## Charity Trust

# 1.0 Introduction

# 1.1 Functional Requirements

Charity Trust is a registered non-profit organization located in Colombo City with the aim of assisting underprivileged people. It has started working for individuals who are deeply affected by poverty and social backwardness. The organization provides assistance to those in need, offers education for impoverished children, and allocates funds for good causes. In order to enhance their services, they require a web application to manage donation details.

Create a Spring MVC Spring Boot Application to fulfill this requirement. Design a show page to display donation details. When users click the '**Add Fund**' button, the application should add the donation details to the list.

# 1.2. Use Case Diagram

**View funds**

**Get fund details**

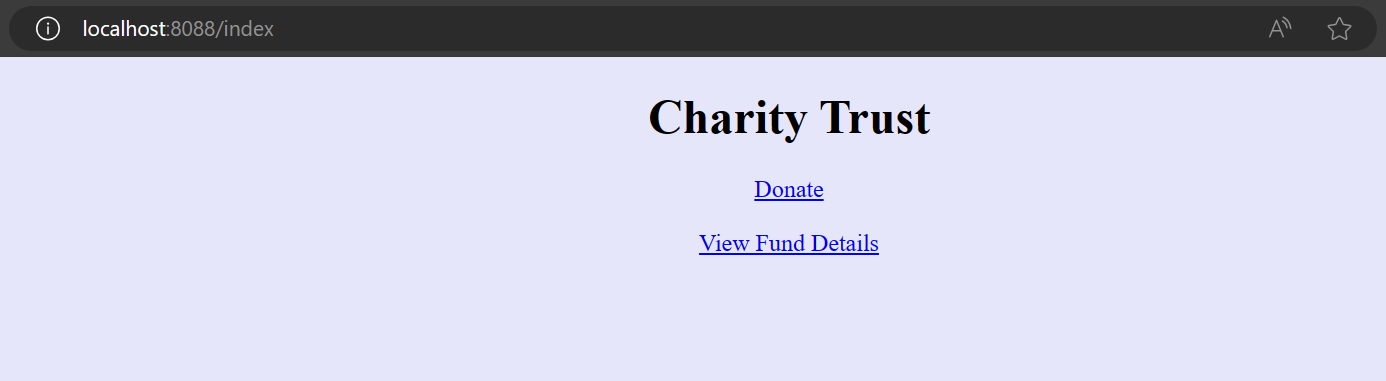
**Add Fund**

# Actor

# 2.0 Technical Specifications

# 2.1 Process Flow

* The first page of the application is **index.jsp** page, this page should have hyperlinks to navigate **showPage.jsp** and **viewList.jsp**



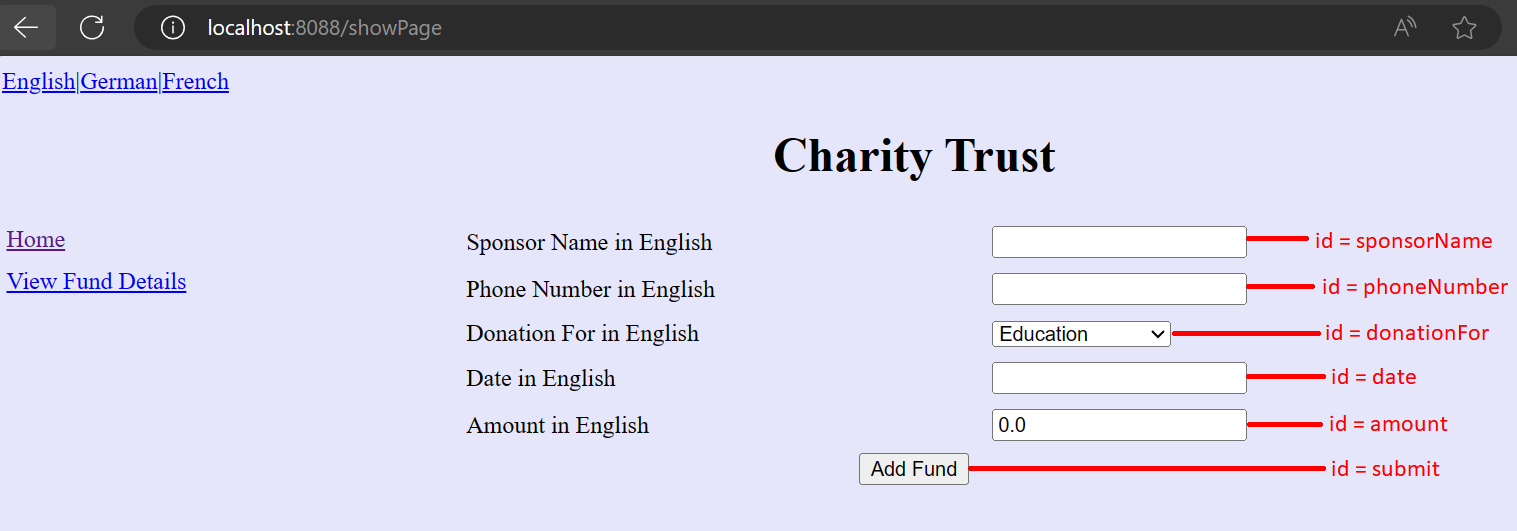
* In **showPage.jsp**, the page should have the below form components.

