SOURCE CODE

1. AppointmentDao

```
public class AppointmentDao {
public static boolean insert(AppointmentVO obj){

		Connection con=DBService.getConnection();
		PreparedStatement ps=null;

		String sql = "insert into appointment(patient_id, patient_name, doctor_name,
app_date, start_time, "
				+ "end_time, booking_emp_id, status)values(?,?,?,?,?,?,?,?)";

		try{

			ps=con.prepareStatement(sql);
			ps.setInt(1, obj.getPatientId());
			ps.setString(2, obj.getpatientName());
			ps.setString(3, obj.getDoctorName());
			ps.setString(4, obj.getAppDate());
			ps.setString(5, obj.getStartTime());
			ps.setString(6, obj.getEndTime());
			ps.setInt(7, obj.getBookingEmpId());
			ps.setString(8, obj.getStatus());
			ps.executeUpdate();
			return true;
		}catch (Exception e) {
			e.printStackTrace();
		}finally{
			DBService.close(null, null, ps, con);
		}return false;
}

public static boolean delete(int appointmentId) {
	Connection con = DBService.getConnection();
	PreparedStatement ps = null;
	try {
		ps = con.prepareStatement("delete from appointment where appointment_id = ?");
		ps.setInt(1, appointmentId);
		ps.executeUpdate();

		return true;
	} catch (Exception e) {
		e.printStackTrace();
	} finally {
		DBService.close(null, null, ps, con);
	}
	return false;

}
```

```
public static List getAppointmentList() {
	List lst = new ArrayList();

	Connection con=DBService.getConnection();
	PreparedStatement ps = null;
	ResultSet rs = null;

	try {
		ps=con.prepareStatement("select * from appointment");
		rs=ps.executeQuery();

		while(rs.next()){
			int appointmentId = rs.getInt("appointment_id");
			int patientId = rs.getInt("patient_id");
			String patientName = rs.getString("patient_name");
			String doctorName = rs.getString("doctor_name");
			String appDate = rs.getString("app_date");
			String startTime = rs.getString("start_time");
			String endTime = rs.getString("end_time");
			int bookingEmpId = rs.getInt("booking_emp_id");
			String status = rs.getString("status");

			AppointmentVO vo = new AppointmentVO(appointmentId, patientId,
patientName, doctorName, appDate, startTime, endTime, bookingEmpId, status );
			lst.add(vo);

		}
	} catch (Exception e) {
		e.printStackTrace();
	}finally{
		DBService.close(rs, null, ps, con);
	}
	return lst;

}

public static List getAppointmentByDoctor() {
	List lst = new ArrayList();

	Connection con = DBService.getConnection();
	PreparedStatement ps = null;
	ResultSet rs = null;

	try {
		ps = con.prepareStatement("SELECT doctor_name, count(appointment_id)
appointmentCount FROM appointment group by doctor_name");
		rs = ps.executeQuery();

		while (rs.next()) {
			String doctorName = rs.getString("doctor_name");
			int appointmentCount = rs.getInt("appointmentCount");
```

```
                AppointmentVO vo = new AppointmentVO(doctorName,
appointmentCount);
                lst.add(vo);
            }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            DBService.close(null, null, ps, con);
        }
        return lst;
}

}
```

