

----- loan eligibility .jsp

```
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags"%>
<%@ taglib prefix="sf" uri="http://www.springframework.org/tags/form"%>
<%@ taglib uri="http://www.springframework.org/tags/form" prefix="form"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1" isELIgnored="false"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>

<head>
<title>Loan Bazaar : Loan Eligibility Check</title>
</head>

<body style="background-color: rgb(200, 250, 300)">

    <!-- ADD YOUR CODE HERE -->
    <center>
        <h3 style="color: blue">Loan Bazaar - Find the Best Loan For
You</h3>
        <form:form modelAttribute="customerDetails"
action="eligibilityCheck"
method="post" id="loanEligibilityForm">
            <table>
                <tr>
                    <td>Name</td>
                    <td><form:input type="text" path="name" id="name"
/></td>
                    <td><form:errors path="name"
cssStyle="color:red"></form:errors></td>
                </tr>
                <tr>
                    <td>Gender</td>
                    <td><form:radiobuttons path="gender"
items="${genderList}" /></td>
                    <td><form:errors path="gender"
cssStyle="color:red"></form:errors>
                    </td>
                </tr>
                <tr>
                    <td>Email</td>
                    <td><form:input type="text" path="email"
id="email" /></td>
                    <td><form:errors path="email"
cssStyle="color:red" ></form:errors></td>
                </tr>
                <tr>
                    <td>Customer City</td>
                    <td><form:select path="customerCity"
items="${cityList}"
id="customerCity"/></td>
                    <td><form:errors path="customerCity"
cssStyle="color:red"></form:errors></td>
```

```

        </tr>
        <tr>
            <td>Employment Type</td>
            <td><form:select path="employmentType"
items="${employeeList}"
                                id="employmentType" /></td>
            <td><form:errors path="employmentType"
cssStyle="color:red"></form:errors></td>
        </tr>
        <tr>
            <td>Monthly Income in INR</td>
            <td><form:input type="text" path="monthlyIncome"
                                id="monthlyIncome" /></td>
            <td><form:errors path="monthlyIncome"
cssStyle="color:red"></form:errors></td>
        </tr>
        <tr>
            <td>Desired Loan Amount in INR</td>
            <td><form:input type="text"
path="desiredLoanAmount"
                                id="desiredLoanAmount" /></td>
            <td><form:errors path="desiredLoanAmount"
cssStyle="color:red"></form:errors></td>
        </tr>
    </table>
    <button type="submit" id="submit">Submit</button>
</form:form>
</center>

</body>

</html>

```

----- Results.jsp

```

<%@ taglib prefix="spring" uri="http://www.springframework.org/tags"%>
<%@ taglib prefix="sf" uri="http://www.springframework.org/tags/form"%>
<%@ taglib uri="http://www.springframework.org/tags/form" prefix="form"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1" isELIgnored="false"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>

<head>
<title>Loan Bazaar : Loan Eligibility Check Results</title>
</head>

<body style="background-color: rgb(200, 250, 300)">

    <!-- ADD YOUR CODE HERE -->
    <h3>${msg}</h3>
    <form:form id="results">
        <c:if test="${not empty data}">
            <table id="resultsTable" border="1">

```

```

        <tr>
            <td>Bank Name</td>
            <td>Loan Product Name</td>
            <td>Max Eligible Loan Amount in Rupees</td>
            <td>Tenure</td>
            <td>Interest</td>
            <td>EMI in Rupees</td>
        </tr>
        <c:forEach items="${data}" var="a">
            <tr>
                <td>${a.bankName}</td>
                <td>${a.loanProductName}</td>
                <td>${a.maxLoanAmount}</td>
                <td>${a.tenure}</td>
                <td>${a.interest}</td>
                <td>${a.monthlyInstallment}</td>
            </tr>
        </c:forEach>
    </table>
</c:if>
</form:form>
</html>

```

----- Error.jsp

```

<%@ taglib prefix="spring" uri="http://www.springframework.org/tags"%>
<%@ taglib prefix="sf" uri="http://www.springframework.org/tags/form"%>
<%@ taglib uri="http://www.springframework.org/tags/form" prefix="form"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1" isELIgnored="false"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>

<head>
<title>Loan Bazaar : Loan Eligibility Errors</title>
</head>

<body style="background-color: rgb(200, 250, 300)">

<!-- ADD YOUR CODE HERE -->
<form:form id="error">
<h3>Unable to retrieve loan information. Below are the error details:</h3>
<h3>Response Code: ${code }</h3>
<h3>Error Message: ${message }</h3>
<h3>Error Occurred on: ${curtime }</h3>
</form:form>

</body>
</html>

