

# Source Code

## ICIN BANK

### Admin Portal:

#### Login.component.html

```
<div class="wrapper">
  <form class="form-signin" (ngSubmit)="onSubmit()"
[hidden]="loggedIn">
    <h2 class="clean-font">Please login</h2>

    <input type="text" class="form-control" name="username"
[(ngModel)]="username" placeholder="Username" name="username" required
autofocus="" />
    <br />

    <input type="password" class="form-control" name="password"
[(ngModel)]="password" placeholder="Password" required />
    <div class="form-group">
      <br/>
      <label >
        <input type="checkbox" name="remember-me" id="remember-me"
/>&nbsp;<span class="clean-font">Remember me</span>
      </label>
    </div>
    <button class="btn btn-primary btn-block" type="submit">Login</button>
  </form>
  <div [hidden]="!loggedIn">
    <h2>Welcome to Admin Portal!</h2>
  </div>
</div>
```

#### Index.html

```
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <title>ICIN Bank</title>
  <base href="/">

  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" type="image/x-icon" href="favicon.ico">
</head>
```

```

<meta charset="utf-8">
<title>ICIN Bank: Admin Portal</title>
<base href="/"> <meta name="viewport" content="width=device-width, initial-
scale=1">
<link rel="icon" type="image/x-icon" href="favicon.ico">
<link href="starter-template.css">
<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"
integrity="sha384-
BVYiISIFeK1dGmJRAkycuHAHRg320mUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"
crossorigin="anonymous">
</head>
  <link rel="stylesheet" type="text/css" href="assets/css/bootstrap.min.css">
  <link rel="stylesheet" type="text/css" href="styles.css">

</head>
<body>
  <app-root>Loading...</app-root>
  <script type="text/javascript" src="assets/js/jquery.js"></script>
  <script type="text/javascript" src="assets/js/bootstrap.min.js"></script>

</body>
</html>

```

### User Account.component.html

```

<h1>User Accounts</h1>

<table id="userTable" class="table table-striped" cellpadding="0"
width="100%">
  <thead>
    <tr>
      <th>User Name</th>
      <th>First Name</th>
      <th>Last Name</th>
      <th>Email</th>
      <th>Phone</th>
      <th>Primary Account</th>
      <th>Savings Account</th>
      <th>Enabled</th>
      <th>Action</th>
    </tr>
  </thead>
  <tbody>
    <tr *ngFor="let user of userList">
      <td>{{user.username}}</td>
      <td>{{user.firstName}}</td>

```

```

        <td>{{user.lastName}}</td>
        <td>{{user.email}}</td>
        <td>{{user.phone}}</td>
        <td><a (click)="onSelectPrimary(user.username)" style="cursor:
pointer;">{{user.primaryAccount.accountBalance}}</a></td>
        <td><a (click)="onSelectSavings(user.username)" style="cursor:
pointer;">{{user.savingsAccount.accountBalance}}</a></td>
        <td>{{user.enabled}}</td>
        <td [hidden]="user.enabled"><a (click)="enableUser(user.username)"
style="cursor: pointer;">Enable</a></td>
        <td [hidden]="!user.enabled"><a
(click)="disableUser(user.username)" style="cursor: pointer;">Disable</a></td>
    </tr>
</tbody>
</table>

```

### App.component.html

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body style="background-color: #98a874;">

    <div class="container">
        <app-navbar></app-navbar>
        <router-outlet></router-outlet>
    </div>

</body>
</html>

```

### CheckBookRequest.compnent.html

```

<h1>Cheque Book Requests</h1>

<table class="table table-striped">
    <thead>
        <tr>
            <th>Request Id</th>
            <th>User Name</th>

```

```

        <th>Description</th>
        <th>Confirmed?</th>
        <th>Action</th>
    </tr>
</thead>
<tbody>
    <tr *ngFor="let chequeBookRequest of chequeBookRequestList">
        <td>{{chequeBookRequest.id}}</td>
        <td>{{chequeBookRequest.user.username}}</td>

        <td>{{chequeBookRequest.description}}</td>
        <td>{{chequeBookRequest.confirmed}}</td>
        <td [hidden]="chequeBookRequest.confirmed"><a
(click)="confirmChequeBookRequest(chequeBookRequest.id)" style="cursor:
pointer;">Confirm</a></td>
        <!-- <td [hidden]="!user.enabled"><a
(click)="disableUser(user.username)" style="cursor: pointer;">Disable</a></td>
-->

    </tr>
</tbody>
</table>

```

### Primary Transaction.component.html

```

<h1>Primary Account Transaction List</h1>

<table class="table table-striped">
    <thead>
        <tr>
            <th>Post Date</th>
            <th>Description</th>
            <th>Type</th>
            <th>Status</th>
            <th>Amount</th>
            <th>Available Balance</th>
        </tr>
    </thead>
    <tbody>
        <tr *ngFor="let primaryTransaction of primaryTransactionList">
            <td>{{primaryTransaction.date | date: 'MM/dd/yyyy'}} </td>
            <td>{{primaryTransaction.description}}</td>
            <td>{{primaryTransaction.type}}</td>
            <td>{{primaryTransaction.status}}</td>
            <td>{{primaryTransaction.amount}}</td>
            <td>{{primaryTransaction.availableBalance}}</td>
        </tr>
    </tbody>
</table>

```

## SavingTransaction.component.html

```
h1>Savings Account Transaction List</h1>

<table class="table table-striped">
  <thead>
    <tr>
      <th>Post Date</th>
      <th>Description</th>
      <th>Type</th>
      <th>Status</th>
      <th>Amount</th>
      <th>Available Balance</th>
    </tr>
  </thead>
  <tbody>
    <tr *ngFor="let savingsTransaction of savingsTransactionList">
      <td>{{savingsTransaction.date | date: 'MM/dd/yyyy'}} </td>
      <td>{{savingsTransaction.description}}</td>
      <td>{{savingsTransaction.type}}</td>
      <td>{{savingsTransaction.status}}</td>
      <td>{{savingsTransaction.amount}}</td>
      <td>{{savingsTransaction.availableBalance}}</td>
    </tr>
  </tbody>
</table>
```

## Navbar.component.html

```
<nav class="navbar navbar-clean">
  <div class="container-fluid">

    <div class="navbar-header">
      <button type="button" class="navbar-toggle collapsed" data-
toggle="collapse" data-target="#bs-example-navbar-collapse-1">
        <span class="sr-only">Toggle navigation</span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
      </button>
      <a class="navbar-brand" routerLink="/login"
routerLinkActive="active">Admin Portal</a>
    </div>

    <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
      <ul class="nav navbar-nav">
```

```

        <li [style.display]="getDisplay()"><a routerLink="/userAccount"
routerLinkActive="active"> User Account <span class="sr-
only">(current)</span></a></li>
        <li [style.display]="getDisplay()"><a routerLink="/chequeBookRequest"
routerLinkActive="active"> Cheque Book Requests <span class="sr-
only">(current)</span></a></li>
    </ul>
    <ul class="nav navbar-nav navbar-right">
        <li [style.display]="getDisplay()"><a (click)="logout()"
style="cursor: pointer;">Logout</a></li>
    </ul>
</div>
</div>
</nav>

```

### **User Portal:**

#### **ICIN Bank Application**

```
package com.icinbank;
```

```
import org.springframework.boot.SpringApplication;
```

```
import org.springframework.boot.autoconfigure.SpringBootApplication;
```

```
@SpringBootApplication
```

```
public class ICINBankApplication {
```

```
    public static void main(String[] args) {
```

```
        SpringApplication.run(ICINBankApplication.class, args);
```

```
    }
```

```
}
```

### **Controller:**

#### **Account Controller:**

```
package com.icinbank.controller;
```

```
import java.security.Principal;
```

```
import java.util.List;
```

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import com.icinbank.domain.PrimaryAccount;
import com.icinbank.domain.PrimaryTransaction;
import com.icinbank.domain.SavingsAccount;
import com.icinbank.domain.SavingsTransaction;
import com.icinbank.domain.User;
import com.icinbank.service.AccountService;
import com.icinbank.service.TransactionService;
import com.icinbank.service.UserService;
```

```
@Controller
```

```
@RequestMapping("/account")
```

```
public class AccountController {
```

```
@Autowired
```

```
private UserService userService;
```

```
@Autowired
```

```
private AccountService accountService;
```

```
@Autowired
```

```
private TransactionService transactionService;
```

```
@RequestMapping("/primaryAccount")
```

```
public String primaryAccount(Model model, Principal principal) {  
  
    List<PrimaryTransaction> primaryTransactionList =  
    transactionService.findPrimaryTransactionList(principal.getName());  
  
    User user = userService.findByUsername(principal.getName());  
    PrimaryAccount primaryAccount = user.getPrimaryAccount();  
  
    model.addAttribute("primaryAccount", primaryAccount);  
    model.addAttribute("primaryTransactionList", primaryTransactionList);  
  
    return "primaryAccount";  
}
```

```
@RequestMapping("/savingsAccount")  
  
public String savingsAccount(Model model, Principal principal) {  
  
    List<SavingsTransaction> savingsTransactionList =  
    transactionService.findSavingsTransactionList(principal.getName());  
  
    User user = userService.findByUsername(principal.getName());  
    SavingsAccount savingsAccount = user.getSavingsAccount();  
  
    model.addAttribute("savingsAccount", savingsAccount);  
    model.addAttribute("savingsTransactionList", savingsTransactionList);  
  
    return "savingsAccount";  
}
```

```
@RequestMapping(value = "/deposit", method = RequestMethod.GET)  
  
public String deposit(Model model) {  
  
    model.addAttribute("accountType", "");  
    model.addAttribute("amount", "");  