|  |  |  |
| --- | --- | --- |
| **Component** | **Component Type** | **Component ID** |
| Sponsor Name | Textbox | sponsorName |
| Phone Number | Textbox | phoneNumber |
| Donation For | Drop down | donationFor |
| Date | Textbox | date |
| Amount | Textbox | amount |
| Submit | Submit | submit |

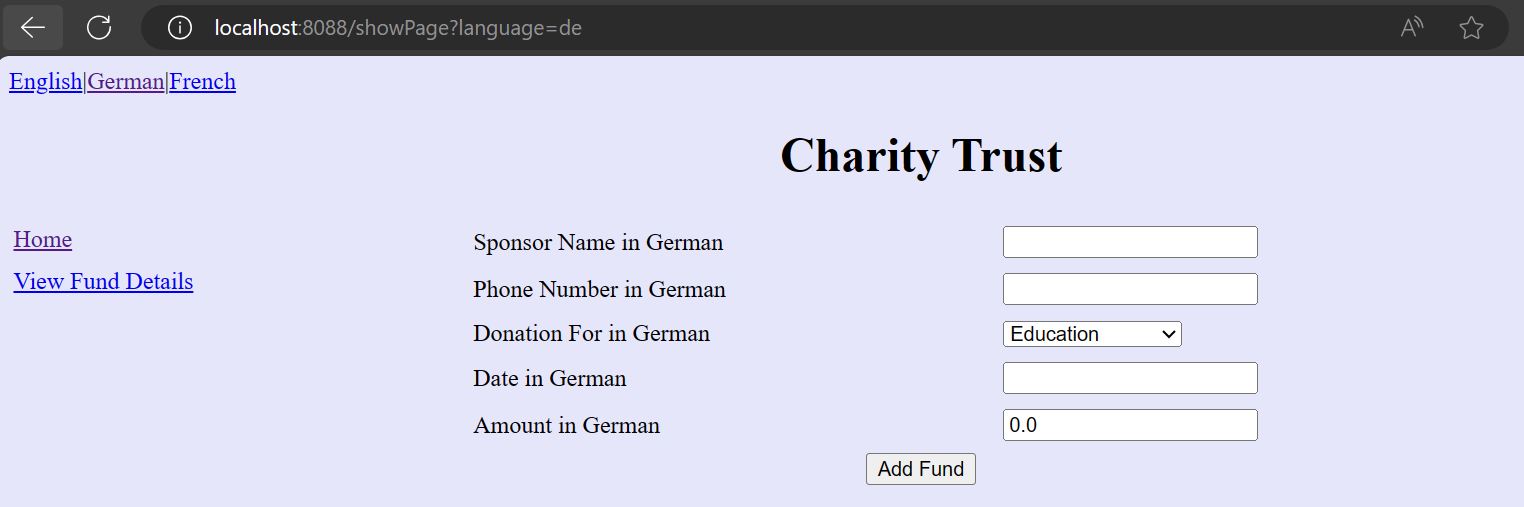
* In the same page we have the provision to choose language English, German and French. By default, the page content will be displayed in English. If we choose the language as French the label content and the error message in the page should displayed in French and if we choose the language as German the label content and the error message in the page should displayed in German. Ensure you implement the concept of Internationalization.
* In the **showPage.jsp** page, ensure you use the component type and id correctly as specified in the above table.
* The values in the **donationFor** drop-down must be auto populated from the **controller** with the values as given in the below table. They should not be populated/hardcoded inside the JSP.