2. AllocateWardDao

```
public class AllocateWardDao {
public static boolean insert(AllocateWardVO obj){

        Connection con=DBService.getConnection();
        PreparedStatement ps=null;

        String sql = "insert into allocateWard(patient_id, patient_name, ward_type, ward_no,
rate, days, booking_emp_id, status)values(?,?,?,?,?,?,?,?)";

        try{

            ps=con.prepareStatement(sql);
            ps.setInt(1, obj.getPatientId());
            ps.setString(2, obj.getPatientName());
            ps.setString(3, obj.getWardType());
            ps.setString(4, obj.getWardNo());
            ps.setString(5, obj.getRate());
            ps.setString(6, obj.getDays());
            ps.setInt(7, obj.getBookingEmpId());
            ps.setString(8, obj.getStatus());

            ps.executeUpdate();
            return true;
        }catch (Exception e) {
            e.printStackTrace();
        }finally{
            DBService.close(null, null, ps, con);
        }return false;

}
public static boolean delete(int allocateWardId) {
    Connection con = DBService.getConnection();
    PreparedStatement ps = null;
    try {
        ps = con.prepareStatement("delete from allocateWard where allocate_ward_id = ?");
```

```
                ps.setInt(1, allocateWardId);
                ps.executeUpdate();

                return true;
        } catch (Exception e) {
                e.printStackTrace();
        } finally {
                DBService.close(null, null, ps, con);
        }
        return false;

}

public static List getAllocateWardList() {
        List lst = new ArrayList();

        Connection con=DBService.getConnection();
        PreparedStatement ps = null;
        ResultSet rs = null;

        try {
                ps=con.prepareStatement("select * from allocateWard");
                rs=ps.executeQuery();

                while(rs.next()){
                        int allocateWardId = rs.getInt("allocate_ward_id");
                        int patientId = rs.getInt("patient_id");
                        String patientName = rs.getString("patient_name");
                        String wardType = rs.getString("ward_type");
                        String wardNo = rs.getString("ward_no");
                        String rate = rs.getString("rate");
                        String days = rs.getString("days");
                        int bookingEmpId = rs.getInt("booking_emp_id");
                        String status = rs.getString("status");

                        AllocateWardVO vo = new AllocateWardVO(allocateWardId, patientId,
patientName, wardType, wardNo, rate, days, bookingEmpId, status );
                        lst.add(vo);

                }
        } catch (Exception e) {
                e.printStackTrace();
        }finally{
                DBService.close(rs, null, ps, con);
        }
        return lst;

}
}
```

3. AssignNurseDao

```
public class AssignNurseDao {
public static boolean insert(AssignNurseVO obj){

		Connection con=DBService.getConnection();
		PreparedStatement ps=null;

		String sql = "insert into assignNurse(patient_id, patient_name, ward_no, nurse_id,
nurse_name, status)values(?,?,?,?,?,?)";

		try{

			ps=con.prepareStatement(sql);
			ps.setInt(1, obj.getPatientId());
			ps.setString(2, obj.getPatientName());
			ps.setString(3, obj.getWardNo());
			ps.setInt(4, obj.getNurseId());
			ps.setString(5, obj.getNurseName());
			ps.setString(6, obj.getStatus());

			ps.executeUpdate();
			return true;
		}catch (Exception e) {
			e.printStackTrace();
		}finally{
			DBService.close(null, null, ps, con);
		}return false;

}
public static boolean delete(int assignNurseId) {
	Connection con = DBService.getConnection();
	PreparedStatement ps = null;
	try {
		ps = con.prepareStatement("delete from assignNurse where assign_nurse_id = ?");
		ps.setInt(1, assignNurseId);
		ps.executeUpdate();

		return true;
	} catch (Exception e) {
		e.printStackTrace();
	} finally {
		DBService.close(null, null, ps, con);
	}
	return false;

}

public static List getAssignNurseList() {
	List lst = new ArrayList();

	Connection con=DBService.getConnection();
```

```
		PreparedStatement ps = null;
		ResultSet rs = null;

		try {
			ps=con.prepareStatement("select * from assignNurse");
			rs=ps.executeQuery();

			while(rs.next()){
				int assignNurseId = rs.getInt("assign_Nurse_id");
				int patientId = rs.getInt("patient_id");
				String patientName = rs.getString("patient_name");
				String wardNo = rs.getString("ward_no");
				int nurseId = rs.getInt("nurse_id");
				String nurseName = rs.getString("nurse_name");
				String status = rs.getString("status");

				AssignNurseVO vo = new AssignNurseVO(assignNurseId, patientId,
patientName, wardNo, nurseId, nurseName, status );
				lst.add(vo);

			}
		} catch (Exception e) {
			e.printStackTrace();
		}finally{
			DBService.close(rs, null, ps, con);
		}
		return lst;

}

}
```

4. GenerateBillDao

```
public class GenerateBillDao {
public static boolean insert(GenerateBillVO obj){

			Connection con=DBService.getConnection();
			PreparedStatement ps=null;

			String sql = "insert into bill(patient_id, patient_name, bill_type, treatment_type,
treatment_amount, "
					+ "ward_type, ward_rate, days)values(?,?,?,?,?,?,?,?)";

			try{

				ps=con.prepareStatement(sql);
				ps.setInt(1, obj.getPatientId());
				ps.setString(2, obj.getPatientName());
				ps.setString(3, obj.getBillType());
				ps.setString(4, obj.getTreatmentType());
				ps.setString(5, obj.getTreatmentAmount());
```

```
                ps.setString(6, obj.getWardType());
                ps.setString(7, obj.getWardRate());
                ps.setInt(8, obj.getDays());
                ps.executeUpdate();
                return true;
            }catch (Exception e) {
                e.printStackTrace();
            }finally{
                DBService.close(null, null, ps, con);
            }return false;
    }

    public static boolean delete(int billId) {
        Connection con = DBService.getConnection();
        PreparedStatement ps = null;
        try {
            ps = con.prepareStatement("delete from bill where bill_id = ?");
            ps.setInt(1, billId);
            ps.executeUpdate();

            return true;
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            DBService.close(null, null, ps, con);
        }
        return false;

    }

    public static List getAllBillList() {
        List lst = new ArrayList();

        Connection con=DBService.getConnection();
        PreparedStatement ps = null;
        ResultSet rs = null;

        try {
            ps=con.prepareStatement("select * from bill");
            rs=ps.executeQuery();

            while(rs.next()){
                int billId = rs.getInt("bill_id");
                int patientId = rs.getInt("patient_id");
                String patientName = rs.getString("patient_name");
                String billType = rs.getString("bill_type");
                String treatmentType = rs.getString("treatment_type");
                String treatmentAmount = rs.getString("treatment_amount");
                String wardType = rs.getString("ward_type");
                String wardRate = rs.getString("ward_rate");
                int days = rs.getInt("days");
```

```
                GenerateBillVO vo = new GenerateBillVO(billId, patientId, patientName,
billType, treatmentType,
                                treatmentAmount, wardType, wardRate, days);
                lst.add(vo);

            }
        } catch (Exception e) {
            e.printStackTrace();
        }finally{
            DBService.close(rs, null, ps, con);
        }
        return lst;

}

}
```

5. MaintainDoctorDao

```
public class MaintainDoctorDao {
public static boolean insert(MaintainDoctorVO obj){

            Connection con=DBService.getConnection();
            PreparedStatement ps=null;

            String sql = "insert into maintainDoctor(doctor_name, speciality, location, days,
visit_time)values(?,?,?,?,?)";

            try{

                ps=con.prepareStatement(sql);
                ps.setString(1, obj.getDoctorName());
                ps.setString(2, obj.getSpeciality());
                ps.setString(3, obj.getLocation());
                ps.setString(4, obj.getDays());
                ps.setString(5, obj.getVisitTime());
                ps.executeUpdate();
                return true;
            }catch (Exception e) {
                e.printStackTrace();
            }finally{
                DBService.close(null, null, ps, con);
            }return false;
}

public static boolean delete(int doctorId) {
        Connection con = DBService.getConnection();
        PreparedStatement ps = null;
        try {
            ps = con.prepareStatement("delete from maintainDoctor where doctor_id = ?");
            ps.setInt(1, doctorId);
            ps.executeUpdate();
```

```
                return true;
        } catch (Exception e) {
                e.printStackTrace();
        } finally {
                DBService.close(null, null, ps, con);
        }
        return false;

}

public static List getDoctorTypeList() {
        List lst = new ArrayList();

        Connection con=DBService.getConnection();
        PreparedStatement ps = null;
        ResultSet rs = null;

        try {
                ps=con.prepareStatement("select * from maintainDoctor");
                rs=ps.executeQuery();

                while(rs.next()){
                        int id = rs.getInt("doctor_id");
                        String doctorName = rs.getString("doctor_name");
                        String speciality = rs.getString("speciality");
                        String location = rs.getString("location");
                        String days = rs.getString("days");
                        String visitTime = rs.getString("visit_time");

                        MaintainDoctorVO vo = new MaintainDoctorVO(id, doctorName, speciality,
location, days, visitTime );
                        lst.add(vo);

                }
        } catch (Exception e) {
                e.printStackTrace();
        }finally{
                DBService.close(rs, null, ps, con);
        }
        return lst;

}

}
```