

Student ID \_\_\_\_\_ Student Name \_\_\_\_\_

## **Web Applications Architecture and Frameworks**

**August 18 2016**

### **PRIVATE AND CONFIDENTIAL**

- 1. Allotted exam duration is 2 hours.**
- 2. Closed book/notes.**
- 3. No personal items including electronic devices (cell phones, computers, calculators, PDAs).**
- 4. Cell phones must be turned in to your proctor before beginning exam.**
- 5. No additional papers are allowed. Sufficient blank paper is included in the exam packet.**
- 6. Exams are copyrighted and may not be copied or transferred.**
- 7. Restroom and other personal breaks are not permitted.**
- 8. Total exam including questions and scratch paper must be returned to the proctor.**

**2 blank pages are provided for writing the solutions and/or scratch paper. All 2 pages must be handed in with the exam**

**BE VERY CAREFUL WITH THE GIVEN 2 HOURS AND USE YOUR TIME WISELY. THE ALLOTTED TIME IS GIVEN FOR EVERY QUESTION.**

**Write your name and student id at the top of this page.**

**Question 1: [10 points] {10 minutes}**

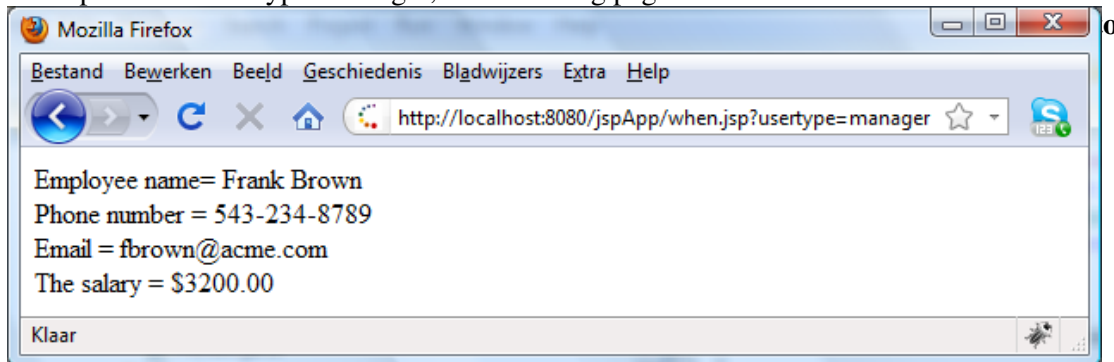
Select true or false for each of the following statements.

	True or False?
SpringMVC supports flash scope	false
The following servlet is thread safe: <code>@WebServlet("/MyServlet") public class MyServlet extends HttpServlet {     private int x=10;      protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {         response.setContentType("text/html");         PrintWriter out = response.getWriter();         out.println("&lt;html&gt;");         out.println("&lt;body&gt;");         out.println("The value of x= "+x);         out.println("&lt;/body&gt;");         out.println("&lt;/html&gt;");     } }</code>	true
The webcontainer stores Cookies in the HttpSession object	false
The HTTP POST method is idempotent	false
Spring WebFlow makes use of a front controller	true
In one Spring WebFlow we can have multiple separate flow definitions	
An HttpSession will end automatically when one closes the browser	false
The following code will do an extra roundtrip between browser and server and the URL is correct  <code>RequestDispatcher rd = request.getRequestDispatcher("page.jsp"); rd.sendRedirect(request, response);</code>	false
The following code will do an extra roundtrip between browser and server and the URL is correct  <code>request.forward("page.jsp");</code>	false
SpringMVC makes use of a front controller called DispatcherServlet	true

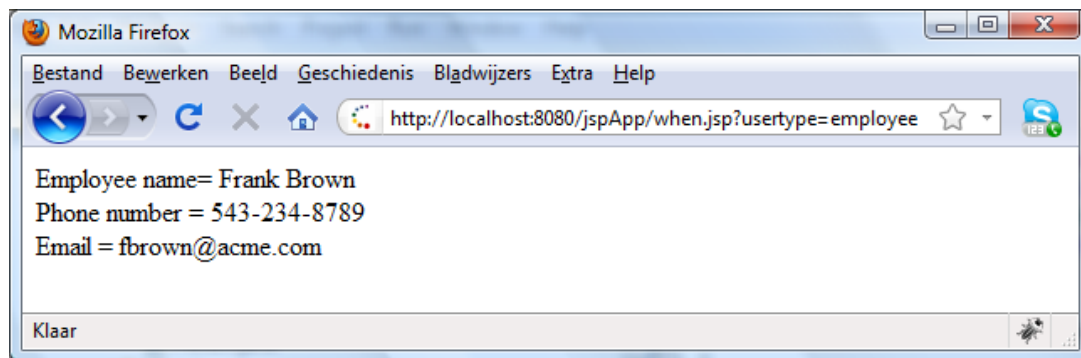
**Question 2: [10 points] {15 minutes}**

