MySqlConn

File Structure

```
+ MySqlConn\
    + emo
        |---  pom.xml
    + emo\src\main\resources
        |---  application.properties
    + emo\src\main\java\com\suji\crudrepo
        |---  CrudRepoDemoApplication.java
        |---  MyRunner.java
    + emo\src\main\java\com\suji\crudrepo\model
        |---  PersonPojo.java
        |---  StudentPojo.java
    + emo\src\main\java\com\suji\crudrepo\configuration
        |---  DateTimeConfig.java
        |---  WebApplicationConfig.java
    + emo\src\main\java\com\suji\crudrepo\control
        |---  GlobalDefaultExceptionHandler.java
        |---  PersonController.java
        |---  StudentController.java
    + emo\src\main\java\com\suji\crudrepo\repository
        |---  PersonRepository.java
        |---  StudentRepository.java
    + emo\src\main\java\com\suji\crudrepo\service
        |---  EmailService.java
        |---  PersonService.java
        |---  StudentService.java
    + emo\src\main\java\com\suji\crudrepo\util
        |---  PrintUtil.java
    + emo\src\main\java\com\suji\crudrepo\configuration\validation
        |---  ContactNumberConstraint.java
        |---  ContactNumberValidator.java
        |---  FieldsValueMatch.java
        |---  FieldsValueMatchValidator.java
        |---  PersonConstraint.java
        |---  PersonValidator.java
        |---  UniquePersonUsernameContraint.java
        |---  UniquePersonUsernameValidator.java
```

Index

```
1. resources\application.properties
2. C:\Users\sujit\Documents\workspace-spring-tool-suite-4-
4.13.1.RELEASE\CrudRepoDemo\pom.xml
3. \model\PersonPojo.java
4. \model\StudentPojo.java
5. \src\main\java\com\suji\crudrepo\configuration\DateTimeConfig.java
6. \src\main\java\com\suji\crudrepo\configuration\validation\ContactNumberConstraint.java
7. \src\main\java\com\suji\crudrepo\configuration\validation\ContactNumberValidator.java
8. \src\main\java\com\suji\crudrepo\configuration\validation\FieldsValueMatch.java
9. \src\main\java\com\suji\crudrepo\configuration\validation\FieldsValueMatchValidator.java
10. \src\main\java\com\suji\crudrepo\configuration\validation\PersonConstraint.java
```

```
11. \src\main\java\com\suji\crudrepo\configuration\validation\PersonValidator.java
12.
\src\main\java\com\suji\crudrepo\configuration\validation\UniquePersonUsernameContraint.java
13.
\src\main\java\com\suji\crudrepo\configuration\validation\UniquePersonUsernameValidator.java
14. \src\main\java\com\suji\crudrepo\configuration\WebApplicationConfig.java
15. \src\main\java\com\suji\crudrepo\control\GlobalDefaultExceptionHandler.java
16. \src\main\java\com\suji\crudrepo\control\PersonController.java
17. \src\main\java\com\suji\crudrepo\control\StudentController.java
18. \src\main\java\com\suji\crudrepo\CrudRepoDemoApplication.java
19. \src\main\java\com\suji\crudrepo\MyRunner.java
20. \src\main\java\com\suji\crudrepo\repository\PersonRepository.java
21. \src\main\java\com\suji\crudrepo\repository\StudentRepository.java
22. \src\main\java\com\suji\crudrepo\service\EmailService.java
23. \src\main\java\com\suji\crudrepo\service\PersonService.java
24. \src\main\java\com\suji\crudrepo\service\StudentService.java
25. \src\main\java\com\suji\crudrepo\util\PrintUtil.java
```

