SECTION V: BACKEND [ADMINISTRATION] SOFTWARE DESIGN DOCUMENTATION

Manage Countries

This module allows performing the following operations:

❑ Adding new countries

❑ Editing existing countries

❑ Viewing existing countries

❑ Deleting existing countries

This module uses the bms.Countries table to perform the above operations.

This module is made up of the following**:**

Type	Name	Description
JSP	manageCountries**.**jsp	The data entry form
Spring		
Controller	CountryController**.**java	The controller class that facilitates data operations
Service Class	CountryService**.**java	The interface for the service layer.
	CountryServiceImpl**.**java	The implementation class for the service layer.
DAO Class	CountryDAO**.**java	The interface for the DAO layer.
	CountryDAOImpl**.**java	The implementation class that performs the actual data operations.
DispatcherServlet	bookshopdispatcher-servlet**.**xml	Central dispatcher for HTTP request handlers/controllers.
Hibernate		
Mapping	hibernate.cfg.xml	The mapping file that holds Model class mapping
Model Class	Country.java	The model class

JSP [manageCountries.jsp]

Diagram 31.1: Manage Countries [Data entry form]

This is a JSP which holds a data entry form, as shown in diagram 31.1. This form appears when the user clicks **Countries**.

This page is not part of the book preview.

This page is not part of the book preview.

This page is not part of the book preview.

This page is not part of the book preview.

For example: The following <form:errors> displays errors [if there are any] for the field country:

```
<form:errors path="country" cssClass="error" />
```

<form:errors>'s **cssClass** attribute refers to a class that is declared in CSS to display error messages in red so that it catches the user's attention.

Controller [CountryController.java]

This is a controller class with the following specifications.

Controller Specifications

Class Name	Package	
CountryController	com.sharanamvaishali.controllers	
Objects		
Object Name	Class Name	Object Type
countryService	CountryService	Service Layer
Methods		
Method Name	Request Mapping	
showManageCountries()	/admin/manageCountries	
saveCountry()	/admin/saveCountry	
deleteCountry()	/admin/deleteCountry/${countryNo}	
editCountry()	/admin/editCountry/${countryNo}	

Code Spec

```
1  package com.sharanamvaishali.controllers;
2
3  import com.sharanamvaishali.model.Country;
4  import com.sharanamvaishali.service.CountryService;
5  import java.util.Map;
6  import javax.validation.Valid;
7  import org.hibernate.exception.ConstraintViolationException;
8  import org.springframework.beans.factory.annotation.Autowired;
9  import org.springframework.dao.DataIntegrityViolationException;
10 import org.springframework.stereotype.Controller;
11 import org.springframework.validation.BindingResult;
12 import org.springframework.web.bind.annotation.ModelAttribute;
13 import org.springframework.web.bind.annotation.PathVariable;
14 import org.springframework.web.bind.annotation.RequestMapping;
15 import org.springframework.web.bind.annotation.RequestMethod;
16
17 @Controller
18 @RequestMapping("/admin")
```

This page is not part of the book preview.

This page is not part of the book preview.

deleteCountry()

In case, the user chooses to delete a country and clicks [image: delete button], **CountryController** holds a mapping to help delete a country record from Manage Countries.

deleteCountry():

❑ Passes **CountryNo** of the country to be deleted [available as a @PathVariable] to CountryService's **removeCountry()**

❑ Checks if the record being deleted is referenced by a record in another database table and accordingly displays the error message indicating that this record cannot be deleted. This is done by catching the DataIntegrityViolation exception

❑ Returns the control to the View which renders Manage Countries with the updated data

editCountry()

CountryController holds a mapping to help the editing of existing countries. In case, the user

chooses to edit a country and clicks [image: edit icon], **CountryController** holds a mapping to help fill up the Manage Countries form with country details.

editCountry():

❑ Retrieves the appropriate Country's details using **CountryNo** as the reference. It calls CountryService's **getCountryById()** which returns the details of the country to be edited as a Country object. **CountryNo** is made available through **@PathVariable** [in the list of

Country rendered by the View] when the user clicks [image: edit icon] to edit a particular country

❑ Returns the control to the View along with the data to be populated in the form. The data to be populated is made available using Map's **put()**, which is used to populate the countries object with the data that **getCountryById()** of CountryService returns

Service Class

Interface [CountryService.java]

This is a service interface with the following specifications.