|  |
| --- |
| **Donation For** |
| Education |
| Food |
| Medicine |
| Cloth |
| MedicalExpense |

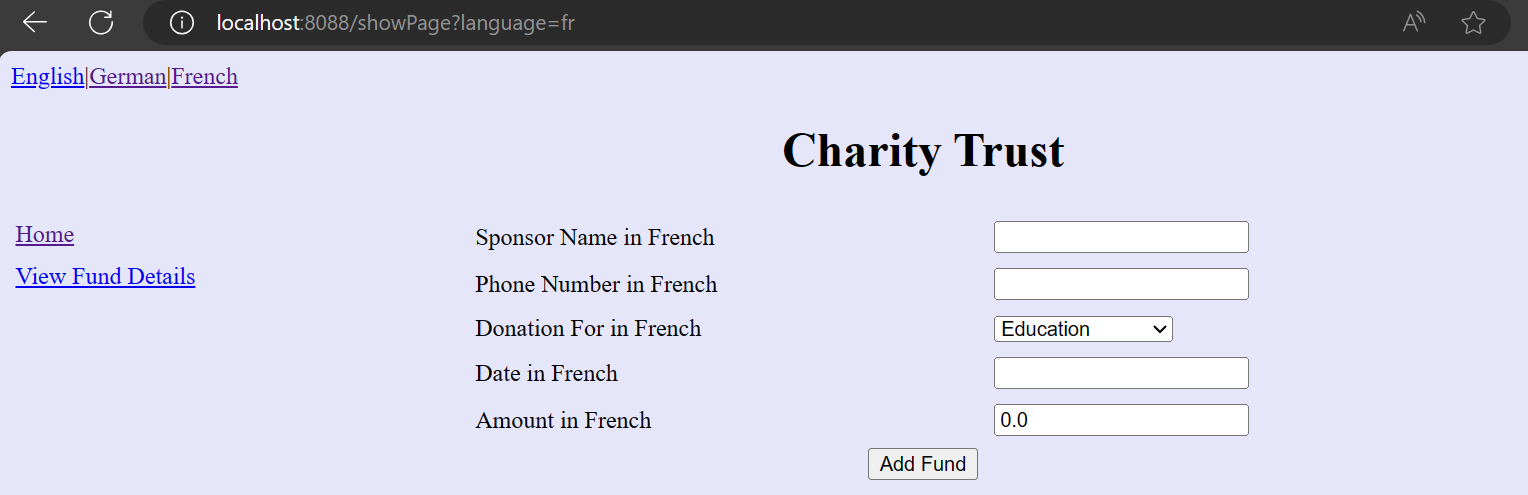
* **CharityService** class contains the attribute **fundList** with type as ArrayList<Sponsor>
* Sample **showPage.jsp** for displaying the page content in English



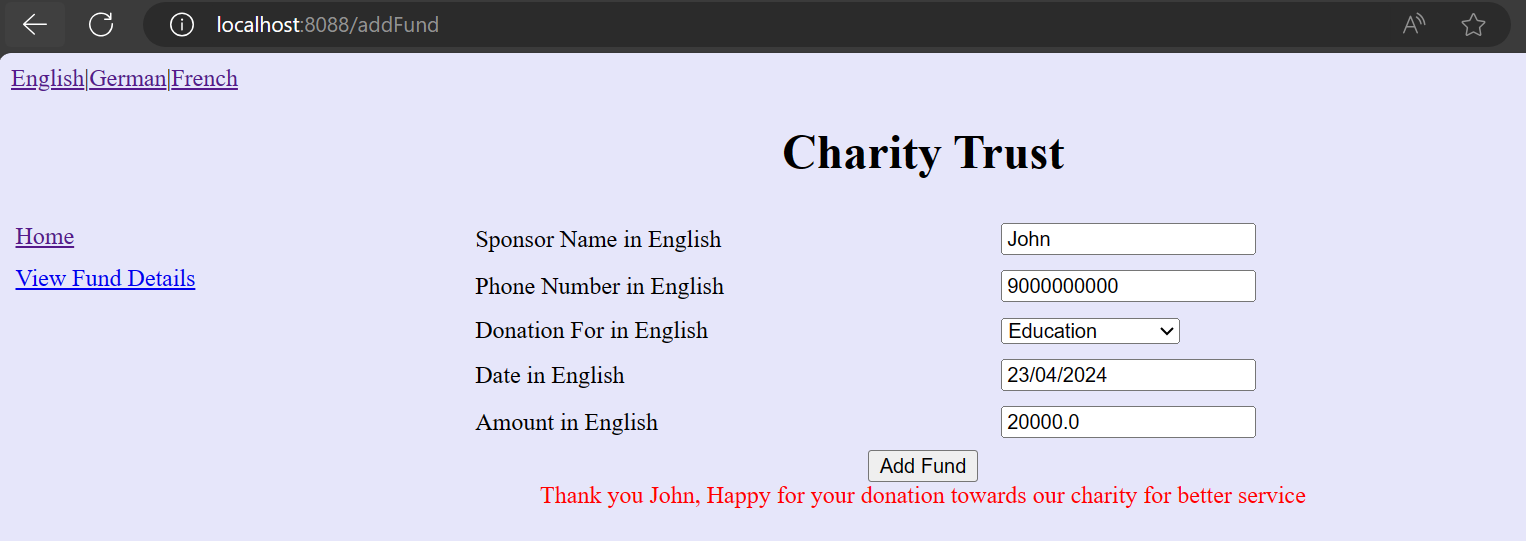
* Sample **showPage.jsp** for displaying the page content in German