---

1. application.properties

resources\application.properties

```
# Datasource

spring.datasource.url=jdbc:mysql://localhost:3306/nitro
spring.datasource.username=root
spring.datasource.password=apple
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

# Hibernate Settings

spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true


# Logging Settings
logging.level.root=INFO
logging.level.springframework.context=debug
logging.level.com.suji.crudrepo=DEBUG


# DateTimeFormatters

spring.mvc.format.date=yyyy-MM-dd
spring.mvc.format.date-time=yyyy-MM-dd HH:mm:ss
spring.mvc.format.time=HH:mm:ss


# Email

spring.main.banner-mode=off

spring.mail.protocol=smtp
spring.mail.host=smtp.gmail.com
```

```
spring.mail.port=587
spring.mail.username=sujithmsjs@gmail.com
spring.mail.password=jSyr3nf2my#Lf7-6
spring.mail.properties.mail.smtp.auth=true
spring.mail.properties.mail.smtp.starttls.enable=true
```

---

2. pom.xml

C:\Users\sujit\Documents\workspace-spring-tool-suite-4-4.13.1.RELEASE\CrudRepoDemo\pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.5.9</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.suji</groupId>
    <artifactId>CrudRepoDemo</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>CrudRepoDemo</name>
    <description>Demo project for Spring Boot</description>
    <properties>
        <java.version>1.8</java.version>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-data-jpa</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-thymeleaf</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-validation</artifactId>
        </dependency>
        <dependency>
```

```
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>

        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <scope>runtime</scope>
        </dependency>
        <dependency>
            <groupId>org.projectlombok</groupId>
            <artifactId>lombok</artifactId>
            <optional>true</optional>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>

         <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-mail</artifactId>
        </dependency>

    </dependencies>

    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
                <configuration>
                    <excludes>
                        <exclude>
                            <groupId>org.projectlombok</groupId>
                            <artifactId>lombok</artifactId>
                        </exclude>
                    </excludes>
                </configuration>
            </plugin>
        </plugins>
    </build>

</project>
```

---

3. PersonPojo.java

\model\PersonPojo.java

```
package com.suji.crudrepo.model;

import java.time.LocalDate;
import java.time.LocalDateTime;
```

```java
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import javax.persistence.Transient;
import javax.validation.constraints.Past;
import javax.validation.constraints.Size;

import org.hibernate.annotations.CreationTimestamp;
import org.springframework.data.annotation.LastModifiedDate;

import com.suji.crudrepo.configuration.validation.FieldsValueMatch;
import com.suji.crudrepo.configuration.validation.PersonConstraint;
import com.suji.crudrepo.configuration.validation.UniquePersonUsernameContraint;

import lombok.Data;

@Data
@Entity
@Table(name = "persons")
@PersonConstraint
public class PersonPojo {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;
    @Size(min = 4, max=16)
    @Column(name = "name", nullable = false)
    private String name;


    @Column(name = "uname", nullable = false)
    @Size(min = 4, max=10)
    private String username;

    @Size(min = 4, max=16)
    @Column(name = "pw", nullable = false)
    private String password;

    @Transient
    private String verifyPassword;

    @Column(name = "sex")
    private char gender;

    @Column(name = "height")
    private double height;
    @Column(name = "ncc")
    private boolean hasNCC;

    @Past
    @Column(name = "date_of_birth")
    private LocalDate dob;
    @Column(name = "registered_on")
```

```
    @CreationTimestamp
    private LocalDateTime registered;
    @Column(name = "last_update")
    @LastModifiedDate
    private LocalDateTime lastUpdate;
}


//name,username,password,gender,dob,height,hasNCC,id,registered, lastUpdate
```

---

4. StudentPojo.java

\model\StudentPojo.java

```
package com.suji.crudrepo.model;

import java.time.LocalDate;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import javax.validation.constraints.Max;
import javax.validation.constraints.Min;
import javax.validation.constraints.NotEmpty;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.Past;
import javax.validation.constraints.Pattern;
import javax.validation.constraints.Size;

import org.springframework.format.annotation.DateTimeFormat;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

@Data
@NoArgsConstructor
@AllArgsConstructor
@Entity
@Table(name = "students")
public class StudentPojo {

