TmsApplication.java

```
package com;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.ComponentScan;

@SpringBootApplication
@ComponentScan("com.*")
public class TmsApplication {
	/**
	 * Starting point of the application
	 *
	 * @param args Arguments passed to the application
	 */
	public static void main(String[] args) {
		SpringApplication.run(TmsApplication.class, args);
	}
}
```

==========

InternationalizationConfig.java

```
package com.controller;

import java.util.Locale;

import org.springframework.context.MessageSource;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import
org.springframework.context.support.ReloadableResourceBundleMessageSource
;
import
org.springframework.validation.beanvalidation.LocalValidatorFactoryBean;
import org.springframework.web.servlet.LocaleResolver;
import
org.springframework.web.servlet.config.annotation.InterceptorRegistry;
import
org.springframework.web.servlet.config.annotation.WebMvcConfigurerAdapter
;
import org.springframework.web.servlet.i18n.LocaleChangeInterceptor;
import org.springframework.web.servlet.i18n.SessionLocaleResolver;

@Configuration
public class InternationalizationConfig extends WebMvcConfigurerAdapter {
	/**
	 * Set default Locale
	 *
	 * @return A bean of LocalResolver
	 */
	@Bean
	public LocaleResolver localeResolver() {
```

```
            SessionLocaleResolver slr = new SessionLocaleResolver();
            slr.setDefaultLocale(Locale.US);

            return slr;
      }

      /**
       * Set path variable name for changing language
       *
       * @return A bean of LocaleChangeInterceptor
       */
      @Bean
      public LocaleChangeInterceptor localeChangeInterceptor() {
            LocaleChangeInterceptor lci = new LocaleChangeInterceptor();
            lci.setParamName("language");

            return lci;
      }

      /**
       * Add interceptor into the registry
       */
      @Override
      public void addInterceptors(InterceptorRegistry registry) {
            registry.addInterceptor(localeChangeInterceptor());
      }

      /**
       * Set base name for messages.properties files Set default encoding
to UTF-8
       *
       * @return A bean of MessageSource
       */
      @Bean
      public MessageSource messageSource() {
            ReloadableResourceBundleMessageSource rrbms = new
ReloadableResourceBundleMessageSource();
            rrbms.setBasename("classpath:messages");
            rrbms.setDefaultEncoding("UTF-8");

            return rrbms;
      }

      /**
       * Set validation message source
       *
       * @return A bean of LocalValidatorFactoryBean
       */
      @Bean
      public LocalValidatorFactoryBean localValidatorFactoryBean() {
            LocalValidatorFactoryBean lvfb = new
LocalValidatorFactoryBean();
            lvfb.setValidationMessageSource(messageSource());
```

```
            return lvfb;
      }
}
```

==========

TaxController.java

```
package com.controller;

import java.util.Arrays;
import java.util.List;

import javax.validation.Valid;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.ModelMap;
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.model.UserClaim;
import com.service.TaxService;

@Controller
public class TaxController {
      @Autowired
      public TaxService taxService;

      /**
       * Display taxclaim.jsp page when a get request is pushed on url
       * /getTaxClaimFormPage
       *
       * @param userClaim Is the UserClaim component
       * @return taxclaim as a jsp page
       * @see UserClaim
       */
      @RequestMapping(value = "/getTaxClaimFormPage", method =
RequestMethod.GET)
      public String claimPage(@ModelAttribute("userClaim") UserClaim
userClaim) {
            return "taxclaim";
      }

      /**
       * Return result.jsp age when validation is successful Otherwise
return back to
       * taxclaim page with error message
       *
       * @param userClaim UserClaim component
       * @param result    BindingResult which validate the user input
       * @param map       ModelMap to put attribute which will be
forwarded to next
```

```
	 *                   page
	 * @return "result.jsp" page if the validation is successful
otherwise
	 *         "taxclaim.jsp" with error included
	 */
	@RequestMapping(value = "/calculateTax", method =
RequestMethod.GET)
	public String calculateTax(@Valid @ModelAttribute("userClaim")
UserClaim userClaim, BindingResult result,
			ModelMap map) {
		if (result.hasErrors()) {
			return "taxclaim";
		}

		double amount = taxService.calculateTax(userClaim);
		map.addAttribute("amount", amount);

		return "result";
	}

	/**
	 * Populate <form:select /> tag in the taxclaim.jsp page
	 *
	 * @return List of expenses
	 */
	@ModelAttribute("expenseList")
	public List<String> populateExpense() {
		return Arrays.asList("MedicalExpense", "TravelExpense",
"FoodExpense");
	}
}
```

==========

UserClaim.java

```
package com.model;

import javax.validation.constraints.NotBlank;
import javax.validation.constraints.PositiveOrZero;
import javax.validation.constraints.Size;

import org.springframework.stereotype.Component;

@Component
public class UserClaim {
	private String expenseType;
	@PositiveOrZero(message = "{error.expenseAmount.negative}")
	private double expenseAmt;
	@NotBlank(message = "{error.employeeId}")
	@Size(min = 5, message = "{error.employeeId.size}")
	private String employeeId;

	public String getExpenseType() {
		return expenseType;
```

```
	}

	public void setExpenseType(String expenseType) {
		this.expenseType = expenseType;
	}

	public double getExpenseAmt() {
		return expenseAmt;
	}

	public void setExpenseAmt(double expenseAmt) {
		this.expenseAmt = expenseAmt;
	}

	public String getEmployeeId() {
		return employeeId;
	}

	public void setEmployeeId(String employeeId) {
		this.employeeId = employeeId;
	}
}
```

==========

TaxService.java

```
package com.service;

import org.springframework.stereotype.Service;

import com.model.UserClaim;

@Service
public interface TaxService {
	/**
	 * Calculate Tax
	 *
	 * @param userClaim UserClaim bean
	 * @return Calculated tax
	 */
	public double calculateTax(UserClaim userClaim);
}
```

==========

TaxServiceImpl.java

```
package com.service;

import org.springframework.stereotype.Service;

import com.model.UserClaim;

@Service
public class TaxServiceImpl implements TaxService {
	/**
```

```
	 * Calculate the tax according to the srs
	 *
	 * @param userClaim UserClaim component to get the values
	 * @return Calculated tax
	 */
	@Override
	public double calculateTax(UserClaim userClaim) {
		String e = userClaim.getExpenseType();
		double a = userClaim.getExpenseAmt();
		double t = 0.0;

		if (e.startsWith("M")) {
			if (a <= 1000) {
				t = 15.0;
			} else if (a > 1000 && a <= 10000) {
				t = 20.0;
			} else if (a > 10000) {
				t = 25.0;
			}
		} else if (e.startsWith("T")) {
			if (a <= 1000) {
				t = 10.0;
			} else if (a > 1000 && a <= 10000) {
				t = 15.0;
			} else if (a > 10000) {
				t = 20.0;
			}
		} else if (e.startsWith("F")) {
			if (a <= 1000) {
				t = 5.0;
			} else if (a > 1000 && a <= 10000) {
				t = 10.0;
			} else if (a > 10000) {
				t = 15.0;
			}
		}

		return a * (t / 100.0);
	}
}
```