```



```
return "deposit";  
}
```

```
@RequestMapping(value = "/deposit", method = RequestMethod.POST)  
public String depositPOST(@ModelAttribute("amount") String amount,  
    @ModelAttribute("accountType") String accountType, Principal principal) {  
    accountService.deposit(accountType, Double.parseDouble(amount), principal);  
  
    return "redirect:/ICINBank";  
}
```

```
@RequestMapping(value = "/withdraw", method = RequestMethod.GET)  
public String withdraw(Model model) {  
    model.addAttribute("accountType", "");  
    model.addAttribute("amount", "");  
  
    return "withdraw";  
}
```

```
@RequestMapping(value = "/withdraw", method = RequestMethod.POST)  
public String withdrawPOST(@ModelAttribute("amount") String amount,  
    @ModelAttribute("accountType") String accountType, Principal principal) {  
    accountService.withdraw(accountType, Double.parseDouble(amount), principal);  
  
    return "redirect:/ICINBank";  
}  
}
```

### **Home Controller:**

```
package com.icinbank.controller;
```

```
import java.security.Principal;
```

```
import java.util.HashSet;
```

```
import java.util.Set;
```

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

```
import org.springframework.stereotype.Controller;
```

```
import org.springframework.ui.Model;
```

```
import org.springframework.web.bind.annotation.ModelAttribute;
```

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

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

```
import com.icinbank.dao.RoleDao;
```

```
import com.icinbank.domain.PrimaryAccount;
```

```
import com.icinbank.domain.SavingsAccount;
```

```
import com.icinbank.domain.User;
```

```
import com.icinbank.domain.security.UserRole;
```

```
import com.icinbank.service.UserService;
```

```
@Controller
```

```
public class HomeController {
```

```
@Autowired
```

```
private UserService userService;
```

```
@Autowired
```

```
private RoleDao roleDao;
```

```
@RequestMapping("/")
```

```
public String home() {  
    return "redirect:/index";  
}
```

```
@RequestMapping("/index")  
public String index() {  
    return "index";  
}
```

```
@RequestMapping(value = "/signup", method = RequestMethod.GET)  
public String signup(Model model) {  
    User user = new User();  
  
    model.addAttribute("user", user);  
  
    return "signup";  
}
```

```
@RequestMapping(value = "/signup", method = RequestMethod.POST)  
public String signupPost(@ModelAttribute("user") User user, Model model) {  
  
    if(userService.checkUserExists(user.getUsername(), user.getEmail())) {  
  
        if (userService.checkEmailExists(user.getEmail())) {  
            model.addAttribute("emailExists", true);  
        }  
  
        if (userService.checkUsernameExists(user.getUsername())) {  
            model.addAttribute("usernameExists", true);  
        }  
    }  
}
```

```

}

return "signup";
} else {
Set<UserRole> userRoles = new HashSet<>();
userRoles.add(new UserRole(user, roleDao.findByName("ROLE_USER")));

userService.createUser(user, userRoles);

return "redirect:/";
}
}

```

```

@RequestMapping("/ICINBank")
public String ICINBank(Principal principal, Model model) {
    User user = userService.findByUsername(principal.getName());
    PrimaryAccount primaryAccount = user.getPrimaryAccount();
    SavingsAccount savingsAccount = user.getSavingsAccount();

    model.addAttribute("primaryAccount", primaryAccount);
    model.addAttribute("savingsAccount", savingsAccount);

    return "ICINBank";
}
}

```

#### **RequestCheckbook Controller:**

```

package com.icinbank.controller;

import java.security.Principal;
import java.text.ParseException;

```

```
import java.text.SimpleDateFormat;
import java.util.Date;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
```

```
import com.icinbank.domain.RequestChequeBook;
import com.icinbank.domain.User;
import com.icinbank.service.RequestChequeBookService;
import com.icinbank.service.UserService;
```

```
@Controller
```

```
@RequestMapping("/requestChequeBook")
```

```
public class RequestChequeBookController {
```

```
    @Autowired
```

```
    private RequestChequeBookService requestChequeBookService;
```

```
    @Autowired
```

```
    private UserService userService;
```

```
    @RequestMapping(value = "/create",method = RequestMethod.GET)
```

```
    public String createRequestChequeBook(Model model) {
```

```
        RequestChequeBook requestChequeBook = new RequestChequeBook();
```

```
        model.addAttribute("requestChequeBook", requestChequeBook);
```

```
        model.addAttribute("dateString", "");
```

```
        return "requestChequeBook";
    }
}
```

```
@RequestMapping(value = "/create",method = RequestMethod.POST)

public String createRequestChequeBookPost(@ModelAttribute("requestChequeBook")
RequestChequeBook requestChequeBook, Model model, Principal principal) throws
ParseException {
```

```
    User user = userService.findByUsername(principal.getName());
    requestChequeBook.setUser(user);
```

```
    requestChequeBookService.createRequestChequeBook(requestChequeBook);
```

```
    return "redirect:/ICINBank";
}
}
```

### **Transfer Controller:**

```
package com.icinbank.controller;
```

```
import java.security.Principal;
```

```
import java.util.List;
```

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

```
import org.springframework.stereotype.Controller;
```

```
import org.springframework.transaction.annotation.Transactional;
```

```
import org.springframework.ui.Model;
```

```
import org.springframework.web.bind.annotation.ModelAttribute;
```

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

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

```
import com.icinbank.domain.PrimaryAccount;
import com.icinbank.domain.Recipient;
import com.icinbank.domain.SavingsAccount;
import com.icinbank.domain.User;
import com.icinbank.service.TransactionService;
import com.icinbank.service.UserService;
```

```
@Controller
```

```
@RequestMapping("/transfer")
```

```
public class TransferController {
```

```
@Autowired
```

```
private TransactionService transactionService;
```

```
@Autowired
```

```
private UserService userService;
```

```
@RequestMapping(value = "/betweenAccounts", method = RequestMethod.GET)
```

```
public String betweenAccounts(Model model) {
```

```
model.addAttribute("transferFrom", "");
```

```
model.addAttribute("transferTo", "");
```

```
model.addAttribute("amount", "");
```

```
return "betweenAccounts";
```

```
}
```

```
@RequestMapping(value = "/betweenAccounts", method = RequestMethod.POST)
public String betweenAccountsPost(
    @ModelAttribute("transferFrom") String transferFrom,
    @ModelAttribute("transferTo") String transferTo,
    @ModelAttribute("amount") String amount,
    Principal principal
) throws Exception {
    User user = userService.findByUsername(principal.getName());
    PrimaryAccount primaryAccount = user.getPrimaryAccount();
    SavingsAccount savingsAccount = user.getSavingsAccount();
    transactionService.betweenAccountsTransfer(transferFrom, transferTo, amount,
    primaryAccount, savingsAccount);

    return "redirect:/ICINBank";
}
```

```
@RequestMapping(value = "/recipient", method = RequestMethod.GET)
public String recipient(Model model, Principal principal) {
    List<Recipient> recipientList = transactionService.findRecipientList(principal);

    Recipient recipient = new Recipient();

    model.addAttribute("recipientList", recipientList);
    model.addAttribute("recipient", recipient);

    return "recipient";
}
```

```
@RequestMapping(value = "/recipient/save", method = RequestMethod.POST)
```



```
public String recipientPost(@ModelAttribute("recipient") Recipient recipient, Principal principal) {
```

```
    User user = userService.findByUsername(principal.getName());
```

```
    recipient.setUser(user);
```

```
    transactionService.saveRecipient(recipient);
```

```
    return "redirect:/transfer/recipient";
```

```
}
```

```
@RequestMapping(value = "/recipient/edit", method = RequestMethod.GET)
```

```
public String recipientEdit(@RequestParam(value = "recipientName") String recipientName, Model model, Principal principal){
```

```
    Recipient recipient = transactionService.findRecipientByName(recipientName);
```

```
    List<Recipient> recipientList = transactionService.findRecipientList(principal);
```

```
    model.addAttribute("recipientList", recipientList);
```

```
    model.addAttribute("recipient", recipient);
```

```
    return "recipient";
```

```
}
```

```
@RequestMapping(value = "/recipient/delete", method = RequestMethod.GET)
```

```
@Transactional
```

```
public String recipientDelete(@RequestParam(value = "recipientName") String recipientName, Model model, Principal principal){
```

```
    transactionService.deleteRecipientByName(recipientName);
```

```
    List<Recipient> recipientList = transactionService.findRecipientList(principal);
```

```

Recipient recipient = new Recipient();
model.addAttribute("recipient", recipient);
model.addAttribute("recipientList", recipientList);
return "recipient";
}

```

```

@RequestMapping(value = "/toSomeoneElse",method = RequestMethod.GET)
public String toSomeoneElse(Model model, Principal principal) {
    List<Recipient> recipientList = transactionService.findRecipientList(principal);

```

```

model.addAttribute("recipientList", recipientList);
model.addAttribute("accountType", "");
return "toSomeoneElse";
}

