Code.NOT\_YET\_OPEN\_ACCOUNT, Constants.Meaning.NOT\_YET\_OPEN\_ACCOUNT);  }  String serialId = UUID.randomUUID().toString().replaceAll("-", "");    Map<String, String> stringStringMap = null;  //时间戳  String timeStamp1 = String.valueOf(System.currentTimeMillis());   //生成签名并MD5加密  String preAutograph = timeStamp1 + "&" + serialId + "&" + name + "&" + idCard + "&" + accountNum + "&" + bankCardNum + "&" + mobile;  String autograph = DigestUtils.md5DigestAsHex(preAutograph.getBytes());   try {  stringStringMap = tieCardClient.tieCard(serialId, name, idCard, accountNum, bankCardNum, mobile, autograph, timeStamp1);  } catch (Exception e) {  tieCardSerial.setId(UUID.randomUUID().toString().replaceAll("-", ""));  tieCardSerial.setUserId(userId);  tieCardSerial.setCardNum(bankCardNum);  tieCardSerial.seteAccountNum(accountNum);  tieCardSerial.setSystem("互联网核心系统");  tieCardSerial.setTime(new Date());  tieCardSerial.setStatus("0");  tieCardSerialMapper.insert(tieCardSerial);  throw new TrainingException(Constants.Code.TIECARD\_SIGN\_VER\_FAIL, Constants.Meaning.TIECARD\_SIGN\_VER\_FAIL);  }  String status = stringStringMap.get("status");  String bankCardNums = stringStringMap.get("bankCardNums");  String timeStamp2 = stringStringMap.get("timeStamp2");  String autograph2 = stringStringMap.get("autograph2");  //生成签名并MD5加密  /\*String backAutograph = timeStamp2 + "&" + status + "&" + bankCardNums;  String checkAutograph = DigestUtils.md5DigestAsHex(backAutograph.getBytes());  //验签  if (!StringUtils.equals(checkAutograph,autograph2)){  throw new TrainingException(Constants.Code.TIECARD\_SIGN\_VER\_FAIL,Constants.Meaning.TIECARD\_SIGN\_VER\_FAIL);  }\*/ // 1成功、2银行卡不存在、3姓名不匹配、4身份证号不匹配、 // 5预留手机号不匹配（与银行卡比较）、6该用户已绑定该卡、7开户信息不匹配，8异常请求   if (StringUtils.equals(status, "1")) {  //todo:绑卡成功后将用户状态码设置为2  sUser.setStatus("2");  sUser.setBankCardNum(bankCardNums);  sUserMapper.updateByPrimaryKeySelective(sUser);  create(bankCardNum, tieCardSerial, userId, accountNum);  return new ResultMessage(Constants.Code.SUCCESS, Constants.Meaning.SUCCESS, status);  }  if (StringUtils.equals(status, "2")) {  create(bankCardNum, tieCardSerial, userId, accountNum);  throw new TrainingException(Constants.Code.BANK\_CARD\_NOT\_EXIST, Constants.Meaning.BANK\_CARD\_NOT\_EXIST);  }  if (StringUtils.equals(status, "3")) {  create(bankCardNum, tieCardSerial, userId, accountNum);  throw new TrainingException(Constants.Code.NAME\_NOT\_MATCHING, Constants.Meaning.NAME\_NOT\_MATCHING);  }  if (StringUtils.equals(status, "4")) {  create(bankCardNum, tieCardSerial, userId, accountNum);  throw new TrainingException(Constants.Code.ID\_CARD\_NOT\_MATCHING, Constants.Meaning.ID\_CARD\_NOT\_MATCHING);  }  if (StringUtils.equals(status, "5")) {  create(bankCardNum, tieCardSerial, userId, accountNum);  throw new TrainingException(Constants.Code.MOBILE\_NOT\_MATCHING, Constants.Meaning.MOBILE\_NOT\_MATCHING);  }  if (StringUtils.equals(status, "6")) {  create(bankCardNum, tieCardSerial, userId, accountNum);  throw new TrainingException(Constants.Code.CARD\_YET\_TIED, Constants.Meaning.CARD\_YET\_TIED);  }  if (StringUtils.equals(status, "7")) {  create(bankCardNum, tieCardSerial, userId, accountNum);  throw new TrainingException(Constants.Code.OPENACCOUNTMSG\_NOT\_TIRCARD, Constants.Meaning.OPENACCOUNTMSG\_NOT\_TIRCARD);  }  if (StringUtils.equals(status, "8")) {  create(bankCardNum, tieCardSerial, userId, accountNum);  throw new TrainingException(Constants.Code.TIECARD\_SIGN\_VER\_FAIL, Constants.Meaning.TIECARD\_SIGN\_VER\_FAIL);  }  //todo:插入流水    return new ResultMessage(Constants.Code.SUCCESS, Constants.Meaning.SUCCESS, "1");  }   private void create(String bankCardNum, TieCardSerial tieCardSerial, String userId, String eAccountNum) {  tieCardSerial.setId(UUID.randomUUID().toString().replaceAll("-", ""));  tieCardSerial.setUserId(userId);  tieCardSerial.setCardNum(bankCardNum);  tieCardSerial.seteAccountNum(eAccountNum);  tieCardSerial.setSystem("互联网核心系统");  tieCardSerial.setTime(new Date());  tieCardSerial.setStatus("1");  tieCardSerialMapper.insert(tieCardSerial);  }  } |
| --- |

FindProductController.java

| package com.i2f.training.controller.products;  import com.i2f.training.common.ResultMessage; import com.i2f.training.model.vm.FindByPagingVm; import com.i2f.training.model.vo.ProductsListVo; import com.i2f.training.service.products.FindProductService; import org.springframework.web.bind.annotation.\*;  import javax.annotation.Resource; import java.util.List;  /\*\*  \* 查询所有产品Controller  \*  \* @package: com.i2f.training.controller.products  \* @author: 第零组---XuQ  \* @date: 2020/3/9 12:05  \*/ @RestController @RequestMapping("product") public class FindProductController {   @Resource  FindProductService findProductService;   /\*\*  \* 查询所有产品  \*  \* @param  \* @return com.i2f.training.common.ResultMessage  \* @date 2020/3/9 13:25  \* @author XuQ  \*/  @GetMapping("find")  public ResultMessage<List<ProductsListVo>> findFinanceProductController() {  return findProductService.findFinanceProductList();  }   /\*\*  \* 分页查询产品  \*  \* @param findByPagingVm 页数，默认第一页  \* @return com.i2f.training.common.ResultMessage<com.github.pagehelper.PageInfo < com.i2f.training.model.base.FinanceProduct>>  \* @date 2020/3/9 12:19  \* @author XuQ  \*/  @PostMapping("findByPaging")  public ResultMessage findByPaging(@RequestBody FindByPagingVm findByPagingVm) {  int size = 6;  int pageInt = Integer.parseInt(findByPagingVm.getPage());  return findProductService.findByPaging(pageInt, size);  } } |
| --- |

FindProductService.java

| package com.i2f.training.service.products;  import com.i2f.training.common.ResultMessage; import com.i2f.training.model.vo.ProductsListVo;  import java.util.List;  /\*\*  \* 查找所有申购记录service层接口  \*  \* @package: com.i2f.training.service.products  \* @author: 第零组---XuQ  \* @date: 2020/3/9 11:51  \*/ public interface FindProductService {   /\*\*  \* 一次查询出所有产品  \*  \* @param  \* @return com.i2f.training.common.ResultMessage<java.util.List<com.i2f.training.model.base.FinanceProduct>>  \* @date 2020/3/9 12:15  \* @author XuQ  \*/  ResultMessage<List<ProductsListVo>> findFinanceProductList();  /\*\*  \* 分页查询所有数据  \*  \* @param page 页数  \* @param size 页大小  \* @return com.i2f.training.common.ResultMessage<com.github.pagehelper.PageInfo<com.i2f.training.model.base.FinanceProduct>>  \* @date 2020/3/9 12:17  \* @author XuQ  \*/  ResultMessage findByPaging(int page, int size); } |
| --- |

