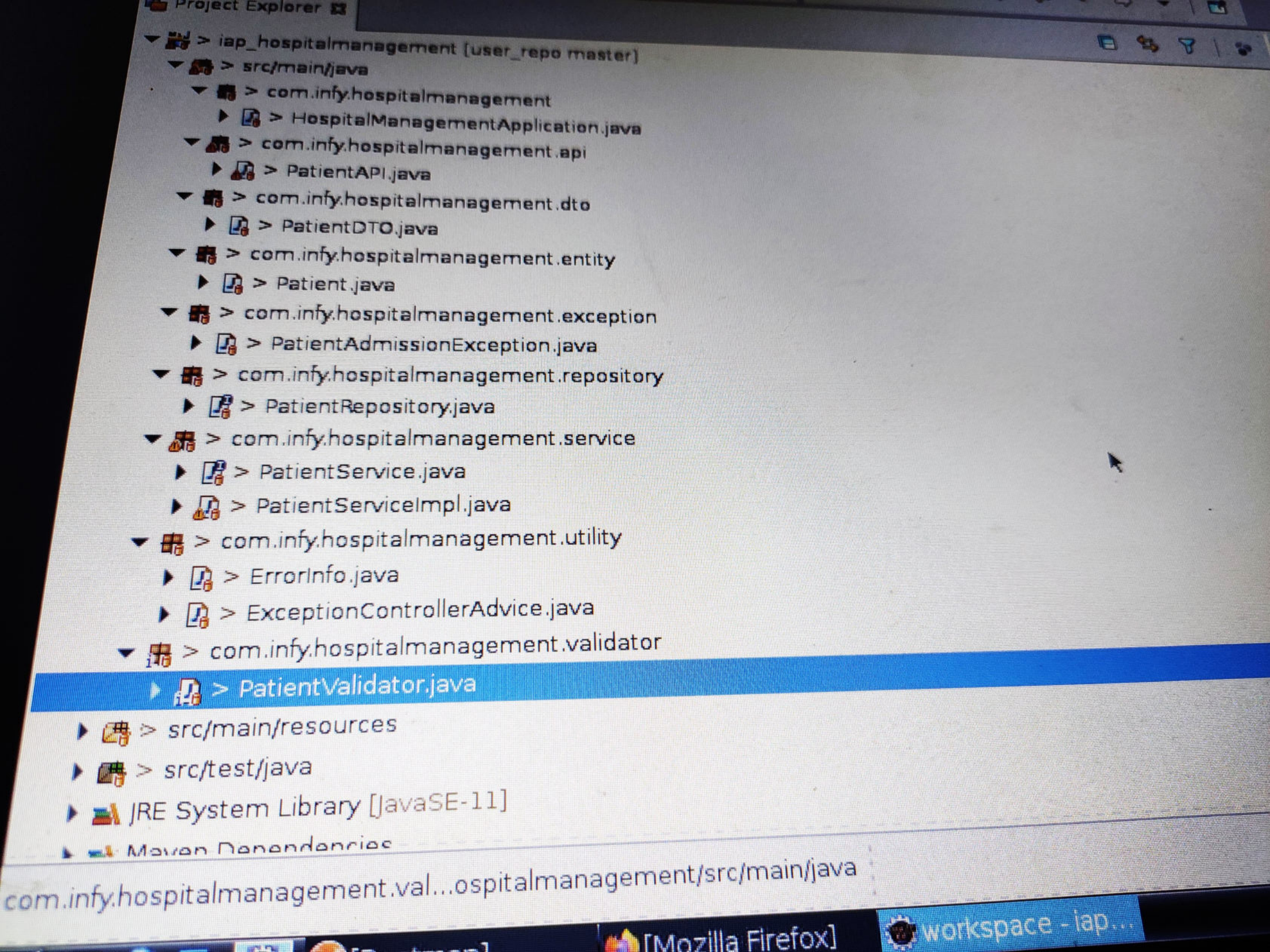
Donate

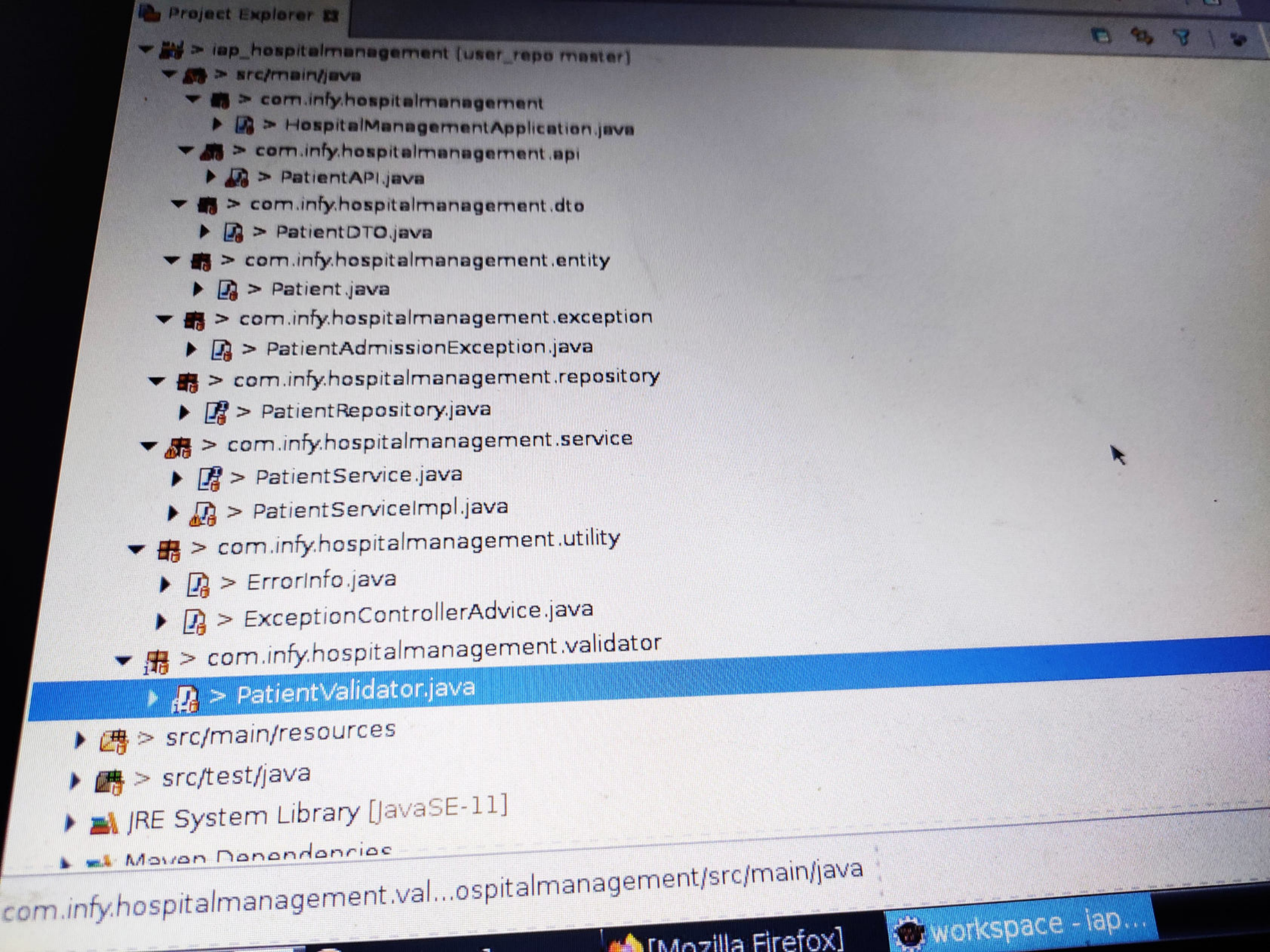
application.properties

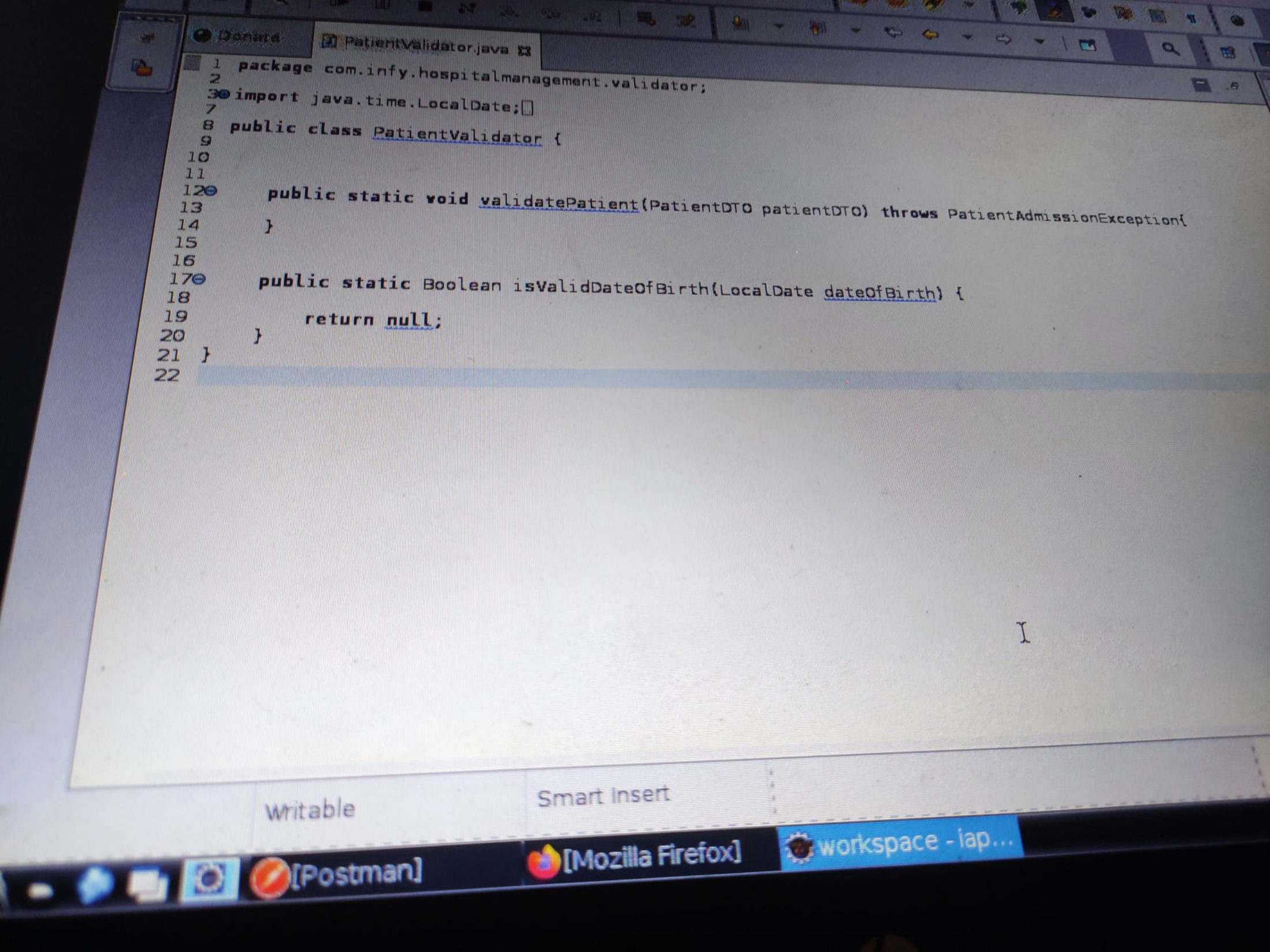
```
1
2 #Database properties
3 spring.datasource.url=jdbc:mysql://localhost:3306/hospital_management_db
4 spring.datasource.username=root
5 spring.datasource.password=root
6 spring.jpa.show-sql=true
7 spring.jpa.properties.hibernate.format_sql=true
8 server.port=8765
9
10
11 #Validator Messages
12 PatientValidator.INVALID_DOB=Date of birth should not be a future date and the age should not be more than 1
13
14 #General Messages
15 General.EXCEPTION_MESSAGE=Request could not be processed due to some issue. Please try again!
16
17 #Service Messages
18 PatientService.PATIENT_UNAVAILABLE=No patient available with given diagnosis
19
```

Writable | Insert

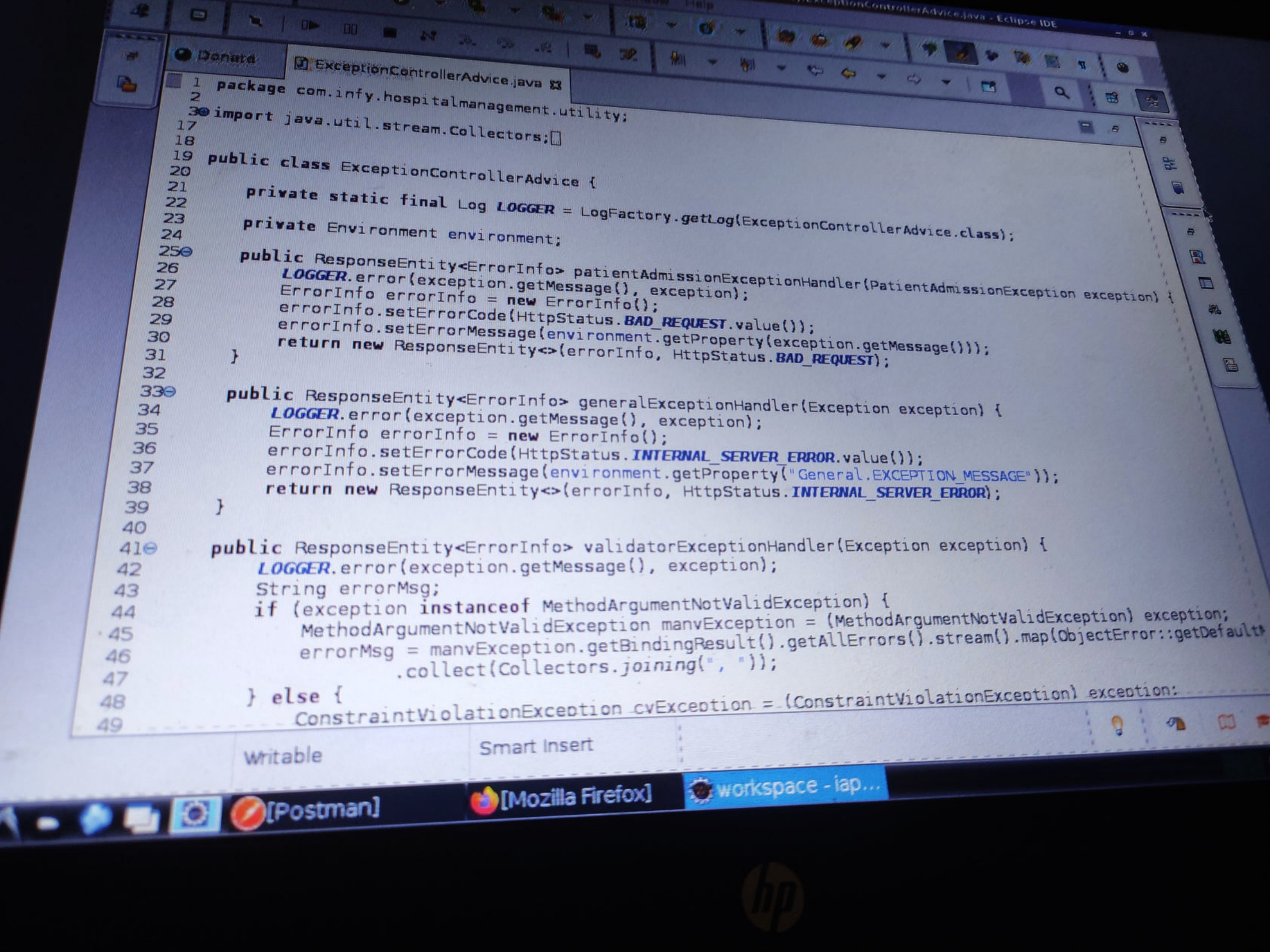
[Postman] [Mozilla Firefox] workspace - iap...