```

```

@RequestMapping(value = "/toSomeoneElse",method = RequestMethod.POST)
public String toSomeoneElsePost(@ModelAttribute("recipientName") String recipientName,
    @ModelAttribute("accountType") String accountType, @ModelAttribute("amount") String
    amount, Principal principal) {

```

```

    User user = userService.findByUsername(principal.getName());
    Recipient recipient = transactionService.findRecipientByName(recipientName);
    transactionService.toSomeoneElseTransfer(recipient, accountType, amount,
    user.getPrimaryAccount(), user.getSavingsAccount());

```

```

return "redirect:/ICINBank
}
}

```

### **User Controller:**

```

package com.icinbank.controller;
import java.security.Principal

```

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
```

```
import com.icinbank.domain.User;
import com.icinbank.service.UserService;
```

```
@Controller
```

```
@RequestMapping("/user")
```

```
public class UserController {
```

```
@Autowired
```

```
private UserService userService;
```

```
@RequestMapping(value = "/profile", method = RequestMethod.GET)
```

```
public String profile(Principal principal, Model model) {
```

```
User user = userService.findByUsername(principal.getName());
```

```
model.addAttribute("user", user);
```

```
return "profile";
```

```
}
```

```
@RequestMapping(value = "/profile", method = RequestMethod.POST)
```

```
public String profilePost(@ModelAttribute("user") User newUser, Model model) {
```

```
User user = userService.findByUsername(newUser.getUsername());
```

```
user.setUsername(newUser.getUsername());
user.setFirstName(newUser.getFirstName());
user.setLastName(newUser.getLastName());
user.setEmail(newUser.getEmail());
user.setPhone(newUser.getPhone());
model.addAttribute("user", user);
userService.saveUser(user);
```

```
return "profile";
```

```
}
```

```
}
```

### **Request Checkbook Resources:**

```
package com.icinbank.resource;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.access.prepost.PreAuthorize;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.icinbank.domain.RequestChequeBook;
import com.icinbank.service.RequestChequeBookService;
```

```
@RestController
```

```
@RequestMapping("/api/chequeBookRequest")
```

```
@PreAuthorize("hasRole('ADMIN')")
```

```
public class RequestChequeBookResource {
```

```
@Autowired
```

```
private RequestChequeBookService requestChequeBookService;
```

```

@RequestMapping("/all")

public List<RequestChequeBook> findRequestChequeBookList() {

List<RequestChequeBook> requestChequeBookList = requestChequeBookService.findAll();


return requestChequeBookList;

}

```

```

@RequestMapping("/{id}/confirm")

public void confirmRequestChequeBook(@PathVariable("id") Long id) {

requestChequeBookService.confirmRequestChequeBook(id);

}

}

```

#### **User Resources:**

```

package com.icinbank.resource;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.access.prepost.PreAuthorize;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;


import com.icinbank.domain.PrimaryTransaction;
import com.icinbank.domain.SavingsTransaction;
import com.icinbank.domain.User;
import com.icinbank.service.TransactionService;
import com.icinbank.service.UserService;

```

```
@RestController
```

```
@RequestMapping("/api")
```

```
@PreAuthorize("hasRole('ADMIN')")
```

```
public class UserResource {
```

```
@Autowired
```

```
private UserService userService;
```

```
@Autowired
```

```
private TransactionService transactionService;
```

```
@RequestMapping(value = "/user/all", method = RequestMethod.GET)
```

```
public List<User> userList() {
```

```
    return userService.findUserList();
```

```
}
```

```
@RequestMapping(value = "/user/primary/transaction", method = RequestMethod.GET)
```

```
public List<PrimaryTransaction> getPrimaryTransactionList(@RequestParam("username")  
String username) {
```

```
    return transactionService.findPrimaryTransactionList(username);
```

```
}
```

```
@RequestMapping(value = "/user/savings/transaction", method = RequestMethod.GET)
```

```
public List<SavingsTransaction> getSavingsTransactionList(@RequestParam("username")  
String username) {
```

```
    return transactionService.findSavingsTransactionList(username);
```

```
@RequestMapping("/user/{username}/enable")
```

```
public void enableUser(@PathVariable("username") String username) {
```

```

userService.enableUser(username);
}

@RequestMapping("/user/{username}/disable")
public void disableUser(@PathVariable("username") String username) {
    userService.disableUser(username);
}
}

```

### **Services:**

#### **Account Services Impl :**

```

package com.icinbank.service.UserServiceImpl;

import java.math.BigDecimal;
import java.security.Principal;
import java.util.Date;

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

import com.icinbank.dao.PrimaryAccountDao;
import com.icinbank.dao.SavingsAccountDao;
import com.icinbank.domain.PrimaryAccount;
import com.icinbank.domain.PrimaryTransaction;
import com.icinbank.domain.SavingsAccount;
import com.icinbank.domain.SavingsTransaction;
import com.icinbank.domain.User;
import com.icinbank.service.AccountService;
import com.icinbank.service.TransactionService;
import com.icinbank.service.UserService;

@Service

public class AccountServiceImpl implements AccountService {

```

```
private static int nextAccountNumber = 11223145;
```

```
@Autowired
```

```
private PrimaryAccountDao primaryAccountDao;
```

```
@Autowired
```

```
private SavingsAccountDao savingsAccountDao;
```

```
@Autowired
```

```
private UserService userService;
```

```
@Autowired
```

```
private TransactionService transactionService;
```

```
public PrimaryAccount createPrimaryAccount() {
```

```
    PrimaryAccount primaryAccount = new PrimaryAccount();
```

```
    primaryAccount.setAccountBalance(new BigDecimal(0.0));
```

```
    primaryAccount.setAccountNumber(accountGen());
```

```
    primaryAccountDao.save(primaryAccount);
```

```
    return primaryAccountDao.findByAccountNumber(primaryAccount.getAccountNumber());
```

```
}
```

```
public SavingsAccount createSavingsAccount() {
```

```
    SavingsAccount savingsAccount = new SavingsAccount();
```

```
    savingsAccount.setAccountBalance(new BigDecimal(0.0));
```

```
    savingsAccount.setAccountNumber(accountGen());
```



```
savingsAccountDao.save(savingsAccount);
```

```
return savingsAccountDao.findByAccountNumber(savingsAccount.getAccountNumber());  
}
```

```
public void deposit(String accountType, double amount, Principal principal) {
```

```
User user = userService.findByUsername(principal.getName());
```

```
if (accountType.equalsIgnoreCase("Primary")) {
```

```
PrimaryAccount primaryAccount = user.getPrimaryAccount();
```

```
primaryAccount.setAccountBalance(primaryAccount.getAccountBalance().add(new  
BigDecimal(amount)));
```

```
primaryAccountDao.save(primaryAccount);
```

```
Date date = new Date();
```

```
PrimaryTransaction primaryTransaction = new PrimaryTransaction(date, "Deposit to Primary  
Account", "Account", "Finished", amount, primaryAccount.getAccountBalance(),  
primaryAccount);
```

```
transactionService.savePrimaryDepositTransaction(primaryTransaction);
```

```
} else if (accountType.equalsIgnoreCase("Savings")) {
```

```
SavingsAccount savingsAccount = user.getSavingsAccount();
```

```
savingsAccount.setAccountBalance(savingsAccount.getAccountBalance().add(new  
BigDecimal(amount)));
```

```
savingsAccountDao.save(savingsAccount);
```

```
Date date = new Date();
```

```
SavingsTransaction savingsTransaction = new SavingsTransaction(date, "Deposit to savings  
Account", "Account", "Finished", amount, savingsAccount.getAccountBalance(),  
savingsAccount);
```

```
transactionService.saveSavingsDepositTransaction(savingsTransaction);  
}  
}
```

```
public void withdraw(String accountType, double amount, Principal principal) {  
    User user = userService.findByUsername(principal.getName());
```

```
    if (accountType.equalsIgnoreCase("Primary")) {  
        PrimaryAccount primaryAccount = user.getPrimaryAccount();  
        primaryAccount.setAccountBalance(primaryAccount.getAccountBalance().subtract(new  
            BigDecimal(amount)));  
        primaryAccountDao.save(primaryAccount);
```

```
        Date date = new Date();
```

```
        PrimaryTransaction primaryTransaction = new PrimaryTransaction(date, "Withdraw from  
        Primary Account", "Account", "Finished", amount, primaryAccount.getAccountBalance(),  
        primaryAccount);
```

```
        transactionService.savePrimaryWithdrawTransaction(primaryTransaction);
```

```
    } else if (accountType.equalsIgnoreCase("Savings")) {
```

```
        SavingsAccount savingsAccount = user.getSavingsAccount();
```

```
        savingsAccount.setAccountBalance(savingsAccount.getAccountBalance().subtract(new  
            BigDecimal(amount)));
```

```
        savingsAccountDao.save(savingsAccount);
```

```
        Date date = new Date();
```

```
        SavingsTransaction savingsTransaction = new SavingsTransaction(date, "Withdraw from  
        savings Account", "Account", "Finished", amount, savingsAccount.getAccountBalance(),  
        savingsAccount);
```

```
        transactionService.saveSavingsWithdrawTransaction(savingsTransaction);
```