FindProductServiceImpl.java

| package com.i2f.training.service.products;  import com.github.pagehelper.PageHelper; import com.github.pagehelper.PageInfo; import com.i2f.training.common.Constants; import com.i2f.training.common.ResultMessage; import com.i2f.training.mapper.FinanceProductMapper; import com.i2f.training.model.base.FinanceProduct; import com.i2f.training.model.example.FinanceProductExample; import com.i2f.training.model.vo.ProductsListVo; import org.springframework.stereotype.Service;  import javax.annotation.Resource; import java.math.BigDecimal; import java.text.SimpleDateFormat; import java.util.ArrayList; import java.util.List;  /\*\*  \* @package: com.i2f.training.service.products  \* @author: 第零组---XuQ  \* @date: 2020/3/9 12:01  \*/ @Service public class FindProductServiceImpl implements FindProductService {  @Resource  FinanceProductMapper financeProductMapper;   @Override  public ResultMessage<List<ProductsListVo>> findFinanceProductList() {  List<ProductsListVo> productsListVoList = new ArrayList<>();  FinanceProductExample financeProductExample = new FinanceProductExample();  financeProductExample.createCriteria().andIdIsNotNull().andStatusNotEqualTo("2");  List<FinanceProduct> financeProductList = financeProductMapper.selectByExample(financeProductExample);  findProduct(productsListVoList, financeProductList);  return new ResultMessage<>(Constants.Code.SUCCESS, Constants.Meaning.SUCCESS, productsListVoList);  }    @Override  public ResultMessage findByPaging(int page, int size) {  List<ProductsListVo> productsListVoList = new ArrayList<>();  FinanceProductExample financeProductExample = new FinanceProductExample();  financeProductExample.createCriteria().andIdIsNotNull().andStatusNotEqualTo("2");  PageHelper.startPage(page, size);  List<FinanceProduct> financeProductList = financeProductMapper.selectByExample(financeProductExample);  PageInfo pageInfo = new PageInfo(financeProductList);  findProduct(productsListVoList, pageInfo.getList());  pageInfo.setList(productsListVoList);  return new ResultMessage(Constants.Code.SUCCESS, Constants.Meaning.SUCCESS, pageInfo);  }   private void findProduct(List<ProductsListVo> productsListVoList, List<FinanceProduct> financeProductList) {  ProductsListVo productsListVo;  for (FinanceProduct p : financeProductList) {  productsListVo = new ProductsListVo();  productsListVo.setId(p.getId());  productsListVo.setName(p.getName());  String rateReturn = p.getRateReturn();  String[] split = rateReturn.split("-");  int length = split.length;  String nowRate = split[length - 1];  productsListVo.setRateReturn(nowRate);  SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyy-MM-dd");  productsListVo.setEndDate(simpleDateFormat.format(p.getEndDate()));  productsListVo.setStartDate(simpleDateFormat.format(p.getStartDate()));  productsListVo.setType(p.getType());  productsListVo.setPurchaseAmount(p.getPurchaseAmount());  productsListVo.setStatus(p.getStatus());  if (length == 1) {  productsListVo.setGain(nowRate);  } else {  String yesterday = split[length - 2];  BigDecimal subtract = new BigDecimal(nowRate).subtract(new BigDecimal(yesterday));  productsListVo.setGain(subtract.toString());  }  productsListVo.setLogo(p.getLogo());  productsListVoList.add(productsListVo);  }  } } |
| --- |

FindProductDetailsController.java

| package com.i2f.training.controller.products;  import com.i2f.training.common.ResultMessage; import com.i2f.training.model.vm.FinanceProductIdVm; import com.i2f.training.service.products.FindFinanceProductDetailService; 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 javax.annotation.Resource;  /\*\*  \* @author  \* date 2020-02-20  \*/ @RestController public class FindProductDetailsController {  @Resource  FindFinanceProductDetailService findFinanceProductService;   @PostMapping("findProductDetails")  public ResultMessage findProductDetails(@RequestBody FinanceProductIdVm financeProductId) {  return findFinanceProductService.findProductDetailsService(financeProductId.getFinanceProductId());  } } |
| --- |

FindFinanceProductDetailService.java

| package com.i2f.training.service.products;  import com.i2f.training.common.ResultMessage;  import com.i2f.training.model.vo.FindProductVo;  public interface FindFinanceProductDetailService {    ResultMessage<FindProductVo> findProductDetailsService(String financeProductId);  } |
| --- |

FindFinanceProductDetailServiceImpl.java

| package com.i2f.training.service.products;  import com.i2f.training.common.Constants; import com.i2f.training.common.ResultMessage; import com.i2f.training.mapper.FinanceProductMapper; import com.i2f.training.model.base.FinanceProduct; import com.i2f.training.model.vo.FindProductVo; import lombok.extern.slf4j.Slf4j; import org.springframework.stereotype.Service;  import javax.annotation.Resource; import java.math.BigDecimal; import java.text.SimpleDateFormat; import java.util.ArrayList; import java.util.Calendar; import java.util.Date; import java.util.List;  @Service @Slf4j public class FindFinanceProductDetailServiceImpl implements FindFinanceProductDetailService {  @Resource  FinanceProductMapper sFinanceProductMapper;    /\*\*  \* @author liudaka  \* @date  \*/   @Override  public ResultMessage<FindProductVo> findProductDetailsService(String financeProductId) {  FinanceProduct sFinanceProduct = sFinanceProductMapper.selectByPrimaryKey(financeProductId);  //todo:获取userId  //日期方法  String str = sFinanceProduct.getRateReturn();    List x = getX(7);  List y = getY(7, str);   String[] strArray = str.split("-");  BigDecimal todayIncome = new BigDecimal(strArray[6].trim());  BigDecimal yesterdayIncome = new BigDecimal(strArray[5].trim());  BigDecimal inComeChange = todayIncome.subtract(yesterdayIncome);  String incomeChanges = inComeChange.toString();   sFinanceProduct.setRateReturn(strArray[6].trim());  String rateReturn = sFinanceProduct.getRateReturn();  String name = sFinanceProduct.getName();  String type = sFinanceProduct.getType();  String id = sFinanceProduct.getId();  String purchaseAmount = sFinanceProduct.getPurchaseAmount();  int holdDay = sFinanceProduct.getHoldDay();  String inventoryAmount = sFinanceProduct.getInventoryAmount();  String logo = sFinanceProduct.getLogo();  String status = sFinanceProduct.getStatus();  Date endDate = sFinanceProduct.getEndDate();  //日期进行标准化  Date startDate = sFinanceProduct.getStartDate();  String serviceCharge = sFinanceProduct.getServiceCharge();  String riskType = sFinanceProduct.getRiskType();  String interestStatus = sFinanceProduct.getInterestStatus();  String description = sFinanceProduct.getDescription();  FindProductVo findProductVos = new FindProductVo(x, y, incomeChanges,id, name, rateReturn, inventoryAmount, interestStatus, serviceCharge, riskType, purchaseAmount, description, logo, holdDay, status, type, startDate, endDate);  /\*List<FindProductVo> findProductVoList = new ArrayList<>();  findProductVoList.add(findProductVos);  FindProductDetailVo findProductDetailVo = new FindProductDetailVo(x, y, incomeChanges, findProductVoList);\*/   return new ResultMessage<>(Constants.Code.SUCCESS, Constants.Meaning.SUCCESS, findProductVos);    }   private List getX(int xy) {   List x = null;  x = new ArrayList();  for (int i = 0; i < xy; i++) {  Calendar calendar = Calendar.getInstance();  calendar.set(Calendar.DAY\_OF\_YEAR, calendar.get(Calendar.DAY\_OF\_YEAR) - 6 + i);  Date today = calendar.getTime();  SimpleDateFormat format = new SimpleDateFormat("MM-dd");  String result = format.format(today);  x.add(result);  //return x;  }  return x;  }   private List getY(int xy, String str) {  List y = null;  y = new ArrayList();  for (int i = 0; i < 7; i++) {  String[] strArray = str.split("-");  y.add(strArray[i]);  //y.add(strArray[i].trim());  }  return y;  }  } |
| --- |