```

----- Loan EligibilityController.java

```
/*  
*****
```

* This class LoanEligibilityController is control the views and model objects

*

* DO NOT CHANGE THE CLASS NAME, PUBLIC METHODS, SIGNATURE OF METHODS, EXCEPTION CLAUSES, RETURN TYPES

* YOU CAN ADD ANY NUMBER OF PRIVATE METHODS TO MODULARIZE THE CODE

* DO NOT SUBMIT THE CODE WITH COMPILATION ERRORS

* CHANGE THE RETURN TYPE FROM NULL OF THE METHODS ONCE YOU BUILT THE LOGIC

* DO NOT ADD ANY ADDITIONAL EXCEPTIONS IN THE THROWS CLAUSE OF THE METHOD. IF NEED BE,

* YOU CAN CATCH THEM AND THROW ONLY THE APPLICATION SPECIFIC EXCEPTION AS PER EXCEPTION CLAUSE

* ADD REQUEST MAPPING URI AND RETURN TYPE AS PER DESIGN DOCUMENT

*

```
*****  
*****/
```

```
package com.cts.loanbazaar.loaneligibility.controller;
```

```
import java.util.ArrayList;
```

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

```
import javax.servlet.http.HttpServletRequest;
```

```
import javax.servlet.http.HttpServletResponse;
```

```
import javax.validation.Valid;
```

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

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

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

```
import org.springframework.validation.BindingResult;
```

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

```

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

import com.cts.loanbazaar.loaneligibility.exception.ApplicationException;
import com.cts.loanbazaar.loaneligibility.model.CustomerDetails;
import com.cts.loanbazaar.loaneligibility.model.LoanProduct;
import com.cts.loanbazaar.loaneligibility.service.LoanEligibilityService;

/**
 * Loan Eligibility Controller
 *
 */
@Controller
public class LoanEligibilityController {

    @Autowired
    LoanEligibilityService loanEligibilityService;

    /**
     * @param model
     * @return String
     */
    @RequestMapping(value = "/home", method = RequestMethod.GET)
    public String showHomePage(Model model) {
        CustomerDetails customerDetails = new CustomerDetails();
        model.addAttribute("customerDetails", customerDetails);
        return "loanEligibility";
    }

    /**
     * @param model

```

```

* @param request
* @param response
* @param customerDetails
* @param result
* @return String
* @throws ApplicationException
*/

@RequestMapping(value = "/eligibilityCheck", method = RequestMethod.POST)
public String getLoanProducts(Model model, HttpServletRequest request,
    HttpServletResponse response,
        @Valid CustomerDetails customerDetails, BindingResult result) throws
    ApplicationException {
    if (!result.hasErrors()) {

        List<LoanProduct> data = new ArrayList<LoanProduct>();
        data = loanEligibilityService.checkEligibleLoanProducts(customerDetails);
        if (data.isEmpty()) {
            model.addAttribute("msg", "Sorry, no loan products matching your
profile.");
        } else {
            model.addAttribute("data", data);
            model.addAttribute("msg", "Congratulations. You are Eligible for the
below Loan Products:");
        }
    }

    return (result.hasErrors()?"loanEligibility":"results");

}

/**
* @return List<String>
*/

```

```

@ModelAttribute("cityList")

public List<String> getCities() {

    List<String> cityList = new ArrayList<String>();

    cityList.add("");

    cityList.add("Chennai");

    cityList.add("Mumbai");

    cityList.add("Bangalore");

    cityList.add("Delhi");

    // cityList.add("Pune");

    cityList.add("Kolkatta");

    // cityList.add("Delhi");

    // cityList.add("Bangalore");

    return cityList;

}

/**
 * @return List<String>
 */

@ModelAttribute("employeeList")

public List<String> getEmploymentTypes() {

    List<String> employeeList = new ArrayList<String>();

    employeeList.add("");

    employeeList.add("Salaried");

    employeeList.add("Self-Employed");

    employeeList.add("Contractual Employment");

    employeeList.add("Student");

    employeeList.add("Pensioner");

    return employeeList;

}

/**