```
    }
```

```
}  
  
private int accountGen() {  
    return ++nextAccountNumber;  
}  
  
}
```

### **Request Checkbook Services Impl:**

```
package com.icinbank.service.UserServiceImpl;  
  
import java.util.List;  
  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
  
import com.icinbank.dao.RequestChequeBookDao;  
import com.icinbank.domain.RequestChequeBook;  
import com.icinbank.service.RequestChequeBookService;  
  
@Service  
public class RequestChequeBookServiceImpl implements RequestChequeBookService {  
  
    @Autowired  
    private RequestChequeBookDao requestChequeBookDao;  
  
    public RequestChequeBook createRequestChequeBook(RequestChequeBook  
requestChequeBook) {  
        return requestChequeBookDao.save(requestChequeBook);  
    }  
  
    public List<RequestChequeBook> findAll() {  
        return requestChequeBookDao.findAll();  
    }  
}
```

```
}
```

```
public RequestChequeBook findRequestChequeBook(Long id) {  
    //return requestChequeBookDao.findOne(id);  
    return requestChequeBookDao.findById(id).orElse(null);  
}
```

```
public void confirmRequestChequeBook(Long id) {  
    RequestChequeBook requestChequeBook = findRequestChequeBook(id);  
    requestChequeBook.setConfirmed(true);  
    requestChequeBookDao.save(requestChequeBook);  
}
```

```
    /*@Override
```

```
    public RequestChequeBook findRequestChequeBook(Long id) {  
        // TODO Auto-generated method stub  
        return null;  
    }*/
```

```
}
```

### **Transaction Services Impl:**

```
package com.icinbank.service.UserServiceImpl;
```

```
import java.math.BigDecimal;
```

```
import java.security.Principal;
```

```
import java.util.Date;
```

```
import java.util.List;
```

```
import java.util.stream.Collectors;
```

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

```
import org.springframework.stereotype.Service;
```

```
import com.icinbank.dao.PrimaryAccountDao;
```

```
import com.icinbank.dao.PrimaryTransactionDao;
```

```
import com.icinbank.dao.RecipientDao;
```

```
import com.icinbank.dao.SavingsAccountDao;
```

```
import com.icinbank.dao.SavingsTransactionDao;
```

```
import com.icinbank.domain.PrimaryAccount;
```

```
import com.icinbank.domain.PrimaryTransaction;
```

```
import com.icinbank.domain.Recipient;
```

```
import com.icinbank.domain.SavingsAccount;
```

```
import com.icinbank.domain.SavingsTransaction;
```

```
import com.icinbank.domain.User;
```

```
import com.icinbank.service.TransactionService;
```

```
import com.icinbank.service.UserService;
```

```
@Service
```

```
public class TransactionServiceImpl implements TransactionService {
```

```
@Autowired
```

```
private UserService userService;
```

```
@Autowired
```

```
private PrimaryTransactionDao primaryTransactionDao;
```

```
@Autowired
```

```
private SavingsTransactionDao savingsTransactionDao;
```

```
@Autowired
```

```
private PrimaryAccountDao primaryAccountDao;
```

```
@Autowired
```

```
private SavingsAccountDao savingsAccountDao;
```

```
@Autowired
```

```
private RecipientDao recipientDao;
```

```
public List<PrimaryTransaction> findPrimaryTransactionList(String username){
```

```
User user = userService.findByUsername(username);
```

```
List<PrimaryTransaction> primaryTransactionList =  
user.getPrimaryAccount().getPrimaryTransactionList();
```

```
return primaryTransactionList;
```

```
}
```

```
public List<SavingsTransaction> findSavingsTransactionList(String username) {
```

```
User user = userService.findByUsername(username);
```

```
List<SavingsTransaction> savingsTransactionList =  
user.getSavingsAccount().getSavingsTransactionList();
```

```
return savingsTransactionList;
```

```
}
```

```
public void savePrimaryDepositTransaction(PrimaryTransaction primaryTransaction) {
```

```
primaryTransactionDao.save(primaryTransaction);
```

```
}
```

```
public void saveSavingsDepositTransaction(SavingsTransaction savingsTransaction) {
```

```
savingsTransactionDao.save(savingsTransaction);  
}
```

```
public void savePrimaryWithdrawTransaction(PrimaryTransaction primaryTransaction) {  
    primaryTransactionDao.save(primaryTransaction);  
}
```

```
public void saveSavingsWithdrawTransaction(SavingsTransaction savingsTransaction) {  
    savingsTransactionDao.save(savingsTransaction);  
}
```

```
public void betweenAccountsTransfer(String transferFrom, String transferTo, String amount,  
    PrimaryAccount primaryAccount, SavingsAccount savingsAccount) throws Exception {  
    if (transferFrom.equalsIgnoreCase("Primary") && transferTo.equalsIgnoreCase("Savings")) {  
        primaryAccount.setAccountBalance(primaryAccount.getAccountBalance().subtract(new  
            BigDecimal(amount)));  
        savingsAccount.setAccountBalance(savingsAccount.getAccountBalance().add(new  
            BigDecimal(amount)));  
        primaryAccountDao.save(primaryAccount);  
        savingsAccountDao.save(savingsAccount);  
    }
```

```
    Date date = new Date();
```

```
    PrimaryTransaction primaryTransaction = new PrimaryTransaction(date, "Between account  
    transfer from "+transferFrom+" to "+transferTo, "Account", "Finished",  
    Double.parseDouble(amount), primaryAccount.getAccountBalance(), primaryAccount);  
    primaryTransactionDao.save(primaryTransaction);  
} else if (transferFrom.equalsIgnoreCase("Savings") &&  
    transferTo.equalsIgnoreCase("Primary")) {  
    primaryAccount.setAccountBalance(primaryAccount.getAccountBalance().add(new  
        BigDecimal(amount)));  
}
```

```
savingsAccount.setAccountBalance(savingsAccount.getAccountBalance().subtract(new  
BigDecimal(amount)));
```

```
primaryAccountDao.save(primaryAccount);
```

```
savingsAccountDao.save(savingsAccount);
```

```
Date date = new Date();
```

```
SavingsTransaction savingsTransaction = new SavingsTransaction(date, "Between account  
transfer from "+transferFrom+" to "+transferTo, "Transfer", "Finished",  
Double.parseDouble(amount), savingsAccount.getAccountBalance(), savingsAccount);
```

```
savingsTransactionDao.save(savingsTransaction);
```

```
} else {
```

```
throw new Exception("Invalid Transfer");
```

```
}
```

```
}
```

```
public List<Recipient> findRecipientList(Principal principal) {
```

```
String username = principal.getName();
```

```
List<Recipient> recipientList = recipientDao.findAll().stream()           //convert  
list to stream
```

```
.filter(recipient -> username.equals(recipient.getUser().getUsername())) //filters the line,  
equals to username
```

```
.collect(Collectors.toList());
```

```
return recipientList;
```

```
}
```

```
public Recipient saveRecipient(Recipient recipient) {
```

```
return recipientDao.save(recipient);
```

```
}
```



```
public Recipient findRecipientByName(String recipientName) {  
    return recipientDao.findByName(recipientName);  
}
```

```
public void deleteRecipientByName(String recipientName) {  
    recipientDao.deleteByName(recipientName);  
}
```

```
public void toSomeoneElseTransfer(Recipient recipient, String accountType, String amount,  
    PrimaryAccount primaryAccount, SavingsAccount savingsAccount) {  
    if (accountType.equalsIgnoreCase("Primary")) {  
        primaryAccount.setAccountBalance(primaryAccount.getAccountBalance().subtract(new  
            BigDecimal(amount)));  
        primaryAccountDao.save(primaryAccount);
```

```
        Date date = new Date();
```

```
        PrimaryTransaction primaryTransaction = new PrimaryTransaction(date, "Transfer to  
        recipient "+recipient.getName(), "Transfer", "Finished", Double.parseDouble(amount),  
        primaryAccount.getAccountBalance(), primaryAccount);
```

```
        primaryTransactionDao.save(primaryTransaction);
```

```
    } else if (accountType.equalsIgnoreCase("Savings")) {
```

```
        savingsAccount.setAccountBalance(savingsAccount.getAccountBalance().subtract(new  
            BigDecimal(amount)));
```

```
        savingsAccountDao.save(savingsAccount);
```

```
        Date date = new Date();
```

```
        SavingsTransaction savingsTransaction = new SavingsTransaction(date, "Transfer to  
        recipient "+recipient.getName(), "Transfer", "Finished", Double.parseDouble(amount),  
        savingsAccount.getAccountBalance(), savingsAccount);
```

```
        savingsTransactionDao.save(savingsTransaction);
```

```
}  
}  
}
```

### **User Services Impl:**

```
package com.icinbank.service.UserServiceImpl;
```

```
import java.util.List;
```

```
import java.util.Set;
```

```
import org.slf4j.Logger;
```

```
import org.slf4j.LoggerFactory;
```

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