Service Interface Specifications

Class Name	Package
CountryService	com.sharanamvaishali.service

Code Spec

```
1  package com.sharanamvaishali.service;
2
3  import com.sharanamvaishali.model.Country;
4  import java.util.List;
5
6  public interface CountryService {
7      public void saveCountry(Country country);
8      public List<Country> listCountry();
9      public void removeCountry(Integer countryNo);
10     public Country getCountryById(Integer countryNo);
11 }
```

Implementation Class [CountryService.java]

This is a service class with the following specifications.

Service Implementation Specifications

Class Name	Package	Implements
CountryServiceImpl	com.sharanamvaishali.service	CountryService

Objects		
Object Name	Class Name	Object Type
countryDAO	CountryDAO	Data Access Object Layer

Methods	
Method Name	Arguments
saveCountry()	Country country
listCountry()	- -
removeCountry()	Integer countryNo
getCountryById()	Integer countryNo

Code Spec

```
1  package com.sharanamvaishali.service;
2
3  import com.sharanamvaishali.dao.CountryDAO;
4  import com.sharanamvaishali.model.Country;
5  import java.util.List;
6  import org.springframework.beans.factory.annotation.Autowired;
```

This page is not part of the book preview.

removeCountry()

removeCountry() holds the business logic to delete country details based on CountryNo.

This method in turn invokes CountryDAO's **removeCountry()**.

getCountryById()

getCountryById() holds the business logic to retrieve country details based on CountryNo.

This method in turn invokes CountryDAO's **getCountryById()**.

DAO Class

Interface [CountryDAO.java]

This is a Data Access Object interface with the following specifications.

DAO Interface Specifications

Class Name	Package
CountryDAO	com.sharanamvaishali.dao

Code Spec

```
1  package com.sharanamvaishali.dao;
2
3  import com.sharanamvaishali.model.Country;
4  import java.util.List;
5
6  public interface CountryDAO {
7      public void saveCountry(Country country);
8      public List<Country> listCountry();
9      public void removeCountry(Integer countryNo);
10     public Country getCountryById(Integer countryNo);
11 }
```

Implementation Class [CountryDAOImpl.java]

This is a **D**ata **A**ccess **O**bject class with the following specifications.

This page is not part of the book preview.

This page is not part of the book preview.

This page is not part of the book preview.

Domain Specifications

Class Name	Package	Implements
Country	com.sharanamvaishali.model	java.io.Serializable
Properties		

Property Name	Property Type	Methods	
countryNo	Integer	getCountryNo()	setCountryNo()
country	String	getCountry()	setCountry()

Code Spec

```java
1  package com.sharanamvaishali.model;
2
3  import javax.persistence.*;
4  import org.hibernate.validator.constraints.NotEmpty;
5
6  @Entity
7  @Table(name="COUNTRIES")
8  public class Country implements java.io.Serializable {
9      @Id
10     @GeneratedValue
11     @Column(name="COUNTRYNO")
12     private Integer countryNo;
13     @Column(name="COUNTRY")
14     @NotEmpty(message="Country name is mandatory")
15     private String country;
16
17     public String getCountry() {
18         return country;
19     }
20     public void setCountry(String country) {
21         this.country = country;
22     }
23
24     public Integer getCountryNo() {
25         return countryNo;
26     }
27     public void setCountryNo(Integer countryNo) {
28         this.countryNo = countryNo;
29     }
30 }
```