* Sample **showPage.jsp** for displaying the page content in French

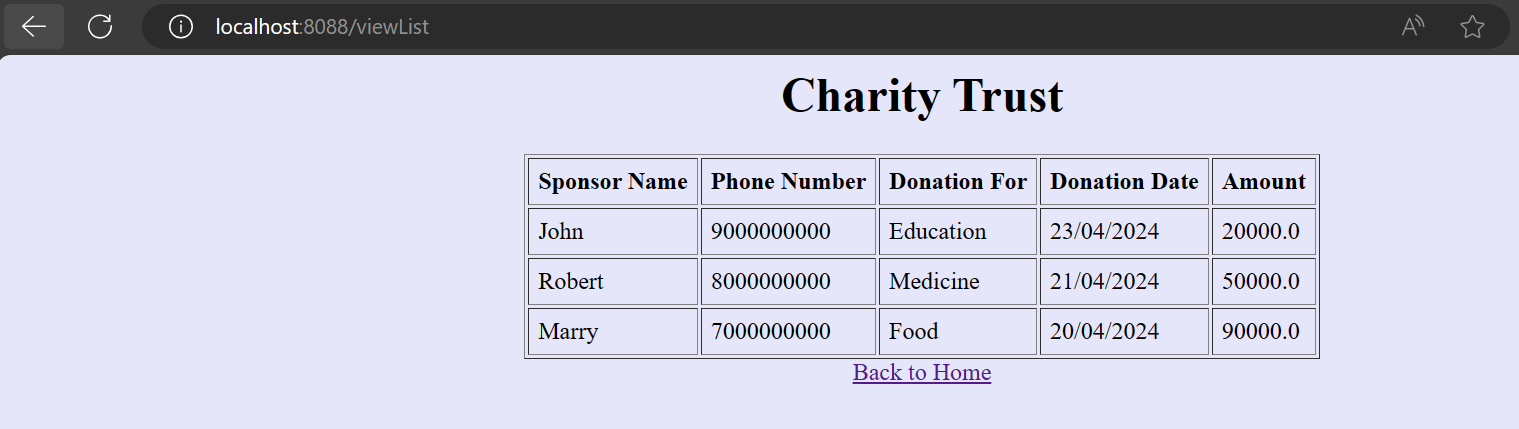


The "**Add Fund**" button, upon click and after all successful validations (Refer 2.2 for validation), should invoke the **addFundDetails** method of the **CharityService** class. This method will add fund details into the fundList.