```

```

        * @return List<String>

        */

        @ModelAttribute("genderList")

        public List<String> getGenderOptions() {

                List<String> genderList = new ArrayList<String>();

                genderList.add("Male");

                genderList.add("Female");

                return genderList;

        }

}

```

-----customer details.java

```

/*****
*****

```

```

* This class CustomerDetails is the value object that must be binded to loanEligibility view
*
* DO NOT CHANGE THE CLASS NAME, PUBLIC METHODS, SIGNATURE OF METHODS, EXCEPTION
CLAUSES, RETURN TYPES
* YOU CAN ADD ANY NUMBER OF PRIVATE METHODS TO MODULARIZE THE CODE
* DO NOT SUBMIT THE CODE WITH COMPILATION ERRORS
* CHANGE THE RETURN TYPE FROM NULL OF THE METHODS ONCE YOU BUILT THE LOGIC
* DO NOT ADD ANY ADDITIONAL EXCEPTIONS IN THE THROWS CLAUSE OF THE METHOD. IF NEED
BE,
* YOU CAN CATCH THEM AND THROW ONLY THE APPLICATION SPECIFIC EXCEPTION AS PER
EXCEPTION CLAUSE
*

```

```

*****
*****/

```

```

package com.cts.loanbazaar.loaneligibility.model;

```

```

import javax.validation.constraints.Size;

```



```
import javax.validation.constraints.Email;
import javax.validation.constraints.NotEmpty;
import javax.validation.constraints.NotNull;

public class CustomerDetails {

    /// DO NOT CHANGE THE VARIABLE NAMES OR DATA TYPES OR ACCESS SPECIFIERS

    @NotNull(message = "must not be null")
    @Size(min = 4, max = 30, message = "size must be between 4 and 30")
    public String name;

    @NotNull(message = "must not be null")
    public String gender;

    @NotEmpty(message = "must not be empty")
    @Email(message = "must be a well-formed email address")
    public String email;

    @NotNull(message = "must not be null")
    public Double monthlyIncome;

    @NotEmpty(message = "must not be empty")
    public String customerCity;

    @NotEmpty(message = "must not be empty")
    public String employmentType;

    @NotNull(message = "must not be null")
    public Double desiredLoanAmount;

    public String getName() {
```

```
        return name;
    }
}
```

```
public void setName(String name) {
    this.name = name;
}
```

```
public String getGender() {
    return gender;
}
```

```
public void setGender(String gender) {
    this.gender = gender;
}
```

```
public String getEmail() {
    return email;
}
```

```
public void setEmail(String email) {
    this.email = email;
}
```

```
public Double getMonthlyIncome() {
    return monthlyIncome;
}
```

```
public void setMonthlyIncome(Double monthlyIncome) {
    this.monthlyIncome = monthlyIncome;
}
```

```

    public String getCustomerCity() {
        return customerCity;
    }

    public void setCustomerCity(String customerCity) {
        this.customerCity = customerCity;
    }

    public String getEmploymentType() {
        return employmentType;
    }

    public void setEmploymentType(String employmentType) {
        this.employmentType = employmentType;
    }

    public Double getDesiredLoanAmount() {
        return desiredLoanAmount;
    }

    public void setDesiredLoanAmount(Double desiredLoanAmount) {
        this.desiredLoanAmount = desiredLoanAmount;
    }
}

```

----- Error response .java

```

/*****
*****

```

* This class ErrorResponse is the value object that must be binded to error view

*

* DO NOT CHANGE THE CLASS NAME, PUBLIC METHODS, SIGNATURE OF METHODS, EXCEPTION CLAUSES, RETURN TYPES

* YOU CAN ADD ANY NUMBER OF PRIVATE METHODS TO MODULARIZE THE CODE

* DO NOT SUBMIT THE CODE WITH COMPILATION ERRORS

* CHANGE THE RETURN TYPE FROM NULL OF THE METHODS ONCE YOU BUILT THE LOGIC

* DO NOT ADD ANY ADDITIONAL EXCEPTIONS IN THE THROWS CLAUSE OF THE METHOD. IF NEED BE,

* YOU CAN CATCH THEM AND THROW ONLY THE APPLICATION SPECIFIC EXCEPTION AS PER EXCEPTION CLAUSE

*

*****/

package com.cts.loanbazaar.loaneligibility.model;