    @Id
    @GeneratedValue(strategy = GenerationType.SEQUENCE)
    private int id;

    @NotNull(message = "LastName can not be null!!")
    @NotEmpty(message = "LastName can not be empty!!")
    private String name;

    @NotNull(message = "Choose the subject count you are going to study!")
    @Min(value = 4, message = "Student should enroll to minimum 4 subjects!!")
    @Max(value = 8, message = "Student can enroll to maximum 8 subjects!!")
```

```
    private int subjectCount;

    @NotNull
    @Min(1)
    @Max(12)
    private int grade;

    @NotNull
    @Size(max = 10, min = 10, message = "Mobile number should be of 10 digits")
    @Pattern(regexp = "[7-9][0-9]{9}", message = "Mobile number is invalid!!")
    private String mobileNo;

    @NotNull(message = "Please enter birth date")
    @Past(message = "Birth date should be less than current date!!")
    @DateTimeFormat(iso = DateTimeFormat.ISO.DATE)
    private LocalDate birthDate;
}
```

---

5. DateTimeConfig.java

\src\main\java\com\suji\crudrepo\configuration\DateTimeConfig.java

```
package com.suji.crudrepo.configuration;

import org.springframework.format.support.FormattingConversionService;
import org.springframework.web.servlet.config.annotation.WebMvcConfigurationSupport;

public class DateTimeConfig extends WebMvcConfigurationSupport {

    @Override
    public FormattingConversionService mvcConversionService() {

        return super.mvcConversionService();
    }
}
```

---

6. ContactNumberConstraint.java

\src\main\java\com\suji\crudrepo\configuration\validation\ContactNumberConstraint.java

```
package com.suji.crudrepo.configuration.validation;

import java.lang.annotation.Documented;
import java.lang.annotation.ElementType;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;

import javax.validation.Constraint;
```

```
import javax.validation.Payload;

@Documented
@Constraint(validatedBy = ContactNumberValidator.class)
@Target({ ElementType.METHOD, ElementType.FIELD })
@Retention(RetentionPolicy.RUNTIME)
public @interface ContactNumberConstraint {

    String message() default "Invalid phone number";

    Class<?>[] groups() default {};

    Class<? extends Payload>[] payload() default {};
}
```

---

7. ContactNumberValidator.java

\src\main\java\com\suji\crudrepo\configuration\validation\ContactNumberValidator.java

```
package com.suji.crudrepo.configuration.validation;

import javax.validation.ConstraintValidator;
import javax.validation.ConstraintValidatorContext;

public class ContactNumberValidator implements ConstraintValidator<ContactNumberConstraint,
String> {

    @Override
    public void initialize(ContactNumberConstraint contactNumber) {
    }

    @Override
    public boolean isValid(String contactField, ConstraintValidatorContext cxt) {

        System.out.println("Out for validation: " + contactField);
        return (contactField != null) && contactField.matches("[0-9]+") &&
(contactField.length() == 10);
    }

}
```

---

8. FieldsValueMatch.java

\src\main\java\com\suji\crudrepo\configuration\validation\FieldsValueMatch.java

```
package com.suji.crudrepo.configuration.validation;

import java.lang.annotation.Documented;
import java.lang.annotation.ElementType;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;
```

```
import javax.validation.Constraint;
import javax.validation.Payload;

@Documented
@Constraint(validatedBy = FieldsValueMatchValidator.class)
@Target({ ElementType.TYPE })
@Retention(RetentionPolicy.RUNTIME)
public @interface FieldsValueMatch {

    String message() default "Fields values don't match!";

    String field();

    String fieldMatch();

    Class<?>[] groups() default {};
    Class<? extends Payload>[] payload() default {};

}
```

---

9. FieldsValueMatchValidator.java

\src\main\java\com\suji\crudrepo\configuration\validation\FieldsValueMatchValidator.java