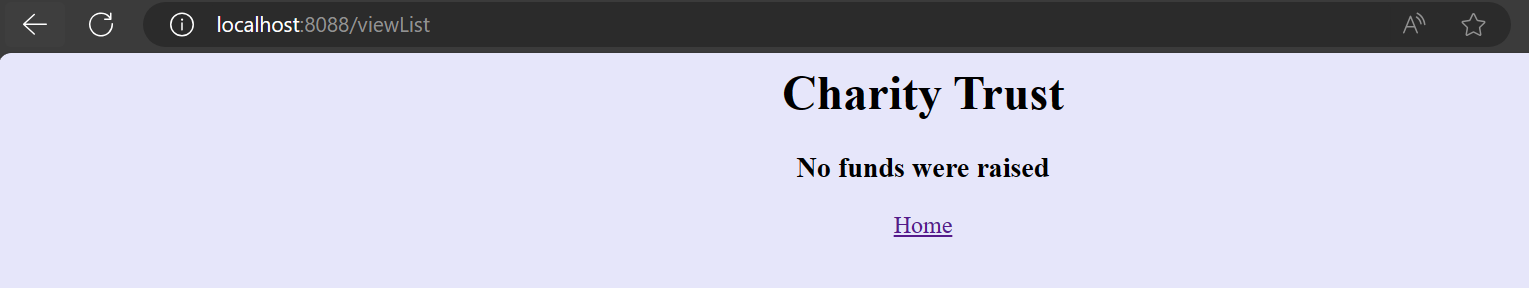
After adding the Sponsor object into the fundList, display the success message **“Thank you <<Sponsor Name>> , Happy for your donation towards our charity for better service”** in the div tag (refer the above screenshot). **div** tag with id as “**message**” is already available as part of code skeleton. While click on the hyper link H**ome**, it should navigate to **index.jsp**. And for V**iew Fund Details** hyper link , it should navigate to **viewList.jsp**

* Render the fund details list in the **viewList.jsp**. If the fundList is empty, then raise an **NoFundsRaisedException** with the message **“No funds were raised”.** Handle this exception using **ExceptionHandler** and render the message in **exceptionPage.jsp**



On clicking “**Back to home**” it should redirect to **index.jsp** page

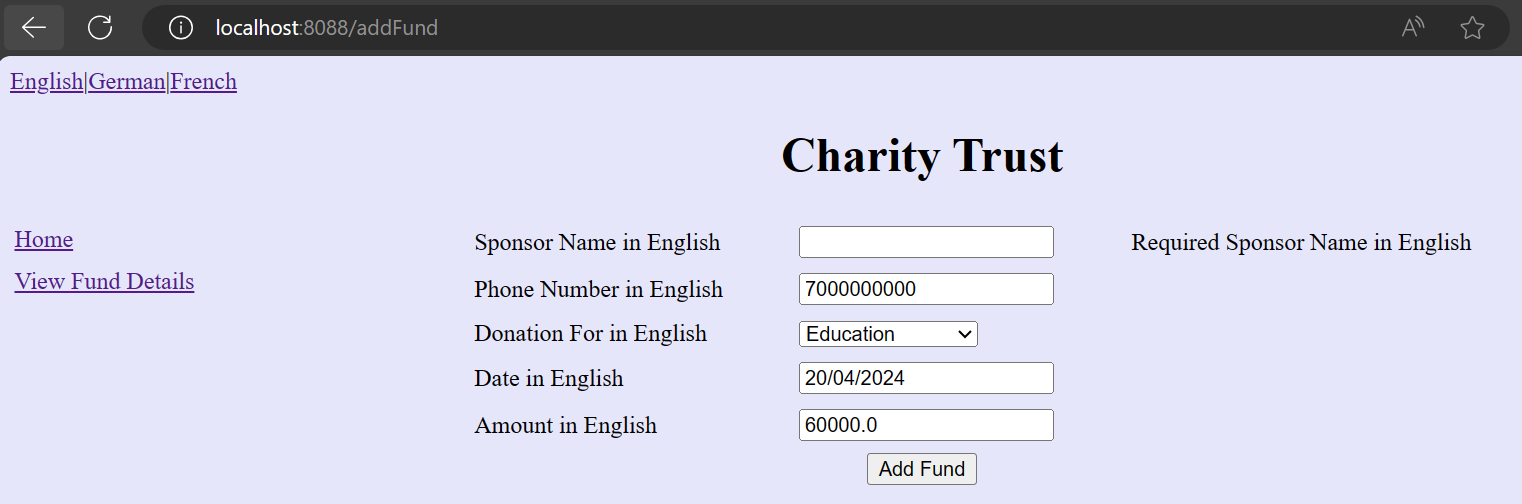
# exceptionPage.jsp (while displaying fund details, if no funds are there then render the message in **exceptionPage.jsp**)