/**

*

*/

public class ErrorResponse {

/// DO NOT CHANGE THE VARIABLE NAMES OR DATA TYPES OR ACCESS SPECIFIERS

private String errorMessage;

private String requestedURI;

public String getErrorMessage() {

return errorMessage;

}

public void setErrorMessage(String errorMessage) {

this.errorMessage = errorMessage;

}

public String getRequestedURI() {

return requestedURI;

```
}
```

```
public void setRequestedURI(String requestedURI) {  
    this.requestedURI = requestedURI;  
}
```

```
}
```

----- Loan product .java

```
/*  
*****
```

* This class ErrorResponse is the value object that must be binded to results view

*

* DO NOT CHANGE THE CLASS NAME, PUBLIC METHODS, SIGNATURE OF METHODS, EXCEPTION CLAUSES, RETURN TYPES

* YOU CAN ADD ANY NUMBER OF PRIVATE METHODS TO MODULARIZE THE CODE

* DO NOT SUBMIT THE CODE WITH COMPILATION ERRORS

* CHANGE THE RETURN TYPE FROM NULL OF THE METHODS ONCE YOU BUILT THE LOGIC

* DO NOT ADD ANY ADDITIONAL EXCEPTIONS IN THE THROWS CLAUSE OF THE METHOD. IF NEED BE,

* YOU CAN CATCH THEM AND THROW ONLY THE APPLICATION SPECIFIC EXCEPTION AS PER EXCEPTION CLAUSE

*

```
*****  
*****/
```

```
package com.cts.loanbazaar.loaneligibility.model;
```

```
public class LoanProduct {
```

```
    ///DO NOT CHANGE THE VARIABLE NAMES OR DATA TYPES OR ACCESS SPECIFIERS
```

```
    public String bankName;
```

```
    public String loanProductName;
```

```
    public Double maxLoanAmount;
```

```
    public Integer tenure;
```

```
    public Double interest;
```

```
public Double monthlyInstallment;

/**
 * @return the bankName
 */
public String getBankName() {
    return bankName;
}

/**
 * @param bankName the bankName to set
 */
public void setBankName(String bankName) {
    this.bankName = bankName;
}

/**
 * @return the loanProductName
 */
public String getLoanProductName() {
    return loanProductName;
}

/**
 * @param loanProductName the loanProductName to set
 */
public void setLoanProductName(String loanProductName) {
    this.loanProductName = loanProductName;
}

/**
 * @return the maxLoanAmount
 */
public Double getMaxLoanAmount() {
    return maxLoanAmount;
}
```

```

/**
 * @param maxLoanAmount the maxLoanAmount to set
 */
public void setMaxLoanAmount(Double maxLoanAmount) {
    this.maxLoanAmount = maxLoanAmount;
}

/**
 * @return the tenure
 */
public Integer getTenure() {
    return tenure;
}

/**
 * @param tenure the tenure to set
 */
public void setTenure(Integer tenure) {
    this.tenure = tenure;
}

/**
 * @return the interest
 */
public Double getInterest() {
    return interest;
}

/**
 * @param interest the interest to set
 */
public void setInterest(Double interest) {
    this.interest = interest;
}

/**

```

```

        * @return the monthlyInstallment
        */
    public Double getMonthlyInstallment() {
        return monthlyInstallment;
    }
    /**
     * @param monthlyInstallment the monthlyInstallment to set
     */
    public void setMonthlyInstallment(Double monthlyInstallment) {
        this.monthlyInstallment = monthlyInstallment;
    }
}

}

----- service

/*****
*****

* This class LoanEligibilityService is to build the eligible loan products for the customer employment
type
*
* DO NOT CHANGE THE CLASS NAME, PUBLIC METHODS, SIGNATURE OF METHODS, EXCEPTION
CLAUSES, RETURN TYPES
* YOU CAN ADD ANY NUMBER OF PRIVATE METHODS TO MODULARIZE THE CODE
* DO NOT SUBMIT THE CODE WITH COMPILATION ERRORS
* CHANGE THE RETURN TYPE FROM NULL OF THE METHODS ONCE YOU BUILT THE LOGIC
* DO NOT ADD ANY ADDITIONAL EXCEPTIONS IN THE THROWS CLAUSE OF THE METHOD. IF NEED
BE,
* YOU CAN CATCH THEM AND THROW ONLY THE APPLICATION SPECIFIC EXCEPTION AS PER
EXCEPTION CLAUSE
*

*****/

package com.cts.loanbazaar.loaneligibility.service;