```
package com.suji.crudrepo.configuration.validation;

import javax.validation.ConstraintValidator;
import javax.validation.ConstraintValidatorContext;

import org.springframework.beans.BeanWrapperImpl;

public class FieldsValueMatchValidator implements ConstraintValidator<FieldsValueMatch,
Object> {

    private String field;
    private String fieldMatch;

    public void initialize(FieldsValueMatch constraintAnnotation) {
        this.field = constraintAnnotation.field();
        this.fieldMatch = constraintAnnotation.fieldMatch();
    }

    public boolean isValid(Object value, ConstraintValidatorContext context) {

        Object fieldValue = new BeanWrapperImpl(value).getPropertyValue(field);
        Object fieldMatchValue = new BeanWrapperImpl(value).getPropertyValue(fieldMatch);

        System.out.println(field+":"+fieldValue+"; "+fieldMatch+":"+fieldMatchValue);

        if (fieldValue != null) {
            System.out.println("Password doesn't match.");
            return fieldValue.equals(fieldMatchValue);
        } else {
            System.out.println("Password matched.");
```

```
            return fieldMatchValue == null;
        }
    }
}
```

---

10. PersonConstraint.java

\src\main\java\com\suji\crudrepo\configuration\validation\PersonConstraint.java

```
package com.suji.crudrepo.configuration.validation;

import java.lang.annotation.ElementType;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;

import javax.validation.Constraint;
import javax.validation.Payload;

@Target(ElementType.TYPE)
@Retention(RetentionPolicy.RUNTIME)
@Constraint(validatedBy = PersonValidator.class)
public @interface PersonConstraint {

    String message() default "Username already Exists";

    Class<?>[] groups() default {};

    Class<? extends Payload>[] payload() default {};
}
```

---

11. PersonValidator.java

\src\main\java\com\suji\crudrepo\configuration\validation\PersonValidator.java

```
package com.suji.crudrepo.configuration.validation;

import javax.validation.ConstraintValidator;
import javax.validation.ConstraintValidatorContext;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;

import com.suji.crudrepo.MyRunner;
import com.suji.crudrepo.model.PersonPojo;
import com.suji.crudrepo.repository.PersonRepository;

public class PersonValidator implements ConstraintValidator<PersonConstraint, PersonPojo> {

    private static final Logger LOG = LoggerFactory.getLogger(MyRunner.class);

    @Autowired
```

```
    private PersonRepository service;

    @Override
    public boolean isValid(PersonPojo person, ConstraintValidatorContext context) {

        String username = person.getUsername();

        boolean isExists = service.existsByUsername(username);
        LOG.info("Username: "+username+"; isExists? "+isExists);

        boolean isPasswordMatched = person.getPassword().equals(person.getVerifyPassword());
        LOG.info("Is Password Matched? "+isPasswordMatched);

        boolean isValid = (!isExists) && isPasswordMatched;
        if(isValid)
            LOG.debug("Username and passwords are valid.");
        else
            LOG.error("Username and passwords aren't valid.");
        return (!isExists) && isPasswordMatched;
    }

}
```

---

12. UniquePersonUsernameContraint.java

\src\main\java\com\suji\crudrepo\configuration\validation\UniquePersonUsernameContraint.java

```
package com.suji.crudrepo.configuration.validation;

import java.lang.annotation.Documented;
import java.lang.annotation.ElementType;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;

import javax.validation.Constraint;
import javax.validation.Payload;

@Documented
@Constraint(validatedBy = UniquePersonUsernameValidator.class)
@Target({ ElementType.METHOD, ElementType.FIELD })
@Retention(RetentionPolicy.RUNTIME)
public @interface UniquePersonUsernameContraint {

    String message() default "Username already Exists";

    Class<?>[] groups() default {};

    Class<? extends Payload>[] payload() default {};
}
```

---

13. UniquePersonUsernameValidator.java

\src\main\java\com\suji\crudrepo\configuration\validation\UniquePersonUsernameValidator.java