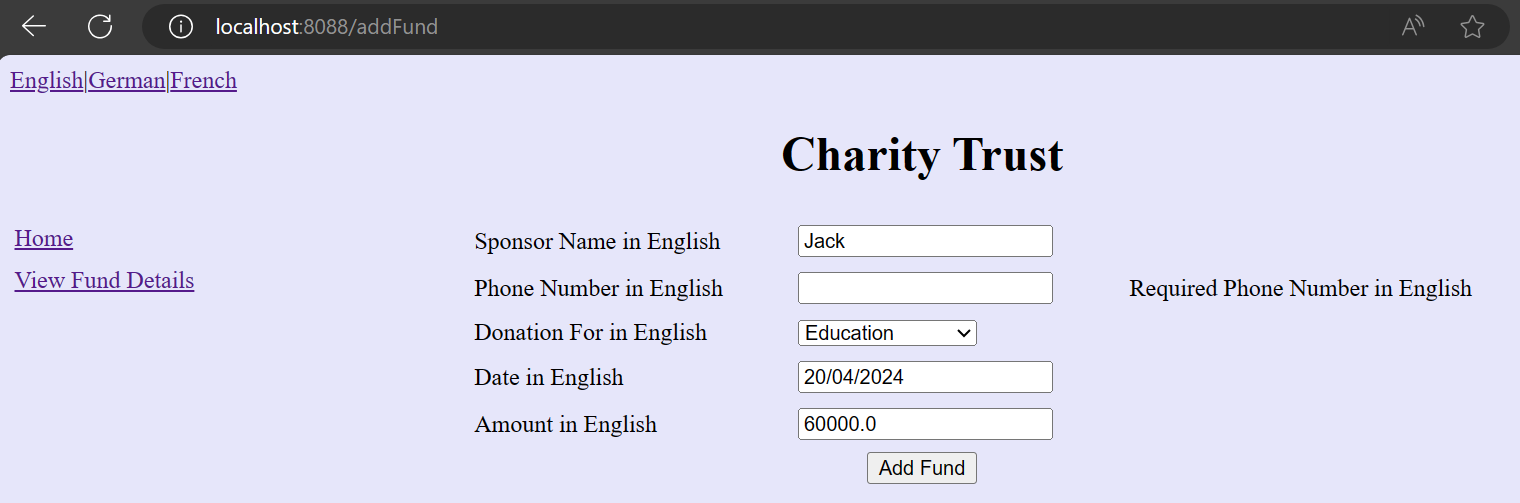
On clicking “**Home**” it should redirect to **index.jsp** page

# 2.2Business Validation

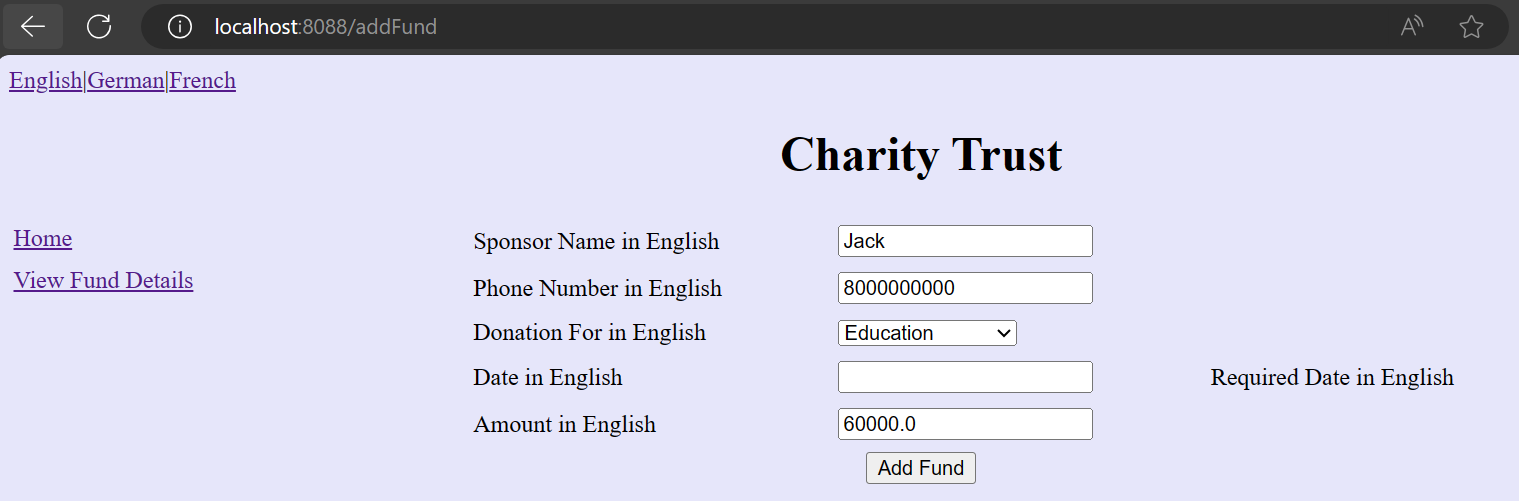
* **Rule:** Sponsor Name should not be blank
* **Message:**  Required Sponsor Name in <<locale>>
  + If validation fails, UI should display the error message - “Required Sponsor Name in <<locale>>”.