Project Explorer
> iap_hospitalmanagement [user_repo master]
> src/main/java
> com.infy.hospitalmanagement
> HospitalManagementApplication.java
> com.infy.hospitalmanagement.api
> PatientAPI.java
> com.infy.hospitalmanagement.dto
> PatientDTO.java
> com.infy.hospitalmanagement.entity
> Patient.java
> com.infy.hospitalmanagement.exception
> PatientAdmissionException.java
> com.infy.hospitalmanagement.repository
> PatientRepository.java
> com.infy.hospitalmanagement.service
> PatientService.java
> PatientServiceImpl.java
> com.infy.hospitalmanagement.utility
> ErrorInfo.java
> ExceptionControllerAdvice.java
> com.infy.hospitalmanagement.validator
> PatientValidator.java
> src/main/resources
> src/test/java
JRE System Library [JavaSE-11]
Maven Dependencies
com.infy.hospitalmanagement.val...ospitalmanagement/src/main/java
[Mozilla Firefox]
workspace - iap...


Project Explorer
> iap_hospitalmanagement [user_repo master]
> src/main/java
> com.infy.hospitalmanagement
> HospitalManagementApplication.java
> com.infy.hospitalmanagement.api
> PatientAPI.java
> com.infy.hospitalmanagement.dto
> PatientDTO.java
> com.infy.hospitalmanagement.entity
> Patient.java
> com.infy.hospitalmanagement.exception
> PatientAdmissionException.java
> com.infy.hospitalmanagement.repository
> PatientRepository.java
> com.infy.hospitalmanagement.service
> PatientService.java
> PatientServiceImpl.java
> com.infy.hospitalmanagement.utility
> ErrorInfo.java
> ExceptionControllerAdvice.java
> com.infy.hospitalmanagement.validator
> PatientValidator.java
> src/main/resources
> src/test/java
JRE System Library [JavaSE-11]
Maven Dependencies
com.infy.hospitalmanagement.val...ospitalmanagement/src/main/java
[Mozilla Firefox]
workspace - iap...


Donate
PatientValidator.java
1 package com.infy.hospitalmanagement.validator;
2
3 import java.time.LocalDate;
7
8 public class PatientValidator {
9
10
11
12 public static void validatePatient(PatientDTO patientDTO) throws PatientAdmissionException{
13
14 }
15
16
17 public static Boolean isValidDateOfBirth(LocalDate dateOfBirth) {
18
19 return null;
20 }
21 }
22
Writable
Smart Insert
[Postman]
[Mozilla Firefox]
workspace - iap...

ExceptionControllerAdvice.java
public ResponseEntity<ErrorInfo> patientAdmissionExceptionHandler(PatientAdmissionException exception)
LOGGER.error(exception.getMessage(), exception);
ErrorInfo errorInfo = new ErrorInfo();
errorInfo.setErrorCode(HttpStatus.BAD_REQUEST.value());
errorInfo.setErrorMessage(environment.getProperty(exception.getMessage()));
return new ResponseEntity<>(errorInfo, HttpStatus.BAD_REQUEST);
public ResponseEntity<ErrorInfo> generalExceptionHandler(Exception exception) {
LOGGER.error(exception.getMessage(), exception);
ErrorInfo errorInfo = new ErrorInfo();
errorInfo.setErrorCode(HttpStatus.INTERNAL_SERVER_ERROR.value());
errorInfo.setErrorMessage(environment.getProperty("General.EXCEPTION_MESSAGE"));
return new ResponseEntity<>(errorInfo, HttpStatus.INTERNAL_SERVER_ERROR);
public ResponseEntity<ErrorInfo> validatorExceptionHandler(Exception exception) {
LOGGER.error(exception.getMessage(), exception);
String errorMsg;
if (exception instanceof MethodArgumentNotValidException) {
MethodArgumentNotValidException manvException = (MethodArgumentNotValidException) exception;
errorMsg = manvException.getBindingResult().getAllErrors().stream().map(ObjectError::getDefault
.collect(Collectors.joining(", "));
} else {
ConstraintViolationException cvException = (ConstraintViolationException) exception;
errorMsg = cvException.getConstraintViolations().stream().map(ConstraintViolation::getMessage)
.collect(Collectors.joining(", "));
}
ErrorInfo errorInfo = new ErrorInfo();
errorInfo.setErrorCode(HttpStatus.BAD_REQUEST.value());
errorInfo.setErrorMessage(errorMsg);
return new ResponseEntity<>(errorInfo, HttpStatus.BAD_REQUEST);
}
}
Writable
Smart Insert
[Postman]
[Mozilla Firefox]
workspace - iap...


ExceptionControllerAdvice.java - Eclipse IDE
Donate
ExceptionControllerAdvice.java
1 package com.infy.hospitalmanagement.utility;
2
3 import java.util.stream.Collectors;
17
18
19 public class ExceptionControllerAdvice {
20
21 private static final Log LOGGER = LogFactory.getLog(ExceptionControllerAdvice.class);
22
23 private Environment environment;
24
25 public ResponseEntity<ErrorInfo> patientAdmissionExceptionHandler(PatientAdmissionException exception) {
26 LOGGER.error(exception.getMessage(), exception);
27 ErrorInfo errorInfo = new ErrorInfo();
28 errorInfo.setErrorCode(HttpStatus.BAD_REQUEST.value());
29 errorInfo.setErrorMessage(environment.getProperty(exception.getMessage()));
30 return new ResponseEntity<>(errorInfo, HttpStatus.BAD_REQUEST);
31 }
32
33 public ResponseEntity<ErrorInfo> generalExceptionHandler(Exception exception) {
34 LOGGER.error(exception.getMessage(), exception);
35 ErrorInfo errorInfo = new ErrorInfo();
36 errorInfo.setErrorCode(HttpStatus.INTERNAL_SERVER_ERROR.value());
37 errorInfo.setErrorMessage(environment.getProperty("General.EXCEPTION_MESSAGE"));
38 return new ResponseEntity<>(errorInfo, HttpStatus.INTERNAL_SERVER_ERROR);
39 }
40
41 public ResponseEntity<ErrorInfo> validatorExceptionHandler(Exception exception) {
42 LOGGER.error(exception.getMessage(), exception);
43 String errorMsg;
44 if (exception instanceof MethodArgumentNotValidException) {
45 MethodArgumentNotValidException manvException = (MethodArgumentNotValidException) exception;
46 errorMsg = manvException.getBindingResult().getAllErrors().stream().map(ObjectError::getDefault
47 .collect(Collectors.joining(", "));
48 } else {
49 ConstraintViolationException cvException = (ConstraintViolationException) exception;
Writable
Smart Insert
[Postman]
[Mozilla Firefox]
workspace - iap...

Donate | ExceptionControllerAdvice.java

```
24
25@ic ResponseEntity<ErrorInfo> patientAdmissionExceptionHandler(PatientAdmissionException exception) {
26 LOGGER.error(exception.getMessage(), exception);
27 ErrorInfo errorInfo = new ErrorInfo();
28 errorInfo.setErrorCode(HttpStatus.BAD_REQUEST.value());
29 errorInfo.setErrorMessage(environment.getProperty(exception.getMessage()));
30 return new ResponseEntity<>(errorInfo, HttpStatus.BAD_REQUEST);
31
32
33@ic ResponseEntity<ErrorInfo> generalExceptionHandler(Exception exception) {
34 LOGGER.error(exception.getMessage(), exception);
35 ErrorInfo errorInfo = new ErrorInfo();
36 errorInfo.setErrorCode(HttpStatus.INTERNAL_SERVER_ERROR.value());
37 errorInfo.setErrorMessage(environment.getProperty("General.EXCEPTION_MESSAGE"));
38 return new ResponseEntity<>(errorInfo, HttpStatus.INTERNAL_SERVER_ERROR);
39
40
41@ic ResponseEntity<ErrorInfo> validatorExceptionHandler(Exception exception) {
42 LOGGER.error(exception.getMessage(), exception);
43 String errorMsg;
44 if (exception instanceof MethodArgumentNotValidException) {
45     MethodArgumentNotValidException manvException = (MethodArgumentNotValidException) exception;
46     errorMsg = manvException.getBindingResult().getAllErrors().stream().map(ObjectError::getDefaultMessage)
47             .collect(Collectors.joining(", "));
48 } else {
49     ConstraintViolationException cvException = (ConstraintViolationException) exception;
50     errorMsg = cvException.getConstraintViolations().stream().map(ConstraintViolation::getMessage)
51             .collect(Collectors.joining(", "));
52 }
53 ErrorInfo errorInfo = new ErrorInfo();
54 errorInfo.setErrorCode(HttpStatus.BAD_REQUEST.value());
55 errorInfo.setErrorMessage(errorMsg);
56 return new ResponseEntity<>(errorInfo, HttpStatus.BAD_REQUEST);
57
58
59
```

Writable | Smart Insert

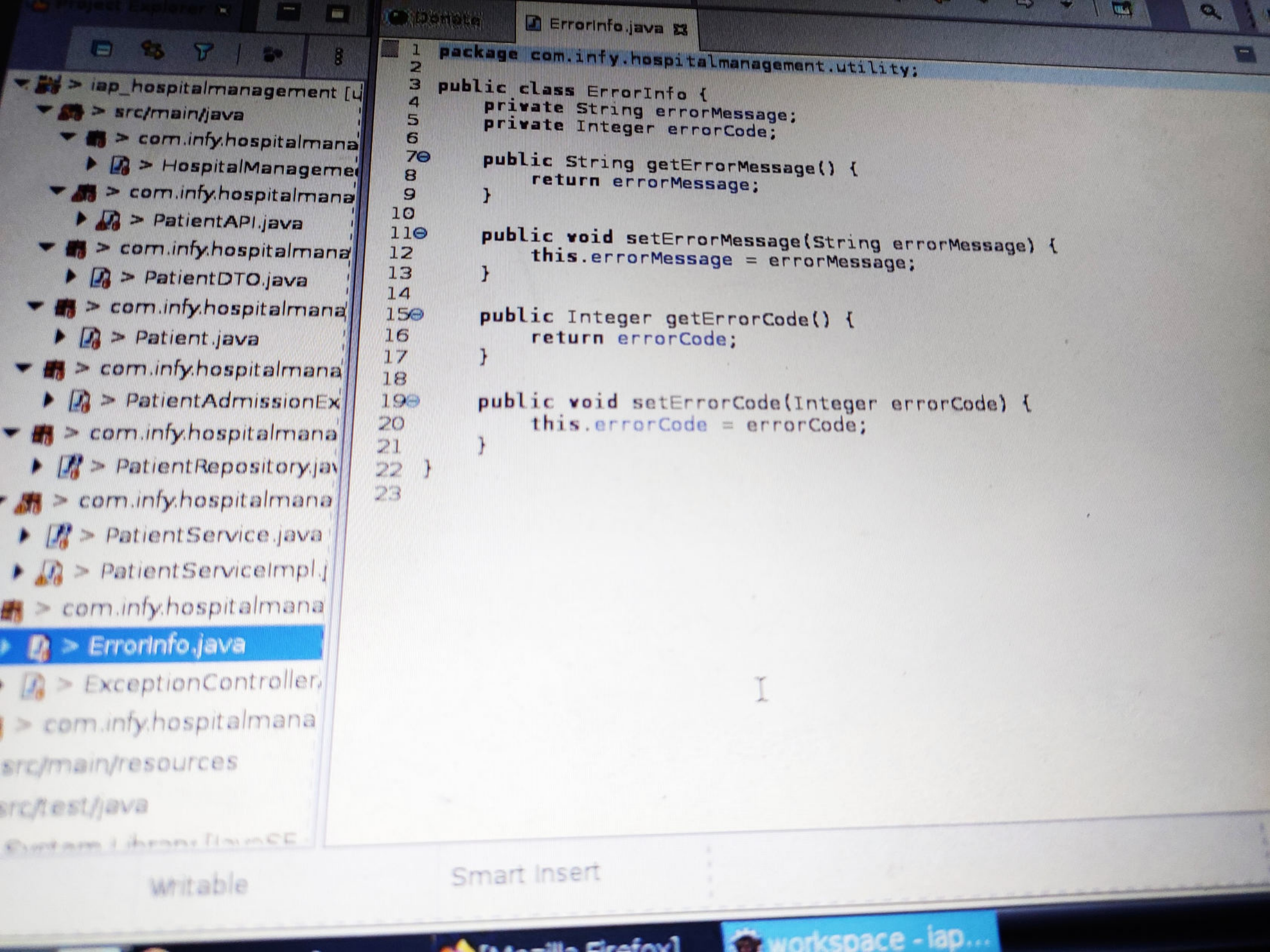
[Postman] [Mozilla Firefox] workspace - iap...

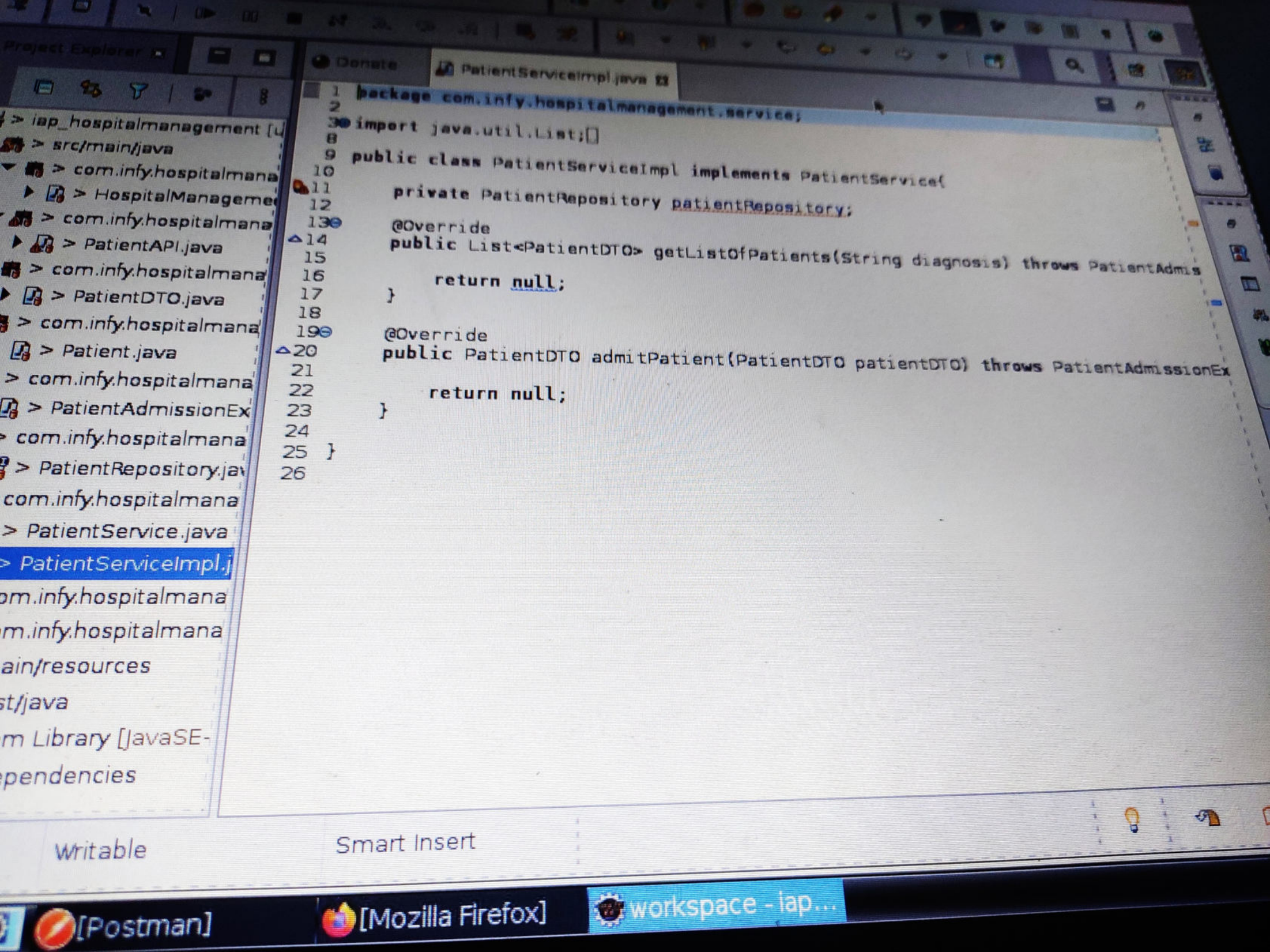
Donate | ExceptionControllerAdvice.java

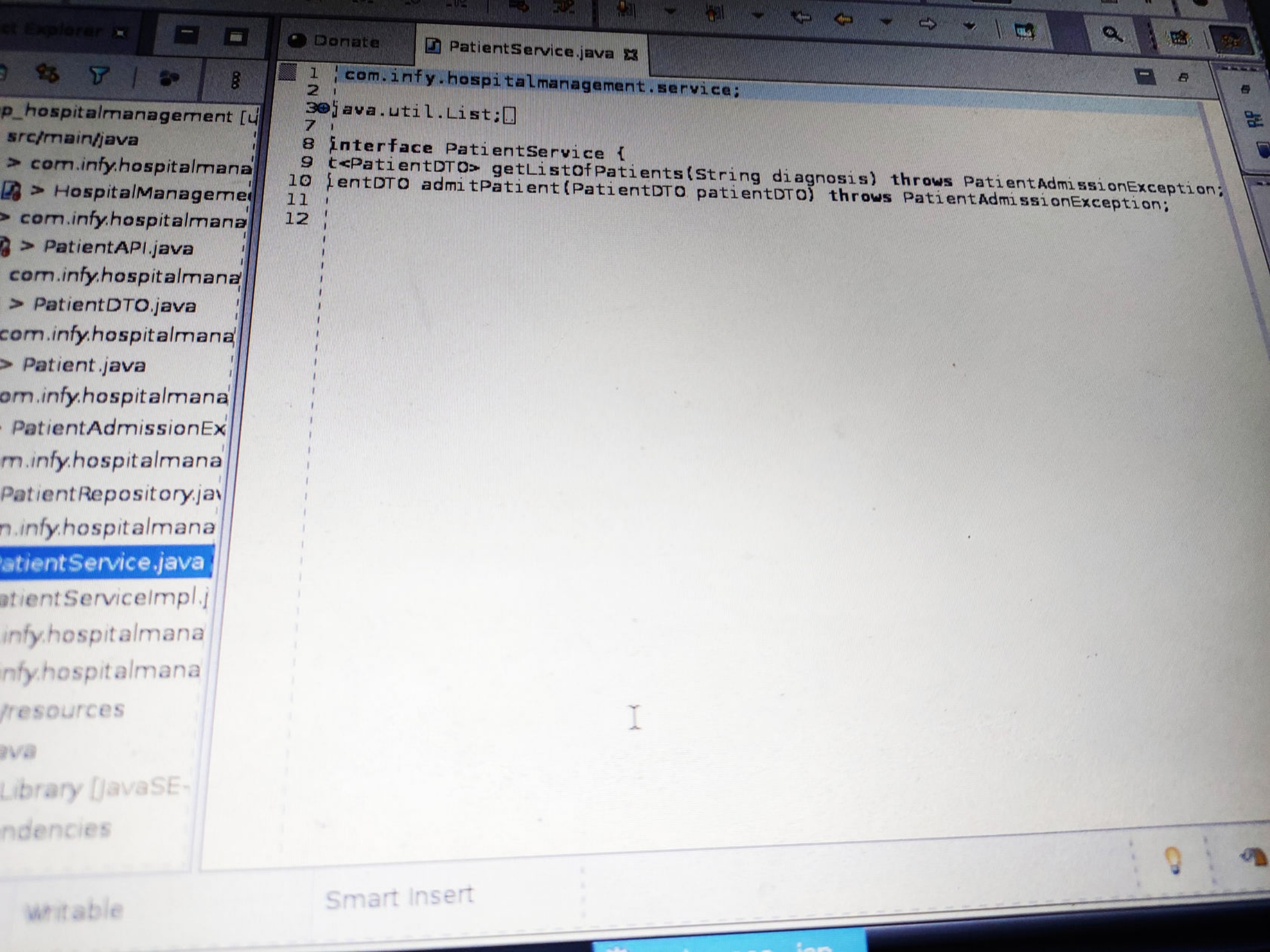
```
1  com.infy.hospitalmanagement.utility;
2
30 ava.util.stream.Collectors;
17
18
19 lass ExceptionControllerAdvice {
20
21 ate static final Log LOGGER = LogFactory.getLog(ExceptionControllerAdvice.class);
22
23 ate Environment environment;
24
25 ic ResponseEntity<ErrorInfo> patientAdmissionExceptionHandler(PatientAdmissionException exception) {
26  LOGGER.error(exception.getMessage(), exception);
27  ErrorInfo errorInfo = new ErrorInfo();
28  errorInfo.setErrorCode(HttpStatus.BAD_REQUEST.value());
29  errorInfo.setErrorMessage(environment.getProperty(exception.getMessage()));
30  return new ResponseEntity<>(errorInfo, HttpStatus.BAD_REQUEST);
31
32
33 ic ResponseEntity<ErrorInfo> generalExceptionHandler(Exception exception) {
34  LOGGER.error(exception.getMessage(), exception);
35  ErrorInfo errorInfo = new ErrorInfo();
36  errorInfo.setErrorCode(HttpStatus.INTERNAL_SERVER_ERROR.value());
37  errorInfo.setErrorMessage(environment.getProperty("General.EXCEPTION_MESSAGE"));
38  return new ResponseEntity<>(errorInfo, HttpStatus.INTERNAL_SERVER_ERROR);
39
40
41 ic ResponseEntity<ErrorInfo> validatorExceptionHandler(Exception exception) {
42  LOGGER.error(exception.getMessage(), exception);
43  String errorMsg;
44  if (exception instanceof MethodArgumentNotValidException) {
45      MethodArgumentNotValidException manvException = (MethodArgumentNotValidException) exception;
46      errorMsg = manvException.getBindingResult().getAllErrors().stream().map(ObjectError::getDefaultMessage)
47              .collect(Collectors.joining(", "));
48  } else {
49      ConstraintViolationException cvException = (ConstraintViolationException) exception;
```

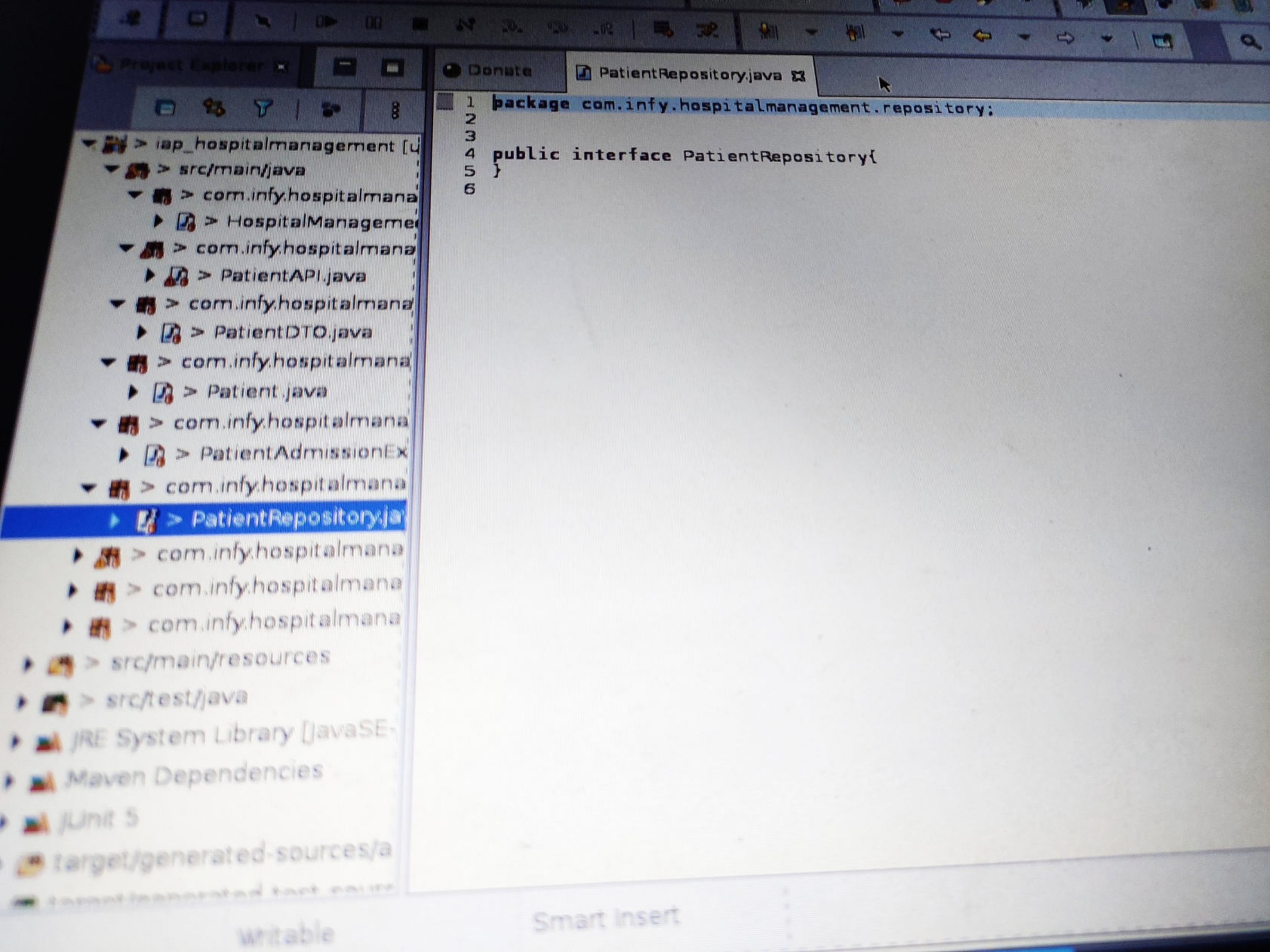
Writable | Smart Insert

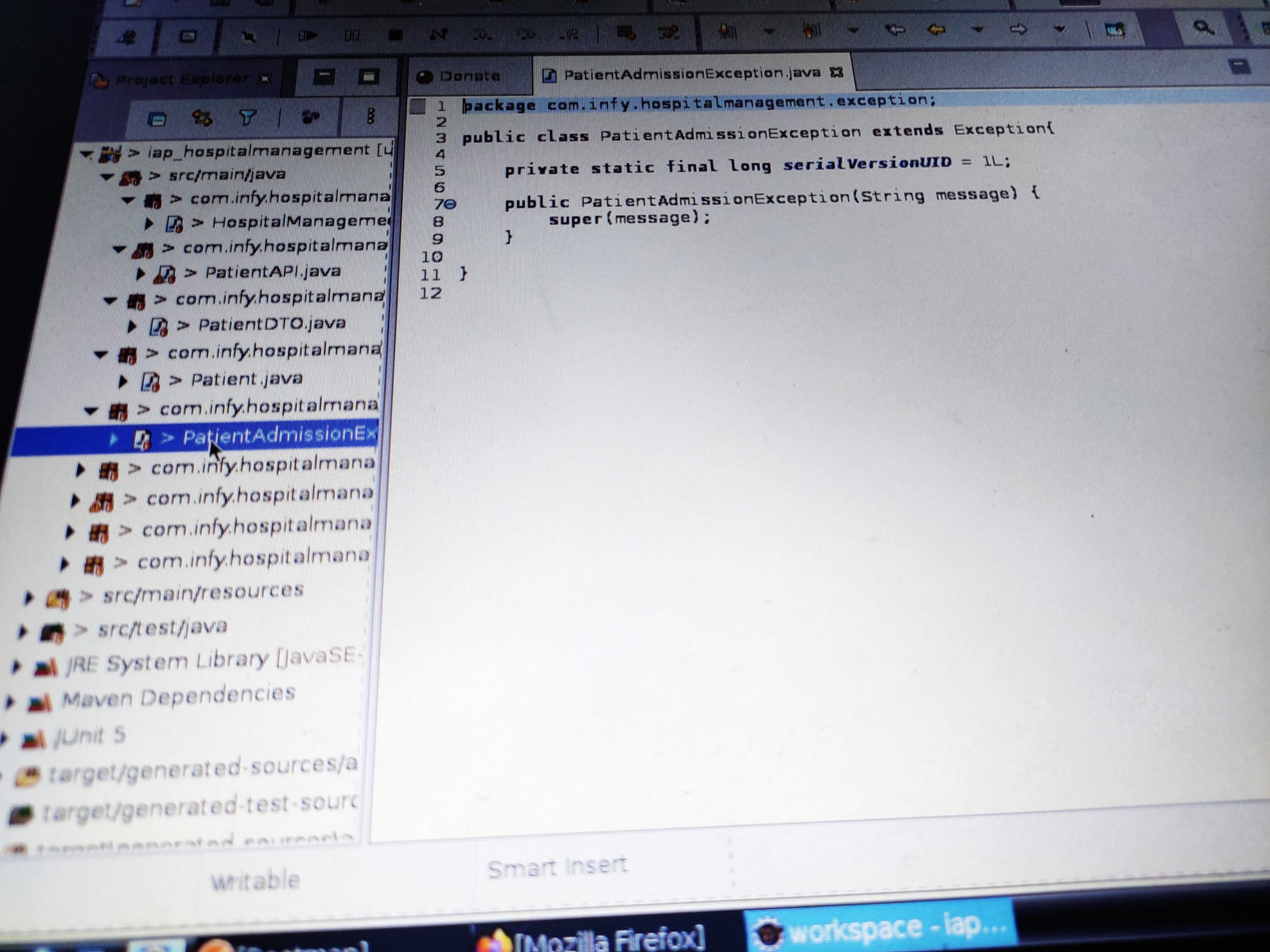
[Postman] [Mozilla Firefox] workspace - iap...

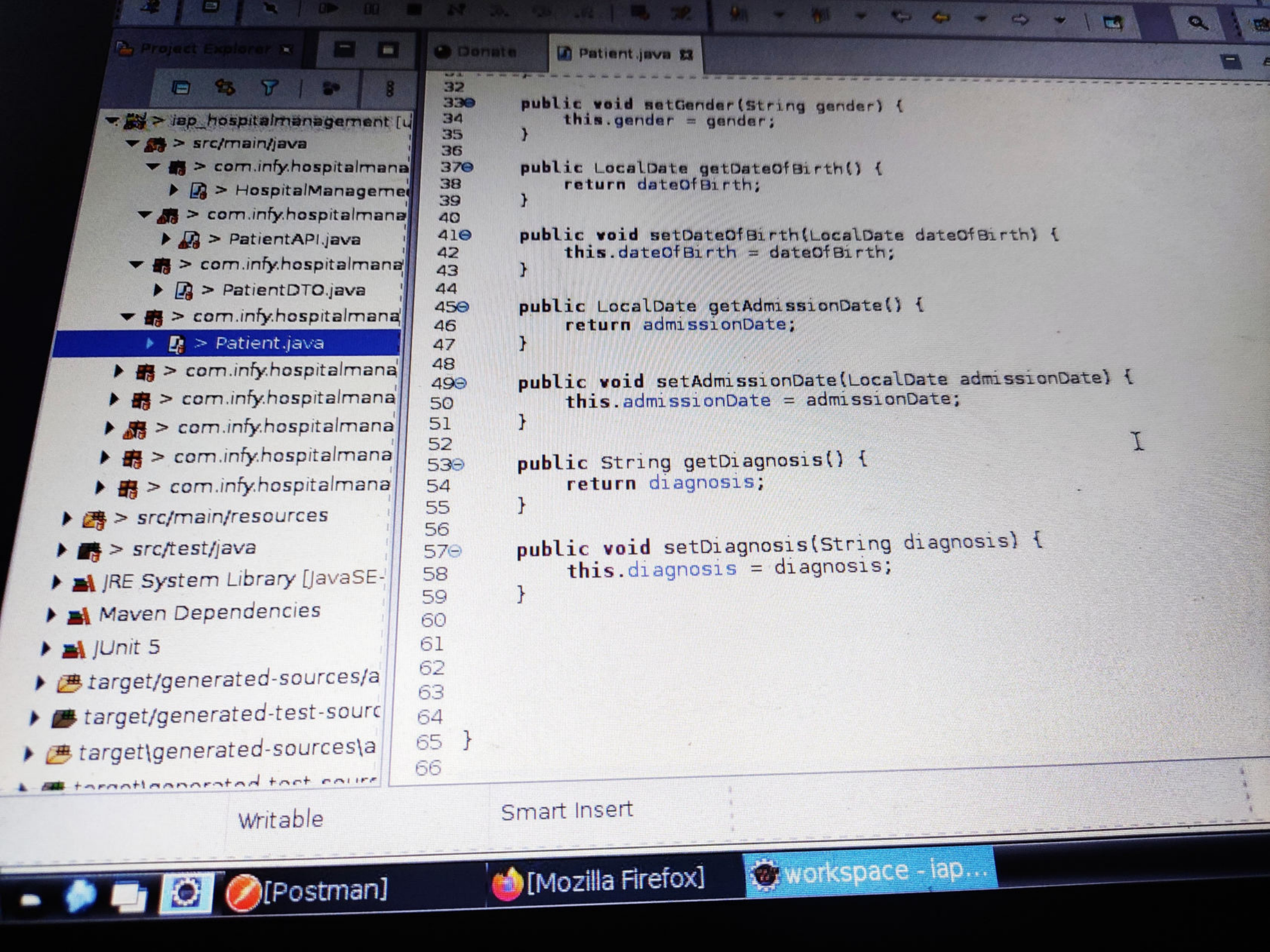

Project Explorer
> iap_hospitalmanagement [u
> src/main/java
> com.infy.hospitalmana
> HospitalManageme
> com.infy.hospitalmana
> PatientAPI.java
> com.infy.hospitalmana
> PatientDTO.java
> com.infy.hospitalmana
> Patient.java
> com.infy.hospitalmana
> PatientAdmissionEx
> com.infy.hospitalmana
> PatientRepository.jav
> com.infy.hospitalmana
> PatientService.java
> PatientServiceImpl.j
> com.infy.hospitalmana
> ErrorInfo.java
> ExceptionController
> com.infy.hospitalmana
src/main/resources
src/test/java
Console
ErrorInfo.java
1 package com.infy.hospitalmanagement.utility;
2
3 public class ErrorInfo {
4 private String errorMessage;
5 private Integer errorCode;
6
7 public String getErrorMessage() {
8 return errorMessage;
9 }
10
11 public void setErrorMessage(String errorMessage) {
12 this.errorMessage = errorMessage;
13 }
14
15 public Integer getErrorCode() {
16 return errorCode;
17 }
18
19 public void setErrorCode(Integer errorCode) {
20 this.errorCode = errorCode;
21 }
22 }
23
Writable
Smart Insert
workspace - iap...

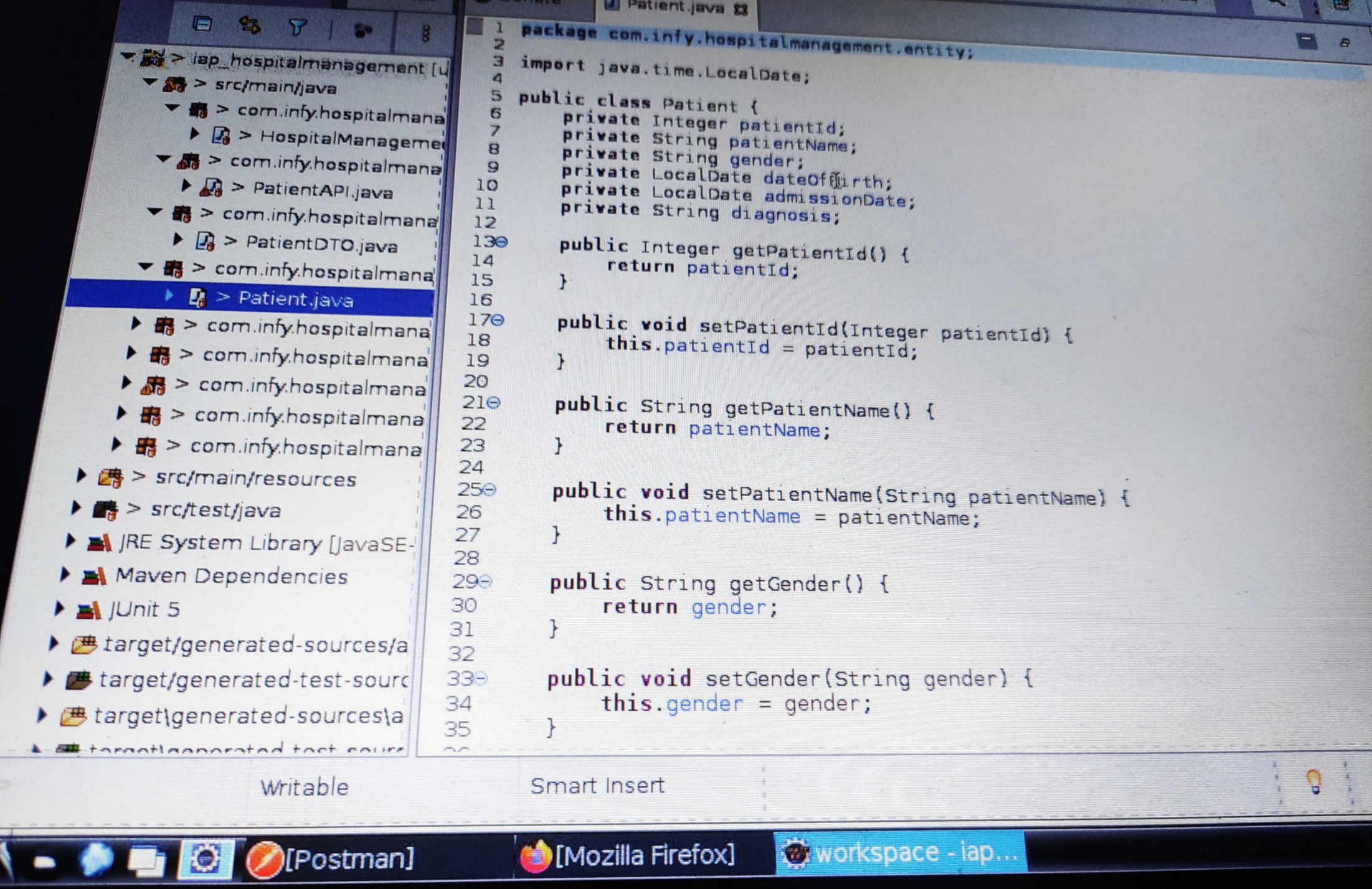

Project Explorer
> iap_hospitalmanagement [u
> src/main/java
> com.infy.hospitalmana
> HospitalManageme
> com.infy.hospitalmana
> PatientAPI.java
> com.infy.hospitalmana
> PatientDTO.java
> com.infy.hospitalmana
> Patient.java
> com.infy.hospitalmana
> PatientAdmissionEx
com.infy.hospitalmana
> PatientRepository.jav
com.infy.hospitalmana
> PatientService.java
> PatientServiceImpl.j
om.infy.hospitalmana
m.infy.hospitalmana
ain/resources
st/java
m Library [JavaSE-
pendencies
Donate
PatientServiceImpl.java
1 package com.infy.hospitalmanagement.service;
2
3 import java.util.List;
8
9 public class PatientServiceImpl implements PatientService{
10
11 private PatientRepository patientRepository;
12
13 @Override
14 public List<PatientDTO> getListOfPatients(String diagnosis) throws PatientAdmis
15
16 return null;
17 }
18
19 @Override
20 public PatientDTO admitPatient(PatientDTO patientDTO) throws PatientAdmissionEx
21
22 return null;
23 }
24
25 }
26
Writable
Smart Insert
[Postman]
[Mozilla Firefox]
workspace - iap...

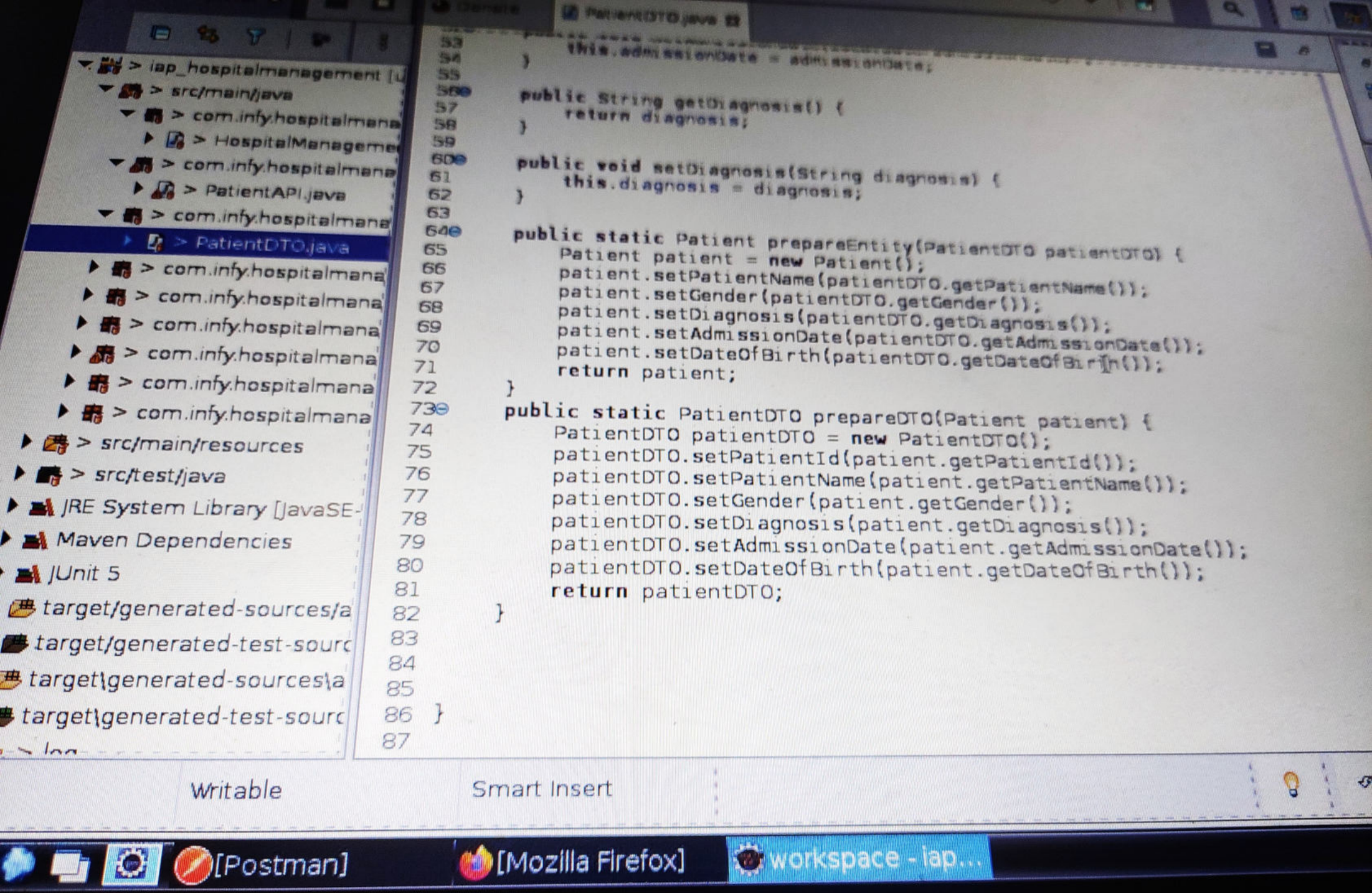

Donate
PatientService.java
1 com.infy.hospitalmanagement.service;
3 java.util.List;
8 interface PatientService {
9 t<PatientDTO> getListOfPatients(String diagnosis) throws PatientAdmissionException;
10 ientDTO admitPatient(PatientDTO patientDTO) throws PatientAdmissionException;
11
12
Smart Insert

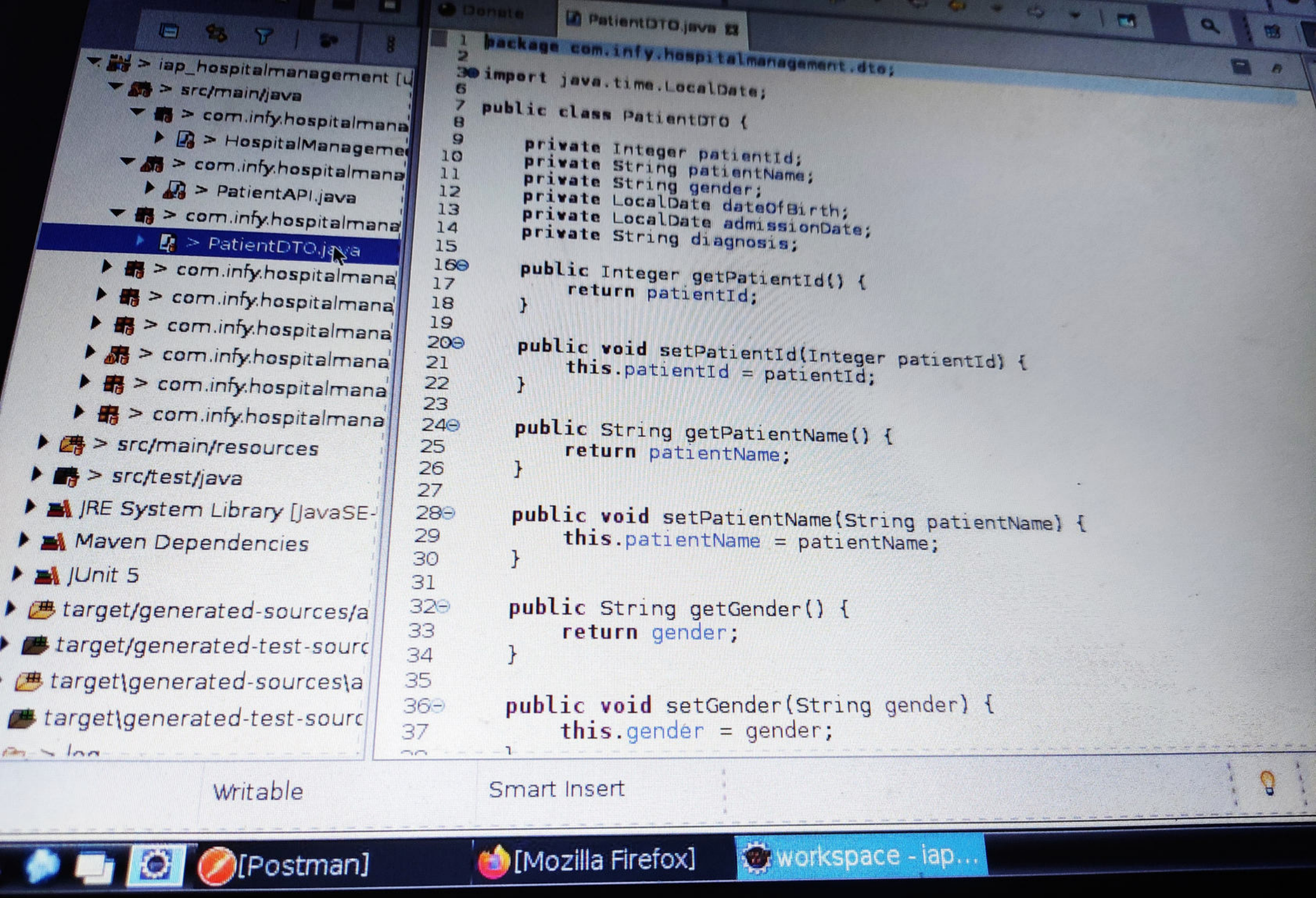

Project Explorer
Donate
PatientRepository.java
1 package com.infy.hospitalmanagement.repository;
2
3
4 public interface PatientRepository{
5 }
6
> iap_hospitalmanagement [u
> src/main/java
> com.infy.hospitalmana
> HospitalManageme
> com.infy.hospitalmana
> PatientAPI.java
> com.infy.hospitalmana
> PatientDTO.java
> com.infy.hospitalmana
> Patient.java
> com.infy.hospitalmana
> PatientAdmissionEx
> com.infy.hospitalmana
> PatientRepository.ja
> com.infy.hospitalmana
> com.infy.hospitalmana
> com.infy.hospitalmana
> src/main/resources
> src/test/java
JRE System Library [JavaSE-
Maven Dependencies
JUnit 5
target/generated-sources/a
Writable
Smart Insert

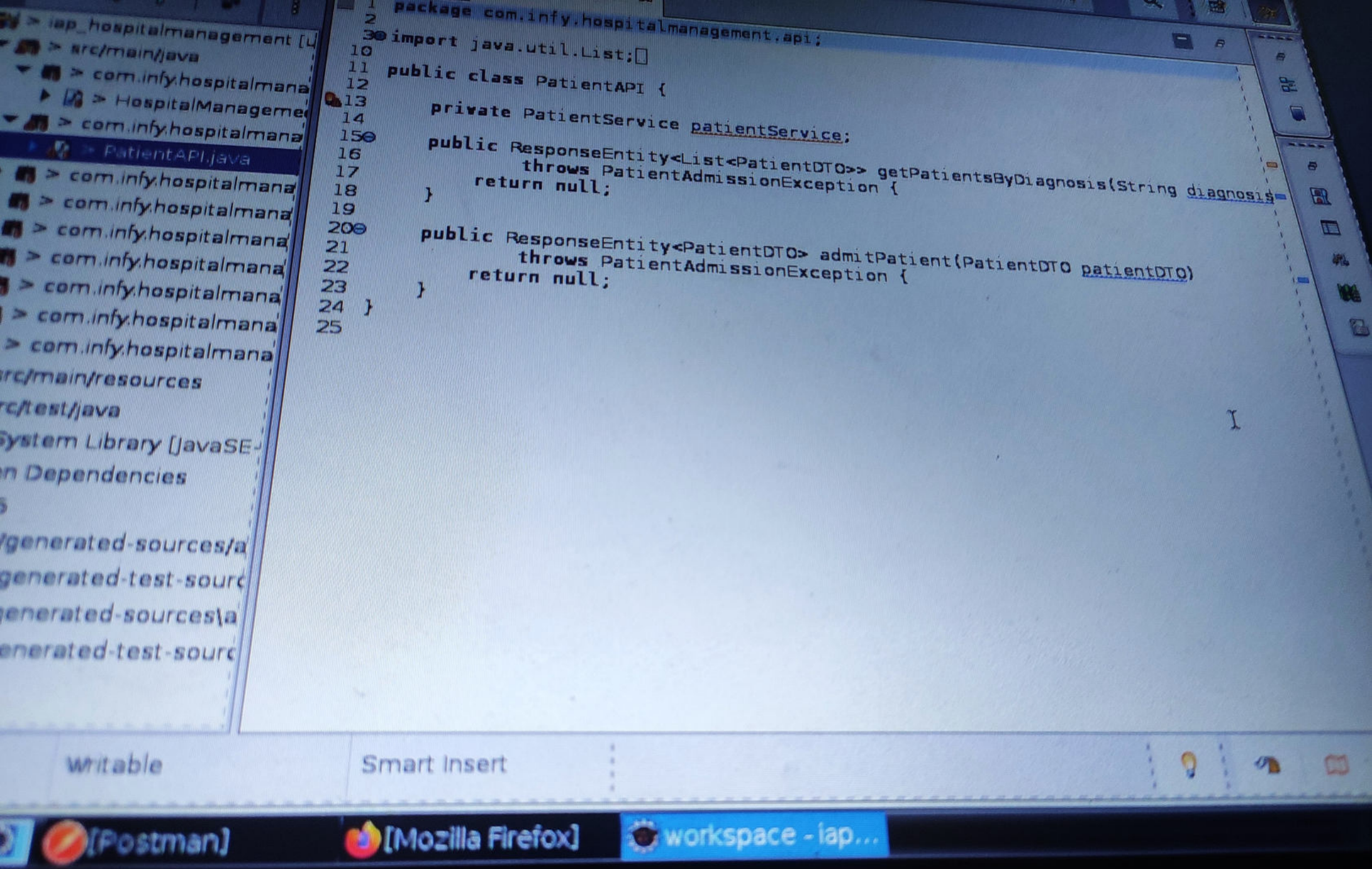

Project Explorer
Donate
PatientAdmissionException.java
1 package com.infy.hospitalmanagement.exception;
2
3 public class PatientAdmissionException extends Exception{
4
5 private static final long serialVersionUID = 1L;
6
7 public PatientAdmissionException(String message) {
8 super(message);
9 }
10
11 }
12
> iap_hospitalmanagement [u
> src/main/java
> com.infy.hospitalmana
> HospitalManageme
> com.infy.hospitalmana
> PatientAPI.java
> com.infy.hospitalmana
> PatientDTO.java
> com.infy.hospitalmana
> Patient.java
> com.infy.hospitalmana
> PatientAdmissionEx
> com.infy.hospitalmana
> com.infy.hospitalmana
> com.infy.hospitalmana
> com.infy.hospitalmana
> src/main/resources
> src/test/java
JRE System Library [JavaSE-
Maven Dependencies
JUnit 5
target/generated-sources/a
target/generated-test-sourc
Writable
Smart Insert
[Mozilla Firefox]
workspace - iap...

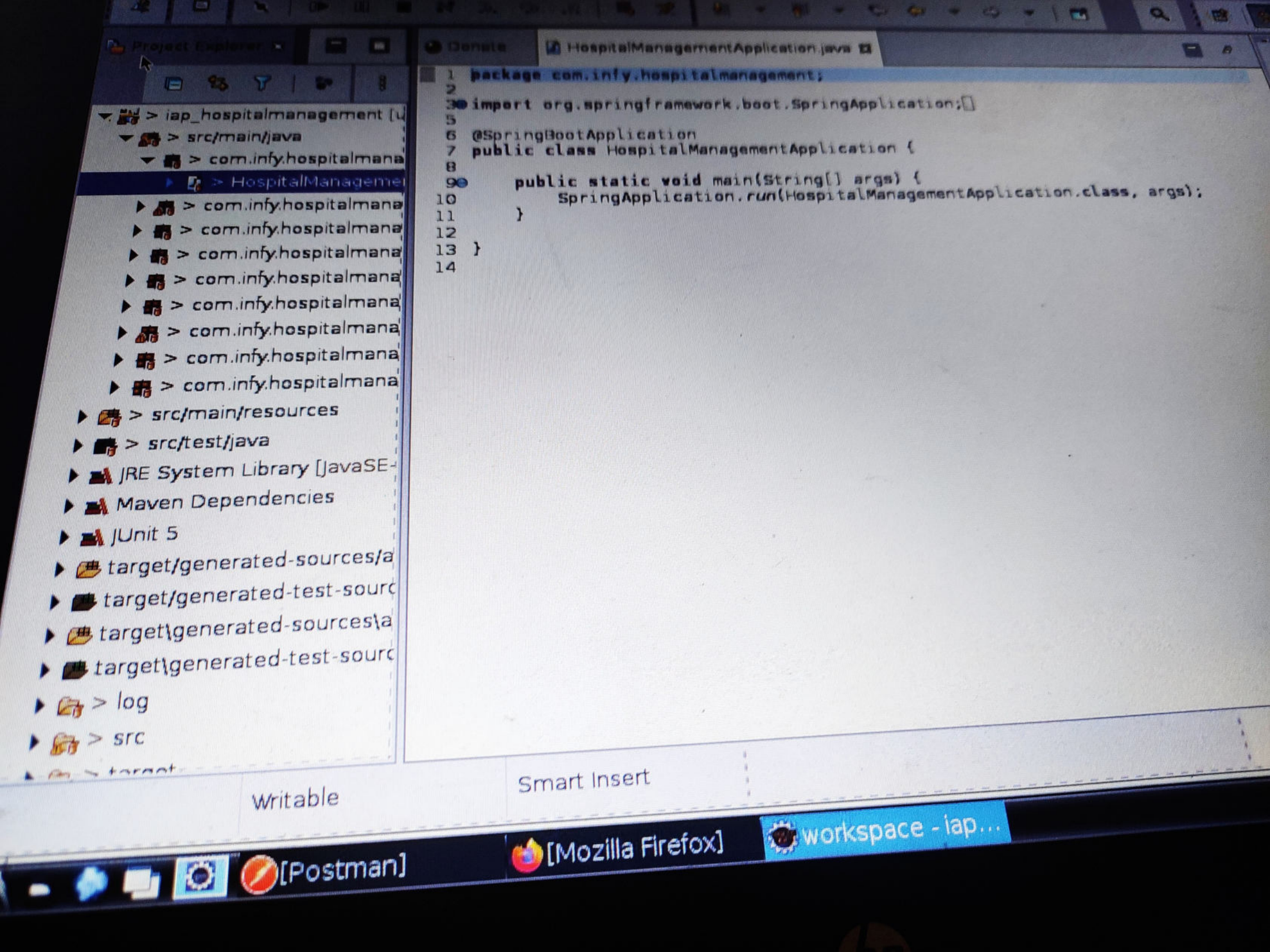

Project Explorer
Donate
Patient.java
> iap_hospitalmanagement [u
> src/main/java
> com.infy.hospitalmana
> HospitalManageme
> com.infy.hospitalmana
> PatientAPI.java
> com.infy.hospitalmana
> PatientDTO.java
> com.infy.hospitalmana
> Patient.java
> com.infy.hospitalmana
> com.infy.hospitalmana
> com.infy.hospitalmana
> com.infy.hospitalmana
> com.infy.hospitalmana
> src/main/resources
> src/test/java
JRE System Library [JavaSE-
Maven Dependencies
JUnit 5
target/generated-sources/a
target/generated-test-sourc
target\generated-sources\a
32
33 public void setGender(String gender) {
34 this.gender = gender;
35 }
36
37 public LocalDate getDateOfBirth() {
38 return dateOfBirth;
39 }
40
41 public void setDateOfBirth(LocalDate dateOfBirth) {
42 this.dateOfBirth = dateOfBirth;
43 }
44
45 public LocalDate getAdmissionDate() {
46 return admissionDate;
47 }
48
49 public void setAdmissionDate(LocalDate admissionDate) {
50 this.admissionDate = admissionDate;
51 }
52
53 public String getDiagnosis() {
54 return diagnosis;
55 }
56
57 public void setDiagnosis(String diagnosis) {
58 this.diagnosis = diagnosis;
59 }
60
61
62
63
64
65 }
66
Writable
Smart Insert
[Postman]
[Mozilla Firefox]
workspace - iap...


Patient.java
1 package com.infy.hospitalmanagement.entity;
2
3 import java.time.LocalDate;
4
5 public class Patient {
6 private Integer patientId;
7 private String patientName;
8 private String gender;
9 private LocalDate dateOfBirth;
10 private LocalDate admissionDate;
11 private String diagnosis;
12
13 public Integer getPatientId() {
14 return patientId;
15 }
16
17 public void setPatientId(Integer patientId) {
18 this.patientId = patientId;
19 }
20
21 public String getPatientName() {
22 return patientName;
23 }
24
25 public void setPatientName(String patientName) {
26 this.patientName = patientName;
27 }
28
29 public String getGender() {
30 return gender;
31 }
32
33 public void setGender(String gender) {
34 this.gender = gender;
35 }
iap_hospitalmanagement
src/main/java
com.infy.hospitalmana
HospitalManageme
com.infy.hospitalmana
PatientAPI.java
com.infy.hospitalmana
PatientDTO.java
com.infy.hospitalmana
Patient.java
com.infy.hospitalmana
com.infy.hospitalmana
com.infy.hospitalmana
com.infy.hospitalmana
com.infy.hospitalmana
src/main/resources
src/test/java
JRE System Library [JavaSE-
Maven Dependencies
JUnit 5
target/generated-sources/a
target/generated-test-sourc
target\generated-sources\a
Writable
Smart Insert
[Postman]
[Mozilla Firefox]
workspace - iap...


> iap_hospitalmanagement
> src/main/java
> com.infy.hospitalmana
> HospitalManageme
> com.infy.hospitalmana
> PatientAPI.java
> com.infy.hospitalmana
> PatientDTO.java
> com.infy.hospitalmana
> com.infy.hospitalmana
> com.infy.hospitalmana
> com.infy.hospitalmana
> com.infy.hospitalmana
> com.infy.hospitalmana
> src/main/resources
> src/test/java
JRE System Library [JavaSE-
Maven Dependencies
JUnit 5
target/generated-sources/a
target/generated-test-sourc
target\generated-sources\a
target\generated-test-sourc
this.admissionDate = admissionDate;
}
public String getDiagnosis() {
return diagnosis;
}
public void setDiagnosis(String diagnosis) {
this.diagnosis = diagnosis;
}
public static Patient prepareEntity(PatientDTO patientDTO) {
Patient patient = new Patient();
patient.setPatientName(patientDTO.getPatientName());
patient.setGender(patientDTO.getGender());
patient.setDiagnosis(patientDTO.getDiagnosis());
patient.setAdmissionDate(patientDTO.getAdmissionDate());
patient.setDateOfBirth(patientDTO.getDateOfBirth());
return patient;
}
public static PatientDTO prepareDTO(Patient patient) {
PatientDTO patientDTO = new PatientDTO();
patientDTO.setPatientId(patient.getPatientId());
patientDTO.setPatientName(patient.getPatientName());
patientDTO.setGender(patient.getGender());
patientDTO.setDiagnosis(patient.getDiagnosis());
patientDTO.setAdmissionDate(patient.getAdmissionDate());
patientDTO.setDateOfBirth(patient.getDateOfBirth());
return patientDTO;
}
}
Writable
Smart Insert
[Postman]
[Mozilla Firefox]
workspace - iap...


> iap_hospitalmanagement [
> src/main/java
> com.infy.hospitalmana
> HospitalManageme
> com.infy.hospitalmana
> PatientAPI.java
> com.infy.hospitalmana
> PatientDTO.java
> com.infy.hospitalmana
> com.infy.hospitalmana
> com.infy.hospitalmana
> com.infy.hospitalmana
> com.infy.hospitalmana
> com.infy.hospitalmana
> src/main/resources
> src/test/java
JRE System Library [JavaSE-
Maven Dependencies
JUnit 5
target/generated-sources/a
target/generated-test-sourc
target\generated-sources\a
target\generated-test-sourc
Donate
PatientDTO.java
1 package com.infy.hospitalmanagement.dto;
3 import java.time.LocalDate;
6
7 public class PatientDTO {
8
9 private Integer patientId;
10 private String patientName;
11 private String gender;
12 private LocalDate dateOfBirth;
13 private LocalDate admissionDate;
14 private String diagnosis;
15
16 public Integer getPatientId() {
17 return patientId;
18 }
19
20 public void setPatientId(Integer patientId) {
21 this.patientId = patientId;
22 }
23
24 public String getPatientName() {
25 return patientName;
26 }
27
28 public void setPatientName(String patientName) {
29 this.patientName = patientName;
30 }
31
32 public String getGender() {
33 return gender;
34 }
35
36 public void setGender(String gender) {
37 this.gender = gender;
Writable
Smart Insert
[Postman]
[Mozilla Firefox]
workspace - iap...


iap_hospitalmanagement [
src/main/java
com.infy.hospitalmana
HospitalManageme
com.infy.hospitalmana
PatientAPI.java
com.infy.hospitalmana
com.infy.hospitalmana
com.infy.hospitalmana
com.infy.hospitalmana
com.infy.hospitalmana
com.infy.hospitalmana
com.infy.hospitalmana
src/main/resources
src/test/java
System Library [JavaSE-
n Dependencies
/generated-sources/a
generated-test-sourc
enerated-sources\a
enerated-test-sourc
1 package com.infy.hospitalmanagement.api;
2
3 import java.util.List;
10
11 public class PatientAPI {
12
13 private PatientService patientService;
14
15 public ResponseEntity<List<PatientDTO>> getPatientsByDiagnosis(String diagnosis
16 throws PatientAdmissionException {
17 return null;
18 }
19
20 public ResponseEntity<PatientDTO> admitPatient(PatientDTO patientDTO)
21 throws PatientAdmissionException {
22 return null;
23 }
24 }
25
Writable
Smart Insert
[Postman]
[Mozilla Firefox]
workspace - iap...


Project Explorer
> iap_hospitalmanagement [u
> src/main/java
> com.infy.hospitalmana
> HospitalManagemer
> com.infy.hospitalmana
> com.infy.hospitalmana
> com.infy.hospitalmana
> com.infy.hospitalmana
> com.infy.hospitalmana
> com.infy.hospitalmana
> com.infy.hospitalmana
> com.infy.hospitalmana
> src/main/resources
> src/test/java
JRE System Library [JavaSE-
Maven Dependencies
JUnit 5
target/generated-sources/a
target/generated-test-sourc
target\generated-sources\a
target\generated-test-sourc
> log
> src
Console
HospitalManagementApplication.java
1 package com.infy.hospitalmanagement;
2
3 import org.springframework.boot.SpringApplication;
5
6 @SpringBootApplication
7 public class HospitalManagementApplication {
8
9 public static void main(String[] args) {
10 SpringApplication.run(HospitalManagementApplication.class, args);
11 }
12
13 }
14
Writable
Smart Insert
[Postman]
[Mozilla Firefox]
workspace - iap...

		"patientName": "Arne Johansson", "gender": "Female", "dateOfBirth": "1995-06-10", "admissionDate": **, } Replace ** with today's date	"errorCode": 400 }

	Invalid test data 7	api/patients
		
		Sample JSON:
		{
		"patientName": "Arne Johansson",
		"gender": "Female",
		"dateOfBirth": "1995-06-10",
		"admissionDate": **,
		"diagnosis": "Cancer,",
		}
		
		Replace ** with today's date
			{
			"errorMessage": "Diagnosis should contain only numbers, alphabets and spaces",
			"errorCode": 400
			}

PREV

	Invalid test data 5	**api/patients**
		
		Sample JSON:
		{
		"patientName": "Arne Johansson",
		"gender": "Female",
		"dateOfBirth": "1995-06-10",
		"admissionDate": "2022-06-14",
		"diagnosis": "Cancer",
		}
			{
			"errorMessage": "Date of admission should be today's date",
			"errorCode": 400
			}

	Invalid test data 6	**api/patients**
		
		Sample JSON:
		{
		"patientName": "Arne Johansson",
		"gender": "Female",
		"dateOfBirth": "1995-06-10",
		"admissionDate": **,
		}
		
		Replace ** with today's date
			{
			"errorMessage": "Diagnosis is mandatory",
			"errorCode": 400
			}

	Invalid test data 7	api/patients Sample JSON:	{ "errorMessage": "Diagnos should contain only numbers,

	Replace ** with today's date	

Invalid test data 3	api/patients
	
	Sample JSON:
	{
	"patientName": "Arne Johansson",
	"gender": "Neutral",
	"dateOfBirth": "1995-06-10",
	"admissionDate": **,
	"diagnosis": "Cancer"
	}
	
	Replace ** with today's date
		{
		"errorMessage": "Gender should be Male, Female or Others",
		"errorCode": 400
		}

Invalid test data 4	api/patients
	
	Sample JSON:
	{
	"patientName": "Arne Johansson",
	"gender": "Female",
	"dateOfBirth": "1880-06-10",
	"admissionDate": **,
	"diagnosis": "Cancer"
	}
	
	Replace ** with today's date
		{
		"errorMessage": "Date of birth should not be a future date and the age should not be more than 100",
		"errorCode": 400
		}

	Invalid test data 1	**api/patients**
		
		Sample JSON:
		{
		"dateOfBirth": "1995-06-10",
		"admissionDate": **,
		"diagnosis": "Cancer"
		}
		
		Replace ** with today's date
			{
			"errorMessage": "Gender is mandatory for admission, Patient name is mandatory",
			"errorCode": 400
			}

	Invalid test data 2	api/patients
		
		Sample JSON:
		{
		"patientName": "Arne2 Johansson",
		"gender": "Female",
		"dateOfBirth": "1995-06-10",
		"admissionDate": **,
		"diagnosis": "Cancer"
		}
		
		Replace ** with today's date
			{
			"errorMessage": "Patient name should have only words and each word should be separated by a space",
			"errorCode": 400
			}

	Invalid test data 3	api/patients
		
		Sample JSON:
		{
		"patientName": "Arne
			{
			"errorMessage": "Gender should be Male, Female or Others
			"errorCode": 400
			}

			{ "errorMessage": "No patient available with given diagnosis", "errorCode": 400 }
Admit a patient	**Valid test data**	**api/patients** **Sample JSON:** { "patientName": "Arne Johansson", "gender": "Female", "dateOfBirth": "1995-06-10", "admissionDate": **, "diagnosis": "Cancer" } Replace ** with today's date	{ "participantId": 1007, "patientName": "Arne Johansson", "gender": "Female", "dateOfBirth": "1995-06-10", "admissionDate": **, "diagnosis": "Cancer", } ** will be replaced by today's date
	Invalid test data 1	api/patients Sample JSON: { "dateOfBirth": "1995-06-10", "admissionDate": **, "diagnosis": "Cancer" } Replace ** with today's date	{ "errorMessage": "Gender is mandatory for admission, Patient name is mandatory", "errorCode": 400 }

	Invalid test data 1	api/patients/Typhoid$	{ "errorMessage": "Diagnosis should contain only numbers, alphabets and spaces", "errorCode": 400 }
	Invalid test data 2	api/patients/Cholera	{ "errorMessage": "No patient available with given diagnosis", "errorCode": 400 }

Admit a patient	Valid test data	api/patients
		
		Sample JSON:
		{
		"patientName": "Arne Johansson",
		"gender": "Female",
		"dateOfBirth": "1995-06-10",
		"admissionDate": **,
		"diagnosis": "Cancer"
		}
		
		Replace ** with today's date
			{
			"participantId": 1007,
			"patientName": "Arne Johansson",
			"gender": "Female",
			"dateOfBirth": "1995-06-10",
			"admissionDate": **,
			"diagnosis": "Cancer",
			}
			
			** will be replaced by today's dat

point			
Get all the patients for the diagnosis	Valid test data	api/patients/Typhoid	[{ "patientId": 1006, "patientName": "Mark", "gender": "Male", "dateOfBirth": "2015-09-12", "admissionDate": "2021-06-14", "diagnosis": "Typhoid" }, { "patientId": 1001, "patientName": "Matthew", "gender": "Male", "dateOfBirth": "1998-08-21", "admissionDate": "2021-01-02", "diagnosis": "Typhoid" }, { "patientId": 1003, "patientName": "John", "gender": "Male", "dateOfBirth": "2015-09-12", "admissionDate": "2020-09-20", "diagnosis": "Typhoid" }]

ConstraintViolationException exceptions

- Add appropriate annotation to make this method as exception handler method for **MethodArgumentNotValidException** and **ConstraintViolationException** exceptions

Please find in the table below sample input and output for REST end points.
Note: The date values have been used considering the present date (today) to be 14th June, 2021. Please change the dates appropriately to test your implementation.

REST end point	Request Type(valid/invalid)	End point URI with sample input	Sample Response
Get all the patients for the diagnosis	Valid test data	api/patients/Typhoid	[{ "patientId": 1006, "patientName": "Mark", "gender": "Male", "dateOfBirth": "2015-09-12", "admissionDate": "2021-06-14", "diagnosis": "Typhoid" }, { "patientId": 1001, "patientName": "Matthew "gender": "Male"

- Inject Environment objects using appropriate annotation
- The body of the methods have been implemented for you

Method Description:

1. **ResponseEntity<ErrorInfo> patientAdmissionExceptionHandler(PatientAdmissionException exception)**
   - This method is for handling **PatientAdmissionException** exception
   - Add appropriate annotation to make this method as exception handler for the PatientAdmissionException
2. **ResponseEntity<ErrorInfo> generalExceptionHandler(Exception exception)**
   - This method is for handling general exceptions
   - Add appropriate annotation to make this method as exception handler for any type of exceptions
3. **ResponseEntity<ErrorInfo> validatorExceptionHandler(Exception exception)**
   - This method is for handling **MethodArgumentNotValidException** and **ConstraintViolationException** exceptions
   - Add appropriate annotation to make this method as exception handler method for **MethodArgumentNotValidException** and **ConstraintViolationException** exceptions

Please find in the table below sample input and output for REST end points.

Note: *The date values have been used considering the present date (today) to be 14th June, 2021. Please ch*

the dates appropriately to test your implementation.

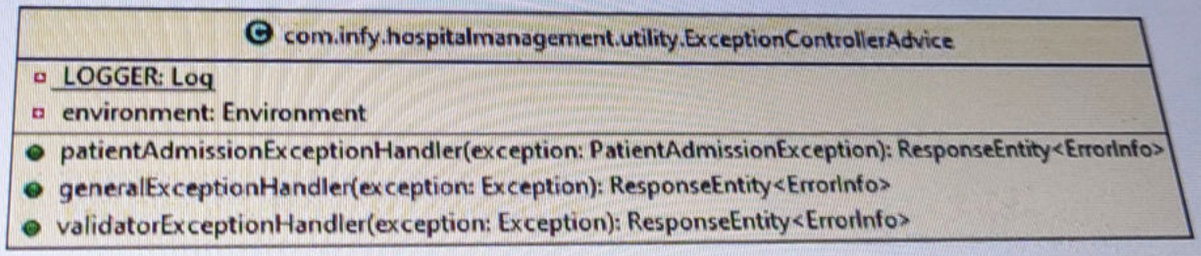
- **admissionDate:**
  If admissionDate is not provided then set **Date of admission is mandatory** as the validation message
  If it is provided, it should be today's date only. If invalid, set **Date of admission should be today's date** as the validation message
- **diagnosis:**
  If diagnosis is not provided then set **Diagnosis is mandatory** as the validation message
  If it is provided, it should contain only numbers, alphabets and spaces as the validation message

- Then invoke the appropriate service class method to admit the patient by passing the **PatientDTO** object which will return the same object with the **patientId** updated.
- Return an object of **ResponseEntity** with the PatientDTO object obtained in the previous step as the response and the HTTP response status as **CREATED**.

7. **Utility Classes**

com.infy.hospitalmanagement.utility.ExceptionControllerAdvice.java

Annotate the following ExceptionControllerAdvice to log and handle all the exceptions of the API class,



- Inject Environment objects using appropriate annotation
- The body of the methods have been implemented for you

- Return an object of **ResponseEntity** with the List<PatientDTO> object obtained in the previous step as a response and the HTTP response status as **OK**.

2. **public ResponseEntity<PatientDTO> admitPatient (PatientDTO patientDTO)**
   - This method admits a patient in the HospitalManagement application
   - It accepts **POST** request with URI **patients**
   - It receives patient details in the form of JSON. The JSON data should be populated in **PatientDTO** parameter of this method after successful validation.
   - The patient details will consist of the patient name, gender, date of birth, admission date and diagnosis. Using **Bean Validation API** validate the details according to the description given below (**Note:** The validation message should not be hard-coded. It should be read from the **ValidationMessages.properties** file),
     - **patientName:**
       If patientName is not provided then set **Patient name is mandatory** as the validation message
       If it is provided, it should contain only words of English alphabets and each word should be separated by a space. If invalid, set **Patient name should have only words and each word should be separated by a space** as the validation message
     - **gender:**
       If gender is not provided then set **Gender is mandatory for admission** as the validation message
       If it is provided,it should be Male, Female or Others. If invalid, set **Gender should be Male, Female or Others** as the validation message
     - **admissionDate:**
       If admissionDate is not provided then set **Date of admission is mandatory** as the validation message
       If it is provided, it should be today's date only. If invalid, set **Date of admission should be t**

Method Description:

1. **ResponseEntity<List<PatientDTO>> getPatientsByDiagnosis (String diagnosis)**
   - This method is to retrieve all the patients having the given diagnosis
   - It accepts **GET** request with URI **patients/{diagnosis}**
   - The path variable diagnosis should be validated as below (use **Bean Validation API** for validation),
     - **diagnosis** can have words containing alphabets and numbers, and each word should be separated by a single space. If invalid, set **Diagnosis should contain only numbers, alphabets and spaces** as the validation message
       Note: The validation message should not be hard-coded. It should be read from the **ValidationMessages.properties** file
   - Once validated, invoke the appropriate service class method by passing the diagnosis to get the **List<PatientDTO>**
   - Return an object of **ResponseEntity** with the List<PatientDTO> object obtained in the previous step as a response and the HTTP response status as **OK**.

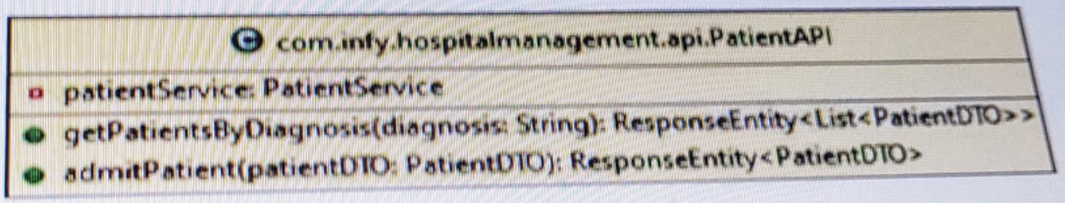
2. ***public* ResponseEntity<PatientDTO> admitPatient (PatientDTO patientDTO)**
   - This method admits a patient in the HospitalManagement application
   - It accepts **POST** request with URI **patients**
   - It receives patient details in the form of JSON. The JSON data should be populated in **PatientDTO** parameter of this method after successful validation.
   - The patient details will consist of the patient name, gender, date of birth, admission date and diagnosis
     Using **Bean Validation API** validate the details according to the description given below (**Note:**
     validation message should not be hard-coded. It should be read from the

2. **PatientDTO admitPatient(PatientDTO patientDTO)**
   - This method is used to admit a new patient details to the database
   - Invoke the validatePatient() method of the PatientValidator class to validate the object of the PatientDTO class received as parameter
   - Create and populate a new object of **Patient** with the appropriate values from the **PatientDTO** object (Hint: You can use the prepareEntity() method of the PatientDTO class for this)
   - Then invoke the appropriate method of the PatientRepository to add the **Patient** object to the database which will return the same object with the auto-generated **patientId**
   - Set the **patientId** of the **Patient** object obtained in the previous step to the existing **PatientDTO** object
   - Return the modified **PatientDTO** object

6. API Class

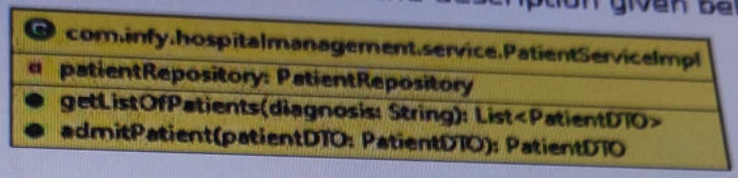
com.infy.hospitalmanagement.api.PatientAPI.java

Implement this class according to the class diagram and desciption given below:



- Annotate this class with appropriate annotation to declare it as REST controller class
- Annotate this class with appropriate annotation so that all its methods are mapped with **api** as base URL
- Annotate this class with appropriate annotation to enable the bean validation support
- Inject instance of PatientService class using appropriate annotation

5. **Service Class**
   com.infy.hospitalmanagement.service.PatientServiceImpl.java
   Implement this class according to the class diagram and the description given below:



- Inject the **PatientRepository** objects using the proper annotation
- Mark it with the appropriate stereotype annotation with the value **patientService**
- Mark this class with necessary annotation to manage transactions

Method Description:

1. **List<PatientDTO> getListOfPatients(String diagnosis)**
   - This method is used to get all the patients having similar diagnosis
   - Invoke the appropriate method of the PatientRepository interface which will return all the patients on the basis of their diagnosis
   - If no patients details are available, throw an object of the **PatientAdmissionException** with the message **PatientService.PATIENT_UNAVAILABLE**
   - Else, populate a list, **List<PatientDTO>** with all the patient details in the descending order of the admission date and return the sorted list

   Hint: You can use the prepareDTO() method of the PatientDTO class to convert the respective objects t
   PatientDTO

1. **void validatePatient(PatientDTO patientDTO)**
   This method receives an object of PatientDTO and calls the respective methods to validate the values and throws an object of PatientAdmissionException with the message, based on the violation, as given below,

Violation for	Message
dateOfBirth	PatientValidator.INVALID_DOB

2. **Boolean IsValidDateOfBirth(LocalDate dateOfBirth)**
   - This method validates the **dateOfBirth** received
   - **dateOfBirth** should not be a future date.
   - **dateOfBirth** should not be a date which is lesser than 100 years from today's date.
   - If the above conditions fail then return false, else return true

4. **Repository**
   com.infy.hospitalmanagement.repository.PatientRepository.java
   - This interface should extend the appropriate repository interface
   - Add the following method to the interface
     - This method should accept the name of the diagnosis and return the list of patients having same valu of **diagnosis**.

5. **Service Class**
   com.infy.hospitalmanagement.service.PatientServiceImpl.java
   Implement this class according to the class diagram and the description given below:

2. **DTO Class**

com.infy.hospitalmanagement.dto.PatientDTO.java

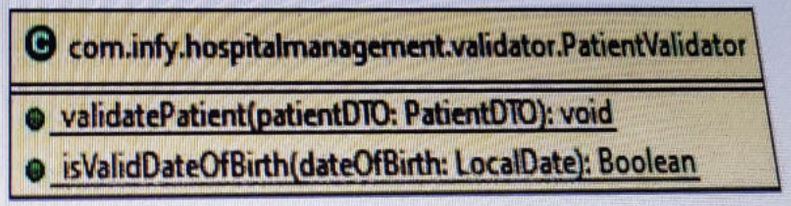
This is a DTO (Data Transfer Object) class that holds patient details such as patient id, patient name, gender, date of birth, admission date and diagnosis.

- *Use appropriate annotations from the Java Bean Validation API for validating patient details attributes of this class. (Refer below admitPatient() method of PatientAPI class for validation rules).*

3. **Validator Class**

com.infy.hospitalmanagement.validator.PatientValidator.java

Implement this class according to the class diagram and the description given below:



Method Description:

1. **void validatePatient(PatientDTO patientDTO)**

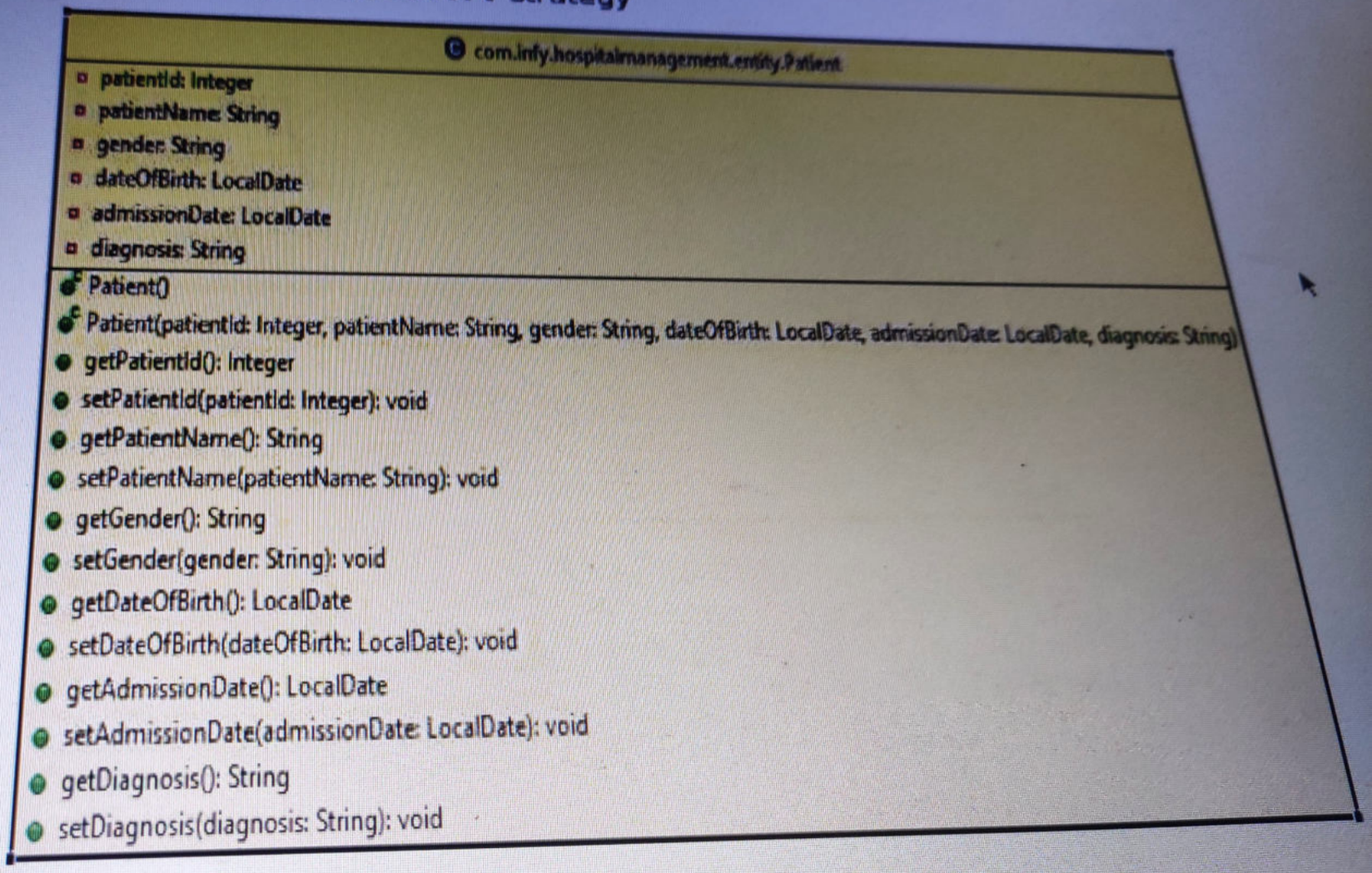
This method receives an object of PatientDTO and calls the respective methods to validate the v
and throws an object of PatientAdmissionException with the message, based on the violation, a
below,

Violation for	Message

1. **Entity Classes**

   com.infy.hospitalmanagement.entity.Patient.java

   - This class has been implemented for you. You have to map this class to the **patient** table with **patientId** as the **primary key**
   - Generate the **patientId** using the **IDENTITY** strategy



File Name	Status
application.properties	Implemented
com.infy.hospitalmanagement.HospitalManagementApplication.java	
com.infy.hospitalmanagement.exception.PatientAdmissionException.java	
com.infy.hospitalmanagement.service.PatientService.java	
com.infy.hospitalmanagement.utility.ErrorInfo.java	
log4j2.properties	
HospitalManagement_TableScripts.sql	
ValidationMessages.properties	
com.infy.hospitalmanagement.dto.PatientDTO.java	To be modified
com.infy.hospitalmanagement.entity.Patient.java	
com.infy.hospitalmanagement.utility.ExceptionControllerAdvice.java	
com.infy.hospitalmanagement.api.PatientAPI.java	To be implemented
com.infy.hospitalmanagement.repository.PatientRepository.java	
com.infy.hospitalmanagement.service.PatientServiceImpl.java	
com.infy.hospitalmanagement.validator.PatientValidator.java	

Infosys Certified Spring Associate Hands-On Assessment

Problem Statement:

In this assessment, you are going to build an **Hospital Management** application which can be used to admit a patient and view all the patients of a given diagnosis. You will have to implement the following functionalities in the application,

1. Admit a new patient
2. Get all patients from a diagnosis

Project Structure provided:

File Name	Status
application.properties	
com.infy.hospitalmanagement.HospitalManagementApplication.java	
com.infy.hospitalmanagement.exception.PatientAdmissionException.java	
com.infy.hospitalmanagement.service.PatientService.java	

Assumptions:

1. All the necessary imports and configurations are done
2. Port number used is 8765

What will be the output when a HTTP POST request is made to the above controller with url as http://localhost:8765/infy and below data?

```
{
    "employeeId": 1,
    "emailId": "andrew@infy.com",
    "employeeName": "Andrew Wyatt",
    "designation": "Senior Associate"
}
```


- ○ EmployeeDTO [employeeId=null, emailId=null, name=null, designation=null]
- ○ EmployeeDTO [employeeId=1, emailId=andrew@infy.com, name=null, designation=Senior Associate]
- ○ EmployeeDTO [employeeId=1, emailId= andrew@infy.com, name=Andrew Wyatt, designation= Senior Associate]
- ○ 404 NOT_FOUND error

Reset **Save**

Question 10

Consider the following EmployeeDTO and EmployeeAPI classes:

```
public class EmployeeDTO {
        private Integer employeeId;
        private String emailId;
        private String name;
        private String designation;

        //getters, setters, toString
}

@RestController
@RequestMapping(value = "/infy")
public class EmployeeAPI {
        @PostMapping
        public void addEmployee(EmployeeDTO employeeDTO) {
                System.out.println(employeeDTO);
        }
}
```

Assumptions:

1. All the necessary imports and configurations are done
2. Port number used is 8765

What will be the output when a HTTP POST request is made to the above controller with url as http://localho

	sequence_name
cust_id	

Assumptions:

1. All necessary configurations are done
2. MySQL database is used

What will be value of the primary key generated when a new customer is added?

- 1001
- 1000
- 2000
- 1

Reset Save

Question 9

16% 84%

Consider the below Entity class and tables:

```
@Entity
public class Customer{
        @Id
        @GeneratedValue
private Integer customerId;
        //rest of the code
}
```

CUSTOMER (**customer_id** is primary key):

customer_id	email_id	name	date_of_birth
1999	martin@infy.com	Martin	1993-10-02

HIBERNATE_SEQUENCE(**sequence_name** is primary key):

sequence_name	next_val
cust_id	1000

Assumptions:

```
    private Integer patientId;
    private String patientName;
    private Long patientPhoneNumber;
    // getters and setters
}
```

Assumption:

1. All necessary imports and configurations are done

What will be the value of pat_id in id_gen table after persisting details of three patients in PATIENT table?

- ○ 110
- ○ 117
- ○ 103
- ○ 115

Reset **Save**

Consider the below Patient entity class where TABLE strategy is used to generate primary key values.

```
@Entity
public class Patient {
        @Id
        @TableGenerator(
            name="pkgen",
            table="id_gen",
            pkColumnName="gen_key",
            valueColumnName="gen_value",
            pkColumnValue="pat_id",
            allocationSize=1)
        @GeneratedValue(generator="pkgen",strategy=GenerationType.TABLE)
        private Integer patientId;
        private String patientName;
        private Long patientPhoneNumber;
        // getters and setters
}
```

Assumption:

1. All necessary imports and configurations are done

What will be the value of pat_id in id_gen table after persisting details of three patients in PATIENT table?

[1 Mark]

Question 8

Consider the following PATIENT table and id_gen table as given below:

PATIENT

patient_id	name	contact_number
51001	Peter	8907645284

id_gen

gen_key	gen_value
pat_id	100

Consider the below Patient entity class where TABLE strategy is used to generate primary key values.

```
@Entity
public class Patient {
```

Focus Area 2

1 2 3

4 5 6

8 9

Before advice is called
In display method
After advice is called

Before advice is called
In display method

In display method
After advice is called

Before advice is called
After advice is called
In display method

Reset Save

Focus Area 2

1 2 3

5 6

8 9

Assumption:

1. All necessary imports and configurations are done.

What is the output when the below code is executed?

```
package com.infy;
@SpringBootApplication
public class SpringBootApplication implements CommandLineRunner {

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

	public void run(String... args) throws Exception {
		display();
	}

	public void display() {
		System.out.println("In display method");
	}
}
```


Before advice is called

[1 Mark]

Question 7

Consider the following LoggingAspect class:

```
package com.infy.utility;
@Component
@Aspect
public class LoggingAspect {

        @Before("execution(* com.infy.SpringBootApplication.*(..))")
        public void before(){
                System.out.println("Before advice is called");
        }

        @After("execution(* com.infy.*Application.*(..))")
        public void after(){
                System.out.println("After advice is called");
        }
}
```

Assumption:

1. All necessary imports and configurations are done.

What is the output when the below code is executed?

What should be inserted at Line 1 so that it creates a correct named query for order using following repository?

```
public interface OrderRepository extends CrudRepository<Order, Integer> {
    String findNameByCustomerEmailId(@Param("emailId") String emailId);
1
}
```

Assumption:

1. All necessary configurations and imports are done

- `@NamedQuery(name="Order.findNameByCustomerEmailId", query="SELECT o.customerName FROM Order o WHERE p.customerEmailId = :emailId")`
- `@NamedQuery(name="Order.findNameByCustomerEmailId", query="SELECT o.customername FROM Order o WHERE p.emailId = :emailId")`
- `@Query(name="Order.findNameByCustomerEmailId", query="SELECT o.customername FROM Order o WHERE p.customerEmailId = :emailId")`
- `@Query(name="Order.findNameByEmailId", query="SELECT o.customername FROM Order o WHERE p.customerEmailId = :emailId")`

Reset | Save

Focus Area 2

1 2 3 4 5 6 8 9

[2 Marks]

Question 6

43%

Consider the following entity class:

```
@Entity
//Line 1
public class Order {
        @Id
        private Integer orderId;
        private String customerName;
        private String customerEmailId;
    //getter and setter methods
}
```

What should be inserted at Line 1 so that it creates a correct named query for Order using following repository?

```
public interface OrderRepository extends CrudRepository<Order, Integer> {
        String findNameByCustomerEmailId(@Param("emailId") String emailId);
}
```

Assumption:

1. All necessary configurations and imports are done

```
        SpringApplication.run(DemoApplication.class, args);
        System.out.println("I am a Spring Boot Application");
    }
    @Override
    public void run(String... args) throws Exception {
        System.out.println(environment.getProperty("welcome")); //Line1
        System.out.println(environment.getProperty("hello"));
    }
}
```


- Welcome to Spring Boot
  Hello World
  I am a Spring Boot Application
- Welcome to DemoApplication
  Hello World
  I am a Spring Boot Application
- I am a Spring Boot Application
  Welcome to DemoApplication
  Hello World
- Exception generated at Line 1 due to ambiguity in the key “welcome”

[2 Marks]

Consider the following properties files present in classpath of the Spring Boot application:

```
application.properties
welcome=Welcome to Spring Boot

messages.properties
welcome=Welcome to DemoApplication
hello=Hello World
```

Assumption:

1. All necessary imports and configurations are done

What will be the output when following code is executed?

```
@SpringBootApplication
@PropertySource("classpath:messages.properties")
public class DemoApplication implements CommandLineRunner {
        @Autowired
        Environment environment;
        public static void main(String[] args) {
                SpringApplication.run(DemoApplication.class, args);
                System.out.println("I am a Spring Boot Application");
        }
```

Question 4

Consider the following class:

```
// Line1
public class FooExceptionHandler {
    // Line2
    public ResponseEntity<ErrorInfo> fromBookingException(BookingException exception) {
        //rest of the code
    }
}
```

Which annotations can be inserted at Line1 and Line2 so that ExceptionHandler class can handle the appropriate exceptions thrown from the application?
Note: Consider all necessary imports are added.

- Line1 - @RestControllerAdvice
  Line2 - @ExceptionHandler(FooBookingException.class)
- Line1 - @RestController
  Line2 - @ExceptionHandler()
- Line1 - @RestControllerAdvice
  Line2 - @ExceptionHandler(FooBookingException.class)
- Line1 - @RestController
  Line2 - @ExceptionHandler(FooBookingException.class)

Question 3

Consider the following controller class:

```
@RestController
@RequestMapping(value="/api")
public class ProductAPI{
   @PutMapping(value = "/products")
   public String updateDetails(@RequestParam Integer id, @RequestParam String name) {
   //rest of the code
   }
}
```

Which of the following HTTP PUT requests are mapped to above controller? Note: Consider all necessary imports are added.

- http://localhost:8080/api/products?id=1001?name=Laptop
- http://localhost:8080/api/products/1001/Laptop
- http://localhost:8080/api/products?id=1001&name=Laptop
- http://localhost:8080/api/products/id=1001/name=Laptop

Reset Save

Assumptions:

1. All necessary imports and configurations are done
2. restTemplate is a reference of RestTemplate class
3. 8765 is the port number used

Which of the following code snippet can be used to hit the getUsers() method in above controller to get the below response?

```
[{userId=101, userName=Nick Eriksen, contactNumber=8945761 32}]
```

- String url = "http://localhost:8765/users/users";
  List<UserDTO> userDTOs = restTemplate.getForObject(url, List<UserDTO>.class);
- String url = "http://localhost:8765/users/users";
  List<UserDTO> userDTOs = restTemplate.getForObject(url, List.class);
- String url = "http://localhost:8765/users";
  List<UserDTO> userDTOs = restTemplate.getForObject(url, List.class);
- String url = "http://localhost:8765/users/users";
  UserDTO userDTOs = restTemplate.getForObject(url, UserDTO.class);

Reset Save

Question 2

Consider the following DTO and REST API classes:

```
public class UserDTO {
        private Integer userId;
        private String userName;
        private String contactNumber;
        //parameterized constructor
        //getters, setters, toString method
}

@RestController
@RequestMapping("/users")
public class UserAPI {
        @GetMapping("/users")
        public ResponseEntity<List<UserDTO>> getUsers(){
                List<UserDTO> userDTOs = new ArrayList<UserDTO>();
                UserDTO userDTO = new UserDTO(101, "Nick Eriksen", "894576132");
                userDTOs.add(userDTO);
                return new ResponseEntity<>(userDTOs, HttpStatus.OK);
        }
}
```

Assumptions:

○ **BOOK:**

book_id	book_name	auth_id
2001	Right Thing	1001
2002	Wonderland	NULL
2003	Potter Bloom	NULL
2004	Key To Success	1002

AUTHOR:

author_id	name
1001	James
1002	Tom

○ **BOOK:**

book_id	book_name	auth_id
2001	Right Thing	1001
2002	Wonderland	1002
2003	Potter Bloom	1002

○ Tables will not be updated and SQLIntegrityConstraintViolationException exception will be thrown

Reset **Save**

```
List<Book> list = new ArrayList<>();
list.add(book);
author.setBooks(list);
authorRepository.save(author);
```


AUTHOR:

author_id	name
1001	James
1002	Tom

BOOK:

book_id	book_name	auth_id
2001	Right Thing	1001
2002	Wonderland	1002
2003	Potter Bloom	1002
2004	Key To Success	1002

AUTHOR:

author_id	name
1001	James
1002	Tom

BOOK:

```
@Entity
public class Author{
        @Id
        private Integer authorId;
    private String name;
    @OneToMany(cascade=CascadeType.ALL)
        @JoinColumn(name="auth_id")
    private List<Book> books;
    //getters and setters
}
```

Assumptions:

1. All necessary configurations and imports are done
2. authorRepository is a valid reference of AuthorRepository which extends CrudRepository interface
3. Transaction is managed by Spring

What will be the state of both the tables after the below code is executed?

```
Author author= authorRepository.findById(1002);
Book book = new Book ();
book.setBookId(2004);
book.setBookName("Key To Success");
List<Book> list = new ArrayList<>();
list.add(book);
author.setBooks(list);
authorRepository.save(author);
```

Question 1

[2 Marks]

Consider the following tables and corresponding entity classes:

29%

AUTHOR (author_id is primary key):

author_id	name
1001	James
1002	Tom

BOOK (book_id is primary key and auth_id is foreign key referring to author_id of AUTHOR table):

book_id	book_name	auth_id
2001	Right Thing	1001
2002	Wonderland	1002
2003	Potter Bloom	1002

```
@Entity
public class Book{
        @Id
        private Integer bookId
        private String bookName;
   //getters and setters
}
```

```
public class Tester2 {
    public static void main(String a[]) {
        List<Customer> list = new ArrayList<Customer>();
        Customer c1 = new Customer(9001, "Patrick");
        Customer c2 = new Customer(9002, "Mary");
        Customer c3 = new Customer(9003, "Anupam");
        list.add(c1);
        list.add(c2);
        list.add(c3);
        Stream<Customer> stream = list.stream().filter(cust -> cust.getCustomerName().length() >= 5);
        stream.forEach((customer) -> System.out.print(customer.getCustomerId() + "  "));
    }
}
```


- ○ 9001
- ○ 9001 9003
- ○ 9001 9002 9003
- ○ 9002 9003

Reset **Save**

Question 5

[2 Marks]

What will be the output of the below code? Assume **Customer class** exists as below

```
class Customer{
    int customerId;
    String customerName;
    Customer(int cId,String name){
        this.customerId=cId;
        this.customerName=name;
    }
    //Getters and setters
}
public class Tester2 {
    public static void main(String a[]) {
        List<Customer> list = new ArrayList<Customer>();
        Customer c1 = new Customer(9001, "Patrick");
        Customer c2 = new Customer(9002, "Mary");
        Customer c3 = new Customer(9003, "Anupam");
        list.add(c1);
        list.add(c2);
        list.add(c3);
        Stream<Customer> stream = list.stream().filter(cust -> cust.getCustomerName().length() >= 5);
        stream.forEach((customer) -> System.out.print(customer.getCustomerId() + "  "));
    }
}
```

```
            return true;
        return false;
    }
}
```

Following is the test case written for testing the above validatePassword() method:

```
public class ValidatorClassTest{
    @Test
    public void validatePasswordTest(){
        // Line1
    }
}
```

Which of the following assertion statements would be appropriate at 'Line1' for valid test cases. Choose TWO correct options.

- [ ] Assertions.assertEquals(Boolean.TRUE,Validator.validatePassword("Abc123"));
- [ ] Assertions.assertTrue(Validator.validatePassword("Abc"));
- [ ] Assertions.assertEquals(TRUE,Validator.validatePassword("Abc123"));
- [ ] Assertions.assertTrue(Validator.validatePassword("Abc1"));

Reset **Save**

Question 4

Given is a Validator class to test a password:

```
public class Validator{
    public static Boolean validatePassword(String password){
        if(password.matches(".*[A-Z].*")&&password.matches(".*[0-9].*"))
            return true;
        return false;
    }
}
```

Following is the test case written for testing the above validatePassword() method:

```
public class ValidatorClassTest{
    @Test
    public void validatePasswordTest(){
        // Line1
    }
}
```

Which of the following assertion statements would be appropriate at 'Line1' for valid test cases. Choose TWO correct options.

- [ ] Assertions.assertEquals(Boolean.TRUE,Validator.validatePassword("Abc123"));

Question 4

Given is a Validator class to test a password:

```
public class Validator{
    public static Boolean validatePassword(String password){
        if(password.matches(".*[A-Z].*")&&password.matches(".*[0-9].*"))
            return true;
        return false;
    }
}
```

Following is the test case written for testing the above validatePassword() method:

```
public class ValidatorClassTest{
    @Test
    public void validatePasswordTest(){
        // Line1
    }
}
```

Which of the following assertion statements would be appropriate at 'Line1' for valid test cases. Choose TWO correct opti

- [ ] Assertions.assertEquals(Boolean.TRUE,Validator.validatePassword("Abc123"));

Area 1

3

```
public static void main(String[] args) {
        Map<String, String> nameMap = new HashMap<String, String>();

        nameMap.put("A", "Andre");
        nameMap.put("B", "Bob");
        nameMap.put(null, null);
        nameMap.put("C", "Catlyn");
        nameMap.put(new String("A"), "Avan");
        System.out.println(nameMap);
}
```

Assumption:

1. All necessary configuration and imports are done

- ○ {null=null, A=Avan, B=Bob, C=Catlyn}
- ○ {A=Avan, B=Bob, C=Catlyn}
- ○ The code will throw run time exception as the same key cannot be used for 2 values.
- ○ The code will throw NullPointerException as the HashMap cannot have null as key value.

Reset **Save**

cus Area 1

2 3

5

Question 3

What will be the output when following code is executed?

```
public static void main(String[] args) {
        Map<String, String> nameMap = new HashMap<String, String>();

        nameMap.put("A", "Andre");
        nameMap.put("B", "Bob");
        nameMap.put(null, null);
        nameMap.put("C", "Catlyn");
        nameMap.put(new String("A"), "Avan");
        System.out.println(nameMap);
}
```

Assumption:

1. All necessary configuration and imports are done

- {null=null, A=Avan, B=Bob, C=Catlyn}
- {A=Avan, B=Bob, C=Catlyn}
- The code will throw run time exception as the same key cannot be used for 2 values.

ocus Area 1

2 3

5

What is the output of the following code?

```
class Customer {
    private String custName;
    Customer(String fName) {
        this.custName = fName;
    }
    public String getCustName() {
        return custName;
    }
    public String toString() {
        return custName;
    }
}

public class Tester1 {
    public static void main(String[] args) {
        Customer cust1 = new Customer("John");
        Customer cust2 = new Customer("Joe");
        Customer cust3 = new Customer("Sam");
        List<Customer> list = new ArrayList<Customer>();
        list.add(cust1);
        list.add(cust2);
        list.add(cust3);
        System.out.print("Before: " + list);
        for (int j = 0; j < list.size(); j++) {
            Customer e = list.get(j);
```


88%

2 3

5

```
        List<Customer> list = new ArrayList<Customer>();
        list.add(cust1);
        list.add(cust2);
        list.add(cust3);
        System.out.print("Before: " + list);
        for (int j = 0; j < list.size(); j++) {
            Customer e = list.get(j);
            if (e.getCustName().equals(cust1.getCustName()))
                list.remove(j+1);
        }
        System.out.println(" After: " + list);
    }
}
```


- ○ Before: [John, Joe, Sam] After: [John, Sam]
- ○ Before: [John, Joe] After: [John]
- ○ Before: [John, Sam] After: [Sam]
- ○ Before: [John, Sam] After: [Joe]

Reset **Save**

s Area 1

2 3

5

What is the output of the following code?

```
class Customer {
    private String custName;
    Customer(String fName) {
        this.custName = fName;
    }
    public String getCustName() {
        return custName;
    }
    public String toString() {
        return custName;
    }
}

public class Tester1 {
    public static void main(String[] args) {
        Customer cust1 = new Customer("John");
        Customer cust2 = new Customer("Joe");
        Customer cust3 = new Customer("Sam");
        List<Customer> list = new ArrayList<Customer>();
        list.add(cust1);
        list.add(cust2);
        list.add(cust3);
        System.out.print("Before: " + list);
        for (int j = 0; j < list.size(); j++) {
            Customer e = list.get(j);
```


88%

Question 1

[1 Mark]

Assuming the below code executes in the main method. What will be the output of the following code?

```
LocalDate localDate = LocalDate.of(2012, 12, 31);
localDate=localDate.plus(2, ChronoUnit.MONTHS);
System.out.println(localDate);
```

- 2013-03-02
- 2013-02-28
- 2013-03-01
- Error: Cannot add month

Reset Save