```
package com.suji.crudrepo.configuration.validation;

import javax.validation.ConstraintValidator;
import javax.validation.ConstraintValidatorContext;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;

import com.suji.crudrepo.MyRunner;
import com.suji.crudrepo.repository.PersonRepository;
import com.suji.crudrepo.service.PersonService;

public class UniquePersonUsernameValidator implements
ConstraintValidator<UniquePersonUsernameContraint, String> {

    private static final Logger logger = LoggerFactory.getLogger(MyRunner.class);

    @Autowired
    private PersonRepository service;

    @Override
    public boolean isValid(String contactField, ConstraintValidatorContext cxt) {

        logger.info(contactField);
        boolean isExists = service.existsByUsername(contactField);

        System.out.println("Username at validation: "+contactField+"; isExists: "+isExists);
         return isExists;
    }

}
```

---

14. WebApplicationConfig.java

\src\main\java\com\suji\crudrepo\configuration\WebApplicationConfig.java

```
package com.suji.crudrepo.configuration;
import org.springframework.boot.web.server.ErrorPage;
import org.springframework.boot.web.server.WebServerFactoryCustomizer;
import org.springframework.boot.web.servlet.server.ConfigurableServletWebServerFactory;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.http.HttpStatus;
import org.springframework.web.servlet.config.annotation.ViewControllerRegistry;
import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;

@Configuration
public class WebApplicationConfig implements WebMvcConfigurer {

    @Override
```

```
    public void addViewControllers(ViewControllerRegistry registry) {
        registry.addViewController("/notFound").setViewName("forward:/error-page.html");
    }

    @Bean
    public WebServerFactoryCustomizer<ConfigurableServletWebServerFactory>
containerCustomizer() {

        return container -> {
            container.addErrorPages(new ErrorPage(HttpStatus.NOT_FOUND, "/notFound"));
        };
    }

}
```

---

15. GlobalDefaultExceptionHandler.java

\src\main\java\com\suji\crudrepo\control\GlobalDefaultExceptionHandler.java

```
package com.suji.crudrepo.control;

import javax.servlet.http.HttpServletRequest;

import org.springframework.core.annotation.AnnotationUtils;
import org.springframework.http.HttpStatus;
import org.springframework.web.bind.annotation.ControllerAdvice;
import org.springframework.web.bind.annotation.ExceptionHandler;
import org.springframework.web.bind.annotation.ResponseStatus;
import org.springframework.web.servlet.ModelAndView;

import com.suji.crudrepo.model.PersonPojo;

@ControllerAdvice
public class GlobalDefaultExceptionHandler {

    @ExceptionHandler(value = Exception.class)
    public ModelAndView defaultErrorHandler(HttpServletRequest req, Exception e) throws
Exception {
        if (AnnotationUtils.findAnnotation(e.getClass(), ResponseStatus.class) != null) {
            throw e;
        }

        ModelAndView mav = new ModelAndView();
        mav.addObject("personPojo", new PersonPojo());
        mav.setViewName("/person/signup");
        return mav;
    }

    @ResponseStatus(HttpStatus.NOT_FOUND) // 404
    public ModelAndView handleConflict() {
        System.out.println("404: Error!");
        ModelAndView mav = new ModelAndView();
        mav.addObject("personPojo", new PersonPojo());
        mav.setViewName("");
        return mav;
```

```
    }

}
```

---

16. PersonController.java

\src\main\java\com\suji\crudrepo\control\PersonController.java

```
package com.suji.crudrepo.control;

import java.util.HashMap;
import java.util.Iterator;
import java.util.List;
import java.util.Map;

import javax.naming.Binding;
import javax.validation.Valid;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.validation.BindingResult;
import org.springframework.validation.FieldError;
import org.springframework.validation.ObjectError;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;

import com.suji.crudrepo.MyRunner;
import com.suji.crudrepo.model.PersonPojo;
import com.suji.crudrepo.service.PersonService;