```



```
import java.util.ArrayList;
```

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

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

```
import com.cts.loanbazaar.loaneligibility.exception.ApplicationException;
```

```
import com.cts.loanbazaar.loaneligibility.model.CustomerDetails;
```

```
import com.cts.loanbazaar.loaneligibility.model.LoanProduct;
```

```
@Service
```

```
public class LoanEligibilityService {
```

```
    /**
```

```
     * @param details
```

```
     * @return List<LoanProduct>
```

```
     * @throws ApplicationException
```

```
    */
```

```
    public List<LoanProduct> checkEligibleLoanProducts(CustomerDetails details) throws  
    ApplicationException {
```

```
        //TODO add your code here
```

```
        if(details.getDesiredLoanAmount()> (12*details.getMonthlyIncome())) {
```

```
            throw new ApplicationException("Customer Not Eligible for the Loan");
```

```
        }
```

```
        List<LoanProduct> data = new ArrayList<LoanProduct>();
```

```
        if(details.getEmploymentType().length()>=4
```

```
&&details.getEmploymentType().length()<=30) {
```

```
            String employmentType = details.getEmploymentType();
```

```
            List<LoanProduct> product = new ArrayList<LoanProduct>();
```

```
            product = buildLoanProducts();
```

```

        for(int i=0;i<product.size();i++) {
            if(product.get(i).getLoanProductName().contains(employmentType)) {
                data.add(product.get(i));
            }
        }
    }
    return data;    //TODO CHANGE THIS RETURN TYPE
}

```

```

/**
 * Use the below method to test the appropriate loan products based on employment type
of the customer

```

```

 * DO NOT CHANGE THE VALUES OF THE LOAN PRODUCT DETAILS

```

```

 * @return List<LoanProduct>

```

```

 */

```

```

private List<LoanProduct> buildLoanProducts() {
    List<LoanProduct> products = new ArrayList<LoanProduct>();

    LoanProduct product1 = new LoanProduct();
    product1.setBankName("MNQ Bank");
    product1.setLoanProductName("Pensioner Pre-Approved Personal Loan");
    product1.setMaxLoanAmount(500000.00);
    product1.setMonthlyInstallment(7000.00);
    product1.setTenure(24);
    product1.setInterest(16.40);
    products.add(product1);

    LoanProduct product2 = new LoanProduct();
    product2.setBankName("PMT Bank");

```

```
product2.setLoanProductName("Student Pre-Approved Education Loan");
product2.setMaxLoanAmount(1200000.00);
product2.setMonthlyInstallment(11000.00);
product2.setTenure(48);
product2.setInterest(12.40);
products.add(product2);
```

```
LoanProduct product3 = new LoanProduct();
product3.setBankName("MNQ Bank");
product3.setLoanProductName("Pre-Approved Personal Loan for Salaried");
product3.setMaxLoanAmount(1000000.00);
product3.setMonthlyInstallment(9000.00);
product3.setTenure(36);
product3.setInterest(15.40);
products.add(product3);
```

```
LoanProduct product4 = new LoanProduct();
product4.setBankName("MNQ Bank");
product4.setLoanProductName("Pre-Approved Personal Loan for Salaried");
product4.setMaxLoanAmount(700000.00);
product4.setMonthlyInstallment(8000.00);
product4.setTenure(24);
product4.setInterest(15.20);
products.add(product4);
```

```
LoanProduct product5 = new LoanProduct();
product5.setBankName("MNQ Bank");
product5.setLoanProductName("Pre-Approved Personal Loan for Self-Employed");
product5.setMaxLoanAmount(2500000.00);
product5.setMonthlyInstallment(34000.00);
product5.setTenure(720);
```

```

        product5.setInterest(11.40);

        products.add(product5);


        return products;

    }

}

----- ApplicationException
/*****
*****

* This class ApplicationException is a user defined exception for the proposed system
*

* DO NOT CHANGE THE CLASS NAME, PUBLIC METHODS, SIGNATURE OF METHODS, EXCEPTION
CLAUSES, RETURN TYPES

* YOU CAN ADD ANY NUMBER OF PRIVATE METHODS TO MODULARIZE THE CODE

* DO NOT SUBMIT THE CODE WITH COMPILATION ERRORS

* CHANGE THE RETURN TYPE FROM NULL OF THE METHODS ONCE YOU BUILT THE LOGIC

* DO NOT ADD ANY ADDITIONAL EXCEPTIONS IN THE THROWS CLAUSE OF THE METHOD. IF NEED
BE,

* YOU CAN CATCH THEM AND THROW ONLY THE APPLICATION SPECIFIC EXCEPTION AS PER
EXCEPTION CLAUSE

* ADD CONSTRUCTORS AS NEEDED

*

*****/

package com.cts.loanbazaar.loaneligibility.exception;

/**