```
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
```

```
import org.springframework.stereotype.Service;
```

```
import org.springframework.transaction.annotation.Transactional;
```

```
import com.icinbank.dao.RoleDao;
```

```
import com.icinbank.dao.UserDao;
```

```
import com.icinbank.domain.User;
```

```
import com.icinbank.domain.security.UserRole;
```

```
import com.icinbank.service.AccountService;
```

```
import com.icinbank.service.UserService;
```

```
@Service
```

```
@Transactional
```

```
public class UserServiceImpl implements UserService{
```

```
private static final Logger LOG = LoggerFactory.getLogger(UserService.class);
```

@Autowired

private UserDao userDao;

@Autowired

private RoleDao roleDao;

@Autowired

private BCryptPasswordEncoder passwordEncoder;

@Autowired

private AccountService accountService;

public void save(User user) {

userDao.save(user);

}

public User findByUsername(String username) {

return userDao.findByUsername(username);

}

public User findByEmail(String email) {

return userDao.findByEmail(email);

}

public User createUser(User user, Set<UserRole> userRoles) {

User localUser = userDao.findByUsername(user.getUsername());

if (localUser != null) {

```

LOG.info("User with username {} already exist. Nothing will be done. ",
user.getUsername());

} else {

String encryptedPassword = passwordEncoder.encode(user.getPassword());
user.setPassword(encryptedPassword);


for (UserRole ur : userRoles) {
roleDao.save(ur.getRole());
}


user.getUserRoles().addAll(userRoles);


user.setPrimaryAccount(accountService.createPrimaryAccount());
user.setSavingsAccount(accountService.createSavingsAccount());


localUser = userDao.save(user);
}


return localUser;
}


public boolean checkUserExists(String username, String email){
if (checkUsernameExists(username) || checkEmailExists(username)) {
return true;
} else {
return false;
}
}


public boolean checkUsernameExists(String username) {

```

```
if (null != findByUsername(username)) {  
    return true;  
}
```

```
return false;  
}
```

```
public boolean checkEmailExists(String email) {  
    if (null != findByEmail(email)) {  
        return true;  
    }
```

```
return false;  
}
```

```
public User saveUser (User user) {  
    return userDao.save(user);  
}
```

```
public List<User> findUserList() {  
    return userDao.findAll();  
}
```

```
public void enableUser (String username) {  
    User user = findByUsername(username);  
    user.setEnabled(true);  
    userDao.save(user);  
}
```

```

public void disableUser (String username) {
    User user = findByUsername(username);
    user.setEnabled(false);
    System.out.println(user.isEnabled());
    userDao.save(user);
    System.out.println(username + " is disabled.");
}
}

```

### **User Security Services:**

```

package com.icinbank.service.UserServiceImpl;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.security.core.userdetails.UserDetailsService;
import org.springframework.security.core.userdetails.UsernameNotFoundException;
import org.springframework.stereotype.Service;

import com.icinbank.dao.UserDao;
import com.icinbank.domain.User;

@Service
public class UserSecurityService implements UserDetailsService {

    /** The application logger */
    private static final Logger LOG = LoggerFactory.getLogger(UserSecurityService.class);

    @Autowired

```

```
private UserDao userDao;
```

```
@Override
```

```
public UserDetails loadUserByUsername(String username) throws  
UsernameNotFoundException {
```

```
User user = userDao.findByUsername(username);
```

```
if (null == user) {
```

```
LOG.warn("Username {} not found", username);
```

```
throw new UsernameNotFoundException("Username " + username + " not found");
```

```
}
```

```
return user;
```

```
}
```

```
}
```

**Security:**

**Authority:**

```
package com.icinbank.domain.security;
```

```
import org.springframework.security.core.GrantedAuthority;
```

```
public class Authority implements GrantedAuthority{
```

```
private final String authority;
```

```
public Authority(String authority) {
```

```
this.authority = authority;
```

```
}
```

```
@Override
```

```
public String getAuthority() {
```

```
return authority;
```

```
}  
}
```

### **Role:**

```
package com.icinbank.domain.security;  
  
import javax.persistence.*;  
  
import java.util.HashSet;  
  
import java.util.Set;  
  
@Entity  
  
public class Role {  
  
    @Id  
  
    // @GeneratedValue(strategy = GenerationType.AUTO)  
  
    private int roleId;  
  
    private String name;  
  
    @OneToMany(mappedBy = "role", cascade = CascadeType.ALL, fetch = FetchType.LAZY)  
    private Set<UserRole> userRoles = new HashSet<>();  
  
    public Role() {  
  
    }  
  
    public int getRoleId() {  
  
        return roleId;  
  
    }  
  
    public void setRoleId(int roleId) {  
  
        this.roleId = roleId;  
  
    }  
  
    public String getName() {  
  
        return name;  
  
    }  
  
    public void setName(String name) {  
  
        this.name = name;  
  
    }  
  
}
```



```

public Set<UserRole> getUserRoles() {
    return userRoles;
}

public void setUserRoles(Set<UserRole> userRoles) {
    this.userRoles = userRoles;
}
}

```

### **User role:**

```

package com.icinbank.domain.security;

import com.icinbank.domain.User;
import javax.persistence.*;

@Entity
@Table(name="user_role")
public class UserRole {

    @Id

    @GeneratedValue(strategy = GenerationType.AUTO)
    private long userRoleId;

    public UserRole(User user, Role role) {
        this.user = user;
        this.role = role;
    }

    @ManyToOne(fetch = FetchType.EAGER)
    @JoinColumn(name = "user_id")
    private User user;

    @ManyToOne(fetch = FetchType.EAGER)
    @JoinColumn(name = "role_id")
    private Role role;

    public UserRole() {}
}

```

```

public long getUserRoleId() {
    return userRoleId;
}

public void setUserRoleId(long userRoleId) {
    this.userRoleId = userRoleId;
}

public User getUser() {
    return user;
}

public void setUser(User user) {
    this.user = user;
}

public Role getRole() {
    return role;
}

public void setRole(Role role) {
    this.role = role;
}
}

```

#### **DAO: Primary Account Dao**

```

package com.icinbank.dao;

import com.icinbank.domain.PrimaryAccount;
import org.springframework.data.repository.CrudRepository;

public interface PrimaryAccountDao extends CrudRepository<PrimaryAccount,Long> {

    PrimaryAccount findByAccountNumber (int accountNumber);}

```

#### **Primary Transaction Dao:**

```

package com.icinbank.dao;

import java.util.List;

import org.springframework.data.repository.CrudRepository;

```

```
import com.icinbank.domain.PrimaryTransaction;

public interface PrimaryTransactionDao extends CrudRepository<PrimaryTransaction, Long>
{
    List<PrimaryTransaction> findAll();
}
```

#### **Recipient Dao:**

```
package com.icinbank.dao;

import java.util.List;

import org.springframework.data.repository.CrudRepository;
import com.icinbank.domain.Recipient;

public interface RecipientDao extends CrudRepository<Recipient, Long> {
    List<Recipient> findAll();
    Recipient findByName(String recipientName);
    void deleteByName(String recipientName);
}
```

#### **Request Check book Dao:**

```
package com.icinbank.dao;

import java.util.List;

import org.springframework.data.repository.CrudRepository;
import com.icinbank.domain.RequestChequeBook;

public interface RequestChequeBookDao extends CrudRepository<RequestChequeBook, Long> {
    List<RequestChequeBook> findAll();
    /*RequestChequeBook findOne(Long id);*/
}
```

#### **Role Dao:**

```
package com.icinbank.dao;

import org.springframework.data.repository.CrudRepository;
import com.icinbank.domain.security.Role;

public interface RoleDao extends CrudRepository<Role, Integer> {
    Role findByName(String name);}
}
```

### **Saving Account Dao:**

```
package com.icinbank.dao;

import com.icinbank.domain.SavingsAccount;

import org.springframework.data.repository.CrudRepository;

public interface SavingsAccountDao extends CrudRepository<SavingsAccount, Long> {

    SavingsAccount findByAccountNumber (int accountNumber);

}
```

### **Saving Transaction Dao:**

```
package com.icinbank.dao;

import java.util.List;

import org.springframework.data.repository.CrudRepository;

import com.icinbank.domain.SavingsTransaction;

public interface SavingsTransactionDao extends CrudRepository<SavingsTransaction, Long>
{

    List<SavingsTransaction> findAll();

}
```

### **User Dao:**

```
package com.icinbank.dao;

import java.util.List;

import org.springframework.data.repository.CrudRepository;

import com.icinbank.domain.User;

public interface UserDao extends CrudRepository<User, Long> {

    User findByUsername(String username);

    User findByEmail(String email);

    List<User> findAll();

}
```

## **Selenium Test Automation:**

### **Admin Automation:**

```
package firsttestngpackage;
```

```

import java.util.concurrent.TimeUnit;
import org.openqa.selenium.Alert;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.BeforeSuite;
import org.testng.annotations.Test;

public class AdminAutomation {

    public String baseUrl = "http://localhost:4201";

    WebDriver driver;

    @BeforeSuite

    public void launchBrowser() {

        System.out.println("Launching chrome Browser");

        System.setProperty("webdriver.chrome.driver", "C:\\Users\\USHA\\Downloads\\ICIN-
        BANK-master\\ICIN-BANK-master\\Selenium-Test-Automation\\AdminPortalAutomation-
        master\\resources\\chromedriver.exe");

        driver = new ChromeDriver();

        driver.get(baseUrl);

        driver.manage().timeouts().implicitlyWait(5, TimeUnit.SECONDS);

    }

    @Test(priority=0) public void login_Pass() {

        driver.manage().timeouts().implicitlyWait(5, TimeUnit.SECONDS);

        driver.findElement(By.name("inputUserName")).sendKeys("madhuri");

        driver.findElement(By.name("password")).sendKeys("mad12345");