Suppose we need to write a **JSP** file with the following behavior:

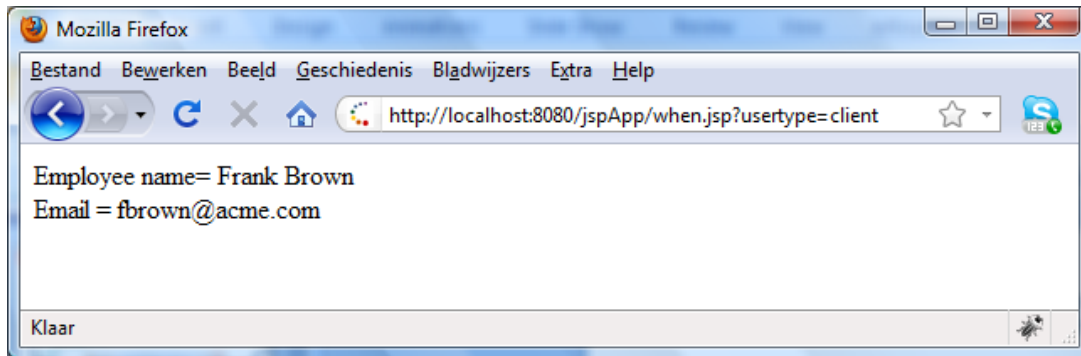
If the parameter `usertype=manager`, the following page is shown:



If the parameter `usertype=employee`, the following page is shown:



If the parameter `usertype=client`, the following page is shown:



You are **NOT** allowed to write java in your JSP page.

Complete the following given code:

```

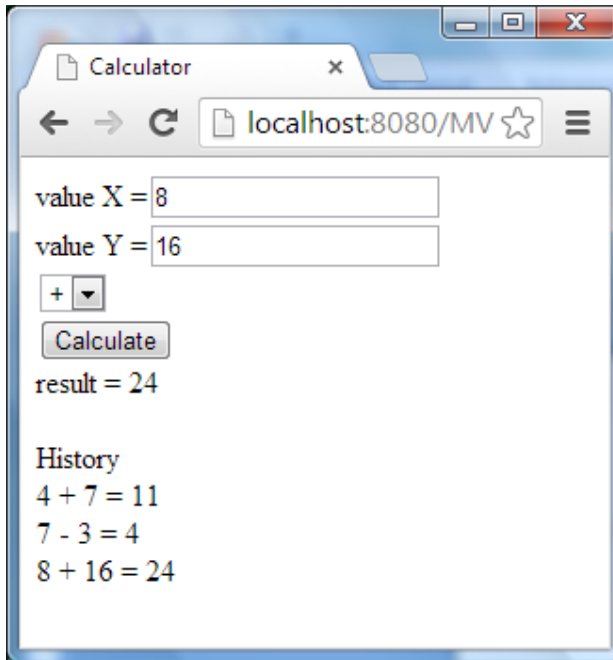
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<html>
<body>
  <% pageContext.setAttribute("employee", "Frank Brown");%>

  Employee name= ${employee}<br>
  <c:choose>
    <c:when test="${param.usertype eq 'manager' }" >
      Phone number = 543-234-8789<br/>
      Email = fbrown@acme.com<br/>
      The salary = $3200.00<br/>
    </c:when>
    <c:when test="${param.usertype eq 'employee' }" >
      Phone number = 543-234-8789<br/>
      Email = fbrown@acme.com<br/>
    </c:when>
    <c:otherwise>
      Email = fbrown@acme.com
    </c:otherwise>
  </c:choose>
</body>
</html>

```

### Question 3. MVC [25 points] {30 minutes}

Write the following calculator application:



When you enter the x and y value, and then select the operator (can be + or -) and click the Calculate button, the page shows the result of the calculation, and also shows the history of all calculations done so far with this application.