```

$$*/$$

```
public ApplicationException(String message) {
    super(message);
}
```

}

```

/*****
*****

```

*

*

***** /

```
package com.cts.loanbazaar.loaneligibility.exception;
```

```
import java.io.IOException;
```

```
import java.net.HttpURLConnection;
```

```
import java.net.URL;
```

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

```
import javax.servlet.http.HttpServletRequest;
```

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

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

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

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

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

```
import org.springframework.web.servlet.ModelAndView;
```

```
import org.springframework.http.HttpStatus;
```

```
import com.cts.loanbazaar.loaneligibility.model.ErrorResponse;
```

```
@ControllerAdvice
```

```
public class ExceptionHandlerControllerAdvice {
```

```
    @ExceptionHandler(ApplicationException.class)
```

```
    @ResponseStatus(HttpStatus.INTERNAL_SERVER_ERROR)
```

```
    @ResponseBody
```

```
    public ModelAndView handleResourceNotFound(final ApplicationException exception, final  
    HttpServletRequest request,
```

```
        final Model model) {
```

```
        // TODO add your code here
```

```
        int statusCode = 0;
```

```
        ErrorResponse err = new ErrorResponse();
```

```

ModelAndView mav = new ModelAndView();

err.setErrorMessage(exception.getMessage());

err.setRequestedURI("http://localhost:8085/" + request.getRequestURI());

try {

    URL url = new URL(err.getRequestedURI());

    HttpURLConnection http = (HttpURLConnection) url.openConnection();

    statusCode = http.getResponseCode();

    mav.addObject("code", statusCode);

    mav.addObject("curtime", new Date());

    mav.addObject("message", err.getErrorMessage());

    mav.setViewName("error");

} catch (IOException e) {

    System.out.println(e);

}

return mav; // TODO change the return type here

}

}

```

----- application

```

/*****
*****

```

- * This class LoanEligibilityServiceApplication is the starter class for Spring Boot
- *
- * DO NOT CHANGE THE CLASS NAME, PUBLIC METHODS, SIGNATURE OF METHODS, EXCEPTION CLAUSES, RETURN TYPES
- * YOU CAN ADD ANY NUMBER OF PRIVATE METHODS TO MODULARIZE THE CODE
- * DO NOT SUBMIT THE CODE WITH COMPILATION ERRORS
- * CHANGE THE RETURN TYPE FROM NULL OF THE METHODS ONCE YOU BUILT THE LOGIC
- * DO NOT ADD ANY ADDITIONAL EXCEPTIONS IN THE THROWS CLAUSE OF THE METHOD. IF NEED BE,

* YOU CAN CATCH THEM AND THROW ONLY THE APPLICATION SPECIFIC EXCEPTION AS PER EXCEPTION CLAUSE

*

```
*****  
*****/
```

```
package com.cts.loanbazaar.loaneligibility.main;
```

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

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

```
import org.springframework.context.annotation.ComponentScan;
```

```
@SpringBootApplication
```

```
@ComponentScan(basePackages =  
{"com.cts.loanbazaar.loaneligibility.controller","com.cts.loanbazaar.loaneligibility.model","com.cts.l  
oanbazaar.loaneligibility.service","com.cts.loanbazaar.loaneligibility.exception"})
```

```
public class LoanEligibilityServiceApplication {
```

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

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

```
    }
```

```
}
```

```
----- app.pro
```

```
#DO NOT CHANGE THE BELOW VALUES
```

```
server.port=8085
```

```
logging.level.org.springframework.web=DEBUG
```

```
#ADD YOUR CODE BELOW
```

```
spring.mvc.view.prefix = /WEB-INF/jsp/
```

```
spring.mvc.view.suffix = .jsp
```