PurchasedController.java

| package com.i2f.training.controller.purchaseOrder;  import com.i2f.training.common.Constants; import com.i2f.training.common.ResultMessage; import com.i2f.training.model.vm.PurchasedDetailVm; import com.i2f.training.model.vo.PurchasedDetailVo; import com.i2f.training.model.vo.TotalPurchasedVo; import com.i2f.training.service.purchaseOrder.FindPurchaseOrderService; import com.i2f.training.service.purchaseOrder.PurchaseDetailService; import org.springframework.web.bind.annotation.\*;  import javax.annotation.Resource;  /\*\*  \* 已购页面需要的消息处理Controller  \*  \* @package: com.i2f.training.controller.purchaseOrder  \* @author: 第零组---XuQ  \* @date: 2020/3/10 19:05  \*/ @RestController @RequestMapping("order") public class PurchasedController {  @Resource  FindPurchaseOrderService findPurchaseOrderService;  @Resource  PurchaseDetailService purchaseDetailService;  @GetMapping("purchased")  public ResultMessage<TotalPurchasedVo> purchased(){  return new ResultMessage<>(Constants.Code.SUCCESS,Constants.Meaning.SUCCESS,findPurchaseOrderService.purchased());  }  @PostMapping("purchasedDetail")  public ResultMessage<PurchasedDetailVo> detailPurchased(@RequestBody PurchasedDetailVm purchasedDetailVm){  return new ResultMessage<>(Constants.Code.SUCCESS,Constants.Meaning.SUCCESS,purchaseDetailService.purchasedDetail(purchasedDetailVm.getPurchaseOrderId()));  } } |
| --- |

FindPurchaseOrderService.java

| package com.i2f.training.service.purchaseOrder;  import com.i2f.training.model.vo.TotalPurchasedVo;  /\*\*  \* 查询所有申购记录的service层接口  \*  \* @package: com.i2f.training.service.products  \* @author: 第零组---XuQ  \* @date: 2020/3/10 17:07  \* holding  \* yesterdayTotalReturn   \*/ public interface FindPurchaseOrderService {  /\*\*  \* 已购页面时，返回昨日总收益和总收益  \*  \* @param  \* @return com.i2f.training.model.vo.PurchasedVo  \* @date 2020/3/10 17:24  \* @author XuQ  \*/  TotalPurchasedVo purchased(); } |
| --- |

FindPurchaseOrderServiceImpl.java

| package com.i2f.training.service.purchaseOrder;  import com.i2f.training.mapper.FinanceProductMapper; import com.i2f.training.mapper.PurchaseOrderMapper; import com.i2f.training.mapper.UserMapper; import com.i2f.training.model.base.FinanceProduct; import com.i2f.training.model.base.PurchaseOrder; import com.i2f.training.model.base.User; import com.i2f.training.model.example.PurchaseOrderExample; import com.i2f.training.model.vo.PurchaseOrderVo; import com.i2f.training.model.vo.TotalPurchasedVo; import org.springframework.stereotype.Service; import org.springframework.web.context.request.RequestContextHolder; import org.springframework.web.context.request.ServletRequestAttributes;  import javax.annotation.Resource; import javax.servlet.http.HttpServletRequest; import java.math.BigDecimal; import java.util.ArrayList; import java.util.List;  /\*\*  \* 已购页面  \*  \* @package: com.i2f.training.service.products  \* @author: 第零组---XuQ  \* @date: 2020/3/10 17:24  \* <p>  \* holding 总持仓 String  \* yesterdayTotalReturn 昨日收益 String  \* <p>  \* purchaseOrderList  \* gain 涨幅 String  \* name 产品名 String  \* rateReturn 收益率 String  \* id 申购记录表中该记录的id String  \* type 产品类型 String  \* yesterdayReturn 昨日已收益金额 String  \*/ @Service public class FindPurchaseOrderServiceImpl implements FindPurchaseOrderService {  @Resource  PurchaseOrderMapper purchaseOrderMapper;  @Resource  FinanceProductMapper financeProductMapper;  @Resource  UserMapper userMapper;   @Override  public TotalPurchasedVo purchased() {  // TODO: 2020/3/10 得知具体用户  HttpServletRequest request = ((ServletRequestAttributes) RequestContextHolder.currentRequestAttributes()).getRequest();  String userId = (String) request.getAttribute("userId");  User user = userMapper.selectByPrimaryKey(userId);  String eAccountNum = user.getEAccountNum();  // TODO: 2020/3/10 获取该用户的所有已购的未赎回的记录  PurchaseOrderExample purchaseOrderExample = new PurchaseOrderExample();  purchaseOrderExample.createCriteria().andEAccountNumEqualTo(eAccountNum).andStatusEqualTo("0");  List<PurchaseOrder> purchaseOrders = purchaseOrderMapper.selectByExample(purchaseOrderExample);  // TODO: 2020/3/10 根据已购记录得到产品信息  //总持仓  BigDecimal holding = BigDecimal.ZERO;  //昨日总收益  BigDecimal yesterdayTotalReturn = BigDecimal.ZERO;  PurchaseOrderVo purchaseOrderVo = null;  TotalPurchasedVo totalPurchasedVo = new TotalPurchasedVo();  List<PurchaseOrderVo> purchaseOrderVoList = new ArrayList<>();   for (PurchaseOrder order : purchaseOrders) {  String totalHolding = order.getTotalHolding();  holding = holding.add(new BigDecimal(totalHolding));  yesterdayTotalReturn = yesterdayTotalReturn.add(new BigDecimal(order.getRetain()));   purchaseOrderVo = new PurchaseOrderVo();  FinanceProduct financeProduct = financeProductMapper.selectByPrimaryKey(order.getProductId());  String[] split = financeProduct.getRateReturn().split("-");  if (split.length < 2) {  purchaseOrderVo.setGain(split[split.length - 1]);  } else {  purchaseOrderVo.setGain((new BigDecimal(split[split.length - 1]).subtract(new BigDecimal(split[split.length - 2]))).toString());  }  purchaseOrderVo.setName(financeProduct.getName());  purchaseOrderVo.setRateReturn(split[split.length - 1]);  purchaseOrderVo.setId(order.getId());  //status为0时开售产品已审购  purchaseOrderVo.setStatus("0");  purchaseOrderVo.setType(financeProduct.getType());  purchaseOrderVo.setYesterdayReturn(order.getRetain());  purchaseOrderVo.setAmount(totalHolding);  purchaseOrderVoList.add(purchaseOrderVo);  }  PurchaseOrderExample purchaseOrderExample1 = new PurchaseOrderExample();  purchaseOrderExample1.createCriteria().andEAccountNumEqualTo(eAccountNum).andStatusEqualTo("2");  List<PurchaseOrder> purchaseOrders1 = purchaseOrderMapper.selectByExample(purchaseOrderExample1);  for (PurchaseOrder order : purchaseOrders1) {  String totalHolding = order.getTotalHolding();  holding = holding.add(new BigDecimal(totalHolding));  yesterdayTotalReturn = yesterdayTotalReturn.add(new BigDecimal(order.getRetain()));   purchaseOrderVo = new PurchaseOrderVo();  FinanceProduct financeProduct = financeProductMapper.selectByPrimaryKey(order.getProductId());  String[] split = financeProduct.getRateReturn().split("-");  if (split.length < 2) {  purchaseOrderVo.setGain(split[split.length - 1]);  } else {  purchaseOrderVo.setGain((new BigDecimal(split[split.length - 1]).subtract(new BigDecimal(split[split.length - 2]))).toString());  }  purchaseOrderVo.setName(financeProduct.getName());  purchaseOrderVo.setRateReturn(split[split.length - 1]);  purchaseOrderVo.setId(order.getId());  //status为2时开售产品已审购  purchaseOrderVo.setStatus("2");  purchaseOrderVo.setType(financeProduct.getType());  purchaseOrderVo.setYesterdayReturn(order.getRetain());  purchaseOrderVo.setAmount(totalHolding);  purchaseOrderVoList.add(purchaseOrderVo);  }     // TODO: 2020/3/10 返回数据  totalPurchasedVo.setHolding(holding.toString());  totalPurchasedVo.setYesterdayTotalReturn(yesterdayTotalReturn.toString());  totalPurchasedVo.setPurchaseOrderList(purchaseOrderVoList);  return totalPurchasedVo;  } } |
| --- |

PurchaseDetailService.java

| package com.i2f.training.service.purchaseOrder;  import com.i2f.training.model.vo.PurchasedDetailVo;  /\*\*  \* 申购记录详情service接口  \*  \* @package: com.i2f.training.service.purchaseOrder  \* @author: 第零组---XuQ  \* @date: 2020/3/10 20:11  \*/ public interface PurchaseDetailService {  /\*\*  \* 查询申购记录详情  \*  \* @param purchaseOrderId 申购记录的id  \* @return com.i2f.training.model.vo.PurchasedDetailVo  \* @date 2020/3/10 20:47  \* @author XuQ  \*/  PurchasedDetailVo purchasedDetail(String purchaseOrderId); } |
| --- |