        //Login Button

        driver.findElement(By.xpath("/html/body/app-root/app-login/div/form/button")).click();
    }

```

```
String actualUrl="http://localhost:4200/user-account";  
String expectedUrl= driver.getCurrentUrl();  
if(actualUrl.equalsIgnoreCase(expectedUrl)) {  
System.out.println("Login Successful"); }  
driver.manage().window().maximize();  
}
```

```
@Test(priority=1) public void useraccount_login_enabling(){
```

```
//User Account Hyperlink
```

```
driver.findElement(By.xpath("/html/body/app-root/div/nav/div/ul/li[1]/a")).click();
```

```
//Enable Button
```

```
driver.findElement(By.xpath("/html/body/app-root/app-user-account/table/tbody/tr[1]/td[6]/button")).click();
```

```
System.out.println("Enabled Login Feature");
```

```
//Disable Button
```

```
driver.findElement(By.xpath("/html/body/app-root/app-user-account/table/tbody/tr[2]/td[7]/button")).click();
```

```
System.out.println("Disabled Login Feature");
```

```
}
```

```
@Test(priority=2) public void useraccount_features(){
```

```
//Click on the dropdown
```

```
driver.findElement(By.xpath("/html/body/app-root/app-user-account/table/tbody/tr/td[9]/select")).click();
```

```
//Select the first option
```

```
driver.findElement(By.xpath("/html/body/app-root/app-user-account/table/tbody/tr/td[9]/select/option[1]")).click();
```

```
//Set button
```

```
driver.findElement(By.xpath("/html/body/app-root/app-user-account/table/tbody/tr/td[9]/button")).click();
```

```
System.out.println("User Roles Changed");
```

```
}
```

```
@Test(priority=4) public void checkbookRequests() {
```

```
//Checkbook Request Hyperlink
```

```
driver.findElement(By.xpath("/html/body/app-root/div/nav/div/ul/li[2]/a")).click();
```

```
//Confirm Request Button
```

```
driver.findElement(By.xpath("/html/body/app-root/app-checkbook-requests/table/tbody/tr[1]/td[7]/button")).click();
```

```
System.out.println("Request Confirmed");
```

```
}
```

```
@Test(priority=3) public void authorization() {
```

```
//Authorization link
```

```
driver.findElement(By.xpath("/html/body/app-root/div/nav/div/ul/li[3]/a")).click();
```

```
//Create Account Button
```

```
driver.findElement(By.xpath("/html/body/app-root/app-authorize-registration/table/tbody/tr/td[9]/button")).click();
```

```
System.out.println("Authorized");
```

```
//Cancel Button
```

```
driver.findElement(By.xpath("/html/body/app-root/app-authorize-registration/table/tbody/tr[2]/td[10]/button")).click();
```

```
System.out.println(" Not Authorized");
```

```
}
```

```
@Test(priority=5) public void logout() {
```

```
//LogOut Button
```

```
driver.findElement(By.xpath("/html/body/app-root/div/nav/div/ul/li[4]/a")).  
click(); System.out.println("Logged Out");
```

```
}
```

```
@Test(priority=6) public void login_Fail() {  
    driver.manage().timeouts().implicitlyWait(5, TimeUnit.SECONDS);  
    driver.findElement(By.name("inputUserName")).sendKeys("madhuri");  
    driver.findElement(By.name("password")).sendKeys("mad");  
    //Login Button  
    driver.findElement(By.xpath("/html/body/app-root/app-login/div/form/button")).click();  
    Alert alert = driver.switchTo().alert();  
    alert.accept();  
    String actualUrl="http://localhost:4200/user-account";  
    String expectedUrl= driver.getCurrentUrl();  
    if(!actualUrl.equalsIgnoreCase(expectedUrl)) {  
        System.out.println("Login UnSuccessful"); }  
    driver.manage().window().maximize();  
}  
}
```

### **User Portal Testing:**

#### **Authentication:**

```
package Authentication;  
  
import org.openqa.selenium.*;  
  
import org.openqa.selenium.chrome.*;  
import org.openqa.selenium.support.ui.ExpectedConditions;  
import org.openqa.selenium.support.ui.WebDriverWait;
```



```
import java.util.concurrent.TimeUnit;
```

```
import org.junit.Test;
```

```
public class Login {
```

```
    @Test
```

```
    public void LoginTest() throws InterruptedException {
```

```
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\USHA\\Downloads\\ICIN-BANK-master\\ICIN-BANK-master\\Selenium-Test-Automation\\AdminPortalAutomation-master\\resources\\chromedriver.exe");
```

```
        WebDriver driver = new ChromeDriver();
```

```
        driver.get("http://localhost:4200/login");
```

```
        Thread.sleep(5000);
```

```
        driver.manage().timeouts().implicitlyWait(14, TimeUnit.SECONDS);
```

```
        driver.findElement(By.xpath("/html/body/app-root/app-login/div/form/div[1]/input")).sendKeys("Novina");
```

```
        driver.findElement(By.xpath("/html/body/app-root/app-login/div/form/div[2]/input")).sendKeys("123456p");
```

```
        driver.findElement(By.xpath("/html/body/app-root/app-login/div/form/div[3]/button")).click();
```

```
        try
```

```
        {
```

```
            WebDriverWait wait=new WebDriverWait(driver, 14);
```

```
            wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath("http://localhost:4200/home")));
```

```
            System.out.println("Login Successful");
```

```

}
catch(Exception e)
{
System.out.println("Error in browser!!\nPlease have a look");
}
Thread.sleep(5000);
driver.quit();
}
}

```

### **User Actions:**

#### **Check Book Request:**

```

package UserActions;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.Select;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.testng.annotations.Test;

public class ChequeBookRequest {

@Test

public void chequeBookRequest() throws InterruptedException {

System.setProperty("webdriver.chrome.driver", "C:\\\\Users\\\\USHA\\\\Downloads\\\\ICIN-
BANK-master\\\\ICIN-BANK-master\\\\Selenium-Test-Automation\\AdminPortalAutomation-
master\\resources\\chromedriver.exe");

```

```
WebDriver driver = new ChromeDriver();
```

```
driver.get("http://localhost:4200/login");
```

```
Thread.sleep(5000);
```

```
driver.manage().timeouts().implicitlyWait(14, TimeUnit.SECONDS);
```

```
driver.findElement(By.xpath(
```

```
"/html/body/app-root/app-login/div/form/div[1]/input")).sendKeys("Novina");
```

```
driver.findElement(By.xpath(
```

```
"/html/body/app-root/app-login/div/form/div[2]/input")).sendKeys("Novina123")
```

```
;
```

```
driver.findElement(By.xpath(
```

```
"/html/body/app-root/app-login/div/form/div[3]/button")).click();
```

```
try
```

```
{
```

```
WebDriverWait wait=new WebDriverWait(driver, 14);
```

```
wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath(
```

```
"/html/body/app-root/app-home/div[1]/h2"))));
```

```
System.out.println("Login Successfull");
```

```
driver.findElement(By.xpath("/html/body/app-root/nav/ul/li[4]/a")).click();
```

```
driver.findElement(By.xpath("/html/body/app-root/app-cheque-book-  
request/div/div[2]/select")).click();
```

```
driver.findElement(By.xpath("/html/body/app-root/app-cheque-book-
request/div/div[2]/select/option[1]")).click();
```

```
driver.findElement(By.xpath("/html/body/app-root/app-cheque-book-
request/div/form/button")).click();
```

```
System.out.println("Cheque Book Issued Successfully!!");
```

```
//          driver.findElement(By.xpath("/html/body/app-
root/nav/ul/li[5]/div/a[2]")).click();
```

```
//
        wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath("/html/body/app
-root/app-login/div/form/h3")));
```

```
//          System.out.println("Signed Out");
```

```
}
```

```
catch(Exception e)
```

```
{
```

```
System.out.println("Error in browser!!\nPlease have a look");
```

```
}
```

```
Thread.sleep(5000);
```

```
driver.quit();
```

```
}
```

```
}
```

### **Edit Profile:**

```
package UserActions;
```

```
import java.util.concurrent.TimeUnit;
```

```
import org.openqa.selenium.By;
```

```
import org.openqa.selenium.WebDriver;
```

```
import org.openqa.selenium.chrome.ChromeDriver;
```

```
import org.openqa.selenium.firefox.FirefoxDriver;
```

```
import org.openqa.selenium.support.ui.ExpectedConditions;
```

```
import org.openqa.selenium.support.ui.Select;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.testng.annotations.Test;
```

```
public class EditProfile {
```

```
@Test
```

```
public void EditPorfile() throws InterruptedException {
```

```
System.setProperty("webdriver.chrome.driver", "C:\\Users\\USHA\\Downloads\\ICIN-  
BANK-master\\ICIN-BANK-master\\Selenium-Test-Automation\\AdminPortalAutomation-  
master\\resources\\chromedriver.exe");
```

```
WebDriver driver= new ChromeDriver();
```

```
driver.get("http://localhost:4200/home");
```

```
/*
```

```
* System.setProperty(
```

```
* "webdriver.chrome.driver","D:\\Novina_BNP\\Simplilearn_Projects\\Project 4\\Chrome  
Driver\\chromedriver.exe"
```