@Controller
@RequestMapping("/person")
public class PersonController {

    private static final Logger LOG = LoggerFactory.getLogger(MyRunner.class);

    @Autowired
    private PersonService personService;

    @RequestMapping("/signup")
    public String getSignupForm(PersonPojo person) {
        System.out.println("Loading Form: " + person);
        return "person/signup";
    }

    @PostMapping("/save-person")
    public String saveFormData(@Valid PersonPojo person, BindingResult results) {
        LOG.info("Inside /save-person, errors has been thrown.");

        if (results.hasErrors()) {
            generateFieldErrors(results);
            LOG.error("Form validation failed...");
            return "person/signup";
```

```
        }

        PersonPojo savedPerson = personService.savePerson(person);
        LOG.debug("Person data saved: " + savedPerson);
        return "/person/welcome";
    }

    private void generateFieldErrors(BindingResult result) {

        List<ObjectError> globalError = result.getGlobalErrors();


        for (ObjectError objectError : globalError) {

            LOG.error("Global Errors: " + objectError.getCode());


            if (objectError.getCode().equalsIgnoreCase("PersonConstraint")) {
                FieldError fc = new FieldError(objectError.getObjectName(),
"verifyPassword", objectError.getDefaultMessage());
                FieldError fc2 = new FieldError(objectError.getObjectName(), "password",
                        objectError.getDefaultMessage());

                System.out.println(fc);
                result.addError(fc);
                result.addError(fc2);
                Object[] arguments = objectError.getArguments();
                for (Object obj : arguments) {
                    System.out.println("Arg: " + obj);
                }
            }
        }
    }

    private Map<String, String> generateObjectErrors(BindingResult result) {
        Map<String, String> fieldErrorMap = new HashMap<>();
        result.getGlobalErrors().forEach(objectError -> {
            if (objectError.getCode().contains("ConfirmPassword")) {
                fieldErrorMap.put("confirmPassword", objectError.getDefaultMessage());
            }
        });
        return fieldErrorMap;
    }

}
```

---

17. StudentController.java

\src\main\java\com\suji\crudrepo\control\StudentController.java

```
package com.suji.crudrepo.control;

import java.util.List;

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.validation.FieldError;
import org.springframework.web.bind.annotation.RequestMapping;

import com.suji.crudrepo.model.StudentPojo;
import com.suji.crudrepo.service.StudentService;
import com.suji.crudrepo.util.PrintUtil;

@Controller
@RequestMapping("student")
public class StudentController {

    @Autowired
    private StudentService service;

    @RequestMapping("/signup")
    public String getSignupForm(StudentPojo student, Model model) {
        return "student/signup";
    }

    @RequestMapping("/save-student")
    public String saveStudentForm(@Valid StudentPojo student, BindingResult results, Model
model) {

        if(results.hasErrors()) {
            PrintUtil.showBindingResult(results);
            return "student/signup";
        }

        StudentPojo savedStudent = service.saveStudent(student);
        model.addAttribute("message", savedStudent.getName()+" You're id id:
"+savedStudent.getId());
        return "student/welcome";
    }
}
```

---

18. CrudRepoDemoApplication.java

\src\main\java\com\suji\crudrepo\CrudRepoDemoApplication.java

```
package com.suji.crudrepo;

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

@SpringBootApplication
public class CrudRepoDemoApplication {

    public static void main(String[] args) {
        SpringApplication.run(CrudRepoDemoApplication.class, args);
    }
```

```
}
```

---

19. MyRunner.java

\src\main\java\com\suji\crudrepo\MyRunner.java

```
package com.suji.crudrepo;


import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.stereotype.Component;

import com.suji.crudrepo.service.EmailService;

@Component
public class MyRunner  implements CommandLineRunner{

    private static final Logger logger = LoggerFactory.getLogger(MyRunner.class);

