---

1. JDBC / Database applications

SQL: create Product table

```
CREATE TABLE Product (
  Product_Id INT PRIMARY KEY,
  Product_Name VARCHAR(100),
  Product_Qty NUMERIC,
  Product_Price NUMERIC(10,2)
);
```

1.1 — Connect using JDBC and print message

```
import java.sql.*;


public class JDBCConnect {
    public static void main(String[] args) {
        String url = "jdbc:mysql://localhost:3306/yourdb"; // change
        String user = "youruser";
        String pass = "yourpass";
        try {
            Class.forName("com.mysql.cj.jdbc.Driver"); // or other driver
            Connection conn = DriverManager.getConnection(url, user, pass);
            if (conn != null && !conn.isClosed()) {
                System.out.println("Connection Established");
                conn.close();
            } else {
                System.out.println("Connection failed");
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

```
}
```

1.2 — Console app: retrieve & display a product record

```
import java.sql.*;
import java.util.Scanner;


public class RetrieveProduct {
   public static void main(String[] args) throws Exception {
      String url = "jdbc:mysql://localhost:3306/yourdb";
      String user = "youruser";
      String pass = "yourpass";
      try (Connection conn = DriverManager.getConnection(url, user, pass);
           Scanner sc = new Scanner(System.in)) {
         System.out.print("Enter Product_Id: ");
         int id = Integer.parseInt(sc.nextLine());
         String sql = "SELECT * FROM Product WHERE Product_Id = ?";
         try (PreparedStatement ps = conn.prepareStatement(sql)) {
            ps.setInt(1, id);
            try (ResultSet rs = ps.executeQuery()) {
               if (rs.next()) {
                  System.out.println("ID: " + rs.getInt("Product_Id"));
                  System.out.println("Name: " + rs.getString("Product_Name"));
                  System.out.println("Qty: " + rs.getBigDecimal("Product_Qty"));
                  System.out.println("Price: " + rs.getBigDecimal("Product_Price"));
               } else {
                  System.out.println("Product not found.");
               }
            }
         }
      }
   }
```

```
}
```

1.3 — CRUD operations (Insert, Delete, Update, Select) — console program

```
import java.sql.*;
import java.util.Scanner;

public class ProductCRUD {
   private static final String URL = "jdbc:mysql://localhost:3306/yourdb";
   private static final String USER = "youruser";
   private static final String PASS = "yourpass";

   public static void main(String[] args) throws Exception {
      try (Connection conn = DriverManager.getConnection(URL, USER, PASS);
           Scanner sc = new Scanner(System.in)) {
         while (true) {
            System.out.println("\n1.Insert 2.Delete 3.Update 4.SelectAll 5.Exit");
            System.out.print("Choice: ");
            int c = Integer.parseInt(sc.nextLine());
            if (c == 1) insert(conn, sc);
            else if (c == 2) delete(conn, sc);
            else if (c == 3) update(conn, sc);
            else if (c == 4) selectAll(conn);
            else break;
         }
      }
   }

   static void insert(Connection conn, Scanner sc) throws SQLException {
      System.out.print("Id: "); int id = Integer.parseInt(sc.nextLine());
      System.out.print("Name: "); String name = sc.nextLine();
      System.out.print("Qty: "); String qty = sc.nextLine();
```

```
        System.out.print("Price: "); String price = sc.nextLine();
        String sql = "INSERT INTO Product(Product_Id, Product_Name, Product_Qty, Product_Price)
VALUES(?,?,?,?)";
        try (PreparedStatement ps = conn.prepareStatement(sql)) {
            ps.setInt(1, id);
            ps.setString(2, name);
            ps.setBigDecimal(3, new java.math.BigDecimal(qty));
            ps.setBigDecimal(4, new java.math.BigDecimal(price));
            System.out.println(ps.executeUpdate() + " row(s) inserted.");
        }
    }