```
* ); WebDriver driver = new ChromeDriver();
```

```
*
```

```
* driver.get("http://localhost:4200/login");
```

```
*/
```

```
Thread.sleep(5000);
```

```

try
{

/*
* WebDriverWait wait=new WebDriverWait(driver, 14);
* wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath(
* "/html/body/app-root/app-home/div[1]/h2")));
* System.out.println("Login Successfull");
*/

driver.manage().timeouts().implicitlyWait(14, TimeUnit.SECONDS);
driver.findElement(By.xpath("//*[@id=\"dropdown04\"]")).click();
driver.findElement(By.xpath("/html/body/app-root/nav/div/ul/li[5]/div/a[1]")).click();

driver.findElement(By.xpath("/html/body/app-root/app-edit-
profile/div[1]/form/div/div[1]/input")).sendKeys("7666854389");
driver.findElement(By.xpath("/html/body/app-root/app-edit-
profile/div[1]/form/div/div[2]/input")).sendKeys("Mumbai");
driver.findElement(By.xpath("/html/body/app-root/app-edit-
profile/div[1]/form/div/div[2]/div[1]/input")).sendKeys("novinarayudu.97@gmail.com");
driver.findElement(By.xpath("/html/body/app-root/app-edit-
profile/div[1]/form/div/div[2]/div[2]/input")).sendKeys("123P1234");
driver.findElement(By.xpath("/html/body/app-root/app-edit-
profile/div[1]/form/div/div[2]/div[3]/input")).sendKeys("123P1234");
//driver.findElement(By.xpath("/html/body/app-root/app-edit-
profile/div[1]/form/div/div[3]/button")).click();
System.out.println("Profile edited");
driver.manage().timeouts().implicitlyWait(14, TimeUnit.SECONDS);
driver.findElement(By.xpath("//*[@id=\"dropdown04\"]")).click();
driver.findElement(By.xpath("/html/body/app-root/nav/div/ul/li[5]/div/a[2]")).click();
//wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath("/html/body/app-
root/app-login/div/form/h3")));

```

```
System.out.println("Signed Out");  
}  
catch(Exception e)  
{  
System.out.println("Error in browser!!\nPlease have a look");  
}
```

```
Thread.sleep(5000);  
}  
}
```

### **Register:**

```
package UserActions;  
  
import org.openqa.selenium.By;  
import org.openqa.selenium.WebDriver;  
import org.openqa.selenium.WebElement;  
import org.openqa.selenium.chrome.ChromeDriver;  
import org.openqa.selenium.support.ui.ExpectedConditions;  
import org.openqa.selenium.support.ui.Select;  
import org.openqa.selenium.support.ui.WebDriverWait;  
import org.testng.annotations.Test;  
  
public class Register {  
  
    @Test  
    public void register() throws InterruptedException {  
  
        System.setProperty("webdriver.chrome.driver", "C:\\\\Users\\\\USHA\\\\Downloads\\\\ICIN-  
BANK-master\\\\ICIN-BANK-master\\\\Selenium-Test-Automation\\\\AdminPortalAutomation-  
master\\resources\\chromedriver.exe");  
  
        WebDriver driver = new ChromeDriver();
```

```

driver.get("http://localhost:4200/login");

Thread.sleep(5000);

try {
//WebDriverWait wait=new WebDriverWait(driver, 14);

driver.findElement(By.xpath("/html/body/app-root/app-login/div/form/div[3]/a")).click();

driver.findElement(By.xpath("/html/body/app-root/app-register/div/form/div/div[1]/input")).sendKeys("Novina");

driver.findElement(By.xpath("/html/body/app-root/app-register/div/form/div/div[2]/input")).sendKeys("Rayudu");

driver.findElement(By.xpath("/html/body/app-root/app-register/div/form/div/div[3]/input")).sendKeys("Novina");

driver.findElement(By.xpath("/html/body/app-root/app-register/div/form/div/div[4]/input")).sendKeys("123456p");

driver.findElement(By.xpath("/html/body/app-root/app-register/div/form/div/div[5]/input")).sendKeys("17/08/2020");

driver.findElement(By.xpath("/html/body/app-root/app-register/div/form/div/div[6]/input")).sendKeys("7666854389");

driver.findElement(By.xpath("/html/body/app-root/app-register/div/form/div/div[7]/input")).sendKeys("Mumbai");

Select id=new Select(driver.findElement(By.xpath("/html/body/app-root/app-register/div/form/div/div[8]/select")));

id.selectByIndex(2);

WebElement fileInput = driver.findElement(By.xpath("/html/body/app-root/app-register/div/form/div/div[9]/input"));

fileInput.sendKeys("C:\\Users\\USHA\\Desktop\\MAVURI USHASRI\\KYC\\demo.txt");

driver.findElement(By.xpath("/html/body/app-root/app-register/div/form/div/div[10]/input")).sendKeys("abcde1234");

driver.findElement(By.xpath("/html/body/app-root/app-register/div/form/div/div[11]/input")).sendKeys("novinarayudu.97@gmail.com");

driver.findElement(By.xpath("/html/body/app-root/app-register/div/form/div/div[12]/button")).click();

```



```

Thread.sleep(5000);

//wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath("/html/body/app-
root/app-login/div/form/h3"))));

System.out.println("Registration Successfull");
}

catch(Exception e) {

System.out.println("Erro in web browser\nPlease have a look");

}

Thread.sleep(5000);

driver.quit();

}

}

```

### **Transaction History:**

```

package UserActions;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.openqa.selenium.support.ui.ExpectedConditions;

import org.openqa.selenium.support.ui.WebDriverWait;

import org.testng.annotations.Test;


public class TransactionHistory {

@Test

public void TransactionHistory() throws InterruptedException {

```

```
System.setProperty("webdriver.chrome.driver", "C:\\Users\\USHA\\Downloads\\ICIN-BANK-master\\ICIN-BANK-master\\Selenium-Test-Automation\\AdminPortalAutomation-master\\resources\\chromedriver.exe");
```

```
WebDriver driver = new ChromeDriver();
```

```
driver.get("http://localhost:4200/home");
```

```
/*
```

```
* System.setProperty("webdriver.chrome.driver",
```

```
* "D:\\Novina_BNP\\Simplilearn_Projects\\Project 4\\Chrome Driver\\chromedriver.exe"
```

```
* ); WebDriver driver = new ChromeDriver();
```

```
*
```

```
* driver.get("http://localhost:4200/login");
```

```
*/
```

```
Thread.sleep(5000);
```

```
/*
```

```
* driver.manage().timeouts().implicitlyWait(14, TimeUnit.SECONDS);
```

```
* driver.findElement(By.xpath(
```

```
* "/html/body/app-root/app-login/div/form/div[1]/input")).sendKeys("Novina");
```

```
* driver.findElement(By.xpath(
```

```
* "/html/body/app-root/app-login/div/form/div[2]/input")).sendKeys("Novina123")
```

```
* ;
```

```
*
```

```
* driver.findElement(By.xpath(
```

```
* "/html/body/app-root/app-login/div/form/div[3]/button")).click();
```

```
*/
```

```
try
```

```
{
```

```

/*
* WebDriverWait wait=new WebDriverWait(driver, 14);
* wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath(
* "/html/body/app-root/app-home/div[1]/h2")));
* System.out.println("Login Successfull");
*/

driver.manage().timeouts().implicitlyWait(14, TimeUnit.SECONDS);
driver.findElement(By.xpath("/html/body/app-root/nav/ul/li[1]/a")).click();
System.out.println("Transaction History Displayed");

```

```

driver.findElement(By.xpath("//*[@id=\"navbardrop\"]")).click();
driver.findElement(By.xpath("/html/body/app-root/nav/ul/li[5]/div/a[2]")).click();
// wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath("/html/body/app-
root/app-login/div/form/h3")));
System.out.println("Signed Out");
}
catch(Exception e)
{
System.out.println("Error in browser!!\nPlease have a look");
}

Thread.sleep(5000);
driver.quit();

```

```

}
}

```

### **Transfer history:**

```
package UserActions;
```

```

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.testng.annotations.Test;

public class TransferHistory {

    @Test
    public void TransactionHistory() throws InterruptedException {

        System.setProperty("webdriver.chrome.driver", "C:\\Users\\USHA\\Downloads\\ICIN-
        BANK-master\\ICIN-BANK-master\\Selenium-Test-Automation\\AdminPortalAutomation-
        master\\resources\\chromedriver.exe");

        WebDriver driver = new ChromeDriver();

        driver.get("http://localhost:4200/home");

        Thread.sleep(5000);

        driver.manage().timeouts().implicitlyWait(14, TimeUnit.SECONDS);

        /*
        * driver.findElement(By.xpath(
        * "/html/body/app-root/app-login/div/form/div[1]/input")).sendKeys("Novina");
        * driver.findElement(By.xpath(

```

```

* "/html/body/app-root/app-login/div/form/div[2]/input")).sendKeys("Novina123")
* ;
*
* driver.findElement(By.xpath(
* "/html/body/app-root/app-login/div/form/div[3]/button")).click();
*/
try
{
/*
* WebDriverWait wait=new WebDriverWait(driver, 14);
* wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath(
* "/html/body/app-root/app-home/div[1]/h2"))));
* System.out.println("Login Successfull");
*/

driver.manage().timeouts().implicitlyWait(14, TimeUnit.SECONDS);
driver.findElement(By.xpath("/html/body/app-root/nav/ul/li[2]/a")).click();
System.out.println("Transfer History Displayed");