PurchaseDetailServiceImpl.java

| package com.i2f.training.service.purchaseOrder;  import com.i2f.training.mapper.FinanceProductMapper; import com.i2f.training.mapper.IncomeDetailedMapper; import com.i2f.training.mapper.PurchaseOrderMapper; import com.i2f.training.model.base.FinanceProduct; import com.i2f.training.model.base.IncomeDetailed; import com.i2f.training.model.base.PurchaseOrder; import com.i2f.training.model.example.IncomeDetailedExample; import com.i2f.training.model.vo.PurchasedDetailVo; import org.springframework.stereotype.Service;  import javax.annotation.Resource; import java.text.SimpleDateFormat; import java.util.ArrayList; import java.util.Calendar; import java.util.Date; import java.util.List;  /\*\*  \* 申购详情service实现类  \*  \* @package: com.i2f.training.service.purchaseOrder  \* @author: 第零组---XuQ  \* @date: 2020/3/10 20:46  \* <p>  \* name 产品名  \* rateReturn 收益率  \* description 产品介绍  \* amount 持仓  \* endTime 到期时间  \* yetReturn 已收益金额  \* riskType 风险类型  \* interestStatus 利滚利  \* serviceCharge 手续费  \* type 申购产品的类型  \*/ @Service public class PurchaseDetailServiceImpl implements PurchaseDetailService {   @Resource  PurchaseOrderMapper purchaseOrderMapper;  @Resource  FinanceProductMapper financeProductMapper;  @Resource  IncomeDetailedMapper incomeDetailedMapper;   @Override  public PurchasedDetailVo purchasedDetail(String purchaseOrderId) {  /\*\*  \*HttpServletRequest request = ((ServletRequestAttributes) RequestContextHolder.currentRequestAttributes()).getRequest();  \*String userId = (String) request.getAttribute("userId");  \*\*/  PurchasedDetailVo purchasedDetailVo = new PurchasedDetailVo();  // TODO: 2020/3/10 该条申购记录  PurchaseOrder purchaseOrder = purchaseOrderMapper.selectByPrimaryKey(purchaseOrderId);  // TODO: 2020/3/10 该条申购记录的信息，及其对应的产品 和 其信息  FinanceProduct financeProduct = financeProductMapper.selectByPrimaryKey(purchaseOrder.getProductId());  purchasedDetailVo.setName(financeProduct.getName());  String[] split = financeProduct.getRateReturn().split("-");  int length = split.length;  purchasedDetailVo.setRateReturn(split[split.length - 1]);  purchasedDetailVo.setDescription(financeProduct.getDescription());  purchasedDetailVo.setAmount(purchaseOrder.getTotalHolding());  purchasedDetailVo.setEndDate(purchaseOrder.getEndTime());  purchasedDetailVo.setYetReturn(purchaseOrder.getRetainA());  purchasedDetailVo.setRiskType(financeProduct.getRiskType());  purchasedDetailVo.setInterestStatus(financeProduct.getInterestStatus());  purchasedDetailVo.setServiceCharge(financeProduct.getServiceCharge());  purchasedDetailVo.setYesterdayReturn(purchaseOrder.getRetain());  purchasedDetailVo.setType(financeProduct.getType());  // TODO: 2020/3/10 在收入详情表中查询最近七天的数据，日期和收益率  IncomeDetailedExample incomeDetailedExample = new IncomeDetailedExample();  Calendar instance = Calendar.getInstance();  instance.set(Calendar.HOUR\_OF\_DAY, 0);  instance.set(Calendar.MINUTE, 0);  instance.set(Calendar.MILLISECOND, 0);  Date yesterday = instance.getTime();  instance.add(Calendar.DAY\_OF\_YEAR, -7);  Date lastWeek = instance.getTime();  //todo 设置排序规则，此处的底层sql为： order by date  incomeDetailedExample.setOrderByClause("date");  incomeDetailedExample.createCriteria().andPurchaseOrderIdEqualTo(purchaseOrderId).andDateBetween(lastWeek, // yesterday  new Date()  );  List<IncomeDetailed> incomeDetailedList = incomeDetailedMapper.selectByExample(incomeDetailedExample);   // TODO: 2020/3/17 获取收益折线图数据  List<String> x = new ArrayList<>();  List<String> y = new ArrayList<>();  SimpleDateFormat format = new SimpleDateFormat("MM-dd");  for (IncomeDetailed i : incomeDetailedList) {  String result = format.format(i.getDate());  x.add(result);  y.add(i.getDaliyIncome());  }  purchasedDetailVo.setX(x);  purchasedDetailVo.setY(y);   // TODO: 2020/3/17 获取收益率折线图数据  Calendar instance1 = Calendar.getInstance();  List<String> x1 = new ArrayList<>();  List<String> y1 = new ArrayList<>();  for (int i = 0; i < length; i++) {  instance1.add(Calendar.DAY\_OF\_YEAR, -1);  String horizontal = format.format(instance1.getTime());  String vertical = split[i];  x1.add(horizontal);  y1.add(vertical);  }  purchasedDetailVo.setX1(x1);  purchasedDetailVo.setY1(y1);   return purchasedDetailVo;  } } |
| --- |

RechargeController.java

| package com.i2f.training.controller.recharge;  import com.i2f.training.common.ResultMessage; import com.i2f.training.model.vm.RechargeVm; import com.i2f.training.service.recharge.RechargeService; import org.springframework.stereotype.Controller; import org.springframework.web.bind.annotation.PostMapping; import org.springframework.web.bind.annotation.RequestBody; import org.springframework.web.bind.annotation.RequestHeader; import org.springframework.web.bind.annotation.ResponseBody;  import javax.annotation.Resource;  /\*\*  \* @author Yutao Ma  \* @date 2020/3/10  \*/ @Controller public class RechargeController {  @Resource  RechargeService rechargeService;  @PostMapping("recharge")  @ResponseBody  public ResultMessage rechargeController(@RequestBody RechargeVm rechargeVm) {  return rechargeService.recharge(rechargeVm.getCardNum(), rechargeVm.getPassword(), rechargeVm.getRechargeAmount(), rechargeVm.getBankName());  } } |
| --- |

RechargeService.java

| package com.i2f.training.service.recharge;  import com.i2f.training.common.ResultMessage; import org.apache.commons.lang3.StringUtils;  public interface RechargeService {  ResultMessage recharge(String cardNum, String password, String rechargeAmount, String bankName); } |
| --- |