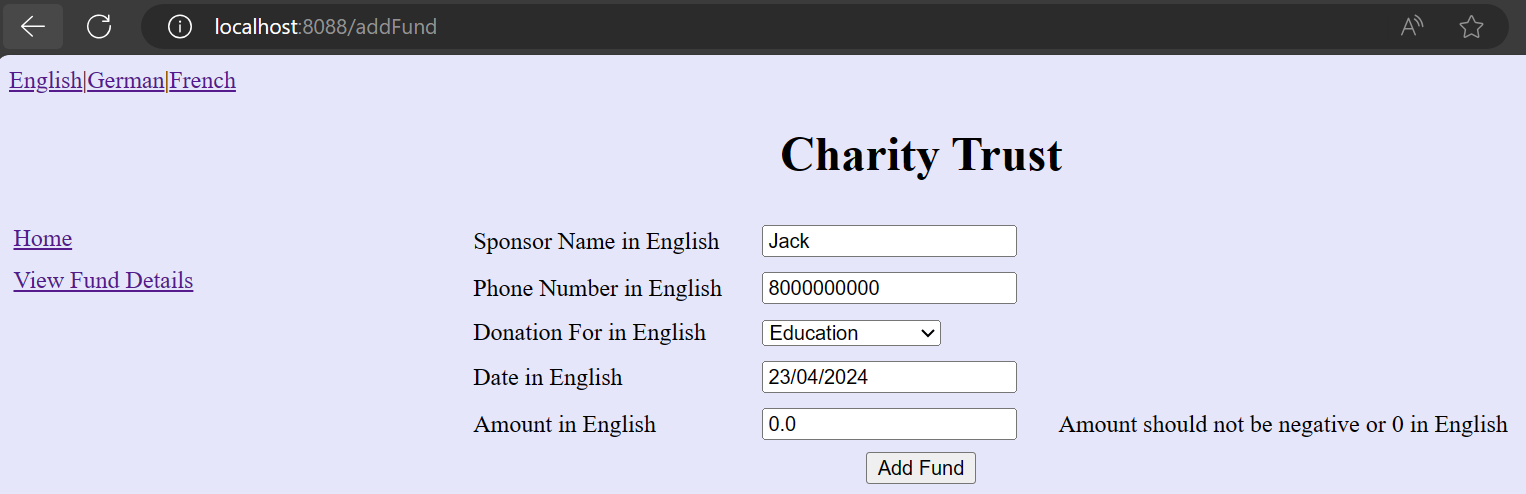
* **Rule:** Phone Number should not be blank
* **Message:** Required Phone Number in <<locale>>
  + If validation fails, UI should display the error message - “Required Phone Number in <<locale>>”.



* **Rule:** Date should not be empty
* **Message:** Required Date in <<locale>>
  + If validation fails, UI should display the error message - “Required Date in <<locale>>”.



* **Rule:** Amount should not be negative or 0
* **Message:**  Amount should not be negative or 0 in <<locale>>
  + If validation fails, UI should display the error message - “Amount should not be negative or 0 in <<locale>>”.



* **Note:** The messages have to be retrieved from the appropriate properties file. The properties file has been given as part of code skeleton. Do not change/modify the property file

# 2.2Control Flow

* By giving the request **/index** from the browser (**Eg: http://localhost:8080/index**),the user should be redirected to the **index.jsp**
* Get input for Sponsor name,phone number, donation for, date and amount
* On clicking **Add Fund** button, do the validation for UI components
* Add the fund details (Sponsor object) object into the findList
* On clicking View Fund Details, if the fundList is empty then raise **NoFundsRaisedException**. Else It should navigate to **viewList.jsp** page