Your implementation should follow the following requirements:

1. The calculator can only **add** and **subtract** 2 numbers
2. The application should follow the correct **Model-View-Controller** principles using Servlets, JSP's and Java classes.
3. It is **not** allowed to write JAVA code in the JSP page.
4. You can assume the user only enters integers for the x and y value on the webpage. You do not need to validate the input.
5. You can see only the history of your own calculations and not the history of someone else's calculations.

Complete the partial given code:

### calculator.jsp:

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@ page import="java.util.*" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">

<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Calculator</title>
  </head>
  <body>
    <form method=GET action=CalcServlet>
      value X =<input type=text name=x><br/>
      value Y =<input type=text name=y><br/>
      <select name="operator">
        <option label="+" value="+">+</option>
        <option label="-" value="-">-</option>
      </select>
      <br/>
      <input type=submit value='Calculate'><br/>
    </form>
    result = ${result}
    <br/>
    <br/>
    History
    <br/>
    <c:forEach var="calculation" items="${history}">
      ${calculation}<br/>
    </c:forEach>
  </body>
</html>
```

## CalcServlet.java

```
public class CalcServlet extends HttpServlet {

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html");

        String strX = request.getParameter("x");
        String strY = request.getParameter("y");
        String operator = request.getParameter("operator");
        int firstnumber = Integer.parseInt(strX);
        int secondnumber = Integer.parseInt(strY);
        int result = 0;
        Calculator calculator = new Calculator();
        result = calculator.calculate(operator, firstnumber, secondnumber);

        request.setAttribute("result", result);

        HttpSession session = request.getSession();
        List<String> historylist = (List<String>) session.getAttribute("history");
        if (historylist == null) {
            historylist = new ArrayList<String>();
            session.setAttribute("history", historylist);
        }
        String calculation = strX + " " + operator + " " + strY + " = " + result;
        historylist.add(calculation);

        RequestDispatcher view = request.getRequestDispatcher("calculator.jsp");
        view.forward(request, response);
    }
}
```

## Calculator.java

```
public class Calculator {

    public int add(int x, int y) {
        return x + y;
    }

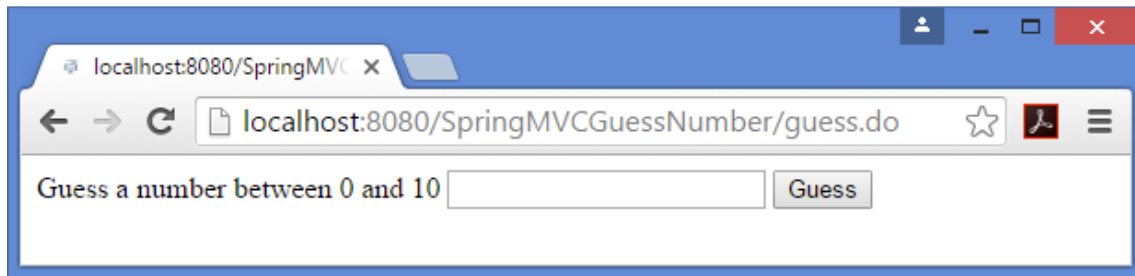
    public int subtract(int x, int y) {
        return x - y;
    }