RechargeServiceImpl.java

| package com.i2f.training.service.recharge;  import com.i2f.training.client.RechargeClient; import com.i2f.training.common.Constants; import com.i2f.training.common.ResultMessage; import com.i2f.training.common.exception.TrainingException; import com.i2f.training.common.util.RandomUtils; import com.i2f.training.common.util.TokenUtils; import com.i2f.training.mapper.RechargeOrderMapper; import com.i2f.training.mapper.RechargeSerialMapper; import com.i2f.training.mapper.UserMapper; import com.i2f.training.model.base.RechargeOrder; import com.i2f.training.model.base.RechargeSerial; import com.i2f.training.model.base.User; import lombok.extern.slf4j.Slf4j; import org.apache.commons.lang3.StringUtils; import org.springframework.stereotype.Service; import org.springframework.transaction.annotation.Transactional; import org.springframework.web.context.request.RequestContextHolder; import org.springframework.web.context.request.ServletRequestAttributes;  import javax.annotation.Resource; import javax.servlet.http.HttpServletRequest; import java.nio.charset.StandardCharsets; import java.util.Date; import java.util.Map;  /\*\*  \* @author Yutao Ma  \* @date 2020/3/10  \*/ @Service @Slf4j public class RechargeServiceImpl implements RechargeService{  @Resource  UserMapper userMapper;  @Resource  RechargeClient rechargeClient;  @Resource  RechargeOrderMapper rechargeOrderMapper;  @Resource  RechargeSerialMapper rechargeSerialMapper;   @Override  @Transactional(rollbackFor = {RuntimeException.class})  public ResultMessage recharge(String cardNum, String password, String rechargeAmount, String bankName){  /\*获取userId\*/  ServletRequestAttributes servletRequestAttributes = (ServletRequestAttributes) RequestContextHolder.currentRequestAttributes();  HttpServletRequest request = servletRequestAttributes.getRequest();  String userId = (String) request.getAttribute("userId");   User user = userMapper.selectByPrimaryKey(userId);  String userStatus = user.getStatus();  String eAccountNum = user.getEAccountNum();  /\*判断用户权限\*/  if (!StringUtils.equals(userStatus, "2")) {  log.info("用户权限不足");  throw new TrainingException(Constants.Code.PERMISSION\_DENIED,Constants.Meaning.PERMISSION\_DENIED);  }   String serialId = RandomUtils.getRandom() + System.currentTimeMillis();  Map<String, String> resMap;  try {  resMap = rechargeClient.rechargeClient(serialId, cardNum, password, rechargeAmount, eAccountNum);  } catch (Exception e) {  throw new TrainingException(Constants.Code.SYSTEM\_EXCEPTION, Constants.Meaning.SYSTEM\_EXCEPTION);  }  // 0.充值成功，1.一类户限额不足，2.余额不足，3异常请求，4密码错误  String resultStatus = resMap.get("status");  String [] res = new String[]{"充值成功", "一类户限额不足", "余额不足", "异常请求", "密码错误"};   if(!StringUtils.equals(resultStatus, "0")){  log.info(userId + " " + res[Integer.valueOf(resultStatus)]);  return new ResultMessage("999999", res[Integer.valueOf(resultStatus)], resultStatus);  }   /\*插入充值记录\*/  RechargeOrder rechargeOrder = new RechargeOrder();  rechargeOrder.setAmount(rechargeAmount);  rechargeOrder.setBankCardNum(cardNum);  rechargeOrder.seteAccountNum(eAccountNum);  String rechargeOrderId = RandomUtils.getRandom() + System.currentTimeMillis();  rechargeOrder.setId(rechargeOrderId);  rechargeOrder.setRechargeTime(new Date());  rechargeOrder.setBankName(bankName);  rechargeOrderMapper.insert(rechargeOrder);  log.info(userId + " " + "插入充值记录");  /\*插入流水记录\*/  RechargeSerial rechargeSerial = new RechargeSerial();  rechargeSerial.setAmount(rechargeAmount);  rechargeSerial.setBankCardNum(cardNum);  rechargeSerial.seteAccountNum(eAccountNum);  rechargeSerial.setId(serialId);  rechargeSerial.setSystem(new String("银行互联网核心系统".getBytes(StandardCharsets.UTF\_8)));  rechargeSerial.setTime(new Date());  rechargeSerial.setStatus("1");  rechargeSerial.setRechargeOrderId(rechargeOrderId);  rechargeSerialMapper.insert(rechargeSerial);  log.info(userId + " " + "插入流水记录");   log.info(userId + " " + "充值成功");  return new ResultMessage("000000", "充值成功", "0");  } } |
| --- |

RedemptionController.java

| package com.i2f.training.controller.redemption;  import com.i2f.training.common.ResultMessage; import com.i2f.training.model.vm.RechargeVm; import com.i2f.training.model.vm.RedemptionVm; import com.i2f.training.service.redemption.RedemptionService; import org.springframework.web.bind.annotation.\*;  import javax.annotation.Resource;  /\*\*  \* @author 零组 chenxu  \* @date 2020/3/11 11:06  \*\*/ @RestController public class RedemptionController {  @Resource  RedemptionService redemptionService;   @PostMapping("redemption")  public ResultMessage<String> rechargeController(@RequestBody RedemptionVm redemptionVm) {  return redemptionService.redemptionService(redemptionVm.getPurchaseOrderId(),redemptionVm.getInterestStatus(),redemptionVm.getPassword(),redemptionVm.getAmount());  } } |
| --- |

RedemptionService.java

| package com.i2f.training.service.redemption;  import com.i2f.training.common.ResultMessage;  /\*\*  \* @author 零组 chenxu  \* @date 2020/3/11 10:23  \*\*/ public interface RedemptionService {   public ResultMessage<String> redemptionService(String purchaseOrderId,String interestStatus,String password,String amount); } |
| --- |

RedemptionServiceImpl.java

| package com.i2f.training.service.redemption;  import com.i2f.training.client.RedemptionClient; import com.i2f.training.common.Constants; import com.i2f.training.common.ResultMessage; import com.i2f.training.common.exception.TrainingException; import com.i2f.training.common.util.RandomUtils; import com.i2f.training.mapper.\*; import com.i2f.training.model.base.\*; import com.i2f.training.model.example.FinanceProductExample; import lombok.extern.slf4j.Slf4j; import org.apache.commons.lang3.StringUtils; import org.springframework.stereotype.Service; import org.springframework.transaction.annotation.Transactional; import org.springframework.util.DigestUtils; import org.springframework.web.context.request.RequestContextHolder; import org.springframework.web.context.request.ServletRequestAttributes;  import javax.annotation.Resource; import javax.servlet.http.HttpServletRequest; import java.math.BigDecimal; import java.util.Date; import java.util.Map; import java.util.UUID;  /\*\*  \* @author 零组 chenxu  \* @date 2020/3/11 10:44  \*\*/ @Slf4j @Service  public class RedemptionServiceImpl implements RedemptionService{   @Resource  PurchaseOrderMapper purchaseOrderMapper;  @Resource  RedemptionOrderMapper redemptionOrderMapper;  @Resource  RedemptionSerialMapper redemptionSerialMapper;  @Resource  UserMapper userMapper;  @Resource  RedemptionClient redemptionClient;  @Resource  FinanceProductMapper financeProductMapper;   @Override  @Transactional(rollbackFor = Exception.class)  public ResultMessage<String> redemptionService(String purchaseOrderId, String interestStatus, String password, String amount) {   ServletRequestAttributes servletRequestAttributes = (ServletRequestAttributes) RequestContextHolder.currentRequestAttributes();  HttpServletRequest request = servletRequestAttributes.getRequest();  String userId = (String) request.getAttribute("userId");  User user = userMapper.selectByPrimaryKey(userId);  String redemptionOrderId = UUID.randomUUID().toString().replaceAll("-", "").toString();  Map<String, String> stringStringMap = null;   PurchaseOrder purchaseOrder = purchaseOrderMapper.selectByPrimaryKey(purchaseOrderId);  //获得当前持仓  String totalHolding = purchaseOrder.getTotalHolding();  //获得当前总利润  String retainA = purchaseOrder.getRetainA();   String reallyAmount = "0";   int i = new BigDecimal(amount).compareTo(new BigDecimal(totalHolding));  //当赎回金额大于持仓金额  if (i>0){  throw new TrainingException(Constants.Code.REDEMPTION\_AMOUNT\_OUT\_HOLD,Constants.Meaning.REDEMPTION\_AMOUNT\_OUT\_HOLD);  }  //当赎回金额小于持仓金额  if (i <0){  reallyAmount = amount;  }  //  if (StringUtils.equals(interestStatus,"0")){  //非利滚利处理  //当赎回金额等于持仓金额  if (i == 0){  reallyAmount = new BigDecimal(amount).add(new BigDecimal(retainA)).toString();  purchaseOrder.setStatus("1");  }   }else {  //利滚利处理  //当赎回金额等于持仓金额  if (i == 0){  reallyAmount = new BigDecimal(amount).toString();  purchaseOrder.setStatus("1");  }  }  String serialId = RandomUtils.getRandom()+String.valueOf(System.currentTimeMillis());   FinanceProduct financeProduct = financeProductMapper.selectByPrimaryKey(purchaseOrder.getProductId());  String publishAccountNum =financeProduct.getPublishAccountNum();  RedemptionSerial redemptionSerial = new RedemptionSerial();  try {  stringStringMap = redemptionClient.redemptionClient(serialId, password, publishAccountNum, reallyAmount, user.getEAccountNum());  } catch (Exception e) {  insertSerial(amount, user.getEAccountNum(), redemptionSerial, purchaseOrder.getProductId(), serialId, "0",null);  throw new TrainingException(Constants.Code.SYSTEM\_EXCEPTION, Constants.Meaning.SYSTEM\_EXCEPTION);  }  insertSerial(amount, user.getEAccountNum(), redemptionSerial, purchaseOrder.getProductId(), serialId, "1",redemptionOrderId);   String status = stringStringMap.get("status");  String timeStamp = stringStringMap.get("timeStamp");  String autograph = stringStringMap.get("autograph");  String s = timeStamp + "&" + status;  String checkAutograph = DigestUtils.md5DigestAsHex(s.getBytes());  if (StringUtils.equals(status,"2")){  throw new TrainingException(Constants.Code.EXCEPTION\_REQUEST,Constants.Meaning.EXCEPTION\_REQUEST);  }  if (StringUtils.isBlank(checkAutograph)||!StringUtils.equals(checkAutograph,autograph)){  throw new TrainingException(Constants.Code.EXCEPTION\_REQUEST,Constants.Meaning.EXCEPTION\_REQUEST);  }  if (StringUtils.equals(status,"3")){  throw new TrainingException(Constants.Code.PASS\_NOT\_EXIST,Constants.Meaning.PASS\_NOT\_EXIST);  }  String s1 = new BigDecimal(totalHolding).subtract(new BigDecimal(amount)).toString();  purchaseOrder.setTotalHolding(s1);  purchaseOrderMapper.updateByPrimaryKeySelective(purchaseOrder);  RedemptionOrder redemptionOrder = new RedemptionOrder(redemptionOrderId,user.getEAccountNum(),reallyAmount,new Date(System.currentTimeMillis()),amount,purchaseOrderId);  int i1 = redemptionOrderMapper.insert(redemptionOrder);  return new ResultMessage<>(Constants.Code.SUCCESS,Constants.Meaning.SUCCESS,"i1");  }    private void insertSerial(String amount, String eAccountNum, RedemptionSerial redemptionSerial, String productId, String serialId, String status,String orderId) {  redemptionSerial.setAmount(amount);  redemptionSerial.seteAccountNum(eAccountNum);  redemptionSerial.setId(serialId);  redemptionSerial.setProductId(productId);  redemptionSerial.setStatus(status);  redemptionSerial.setSystem("银行互联网核心系统");  redemptionSerial.setTime(new Date());  redemptionSerial.setRedemptionOrderId(orderId);  redemptionSerialMapper.insert(redemptionSerial);  } } |
| --- |

ToMineController.java

| package com.i2f.training.controller.toMine;  import com.i2f.training.common.Constants; import com.i2f.training.common.ResultMessage; import com.i2f.training.model.vo.OutToMineVo; import com.i2f.training.service.toMine.ToMineService; import org.springframework.stereotype.Controller; import org.springframework.web.bind.annotation.GetMapping; import org.springframework.web.bind.annotation.RequestAttribute; import org.springframework.web.bind.annotation.RequestHeader; import org.springframework.web.bind.annotation.ResponseBody;  import javax.annotation.Resource;  /\*\*  \* @author Yutao Ma  \* @date 2020/3/10  \*/ @Controller public class ToMineController {  @Resource  ToMineService toMineService;  @GetMapping("/toMine")  @ResponseBody  public ResultMessage<OutToMineVo> toMinePage(){  return new ResultMessage<>(Constants.Code.SUCCESS,Constants.Meaning.SUCCESS,toMineService.toMine());  } } |
| --- |

ToMineService.java

| package com.i2f.training.service.toMine;  import com.i2f.training.model.vo.OutToMineVo;  public interface ToMineService {  OutToMineVo toMine(); } |
| --- |

ToMineServiceImpl.java

| package com.i2f.training.service.toMine;  import com.i2f.training.client.FindEAccountClient; import com.i2f.training.common.Constants; import com.i2f.training.common.exception.TrainingException; import com.i2f.training.mapper.FinanceProductMapper; import com.i2f.training.mapper.PurchaseOrderMapper; import com.i2f.training.mapper.UserMapper; import com.i2f.training.model.base.FinanceProduct; import com.i2f.training.model.base.PurchaseOrder; import com.i2f.training.model.base.User; import com.i2f.training.model.example.PurchaseOrderExample; import com.i2f.training.model.vo.OutToMineVo; import com.i2f.training.model.vo.ProductVo; import lombok.extern.slf4j.Slf4j; import org.apache.commons.lang3.StringUtils; import org.springframework.stereotype.Service; import org.springframework.web.context.request.RequestContextHolder; import org.springframework.web.context.request.ServletRequestAttributes;  import javax.annotation.Resource; import javax.servlet.http.HttpServletRequest; import java.math.BigDecimal; import java.util.ArrayList; import java.util.List; import java.util.Map; import java.util.UUID;  /\*\*  \* @author Yutao Ma  \* @date 2020/3/10  \*/ @Service @Slf4j public class ToMineServiceImpl implements ToMineService{  @Resource  UserMapper userMapper;  @Resource  FinanceProductMapper financeProductMapper;  @Resource  PurchaseOrderMapper purchaseOrderMapper;  @Resource  FindEAccountClient findEAccountClient;  @Override  public OutToMineVo toMine() {  /\*获取userId\*/  ServletRequestAttributes servletRequestAttributes = (ServletRequestAttributes) RequestContextHolder.currentRequestAttributes();  HttpServletRequest request = servletRequestAttributes.getRequest();  String userId = (String) request.getAttribute("userId");  /\*数据声明\*/  OutToMineVo outToMineVo = new OutToMineVo();  String realName;  String accountNum;  String balance;  List<ProductVo> productList = new ArrayList<>();  User user = userMapper.selectByPrimaryKey(userId);  accountNum = user.getEAccountNum();  /\*判断用户权限\*/  if(StringUtils.equals(user.getStatus(), "0")){  log.info("用户权限不足");  throw new TrainingException(Constants.Code.PERMISSION\_DENIED,Constants.Meaning.PERMISSION\_DENIED);  }  /\*与银行核心交互获得account信息\*/  Map<String, String> account;  try {  account = findEAccountClient.getAccount(UUID.randomUUID().toString().replaceAll("-", ""), accountNum);  } catch (Exception e) {  throw new TrainingException(Constants.Code.SYSTEM\_EXCEPTION, Constants.Meaning.SYSTEM\_EXCEPTION);  }  if(StringUtils.equals(account.get("status"), "1")){  log.info("失败的操作");  throw new TrainingException("999999", "操作失败");  }  balance = account.get("balance");  realName = account.get("realName");  /\*\*  \* 获得产品名称与本息和  \*/  PurchaseOrderExample purchaseOrderExample = new PurchaseOrderExample();  purchaseOrderExample.createCriteria().andEAccountNumEqualTo(accountNum);  List<PurchaseOrder> purchaseOrderList = purchaseOrderMapper.selectByExample(purchaseOrderExample);  if(purchaseOrderList.isEmpty()){  log.info(userId + ":名下没有产品");  }else {  for(PurchaseOrder purchaseOrder : purchaseOrderList){  if(StringUtils.equals(purchaseOrder.getStatus(), "1")){  log.info(purchaseOrder.getProductId() + ":该产品已被赎回");  continue;  }  FinanceProduct financeProduct = financeProductMapper.selectByPrimaryKey(purchaseOrder.getProductId());  ProductVo productVo = new ProductVo();  productVo.setName(financeProduct.getName());  BigDecimal totalHolding = new BigDecimal(purchaseOrder.getTotalHolding());  BigDecimal retain\_a = new BigDecimal(purchaseOrder.getRetainA());  BigDecimal total = totalHolding.add(retain\_a);  productVo.setValue(total.toString());  productList.add(productVo);  }  }  /\*返回结果集\*/  outToMineVo.setRealName(realName);  outToMineVo.setBalance(balance);  outToMineVo.setAccountNum(accountNum);  outToMineVo.setProductList(productList);  return outToMineVo;  } } |
| --- |

RedemptionService.java

| package com.i2f.training.service.redemption;  import com.i2f.training.common.ResultMessage;  /\*\*  \* @author 零组 chenxu  \* @date 2020/3/11 10:23  \*\*/ public interface RedemptionService {   public ResultMessage<String> redemptionService(String purchaseOrderId,String interestStatus,String password,String amount); } |
| --- |

TransactionDetailsService.java

| package com.i2f.training.service.transactionDetails;  import com.i2f.training.common.ResultMessage; import com.i2f.training.model.vo.TransactionDetailsVo;  /\*\*  \* @package: com.i2f.training.service.transactionDetails  \* @author: 第零组---FengBoTong  \* @date: 2020/3/10 14:25  \* @description: 交易记录明细  \*/ public interface TransactionDetailsService {  ResultMessage<TransactionDetailsVo> transactionDetails(String transactionId,String type); } |
| --- |