**3.0Component Specification**

# 3.0.1Controller

* The **CharityController** should be configured via annotation as a Controller

|  |  |  |  |
| --- | --- | --- | --- |
| **CharityController** | | | |
| **Attribute Name** | **Attribute Type** | **Access Specifier** | **Constraints** |
| service | CharityService | private | Use annotation to autowire |

|  |  |  |  |
| --- | --- | --- | --- |
| **Method Name** | **Method Argument name:type** | **Return type** | **RequestMapping URL & Request Method** |
| showIndex |  | String | /index& GET |
| showPage | modelAttribute “donate”: Sponsor | String | /showPage & GET |
| addFundDetails | modelAttribute “donate”: Sponsor, result:BindingResult, model:ModelMap | String | /addFund & POST |
| viewFundList | model:ModelMap | String | /viewList & GET |
| populateDonationList |  | List<String> | Should be annotated with ModelAttribute with name “donationFor” |
| exceptionHandler | Exception | ModelAndView | Use annotation to handle the user defined Exception |

* Inside the **populateDonationList** method you should add the **donationFor** into the List and return the List.
* **CharityService** class should be autowired inside the Controller via annotations.

# 3.0.2 Service

* **CharityService** class should be configured via annotation as Service

|  |  |  |  |
| --- | --- | --- | --- |
| **CharityService** | | | |
| **Attribute Name** | **Attribute Type** | **Access Specifier** | **Constraints** |
| fundList | ArrayList<Sponsor> | private |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Method Name** | **Method Argument name:type** | **Return type** | **Responsibilities** |
| addfundDetails | “sponsor”: Sponsor | boolean | Add the Sponsor object into the fund list |
| viewFundList |  | ArrayList<Sponsor> | If the fundList is empty, raise the user **NoFundsRaisedException**. |

# 3.0.3Model

|  |  |  |
| --- | --- | --- |
| **Class(Model)** | **Property** | **Methods** |
| Sponsor | sponsorName:String  phoneNumber:String  donationFor:String  date : String  amount: float | //getters and setters are provided |

* Include the needed annotations inside the Sponsor class for validations already mentioned. The error messages should be retrieved from the properties file

# 3.0.4 Internationalization

* Override the methods in InternationalizationConfig class provided as part of the code skeleton to support Internationalization.
* **Note**: For the LocaleChangeInterceptor, to internationalize the page based on the new Locale, use the parameter "language".
* **Hint**: Request from the UI (showPage.jsp) for all the three hyperlinks have language parameter added to a request. Do not modify/change the hyper link /the values already provided as part of the code skeleton in the showPage.jsp

# 3.1 Request Mapping Tasks

|  |  |
| --- | --- |
| **/index** | Redirect the user to **index.jsp** |
| **/showPage** | Redirect the user to **showPage.jsp** |
| **/addFund** | Redirect the user to showPage.jsp. Should invoke **addFundDetails** of the Controller after submitting all the valid Fund details from **showPage.jsp**, which in turn should invoke the **addFundDetails** of the CharityService and if there are validation errors it should be redirected to **showPage**.**jsp** page |
| **/viewList** | .Should invoke **viewFundList** of the Controller which in turn should invoke the **viewFundList** of the CharityService. If there is an exception then it should redirect to **exceptionPage.jsp** |

# 4.0 Overall Design Constraints

1. Implement the code using the best design standards.
2. Use Internationalization for all the labels and messages in the Add Fund Details page.
3. showPage should support Internationalization for three languages English, German and French. The error messages and the labels should be displayed from the properties file depending on the locale. **The properties file is already provided as part of the code skeleton.**
4. Spring Validator should be used for all the Validations. Use only annotation for performing the validations.
5. CharityTrustApplication which is the main class, is already provided as part of the code skeleton. Include only the required annotations to auto scan the Controller, Service classes.
6. **Do not change the property name given in the application.properties files, you can change the value and you can include additional property if needed.**
7. **In the pom.xml you are provided with all the dependencies needed for developing the application.**
8. You will not be evaluated based on the UI design (layout, color, formatting, etc.). You are free to have a basic UI with all the required UI components (input fields, buttons, labels, etc.). Expected components with the id alone should be designed as per the requirement.
9. Adhere to the design specifications mentioned in the case study.
10. Do not change or delete the class/method names or return types which are provided to you as a part of the base code skeleton.
11. Also do not change the name or id of the HTML component which are provided to you as a part of the base code skeleton.
12. **Please make sure that your code does not have any compilation errors while submitting your case study solution.**