    int calculate(String operator, int firstnumber, int secondnumber) {
        if (operator.equals("+")) {
            return add(firstnumber, secondnumber);
        }
        if (operator.equals("-")) {
            return subtract(firstnumber, secondnumber);
        }
        return 0;
    }
}
```

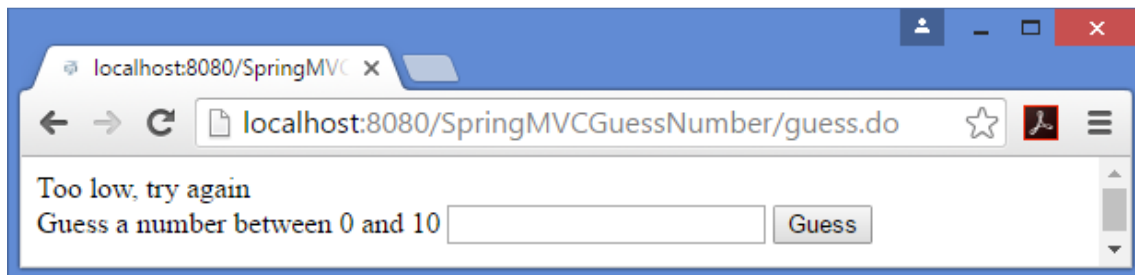


#### Question 4 [25 points ] {25 minutes}

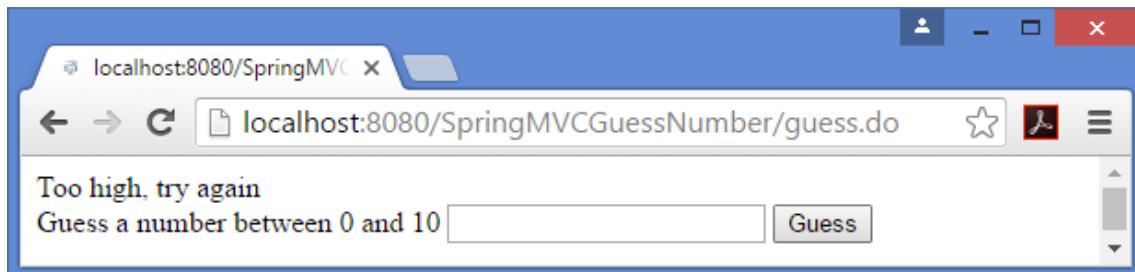
Write a simple guessnumber application with SpringMVC. When you start the application you see the following page:



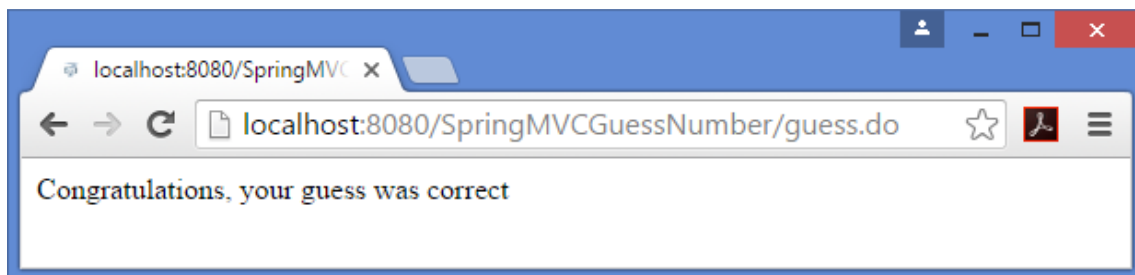
The application asks to enter a number between 0 and 10. If the guess is too low, you see the following page:



If the guess is too high, you see the following page:



If the guess is correct, you see the following page:



Complete the following given code:

```
<%@ page language="java" pageEncoding="ISO-8859-1"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<html>
<body>
    ${result}
    <c:if test="${showform eq 'true' }">
        <form action="guess.do" method="post">
            Guess a number between 0 and 10 <input type="text"
name="guess"
                                size="20"> <input type="submit"
value="Guess"><br />
            </form>
        </c:if>
    </body>
</html>

@Service
public class GuessService {
    private int toBeGuessedNumber=8;

    public String checkGuess(int guess){
        if (guess < 8){
            return "Too low, try again";
        } else if (guess > 8){
            return "Too high, try again";
        } else if (guess == 8){
            return "Congratulations, your guess was correct";
        }
        return "";
    }
}
```

```

@Controller
public class GuessController {

    @Autowired
    GuessService guessService;

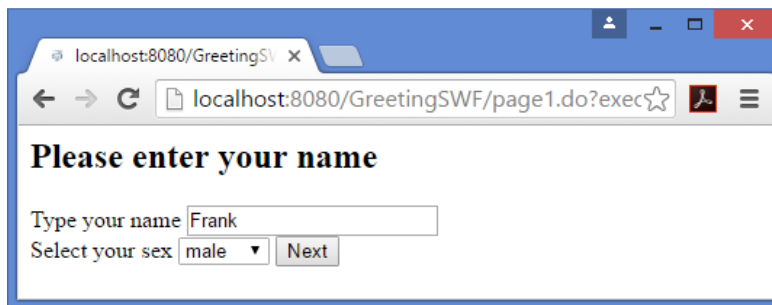
    @RequestMapping(value = "/guess.do", method =
RequestMethod.POST)
    public String entry(@RequestParam("guess") int guess, Map
model) {
        String result=guessService.checkGuess(guess);
        String tryagain="true";
        if (result.equals("Congratulations, your guess was
correct")) ){
            tryagain="false";
        }
        model.put("result", result);
        model.put("showform", tryagain);
        return "guess.jsp";
    }