TransactionDetailsServiceImpl.java

| package com.i2f.training.service.transactionDetails;  import com.i2f.training.common.Constants; import com.i2f.training.common.ResultMessage; import com.i2f.training.common.exception.TrainingException; import com.i2f.training.mapper.\*; import com.i2f.training.model.base.\*; import com.i2f.training.model.example.PurchaseSerialExample; import com.i2f.training.model.example.RechargeSerialExample; import com.i2f.training.model.example.RedemptionSerialExample; import com.i2f.training.model.example.WithdrawSerialExample; import com.i2f.training.model.vo.TransactionDetailsVo; import lombok.extern.slf4j.Slf4j; import org.apache.commons.lang3.StringUtils; import org.springframework.stereotype.Service;  import javax.annotation.Resource; import java.util.Date; import java.util.List;  /\*\*  \* @package: com.i2f.training.service.transactionDetails  \* @author: 第零组---FengBoTong  \* @date: 2020/3/10 14:29  \* @description:  \*/ @Service @Slf4j public class TransactionDetailsServiceImpl implements TransactionDetailsService {  @Resource  RechargeOrderMapper rechargeOrderMapper;  @Resource  RechargeSerialMapper rechargeSerialMapper;  @Resource  WithdrawOrderMapper withdrawOrderMapper;  @Resource  WithdrawSerialMapper withdrawSerialMapper;  @Resource  FinanceProductMapper financeProductMapper;  @Resource  PurchaseOrderMapper purchaseOrderMapper;  @Resource  PurchaseSerialMapper purchaseSerialMapper;  @Resource  RedemptionOrderMapper redemptionOrderMapper;  @Resource  RedemptionSerialMapper redemptionSerialMapper;  @Override  public ResultMessage<TransactionDetailsVo> transactionDetails(String transactionId, String type) {  TransactionDetailsVo transactionDetailsVo = new TransactionDetailsVo();  log.info(type);  log.info(transactionId);  //todo 判断是哪种类型记录 1充值 2提现 3申购 4赎回  if (StringUtils.equals(type,"1")) {  RechargeOrder rechargeOrder = rechargeOrderMapper.selectByPrimaryKey(transactionId);  String bankCardNum = rechargeOrder.getBankCardNum();  String bankName = rechargeOrder.getBankName();  Date rechargeTime = rechargeOrder.getRechargeTime();  //todo 通过订单ID 去流水表 查询订单号  RechargeSerialExample rechargeSerialExample = new RechargeSerialExample();  rechargeSerialExample.createCriteria().andRechargeOrderIdEqualTo(transactionId);  List<RechargeSerial> rechargeSerials = rechargeSerialMapper.selectByExample(rechargeSerialExample);  RechargeSerial rechargeSerial = rechargeSerials.get(0);  String id = rechargeSerial.getId();   transactionDetailsVo.setBankCardName(bankName);  transactionDetailsVo.setBankCardNum(bankCardNum);  transactionDetailsVo.setCompany(null);  transactionDetailsVo.setExplain("银行卡充值到电子账户");  transactionDetailsVo.setProductName(null);  transactionDetailsVo.setSerialId(id);  transactionDetailsVo.setTime(rechargeTime);  return new ResultMessage<>(Constants.Code.SUCCESS,Constants.Meaning.SUCCESS,transactionDetailsVo);  }  if (StringUtils.equals(type,"2")) {  WithdrawOrder withdrawOrder = withdrawOrderMapper.selectByPrimaryKey(transactionId);  String bankCardNum = withdrawOrder.getBankCardNum();  Date withdrawTime = withdrawOrder.getWithdrawTime();  String bankName = withdrawOrder.getBankName();  WithdrawSerialExample withdrawSerialExample = new WithdrawSerialExample();  withdrawSerialExample.createCriteria().andWithdrawOrderIdEqualTo(transactionId);  List<WithdrawSerial> withdrawSerials = withdrawSerialMapper.selectByExample(withdrawSerialExample);  WithdrawSerial withdrawSerial = withdrawSerials.get(0);  String id = withdrawSerial.getId();   transactionDetailsVo.setBankCardName(bankName);  transactionDetailsVo.setBankCardNum(bankCardNum);  transactionDetailsVo.setCompany(null);  transactionDetailsVo.setExplain("电子账户提现到银行卡");  transactionDetailsVo.setProductName(null);  transactionDetailsVo.setSerialId(id);  transactionDetailsVo.setTime(withdrawTime);  return new ResultMessage<>(Constants.Code.SUCCESS,Constants.Meaning.SUCCESS,transactionDetailsVo);  }  if (StringUtils.equals(type,"3")) {  PurchaseOrder purchaseOrder = purchaseOrderMapper.selectByPrimaryKey(transactionId);  Date createTime = purchaseOrder.getCreateTime();  String productId = purchaseOrder.getProductId();  FinanceProduct financeProduct = financeProductMapper.selectByPrimaryKey(productId);  String company = financeProduct.getCompany();  String name = financeProduct.getName();  PurchaseSerialExample purchaseSerialExample = new PurchaseSerialExample();  purchaseSerialExample.createCriteria().andPurchaseOrderIdEqualTo(transactionId);  List<PurchaseSerial> purchaseSerials = purchaseSerialMapper.selectByExample(purchaseSerialExample);  PurchaseSerial purchaseSerial = purchaseSerials.get(0);  String id = purchaseSerial.getId();   transactionDetailsVo.setBankCardName(null);  transactionDetailsVo.setBankCardNum(null);  transactionDetailsVo.setCompany(company);  transactionDetailsVo.setExplain(null);  transactionDetailsVo.setProductName(name);  transactionDetailsVo.setSerialId(id);  transactionDetailsVo.setTime(createTime);  return new ResultMessage<>(Constants.Code.SUCCESS,Constants.Meaning.SUCCESS,transactionDetailsVo);  }  if (StringUtils.equals(type,"4")) {  RedemptionOrder redemptionOrder = redemptionOrderMapper.selectByPrimaryKey(transactionId);  Date redemptionTime = redemptionOrder.getRedemptionTime();  String purchaseOrderId = redemptionOrder.getPurchaseOrderId();  PurchaseOrder purchaseOrder = purchaseOrderMapper.selectByPrimaryKey(purchaseOrderId);  String productId = purchaseOrder.getProductId();  FinanceProduct financeProduct = financeProductMapper.selectByPrimaryKey(productId);  String company = financeProduct.getCompany();  String name = financeProduct.getName();  RedemptionSerialExample redemptionSerialExample = new RedemptionSerialExample();  redemptionSerialExample.createCriteria().andRedemptionOrderIdEqualTo(transactionId);  List<RedemptionSerial> redemptionSerials = redemptionSerialMapper.selectByExample(redemptionSerialExample);  RedemptionSerial redemptionSerial = redemptionSerials.get(0);  String id = redemptionSerial.getId();   transactionDetailsVo.setBankCardName(null);  transactionDetailsVo.setBankCardNum(null);  transactionDetailsVo.setCompany(company);  transactionDetailsVo.setExplain(null);  transactionDetailsVo.setProductName(name);  transactionDetailsVo.setSerialId(id);  transactionDetailsVo.setTime(redemptionTime);  return new ResultMessage<>(Constants.Code.SUCCESS,Constants.Meaning.SUCCESS,transactionDetailsVo);  }  //todo 返还的type数据不对 判断为异常请求  throw new TrainingException(Constants.Code.SYSTEM\_EXCEPTION,Constants.Meaning.EXCEPTION\_REQUEST);  } } |
| --- |

applyProductRecordDetails.jsp

| <html> <head>  <title>欢迎登录</title>  <meta http-equiv="content-type" content="text/html; charset=UTF-8"> </head> <body> <body>  #foreach($applyProductRecordDetails in $applyProductRecordDetailss)   <h3>产品名称：$info.productNum</h3>  <h3>产品类型：#if($info.type == '0')活期 #else 死期 #end</h3>  <h3>起购金额$info.amount 元</h3>  <h3>产品库存：$info.inventoryAmount</h3>  <h3>预售开始时间：$info.advanceStartDate</h3>  <h3>预售结束时间：$info.advanceEndDate</h3>  <h3>开售日期：$info.startDate</h3>  <h3>停售日期：$info.endDate</h3>  <h3>产品特点：$info.riskType 和 #if($info.interestStatus == '0')利滚利 #else 非利滚利 #end</h3>  <h3>产品描述：$info.description</h3>  <h3>产品所属公司：$info.company</h3>  <h3>产品图片：</h3>  <h3>预计收益率：$info.rateReturn</h3></td>  #end </body> </html> |
| --- |

applyProductRecordDetails.jsp

| <html> <head>  <title>欢迎登录</title>  <meta http-equiv="content-type" content="text/html; charset=UTF-8"> </head> <body> <body>  #foreach($applyProductRecordDetails in $applyProductRecordDetailss)   <h3>产品名称：$info.productNum</h3>  <h3>产品类型：#if($info.type == '0')活期 #else 死期 #end</h3>  <h3>起购金额$info.amount 元</h3>  <h3>产品库存：$info.inventoryAmount</h3>  <h3>预售开始时间：$info.advanceStartDate</h3>  <h3>预售结束时间：$info.advanceEndDate</h3>  <h3>开售日期：$info.startDate</h3>  <h3>停售日期：$info.endDate</h3>  <h3>产品特点：$info.riskType 和 #if($info.interestStatus == '0')利滚利 #else 非利滚利 #end</h3>  <h3>产品描述：$info.description</h3>  <h3>产品所属公司：$info.company</h3>  <h3>产品图片：</h3>  <h3>预计收益率：$info.rateReturn</h3></td>  #end </body> </html> |
| --- |

applyProductRecordDetails.jsp

| <html> <head>  <title>欢迎登录</title>  <meta http-equiv="content-type" content="text/html; charset=UTF-8"> </head> <body> <body>  #foreach($applyProductRecordDetails in $applyProductRecordDetailss)   <h3>产品名称：$info.productNum</h3>  <h3>产品类型：#if($info.type == '0')活期 #else 死期 #end</h3>  <h3>起购金额$info.amount 元</h3>  <h3>产品库存：$info.inventoryAmount</h3>  <h3>预售开始时间：$info.advanceStartDate</h3>  <h3>预售结束时间：$info.advanceEndDate</h3>  <h3>开售日期：$info.startDate</h3>  <h3>停售日期：$info.endDate</h3>  <h3>产品特点：$info.riskType 和 #if($info.interestStatus == '0')利滚利 #else 非利滚利 #end</h3>  <h3>产品描述：$info.description</h3>  <h3>产品所属公司：$info.company</h3>  <h3>产品图片：</h3>  <h3>预计收益率：$info.rateReturn</h3></td>  #end </body> </html> |
| --- |

auditorapply.jsp

| <html>  <head>  <title>欢迎您！</title>  <meta http-equiv="content-type" content="text/html; charset=UTF-8">  </head>  <body>  <div align="left">  <br>  <form action="/find.do" method="post">  <b style="color: red">&nbsp;&nbsp;&nbsp;</b>  <input name="productname" placeholder="产品名称" style="width: 150px;height: 25px;"/>  <input name="applyProductRecords.id" type="hidden" value="${applyProductRecords.id}">&nbsp;&nbsp;&nbsp;&nbsp;  <input type="submit" value="查询">  </form>  </div>   <table width="88%" height="73px" bordercolor="#E6E6E6" cellpadding="0" cellspacing="0" bgcolor="#999999" align="center" border="1">  <tr>  <td height="40px" bgcolor="#FAFAFA" nowrap="nowrap">  <div align="center"><font >产品编号</font></div>  </td>  <td height="40px" bgcolor="#FAFAFA" nowrap="nowrap">  <div align="center"><font >产品名称</font></div>  </td>  <td height="40px" bgcolor="#FAFAFA" nowrap="nowrap">  <div align="center"><font >产品管理员账号</font></div>  </td>  <td height="40px" bgcolor="#FAFAFA" nowrap="nowrap">  <div align="center"><font >产品管理员</font></div>  </td>  <td height="40px" bgcolor="#FAFAFA" nowrap="nowrap">  <div align="center"><font >产品审核员账号</font></div>  </td>  <td height="40px" bgcolor="#FAFAFA" nowrap="nowrap">  <div align="center"><font >产品审核员</font></div>  </td>  <td height="40px" bgcolor="#FAFAFA" nowrap="nowrap">  <div align="center"><font >申请类型</font></div>  </td>  <td height="40px" bgcolor="#FAFAFA" nowrap="nowrap">  <div align="center"><font >申请时间</font></div>  </td>  <td height="40px" bgcolor="#FAFAFA" nowrap="nowrap">  <div align="center"><font >详情</font></div>  </td>  <td height="40px" bgcolor="#FAFAFA" nowrap="nowrap" colspan="2">  <div align="center"><font >操作</font></div>  </td>  </tr>  #foreach($applyProductRecords in $applyProductRecordss)  <tr bgcolor="#FFFFFF">  <td height="20px" valign="middle">$applyProductRecords.productnum</td>  <td height="20px" valign="middle">$applyProductRecords.name</td>  <td height="20px" valign="middle">$applyProductRecords.adminid  <td height="20px" valign="middle">$applyProductRecords.adminname</td>  <td height="20px" valign="middle">$applyProductRecords.auditorid</td>  <td height="20px" valign="middle">$applyProductRecords.auditorname</td>  <td height="20px" valign="middle">$applyProductRecords.applytype</td>  <td height="20px" valign="middle">$applyProductRecords.applytime</td>  <td><a class="btn btn-default btn-sm" href="/findDetail.do?id=${applyProductRecords.id}">详情</a>&nbsp;</td>  <td><a class="btn btn-default btn-sm" href="/auditorInfo/getFinishedList.do?userId=${applyProductRecords.auditorid}&id=${applyProductRecords.tablerid}" && href="/insert.do?id=${applyProductRecords.tablerid}">通过</a>&nbsp;  <a class="btn btn-default btn-sm" href="/auditorInfo/getFinishedList.do?userId=${applyProductRecords.auditorid}">驳回</a></td>  #end  </tr>  </table> </body> </html> |
| --- |

productAuditRecord.jsp

| <html> <head>  <title>哈哈哈哈哈哈哈哈</title>  <meta http-equiv="content-type" content="text/html; charset=UTF-8"> </head> <script>  </script> <body>  #foreach($super in $aa)  <table>  <tr>  <td>$super.productId</td>  <td>$super.productName</td>  <td>$super.id</td>  <td>$super.name</td>  <td>$super.adminId</td>  <td>$super.adminName</td>  <td>$super.result</td>  <td>$super.applyTime</td>  <td>$super.auditTime</td>  </tr>  </table>  #end  </body> </html> |
| --- |

productList.jsp

| <html> <head>  <title>产品列表</title>  <meta http-equiv="content-type" content="text/html; charset=UTF-8"> </head> <script>  var id =$student2.age </script> <body> <h1>$admin</h1>  #foreach($list1 in $list)  <table>  <tr>  <td>$list1.productNum</td>  <td>$list1.time</td>  <td>$list1.name</td>  <td>$list1.type</td>  <td>$list1.amount</td>  <td>$list1.startDate</td>  <td>$list1.adminName</td>  <td>产品调整</td>  <td>产品下架</td>  </tr>  </table>  #end </body> </html> |
| --- |

applicationInfo.jsp

| <html> <head>  <title>审核信息</title>  <meta http-equiv="content-type" content="text/html; charset=UTF-8"> </head> <script>  var id =$student2.age </script> <body> <form action="/login/applicationInfo.do" method="post">  <input type="text" required="required" placeholder="用户ID生成" name="userId">  <button class="but" type="submit">提交</button> </form>  #foreach($!list1 in $!appVo.list)  <table>  <tr>  <td>$!list1</td>   <td>$!appVo.id</td>  <td>$!appVo.name</td>  <td>$!list1.productNum</td>  <td>$!list1.name</td>  <td>$!list1.applyType</td>  <td>$!list1.applyTime</td>  <td>$!list1.result</td>  </tr>  </table>  #end </body> </html> |
| --- |

findProduct.jsp

| <html> <head>  <title>审核信息</title>  <meta http-equiv="content-type" content="text/html; charset=UTF-8"> </head> <script>  var id =$student2.age </script> <body> <form action="/login/findProduct.do" method="post">  <input type="text" required="required" placeholder="用户ID生成" name="userId">  <input type="text" required="required" placeholder="产品名称" name="productName">  <button class="but" type="submit">提交</button> </form>  #foreach($!list1 in $!findProductVo.list)  <table>  <tr>  <td>$!list1</td>   <td>$!findProductVo.id</td>  <td>$!findProductVo.name</td>  <td>$!list1.productNum</td>  <td>$!list1.name</td>  <td>$!list1.applyType</td>  <td>$!list1.applyTime</td>  <td>$!list1.result</td>  </tr>  </table>  #end </body> </html> |
| --- |