//          driver.findElement(By.xpath("/html/body/app-
root/nav/ul/li[5]/div/a[2]")).click();
//
//          wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath("/html/body/app
-root/app-login/div/form/h3"))));
//          System.out.println("Signed Out");
}
catch(Exception e)
{
System.out.println("Error in browser!!\nPlease have a look");
}

```

```
Thread.sleep(5000);
```

```
driver.quit();
```

```
}
```

```
}
```

### **Transfer Money:**

```
package UserActions;
```

```
import java.util.concurrent.TimeUnit;
```

```
import org.openqa.selenium.By;
```

```
import org.openqa.selenium.WebDriver;
```

```
import org.openqa.selenium.chrome.ChromeDriver;
```

```
import org.openqa.selenium.support.ui.ExpectedConditions;
```

```
import org.openqa.selenium.support.ui.WebDriverWait;
```

```
import org.testng.annotations.Test;
```

```
public class TransferMoney {
```

```
@Test
```

```
public void TransferMoney() throws InterruptedException{
```

```
System.setProperty("webdriver.chrome.driver", "C:\\Users\\USHA\\Downloads\\ICIN-BANK-master\\ICIN-BANK-master\\Selenium-Test-Automation\\AdminPortalAutomation-master\\resources\\chromedriver.exe");
```

```
WebDriver driver = new ChromeDriver();
```

```
driver.get("http://localhost:4200/login");
```

```
Thread.sleep(5000);
```

```

driver.manage().timeouts().implicitlyWait(14, TimeUnit.SECONDS);

driver.findElement(By.xpath("/html/body/app-root/app-
login/div/form/div[1]/input")).sendKeys("Novina");

driver.findElement(By.xpath("/html/body/app-root/app-
login/div/form/div[2]/input")).sendKeys("Novina123");


driver.findElement(By.xpath("/html/body/app-root/app-
login/div/form/div[3]/button")).click();

try
{
WebDriverWait wait=new WebDriverWait(driver, 14);

wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath("/html/body/app-
root/app-home/div[1]/h2")));

System.out.println("Login Successfull");


driver.manage().timeouts().implicitlyWait(14, TimeUnit.SECONDS);

driver.findElement(By.xpath("/html/body/app-root/nav/ul/li[3]/a")).click();


driver.manage().timeouts().implicitlyWait(14, TimeUnit.SECONDS);

driver.findElement(By.xpath("/html/body/app-root/app-transfer-between-
accounts/div/form/div[3]/input")).sendKeys("123456789");

driver.findElement(By.xpath("/html/body/app-root/app-transfer-between-
accounts/div/form/div[4]/input")).sendKeys("1234");

driver.findElement(By.xpath("/html/body/app-root/app-transfer-between-
accounts/div/form/div[5]/input")).sendKeys("1000");


Thread.sleep(5000);

driver.manage().timeouts().implicitlyWait(14, TimeUnit.SECONDS);

driver.findElement(By.xpath("/html/body/app-root/app-transfer-between-
accounts/div/form/div[6]/button")).click();

wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath("/html/body/app-
root/app-home/div[1]/h2")));

```

```
System.out.println("Transfer Money Successfull");
```

```
//          driver.findElement(By.xpath("/html/body/app-  
root/nav/ul/li[5]/div/a[2]")).click();
```

```
//          wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath("/html/body/app  
-root/app-login/div/form/h3")));
```

```
//          System.out.println("Signed Out");
```

```
}
```

```
catch(Exception e)
```

```
{
```

```
System.out.println("Error in browser!!\nPlease have a look");
```

```
}
```

```
Thread.sleep(5000);
```

```
driver.quit();
```

```
}
```

```
}
```

### **Pom.xml:**

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<project xmlns="http://maven.apache.org/POM/4.0.0"  
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0  
https://maven.apache.org/xsd/maven-4.0.0.xsd">
```

```
<modelVersion>4.0.0</modelVersion>
```

```
<parent>
```

```
<groupId>org.springframework.boot</groupId>
```

```
<artifactId>spring-boot-starter-parent</artifactId>
```

```
<version>2.3.1.RELEASE</version>
```

```
<relativePath/> <!-- lookup parent from repository -->
```

```
</parent>
```



```
<groupId>com.example</groupId>
<artifactId>demo</artifactId>
<version>0.0.1-SNAPSHOT</version>
<name>ICINBank</name>
<description>ICIN Bank: Online Banking</description>
```

```
<properties>
<java.version>1.8</java.version>
<maven-jar-plugin.version>3.1.1</maven-jar-plugin.version>
</properties>
```

```
<dependencies>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-web</artifactId>
</dependency>
<!-- https://mvnrepository.com/artifact/org.springframework.boot/spring-boot-starter-
data-jpa -->
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
<!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java -->
<dependency>
  <groupId>org.seleniumhq.selenium</groupId>
  <artifactId>selenium-java</artifactId>
  <version>3.141.59</version>
</dependency>
```

```
<dependency>
```

```
<groupId>org.hibernate</groupId>
<artifactId>hibernate-core</artifactId>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-thymeleaf</artifactId>
</dependency>
<!-- https://mvnrepository.com/artifact/org.hibernate.javax.persistence/hibernate-jpa-2.0-
api -->
<dependency>
<groupId>org.hibernate.javax.persistence</groupId>
<artifactId>hibernate-jpa-2.1-api</artifactId>
<version>1.0.0.Final</version>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-jdbc</artifactId>
</dependency>
<!-- https://mvnrepository.com/artifact/org.springframework.security/spring-security-core -
->
<dependency>
<groupId>org.springframework.security</groupId>
<artifactId>spring-security-core</artifactId>
</dependency>
<!-- https://mvnrepository.com/artifact/org.springframework.boot/spring-boot-starter-test
-->
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-test</artifactId>
<scope>test</scope>
```

</dependency>

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-web</artifactId>

</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework.security/spring-security-config -->

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-config</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-cache</artifactId>

</dependency>

<dependency>

<groupId>net.sf.ehcache</groupId>

<artifactId>ehcache-core</artifactId>

<version>2.6.10</version>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

```

        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-maven-plugin</artifactId>
    </plugin>
</plugins>
</build>
</project>

```

### **testing.xml:**

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="Suite">
    <test thread-count="5" name="Test">
        <classes>
            <class name="com.icinbank.RegistrationForm"/>
            <class name="com.icinbank.Profiledisplay"/>
            <class name="com.icinbank.Deposit"/>
            <class name="com.icinbank.RequestChequebook"/>

            <class name="com.icinbank.TransferbtwAccounts"/>
            <class name="com.icinbank.LoginForm"/>
            <class name="com.icinbank.AddRecipient"/>
            <class name="com.icinbank.Withdraw"/>
        </classes>
    </test> <!-- Test -->
</suite> <!-- Suite -->

```

### **Pom.xml1:**

```

project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">

    <modelVersion>4.0.0</modelVersion>

```

<groupId>AdminPortalAutomation</groupId>

<artifactId>AdminPortalAutomation</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<dependency>

    <groupId>org.testng</groupId>

    <artifactId>testng</artifactId>

    <version>6.14.3</version>

    <scope>test</scope>

</dependency>

<dependency>

    <groupId>org.seleniumhq.selenium</groupId>

    <artifactId>selenium-java</artifactId>

    <version>3.141.59</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.webjars/jquery -->

<dependency>

    <groupId>org.webjars</groupId>

    <artifactId>jquery</artifactId>

    <version>3.5.1</version>

</dependency>

</dependencies>

<build>

    <sourceDirectory>src</sourceDirectory>

    <plugins>

        <plugin>

```

        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-compiler-plugin</artifactId>
        <version>3.8.1</version>

        <configuration>
            <source>1.8</source>
            <target>1.8</target>
            <fork>true</fork>
            <executable>C:\Program Files\Java\jdk1.8.0_171\bin\javac.exe</executable>
        </configuration>
    </plugin>

    <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-surefire-plugin</artifactId>
        <version>3.0.0-M3</version>
    </plugin>
</plugins>
</build>
</project>

```

### **Pom.xml2:**

```

<project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
    https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>UserPortalTesting</groupId>
    <artifactId>UserPortalTesting</artifactId>
    <version>0.0.1-SNAPSHOT</version>

```

<name>UserPortalAuthentication</name>  
<description>User portal testing</description>

<dependencies>

<dependency>  
<groupId>org.seleniumhq.selenium</groupId>  
<artifactId>selenium-java</artifactId>  
<version>3.141.59</version>  
</dependency>

<dependency>  
<groupId>org.webjars</groupId>  
<artifactId>jquery</artifactId>  
<version>3.5.1</version>  
</dependency>

<dependency>  
<groupId>junit</groupId>  
<artifactId>junit</artifactId>  
<version>4.12</version>  
<scope>test</scope>  
</dependency>

<dependency>  
<groupId>org.slf4j</groupId>  
<artifactId>slf4j-api</artifactId>  
<version>1.7.5</version>  
</dependency>

<dependency>

```
<groupId>org.slf4j</groupId>  
<artifactId>slf4j-log4j12</artifactId>  
<version>1.7.5</version>  
</dependency>
```

```
</dependencies>
```

```
</project>
```