    @RequestMapping(value = "/guess.do", method = RequestMethod.GET)
    public String showGuessPage(Map model){
        model.put("result", "");
        model.put("showform", "true");
        return "guess.jsp";
    }
}

```

**Question 5. [25 points] {25 minutes}**

Write the following application with **Spring WebFlow**:



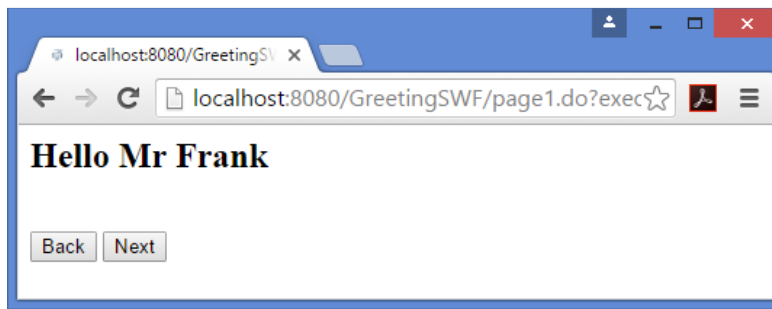
localhost:8080/GreetingSWF/page1.do?exec

**Please enter your name**

Type your name

Select your sex

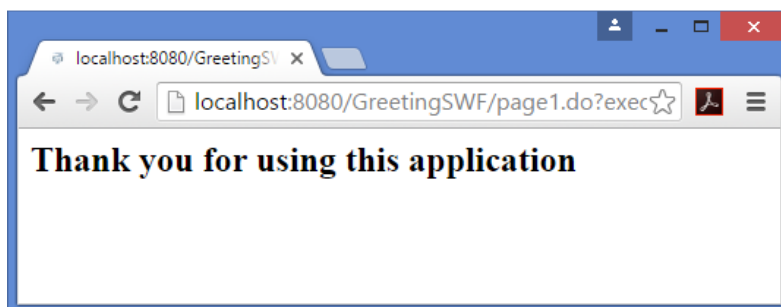
On the first page you enter your name and your sex (male or female). When you click the Next button and you selected that your sex is male, you see the next page:



localhost:8080/GreetingSWF/page1.do?exec

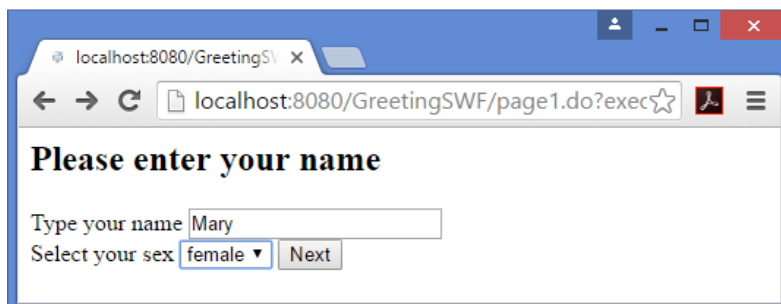
**Hello Mr Frank**

The Back button will show the previous page again, and the Next button will show the following page:



localhost:8080/GreetingSWF/page1.do?exec

**Thank you for using this application**



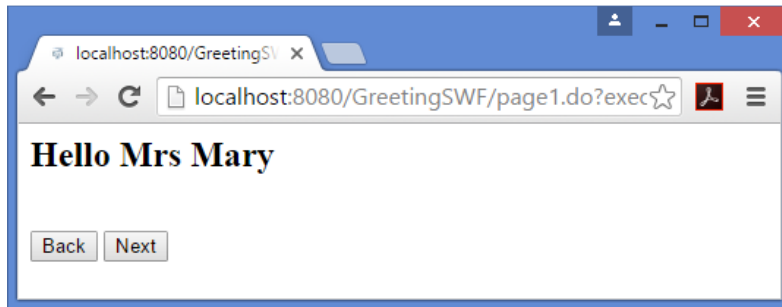
localhost:8080/GreetingSWF/page1.do?exec

**Please enter your name**

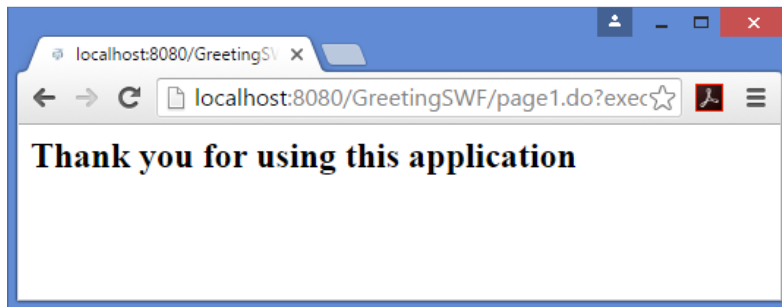
Type your name

Select your sex

But when you select female on the first page and click Next, you see the following page:



The Back button will show the previous page again, and the Next button will show the following page:



This application uses the following internal viewresolver:

```
<bean id="viewResolver"
      class="org.springframework.web.servlet.view.InternalResourceViewResolver">
  <property name="prefix" value=""/>
  <property name="suffix" value=".jsp"/>
</bean>
```

The JSP pages are given below  
Complete the webflow definition in page1.xml:

### entry.jsp:

```
<%@ page language="java" pageEncoding="ISO-8859-1"%>
<%@taglib prefix="form" uri="http://www.springframework.org/tags/form"%>
<html>
<body>
    <h2>Please enter your name</h2>
    <form:form method="post" modelAttribute="person">
Type your name <form:input path="name" size="20" />
        <br />
Select your sex <form:select path="sex">
            <form:option value="male" label="male" />
            <form:option value="female" label="female" />
        </form:select>
        <input type="submit" name="_eventId_next" value="Next" />
    </form:form>
</body>
</html>
```

### malepage.jsp

```
<%@ page language="java" pageEncoding="ISO-8859-1"%>
<%@taglib prefix="form" uri="http://www.springframework.org/tags/form"%>
<html>
<body>
<h2>Hello Mr ${person.name}</h2>
<br/>
<form:form method="post">
<input type="submit" name="_eventId_back" value="Back" />
<input type="submit" name="_eventId_next" value="Next" />
</form:form>
</body>
</html>
```

### femalepage.jsp

```
<%@ page language="java" pageEncoding="ISO-8859-1"%>
<%@taglib prefix="form" uri="http://www.springframework.org/tags/form"%>
<html>
<body>
<h2>Hello Mrs ${person.name}</h2>
<br/>
<form:form method="post">
<input type="submit" name="_eventId_back" value="Back" />
<input type="submit" name="_eventId_next" value="Next" />
</form:form>
</body>
</html>
```

### thankyoupage.jsp

```
<%@ page language="java" pageEncoding="ISO-8859-1"%>
<%@taglib prefix="form" uri="http://www.springframework.org/tags/form"%>
<html>
<body>
<h2>Thank you for using this application</h2>
<br/>
</body>
</html>
```

The flow is defined in the following page:

page1.xml

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

<flow xmlns="http://www.springframework.org/schema/webflow"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.springframework.org/schema/webflow
                          http://www.springframework.org/schema/webflow/spring-webflow-
                          2.0.xsd">

    <var name="person" class="domain.Person" />
    <view-state id="entry" view="entry" model="person">
        <transition on="next" to="checkPerson" />
    </view-state>

    <action-state id="checkPerson">
        <evaluate expression="myController.checkSex(person)" />
        <transition on="male" to="malepage" />
        <transition on="female" to="femalepage" />
    </action-state>

    <view-state id="malepage" view="malepage" model="person">
        <transition on="next" to="thankyoupage" />
        <transition on="back" to="entry" />
    </view-state>

    <view-state id="femalepage" view="femalepage" model="person">
        <transition on="next" to="thankyoupage" />
        <transition on="back" to="entry" />
    </view-state>

    <end-state id="thankyoupage" view="thankyoupage" />

</flow>
```

**Question 6. [5 points] {15 minutes}**

Describe how we can relate **Model-View-Controller** to the principles of SCI. Your answer should be about half a page, but should not exceed one page (handwritten). The number of points you get for this question depend on how well you explain the relationship between **Model-View-Controller** and the principles of SCI.

Your answer: