*Term Project*

*Instructor: Faisal Khan*

|  |  |  |
| --- | --- | --- |
| *Name* | *Student ID* | *Group#* |
| *Rupesh Shrestha* | *C0908441* |  |
| *Prabin Ghimire* | *C0903145* |  |
| *Bijay Sharma* | *C0913527* |  |
| *Milan Chaulagain* | *C0913649* |  |
| *Rupak Paudel* | *C0903770* |  |

1-Use Documentation Comments where needed and use Javadoc to create HTML files for documentation.

2-In your project src folder add this word document with all your group members names and paste the screen shots of your output after each question.

2-Right Click on your Project Folder in Eclipse, Copy Option is available. Create a copy of the Project and upload the zipped folder using Moodle.

Create java desktop application with CRUD functionality using Java Fx Api or Scanner Class. You should create an application for a Doctor’s Office. You have to maintain the record of patients using this application. You will be using a Patient class with the fields like Patientid,Name,Contact,Region,Disease,VisitDate .You have to follow these steps for this application.

a-Create a database named PatientRecords. 10

create database patientrecords;

A screenshot of a computer

Description automatically generated

b-Create table named Patients in the database. 10

CREATE TABLE `patientrecords`.`patients` (

`patientId` INT NOT NULL,

`name` VARCHAR(60) NULL,

`contact` VARCHAR(60) NULL,

`reason` VARCHAR(200) NULL,

`disease` VARCHAR(60) NULL,

`visitDate` DATE NULL,

PRIMARY KEY (`patientId`));

A screenshot of a computer

Description automatically generated

c-Insert 5 or 6 Patient Records in thedatabase 10

INSERT INTO `patientrecords`.`patients` (`patientId`, `name`, `contact`, `reason`, `disease`, `visitDate`)

VALUES

(1, 'Ram Shrestha', '987-654-3210', 'Routine checkup', 'None', '2024-03-29'),

(2, 'Sita Rana', '987-654-3211', 'Headache', 'Migraine', '2024-03-30'),

(3, 'Hari Thapa', '987-654-3212', 'Fever', 'Flu', '2024-03-31'),

(4, 'Parvati Gurung', '987-654-3213', 'Cough', 'Bronchitis', '2024-04-01'),

(5, 'Govinda Pandey', '987-654-3214', 'Abdominal pain', 'Appendicitis', '2024-04-02');

A screenshot of a computer

Description automatically generated

d-Add Jar file for your database to the lib Folder 10

A screenshot of a computer

Description automatically generated

e-Create one Java class as a Model class. 10

Patient.java

**package** model;

**import** java.util.Date;

/\*\*

\* The Patient class represents a patient record in the system.

\*/

**public** **class** Patient {

**private** **int** patientId;

**private** String name;

**private** String contact;

**private** String reason;

**private** String disease;

**private** Date visitDate;

/\*\*

\* Gets the patient ID.

\*

\* **@return** The patient ID

\*/

**public** **int** getPatientId() {

**return** patientId;

}

/\*\*

\* Sets the patient ID.

\*

\* **@param** patientId The patient ID to set

\*/

**public** **void** setPatientId(**int** patientId) {

**this**.patientId = patientId;

}

/\*\*

\* Gets the patient's name.

\*

\* **@return** The patient's name

\*/

**public** String getName() {

**return** name;

}

/\*\*

\* Sets the patient's name.

\*

\* **@param** name The name to set

\*/

**public** **void** setName(String name) {

**this**.name = name;

}

/\*\*

\* Gets the patient's contact information.

\*

\* **@return** The patient's contact information

\*/

**public** String getContact() {

**return** contact;

}

/\*\*

\* Sets the patient's contact information.

\*

\* **@param** contact The contact information to set

\*/

**public** **void** setContact(String contact) {

**this**.contact = contact;

}

/\*\*

\* Gets the reason for the patient's visit.

\*

\* **@return** The reason for the visit

\*/

**public** String getReason() {

**return** reason;

}

/\*\*

\* Sets the reason for the patient's visit.

\*

\* **@param** reason The reason for the visit to set

\*/

**public** **void** setReason(String reason) {

**this**.reason = reason;

}

/\*\*

\* Gets the disease the patient is diagnosed with.

\*

\* **@return** The disease the patient is diagnosed with

\*/

**public** String getDisease() {

**return** disease;

}

/\*\*

\* Sets the disease the patient is diagnosed with.

\*

\* **@param** disease The disease to set

\*/

**public** **void** setDisease(String disease) {

**this**.disease = disease;

}

/\*\*

\* Gets the date of the patient's visit.

\*

\* **@return** The date of the visit

\*/

**public** Date getVisitDate() {

**return** visitDate;

}

/\*\*

\* Sets the date of the patient's visit.

\*

\* **@param** visitDate The date of the visit to set

\*/

**public** **void** setVisitDate(Date visitDate) {

**this**.visitDate = visitDate;

}

/\*\*

\* Constructs a new Patient object.

\*

\* **@param** patientId The patient ID

\* **@param** name The patient's name

\* **@param** contact The patient's contact information

\* **@param** reason The reason for the visit

\* **@param** disease The disease diagnosed

\* **@param** visitDate The date of the visit

\*/

**public** Patient(**int** patientId, String name, String contact, String reason, String disease, Date visitDate) {

**this**.patientId = patientId;

**this**.name = name;

**this**.contact = contact;

**this**.reason = reason;

**this**.disease = disease;

**this**.visitDate = visitDate;

}

/\*\*

\* Returns a string representation of the Patient object.

\*

\* **@return** A string representation of the Patient object

\*/

@Override

**public** String toString() {

**return** "Patient [patientId=" + patientId + ", name=" + name + ", contact=" + contact + ", reason=" + reason

+ ", disease=" + disease + ", visitDate=" + visitDate + "]\n";

}

}

f-Create a separate DAO class for database connectivity.10

g-Create separate methods for connection and all crud operations. 20

PatientDAO.java

**package** dao;

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**import** java.sql.PreparedStatement;

**import** java.sql.ResultSet;

**import** java.sql.SQLException;

**import** java.sql.Statement;

**import** java.util.ArrayList;

**import** java.util.List;

**import** model.Patient;

/\*\*

\* The PatientDAO class handles database operations related to the Patient model.

\*/

**public** **class** PatientDAO {

/\*\*

\* Establishes a database connection.

\*

\* **@return** Connection object representing the database connection

\*/

**public** Connection setupConnection() {

Connection con = **null**;

String url = "jdbc:mysql://localhost:3306/patientrecords?autoReconnect=true&useSSL=false";

String user = "root";

String pwd = "Rupesh@123";

**try** {

Class.*forName*("com.mysql.jdbc.Driver");

con = DriverManager.*getConnection*(url, user, pwd);

System.***out***.println("Connection Successful!");

} **catch** (ClassNotFoundException | SQLException e) {

e.printStackTrace();

}

**return** con;

}

/\*\*

\* Retrieves all patient records from the database.

\*

\* **@return** List of Patient objects representing the records retrieved

\*/

**public** List<Patient> displayRecs() {

List<Patient> patients = **new** ArrayList<>();

Connection con = setupConnection();

String sql = "select \* from patients";

**try** {

Statement stmt = con.createStatement();

ResultSet rs = stmt.executeQuery(sql);

**while** (rs.next()) {

Patient patient = **new** Patient(rs.getInt("patientId"), rs.getString("name"),

rs.getString("contact"), rs.getString("reason"), rs.getString("disease"),

rs.getDate("visitDate"));

patients.add(patient);

}

System.***out***.println(patients);

} **catch** (SQLException e) {

e.printStackTrace();

}

**return** patients;

}

/\*\*

\* Inserts a new patient record into the database.

\*

\* **@param** patient Patient object representing the record to be inserted

\*/

**public** **void** insertRec(Patient patient) {

Connection con = setupConnection();

String sql = "insert into patients (patientId, name, contact, reason, disease, visitDate) values (?, ?, ?, ?, ?, ?)";

**try** {

PreparedStatement pstmt = con.prepareStatement(sql);

pstmt.setInt(1, patient.getPatientId());

pstmt.setString(2, patient.getName());

pstmt.setString(3, patient.getContact());

pstmt.setString(4, patient.getReason());

pstmt.setString(5, patient.getDisease());

pstmt.setDate(6, **new** java.sql.Date(patient.getVisitDate().getTime()));

**int** status = pstmt.executeUpdate();

**if** (status > 0) {

System.***out***.println("Record inserted successfully !!!");

displayRecs();

} **else** {

System.***out***.println("Try again it happens");

}

} **catch** (SQLException e) {

e.printStackTrace();

}

}

/\*\*

\* Retrieves a patient record by ID from the database.

\*

\* **@param** id\_to\_fetch\_rec The ID of the patient record to retrieve

\* **@return** Patient object representing the record retrieved

\*/

**public** Patient getRecById(**int** id\_to\_fetch\_rec) {

Patient patient\_to\_fetch\_byId = **null**;

Connection con = setupConnection();

String sql = "select \* from patients where patientId = ?";

**try** {

PreparedStatement pstmt = con.prepareStatement(sql);

pstmt.setInt(1, id\_to\_fetch\_rec);

ResultSet rs = pstmt.executeQuery();

**if** (rs.next()) {

patient\_to\_fetch\_byId = **new** Patient(rs.getInt("patientId"), rs.getString("name"),

rs.getString("contact"), rs.getString("reason"), rs.getString("disease"),

rs.getDate("visitDate"));

System.***out***.println(patient\_to\_fetch\_byId);

}

} **catch** (SQLException e) {

e.printStackTrace();

}

**return** patient\_to\_fetch\_byId;

}

/\*\*

\* Deletes a patient record by ID from the database.

\*

\* **@param** id\_to\_delete The ID of the patient record to delete

\*/

**public** **void** deleteRecById(**int** id\_to\_delete) {

Connection con = setupConnection();

String sql = "delete from patients where patientId=?";

**try** {

PreparedStatement pstmt = con.prepareStatement(sql);

pstmt.setInt(1, id\_to\_delete);

**int** status = pstmt.executeUpdate();

**if** (status > 0) {

System.***out***.println("Record deleted Successfully");

displayRecs();

} **else** {

System.***out***.println("Try again!");

}

} **catch** (SQLException e) {

e.printStackTrace();

}

}

/\*\*

\* Updates a patient record in the database.

\*

\* **@param** curr\_id The ID of the patient record to update

\* **@param** updatedPatient Patient object representing the updated record

\*/

**public** **void** updateRecById(**int** curr\_id, Patient updatedPatient) {

Connection con = setupConnection();

String sql = "update patients set patientId = ?, name= ?, contact= ?, reason= ?, disease= ?, visitDate = ? where patientId = ?";

**try** {

PreparedStatement pstmt = con.prepareStatement(sql);

pstmt.setInt(1, updatedPatient.getPatientId());

pstmt.setString(2, updatedPatient.getName());

pstmt.setString(3, updatedPatient.getContact());

pstmt.setString(4, updatedPatient.getReason());

pstmt.setString(5, updatedPatient.getDisease());

pstmt.setDate(6, **new** java.sql.Date(updatedPatient.getVisitDate().getTime()));

pstmt.setInt(7, curr\_id);

**int** status = pstmt.executeUpdate();

**if** (status > 0) {

System.***out***.println("Record updated Successfully");

displayRecs();

} **else** {

System.***out***.println("Try again!");

}

} **catch** (SQLException e) {

e.printStackTrace();

}

}

}

h-Create a third separate java class and call these methods in this class to display data on console or Display using Java Fx Api . 20

Controller.java

**package** controller;

**import** java.util.Date;

**import** java.util.Scanner;

**import** dao.PatientDAO;

**import** model.Patient;

/\*\*

\* The Controller class manages the interaction between the user interface and the data access objects.

\*/

**public** **class** Controller {

// Data Access Object for patient records

PatientDAO dao = **new** PatientDAO();

// Scanner object for user input

Scanner sc = **new** Scanner(System.***in***);

/\*\*

\* Display all patient records.

\*/

**public** **void** displayAll() {

System.***out***.println();

dao.displayRecs();

}

/\*\*

\* Insert a new patient record.

\*/

**public** **void** insertRec() {

System.***out***.println("Welcome to Insert Record Option in our Application");

dao.displayRecs();

// Get user input for patient properties

System.***out***.println("Please Enter the Id of new Patient:");

**int** patientId = sc.nextInt();

sc.nextLine(); // Consume the newline character

System.***out***.println("Please Enter the name of new Patient:");

String name = sc.nextLine();

System.***out***.println("Please Enter the contact of new Patient:");

String contact = sc.nextLine();

System.***out***.println("Please Enter the reason of visit:");

String reason = sc.nextLine();

System.***out***.println("Please Enter the Disease name:");

String disease = sc.nextLine();

// Create a patient object

Patient patientToAdd = **new** Patient(patientId, name, contact, reason, disease, **new** Date());

// Insert the patient record

dao.insertRec(patientToAdd);

}

/\*\*

\* Retrieve a single patient record by patient id.

\*/

**public** **void** getASinglePatient() {

System.***out***.println("Please Enter the Id of the patient:");

**int** patientId = sc.nextInt();

sc.nextLine(); // Consume the newline character

dao.getRecById(patientId);

System.***out***.println();

}

/\*\*

\* Delete a patient record by patient id.

\*/

**public** **void** deleteAPatientRecord() {

System.***out***.println("Enter the id of the patient to be deleted:");

**int** patientId = sc.nextInt();

sc.nextLine(); // Consume the newline character

dao.deleteRecById(patientId);

System.***out***.println();

}

/\*\*

\* Update a patient record by patient id.

\*/

**public** **void** updateAPatientRecord() {

System.***out***.println("\nEnter the id of the patient to be updated:");

**int** patientId = sc.nextInt();

sc.nextLine(); // Consume the newline character

dao.getRecById(patientId);

// Get updated data for patient properties

System.***out***.println("Please Enter new Id of the Patient:");

**int** newId = sc.nextInt();

sc.nextLine(); // Consume the newline character

System.***out***.println("Please Enter the new name of the Patient:");

String name = sc.nextLine();

System.***out***.println("Please Enter the new contact of the Patient:");

String contact = sc.nextLine();

System.***out***.println("Please Enter the current reason of visit:");

String reason = sc.nextLine();

System.***out***.println("Please Enter the Disease name:");

String disease = sc.nextLine();

// Create a patient object with updated data

Patient updatedPatient = **new** Patient(newId, name, contact, reason, disease, **new** Date());

// Update the patient record

dao.updateRecById(patientId, updatedPatient);

}

}

Usermenu.java

**package** view;

**import** java.util.Scanner;

**import** controller.Controller;

/\*\*

\* The userMenu class provides a menu interface for interacting with the Patient Information System.

\*/

**public** **class** userMenu {

/\*\*

\* Main method to start the application.

\*

\* **@param** args Command-line arguments (not used).

\*/

**public** **static** **void** main(String[] args) {

// Create an instance of the Controller class

Controller ctrl = **new** Controller();

// Create a Scanner object for user input

Scanner sc = **new** Scanner(System.***in***);

**int** entry = 0;

// Display welcome message

System.***out***.println("\n\nWelcome to the Patient Information System");

// Display menu and prompt user for input

**do** {

System.***out***.println("-----------------------------------------");

System.***out***.println("Enter 1 to display all Records.");

System.***out***.println("Enter 2 to Insert Record");

System.***out***.println("Enter 3 to get a single patient using patient id!!!");

System.***out***.println("Enter 4 to delete patient by Id !!!");

System.***out***.println("Enter 5 to update a record by Id !!!");

System.***out***.println("Enter 6 to Exit the application !!!");

System.***out***.print("Please Type your Entry: a number from 1 to 6 only: ");

// Read user input

entry = sc.nextInt();

// Perform action based on user input

**switch** (entry) {

**case** 1: {

ctrl.displayAll();

**break**;

}

**case** 2: {

ctrl.insertRec();

**break**;

}

**case** 3: {

ctrl.getASinglePatient();

**break**;

}

**case** 4: {

ctrl.deleteAPatientRecord();

**break**;

}

**case** 5: {

ctrl.updateAPatientRecord();

**break**;

}

**case** 6: {

System.***out***.println("Exiting the application...");

System.*exit*(0);

**break**;

}

**default**: {

System.***out***.println("Invalid entry. Please try again.");

}

}

} **while** (entry != 6);

// Close the Scanner object

sc.close();

}

}

**Output:**

**Displaying all patients records.**

A screenshot of a computer

Description automatically generated

**Inserting a patient’s record in the table**

**A screenshot of a computer

Description automatically generated**

**View patients with id**

**A screenshot of a computer

Description automatically generated**

**Deleting a patient**

**A screenshot of a computer screen

Description automatically generated**

**Updating a patient's record**

**A screenshot of a computer

Description automatically generated**

**Exiting the application**

**A screenshot of a computer

Description automatically generated**

**Using Javadoc to create HTML documents**

**A screenshot of a computer

Description automatically generated**

**Generated html files for documentation inside html\_doc folder**

**A screenshot of a computer

Description automatically generated**

**Controller html documentation**

**A screenshot of a computer

Description automatically generated**

**patientDAO html file**

**A screenshot of a computer

Description automatically generated**

**Patient html file for documentation**

**A screenshot of a computer

Description automatically generated**

**Usermenu html file for documentation**

**A screenshot of a computer

Description automatically generated**