1. **Ссылка на гит:**

[AnatolSalau/SpringValidation (github.com)](https://github.com/AnatolSalau/SpringValidation)

1. **Структура приложения**

- Модели и ДТО

@Data  
@NoArgsConstructor  
  
@Entity  
public class Input *{* @Id  
 @GeneratedValue  
 private Long id;  
  
 @Min*(*1*)* @Max*(*10*)* private int numberBetweenOneAndTen;  
  
 @Pattern*(*regexp = "^[0-9]{1,3}\\.[0-9]{1,3}\\.[0-9]{1,3}\\.[0-9]{1,3}$"*)* private String ipAddress;  
*}*

@Data  
  
public class InputDto *{* private Long id;  
 @Min*(*1*)* @Max*(*10*)* private int numberBetweenOneAndTen;  
  
 @Pattern*(*regexp = "^[0-9]{1,3}\\.[0-9]{1,3}\\.[0-9]{1,3}\\.[0-9]{1,3}$"*)* private String ipAddress;  
*}*

- Репозитории

@Repository  
public interface InputRepository extends JpaRepository*<*Input, Long*> {* List*<*Input*>* findAllByIpAddress*(*String ipAdress*)*;  
*}*

* Сервисы

@Service  
@Validated  
public class InputService *{* private final InputRepository repository;  
 private final ModelMapper mapper;  
  
 @Autowired  
 public InputService*(*InputRepository repository, ModelMapper mapper*) {* this.repository = repository;  
 this.mapper = mapper;  
 *}* public InputDto saveInput*(*InputDto inputDto*) {* Input save = repository.save*(*convertDtoToModel*(*inputDto*))*;  
 return convertModelToDto*(*save*)*;  
 *}* @NotNull  
 @Size*(*min = 1*)* public List*<*@NotNull InputDto*>* findAllUser*() {* List*<*Input*>* result = repository.findAll*()*;  
 return result.stream*()* .map*(*input -> mapper.map*(* input, InputDto.class*)  
 )* .toList*()*;  
 *}* @NotNull  
 @Size*(*min = 1*)* public List*<*@NotNull InputDto*>* findAllUsersByIpAddress*(*@NotEmpty String ipAdress*) {* List*<*Input*>* result = repository.findAllByIpAddress*(*ipAdress*)*;  
 return result.stream*()* .map*(*input -> mapper.map*(* input, InputDto.class*)  
 )* .toList*()*;  
 *}* private Input convertDtoToModel*(*InputDto inputDto*) {* Input map = mapper.map*(*inputDto, Input.class*)*;  
 return map;  
 *}* private InputDto convertModelToDto*(*Input input*) {* InputDto map = mapper.map*(*input, InputDto.class*)*;  
 return map;  
 *}  
}*

1. **Интеграционные тесты InputService**

@ExtendWith*(*SpringExtension.class*)*@SpringBootTest  
class InputServiceTest *{* @MockBean  
 InputRepository repository;  
  
 @Autowired  
 private InputService inputService;  
  
 @Test  
 void whenReturnListIsNull\_thenThrowException*() {  
 when(*repository.findAll*())* .thenReturn*(*returnEmptyInputList*())*;  
  
 Assertions.*assertThrows(* ConstraintViolationException.class,inputService::findAllUser  
 *)*;  
 *}* @Test  
 void whenIpAdressIsDone\_thenReturnList*() {* String ipAdress = "128.0.0.1";  
 *when(*repository.findAllByIpAddress*(*ipAdress*))* .thenReturn*(*returnTwoInputs*())*;  
 List*<*InputDto*>* allByIpAddress = inputService.findAllUsersByIpAddress*(*ipAdress*)*;  
 Assertions.*assertEquals(*2,allByIpAddress.size*())*;  
 *}* @Test  
 void whenIpAdressIsNull\_thenThrowException*() {* String ipAdress = null;  
 *when(*repository.findAllByIpAddress*(*ipAdress*))* .thenReturn*(*returnTwoInputs*())*;  
 Assertions.*assertThrows(* ConstraintViolationException.class,*()* -> *{* inputService.findAllUsersByIpAddress*(*ipAdress*)*;  
 *}  
 )*;  
 *}* @Test  
 void whenIpAdressIsEmpty\_thenThrowException*() {* String ipAdress = "";  
 *when(*repository.findAllByIpAddress*(*ipAdress*))* .thenReturn*(*returnTwoInputs*())*;  
 Assertions.*assertThrows(* ConstraintViolationException.class,*()* -> *{* inputService.findAllUsersByIpAddress*(*ipAdress*)*;  
 *}  
 )*;  
 *}* @Test  
 void whenResulListContainNull\_thenThrowException*() {* String ipAdress = "127.0.0.1";  
 *when(*repository.findAllByIpAddress*(*ipAdress*))* .thenReturn*(*returnListInputsWithOneNull*())*;  
 Assertions.*assertThrows(* IllegalArgumentException.class,*()* -> *{* inputService.findAllUsersByIpAddress*(*ipAdress*)*;  
 *}  
 )*;  
 *}* private List*<*Input*>* returnEmptyInputList*() {* List*<*Input*>* result = new ArrayList*<>()*;  
 return result;  
 *}* private List*<*Input*>* returnTwoInputs*() {* Input input1 = new Input*()*;  
 input1.setId*(*1L*)*;  
 input1.setIpAddress*(*"127.0.0.1"*)*;  
 input1.setNumberBetweenOneAndTen*(*1*)*;  
  
 Input input2 = new Input*()*;  
 input2.setId*(*2L*)*;  
 input2.setIpAddress*(*"127.0.0.1"*)*;  
 input2.setNumberBetweenOneAndTen*(*2*)*;  
  
 List*<*Input*>* result = List.*of(*input1, input2*)*;  
 return result;  
 *}* private List*<*Input*>* returnListInputsWithOneNull*() {* Input input1 = new Input*()*;  
 input1.setId*(*1L*)*;  
 input1.setIpAddress*(*"127.0.0.1"*)*;  
 input1.setNumberBetweenOneAndTen*(*1*)*;  
  
 Input input2 = new Input*()*;  
 input2.setId*(*2L*)*;  
 input2.setIpAddress*(*"127.0.0.1"*)*;  
 input2.setNumberBetweenOneAndTen*(*2*)*;  
  
 List*<*Input*>* result = new ArrayList*<>()*;  
 result.add*(*input1*)*;  
 result.add*(*input2*)*;  
 result.add*(*null*)*;  
 return result;  
 *}  
}*

1. **Вывод**