    @Override
    public void run(String... args) throws Exception {

        logger.debug("Email Sent Successfully.");
        logger.info("This is just some info");
        logger.debug("This is just some info");
    }
}
```

---

20. PersonRepository.java

\src\main\java\com\suji\crudrepo\repository\PersonRepository.java

```
package com.suji.crudrepo.repository;

import org.springframework.data.jpa.repository.JpaRepository;

import com.suji.crudrepo.model.PersonPojo;

public interface PersonRepository extends JpaRepository<PersonPojo, Integer> {

    boolean existsByUsername(String value);

}
```

---

21. StudentRepository.java

\src\main\java\com\suji\crudrepo\repository\StudentRepository.java

```
package com.suji.crudrepo.repository;

import org.springframework.data.jpa.repository.JpaRepository;

import com.suji.crudrepo.model.StudentPojo;

public interface StudentRepository extends JpaRepository<StudentPojo, Integer> {

}
```

---

22. EmailService.java

\src\main\java\com\suji\crudrepo\service\EmailService.java

```
package com.suji.crudrepo.service;

import org.springframework.mail.SimpleMailMessage;
import org.springframework.mail.javamail.JavaMailSender;
import org.springframework.stereotype.Service;

@Service
public class EmailService {

    private JavaMailSender javaMailSender;

    public EmailService(JavaMailSender javaMailSender) {
        this.javaMailSender = javaMailSender;
    }

    public void sendMail(String toEmail, String subject, String message) {

        SimpleMailMessage mailMessage = new SimpleMailMessage();

        //Caused by: org.springframework.mail.MailAuthenticationException: Authentication
failed;
        mailMessage.setTo(toEmail);
        mailMessage.setSubject(subject);
        mailMessage.setText(message);

        mailMessage.setFrom("sujthmsjs@gmail.com");

        javaMailSender.send(mailMessage);
    }
}
```

---

23. PersonService.java

\src\main\java\com\suji\crudrepo\service\PersonService.java

```
package com.suji.crudrepo.service;

import org.slf4j.Logger;
```

```
import org.slf4j.LoggerFactory;
import org.springframework.stereotype.Service;

import com.suji.crudrepo.model.PersonPojo;
import com.suji.crudrepo.repository.PersonRepository;

@Service
public class PersonService {

    private static final Logger LOG = LoggerFactory.getLogger(PersonService.class);

    private PersonRepository repository;

    public PersonService(PersonRepository repository) {
        LOG.info("PersonRepository: "+repository);
        this.repository = repository;
    }

    public PersonPojo savePerson(PersonPojo student) {
        PersonPojo save = repository.save(student);
        if(save != null)
            LOG.debug("Persaon Datasaved: "+student);
        else
            LOG.error("Person Data not saved: "+student);

        return save;
    }



}
```

---

24. StudentService.java

\src\main\java\com\suji\crudrepo\service\StudentService.java

```
package com.suji.crudrepo.service;

import org.springframework.stereotype.Service;
import com.suji.crudrepo.model.StudentPojo;
import com.suji.crudrepo.repository.StudentRepository;




@Service
public class StudentService {

    private StudentRepository repository;

    public StudentService(StudentRepository repository) {
        System.out.println("StudentRepository Autowiring...");
        this.repository = repository;
    }
```

```
    public StudentPojo saveStudent(StudentPojo student) {
        return repository.save(student);
    }
}
```

---

25. PrintUtil.java

\src\main\java\com\suji\crudrepo\util\PrintUtil.java

```
package com.suji.crudrepo.util;

import java.util.List;

import org.springframework.validation.BindingResult;
import org.springframework.validation.FieldError;

public class PrintUtil {

    public static void showBindingResult(BindingResult results) {
        List<FieldError> error = results.getFieldErrors();
        for (FieldError fieldError : error) {
            System.out.println(
                    "Field Name: " + fieldError.getField() + "; Error Message : " +
fieldError.getDefaultMessage());
        }
    }

}
```