    static void delete(Connection conn, Scanner sc) throws SQLException {
        System.out.print("Id to delete: "); int id = Integer.parseInt(sc.nextLine());
        try (PreparedStatement ps = conn.prepareStatement("DELETE FROM Product WHERE
Product_Id=?")) {
            ps.setInt(1, id);
            System.out.println(ps.executeUpdate() + " row(s) deleted.");
        }
    }

    static void update(Connection conn, Scanner sc) throws SQLException {
        System.out.print("Id to update: "); int id = Integer.parseInt(sc.nextLine());
        System.out.print("New Name: "); String name = sc.nextLine();
        System.out.print("New Qty: "); String qty = sc.nextLine();
        System.out.print("New Price: "); String price = sc.nextLine();
        String sql = "UPDATE Product SET Product_Name=?, Product_Qty=?, Product_Price=? WHERE
Product_Id=?";
        try (PreparedStatement ps = conn.prepareStatement(sql)) {
            ps.setString(1, name);
            ps.setBigDecimal(2, new java.math.BigDecimal(qty));
            ps.setBigDecimal(3, new java.math.BigDecimal(price));
```

```
            ps.setInt(4, id);
            System.out.println(ps.executeUpdate() + " row(s) updated.");
        }
    }

    static void selectAll(Connection conn) throws SQLException {
        try (Statement st = conn.createStatement();
             ResultSet rs = st.executeQuery("SELECT * FROM Product")) {
            System.out.println("Products:");
            while (rs.next()) {
                System.out.printf("%d | %s | %s | %s%n",
                        rs.getInt("Product_Id"),
                        rs.getString("Product_Name"),
                        rs.getString("Product_Qty"),
                        rs.getString("Product_Price"));
            }
        }
    }
}
```

1.4 — GUI (Swing) that performs Insert / Delete / Update

Simple Swing frame connected to DB. Save as ProductManagerGUI.java.

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.sql.*;

public class ProductManagerGUI extends JFrame {
    private JTextField idF = new JTextField(10);
    private JTextField nameF = new JTextField(20);
    private JTextField qtyF = new JTextField(10);
```

```
    private JTextField priceF = new JTextField(10);
    private JButton insertBtn = new JButton("Insert");
    private JButton updateBtn = new JButton("Update");
    private JButton deleteBtn = new JButton("Delete");
    private Connection conn;

    public ProductManagerGUI() {
        setTitle("Product Manager");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new GridBagLayout());
        GridBagConstraints c = new GridBagConstraints();
        c.insets = new Insets(5,5,5,5);
        c.gridx=0; c.gridy=0; add(new JLabel("ID:"), c); c.gridx=1; add(idF,c);
        c.gridx=0; c.gridy=1; add(new JLabel("Name:"), c); c.gridx=1; add(nameF,c);
        c.gridx=0; c.gridy=2; add(new JLabel("Qty:"), c); c.gridx=1; add(qtyF,c);
        c.gridx=0; c.gridy=3; add(new JLabel("Price:"), c); c.gridx=1; add(priceF,c);
        JPanel p = new JPanel(); p.add(insertBtn); p.add(updateBtn); p.add(deleteBtn);
        c.gridx=0; c.gridy=4; c.gridwidth=2; add(p,c);
        pack();
        setLocationRelativeTo(null);

        try {
            conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/yourdb","youruser","yourpass");
        } catch (SQLException e) {
            JOptionPane.showMessageDialog(this, "DB Conn failed: " + e.getMessage());
            System.exit(1);
        }

        insertBtn.addActionListener(e -> doInsert());
        updateBtn.addActionListener(e -> doUpdate());
```

```
        deleteBtn.addActionListener(e -> doDelete());
    }

    private void doInsert() {
        try (PreparedStatement ps = conn.prepareStatement(
                "INSERT INTO Product(Product_Id, Product_Name, Product_Qty, Product_Price)
VALUES(?,?,?,?)")) {
            ps.setInt(1, Integer.parseInt(idF.getText()));
            ps.setString(2, nameF.getText());
            ps.setBigDecimal(3, new java.math.BigDecimal(qtyF.getText()));
            ps.setBigDecimal(4, new java.math.BigDecimal(priceF.getText()));
            int r = ps.executeUpdate();
            JOptionPane.showMessageDialog(this, r + " row(s) inserted");
        } catch (Exception ex) { JOptionPane.showMessageDialog(this, "Error: " + ex.getMessage()); }
    }

    private void doUpdate() {
        try (PreparedStatement ps = conn.prepareStatement(
                "UPDATE Product SET Product_Name=?, Product_Qty=?, Product_Price=? WHERE
Product_Id=?")) {
            ps.setString(1, nameF.getText());
            ps.setBigDecimal(2, new java.math.BigDecimal(qtyF.getText()));
            ps.setBigDecimal(3, new java.math.BigDecimal(priceF.getText()));
            ps.setInt(4, Integer.parseInt(idF.getText()));
            int r = ps.executeUpdate();
            JOptionPane.showMessageDialog(this, r + " row(s) updated");
        } catch (Exception ex) { JOptionPane.showMessageDialog(this, "Error: " + ex.getMessage()); }
    }

    private void doDelete() {
        try (PreparedStatement ps = conn.prepareStatement("DELETE FROM Product WHERE
Product_Id=?")) {
```

```
            ps.setInt(1, Integer.parseInt(idF.getText()));
            int r = ps.executeUpdate();
            JOptionPane.showMessageDialog(this, r + " row(s) deleted");
        } catch (Exception ex) { JOptionPane.showMessageDialog(this, "Error: " + ex.getMessage()); }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> new ProductManagerGUI().setVisible(true));
    }
}
```

1.5 — Present choices and display price (console)

```
import java.sql.*;
import java.util.*;

public class ChooseProduct {
    public static void main(String[] args) throws Exception {
        String url = "jdbc:mysql://localhost:3306/yourdb";
        String user = "youruser";
        String pass = "yourpass";
        try (Connection conn = DriverManager.getConnection(url,user,pass);
             Scanner sc = new Scanner(System.in)) {
            Map<Integer,String> map = new LinkedHashMap<>();
            try (Statement st = conn.createStatement();
                 ResultSet rs = st.executeQuery("SELECT Product_Id, Product_Name FROM Product")) {
                while (rs.next()) map.put(rs.getInt(1), rs.getString(2));
            }
            System.out.println("Products:");
            map.forEach((id,name) -> System.out.println(id + " -> " + name));
            System.out.print("Choose product id: ");
            int id = Integer.parseInt(sc.nextLine());
```

```
        try (PreparedStatement ps = conn.prepareStatement("SELECT Product_Price FROM Product
WHERE Product_Id=?")) {
            ps.setInt(1, id);
            try (ResultSet rs = ps.executeQuery()) {
               if (rs.next()) System.out.println("Price: " + rs.getBigDecimal(1));
               else System.out.println("Product not found");
            }
         }
      }
   }
}
```

---

2. Servlets

Note: use Servlet 3 annotations or web.xml. Below uses annotations (requires servlet-api library / compatible container like Tomcat).

2.1 — Hello World servlet

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import javax.servlet.annotation.*;

@WebServlet("/hello")
public class HelloServlet extends HttpServlet {
   protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException,
IOException {
      resp.setContentType("text/html;charset=UTF-8");
      try(PrintWriter out = resp.getWriter()) {
         out.println("<h1>Hello World</h1>");
      }
   }
}
```

2.2 — Display all request headers

```
@WebServlet("/headers")
public class HeadersServlet extends HttpServlet {
   protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws IOException {
      resp.setContentType("text/plain");
      PrintWriter out = resp.getWriter();
      var it = req.getHeaderNames();
      while (it.hasMoreElements()) {
         String name = it.nextElement();
         out.println(name + ": " + req.getHeader(name));
      }
   }
}
```

2.3 — Display request parameters

```
@WebServlet("/params")
public class ParamsServlet extends HttpServlet {
   protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws IOException {
      resp.setContentType("text/plain");
      PrintWriter out = resp.getWriter();
      req.getParameterMap().forEach((k,v) -> out.println(k + " = " + String.join(", ", v)));
   }
}
```

2.4 — Serve one of three PDFs based on year param

Place your PDFs under webapp/WEB-INF/resources or a location accessible. Example uses getResourceAsStream.

```
@WebServlet("/syllabus")
public class SyllabusServlet extends HttpServlet {
   protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws IOException {
```

```
        String year = req.getParameter("year"); // expected "1","2","3"
        String path;
        if ("1".equals(year)) path = "/WEB-INF/resources/mca1.pdf";
        else if ("2".equals(year)) path = "/WEB-INF/resources/mca2.pdf";
        else path = "/WEB-INF/resources/mca3.pdf";

        try (InputStream in = getServletContext().getResourceAsStream(path)) {
            if (in == null) {
                resp.sendError(HttpServletResponse.SC_NOT_FOUND, "PDF not found");
                return;
            }
            resp.setContentType("application/pdf");
            resp.setHeader("Content-Disposition", "inline; filename=\"syllabus.pdf\"");
            byte[] buf = new byte[8192];
            int n;
            OutputStream out = resp.getOutputStream();
            while ((n = in.read(buf)) > 0) out.write(buf,0,n);
        }
    }
}
```

2.5 — Authentication servlet (DB) + set session and forward to home.jsp

Assume Login table has columns loginid, password, fullname, address.

```
@WebServlet("/login")
public class LoginServlet extends HttpServlet {
    private String dbUrl = "jdbc:mysql://localhost:3306/yourdb", user="youruser", pass="yourpass";

    protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws
ServletException, IOException {
        String loginid = req.getParameter("loginid");
        String password = req.getParameter("password");
```

```
        try (Connection conn = DriverManager.getConnection(dbUrl, user, pass);
             PreparedStatement ps = conn.prepareStatement("SELECT fullname, address FROM Login WHERE loginid=? AND password=?")) {
            ps.setString(1, loginid);
            ps.setString(2, password);
            try (ResultSet rs = ps.executeQuery()) {
                if (rs.next()) {
                    String fullname = rs.getString("fullname");
                    String address = rs.getString("address");
                    HttpSession session = req.getSession(true);
                    session.setAttribute("loginid", loginid);
                    session.setAttribute("fullname", fullname);
                    session.setAttribute("address", address);
                    // forward to home.jsp
                    req.getRequestDispatcher("/home.jsp").forward(req, resp);
                } else {
                    req.setAttribute("error", "Invalid credentials");
                    req.getRequestDispatcher("/login.jsp").forward(req, resp);
                }
            }
        } catch (SQLException e) {
            throw new ServletException(e);
        }
    }
}
```

And a home.jsp would display session attributes (sample below in JSP section).

2.6 — Interest calculation servlet (process HTML form)

Form sends principal, rate, years to /interest.

```
@WebServlet("/interest")
public class InterestServlet extends HttpServlet {
```

```
    protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws IOException {
        double p = Double.parseDouble(req.getParameter("principal"));
        double r = Double.parseDouble(req.getParameter("rate"));
        double t = Double.parseDouble(req.getParameter("years"));
        double simpleInterest = p * r * t / 100.0;
        double amount = p + simpleInterest;
        resp.setContentType("text/html");
        try (PrintWriter out = resp.getWriter()) {
            out.printf("<h2>Simple Interest: %.2f</h2><p>Amount: %.2f</p>", simpleInterest, amount);
        }
    }
}
```

Example HTML form:

```
<form action="interest" method="post">
  Principal: <input name="principal"><br>
  Rate (%): <input name="rate"><br>
  Years: <input name="years"><br>
  <button type="submit">Calculate</button>
</form>
```

2.7 — Remember last visit using Cookie (stores timestamp)

```
@WebServlet("/lastvisit")
public class LastVisitServlet extends HttpServlet {
    protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws IOException {
        Cookie[] cookies = req.getCookies();
        String last = null;
        if (cookies != null) {
            for (Cookie c : cookies) {
                if ("lastVisit".equals(c.getName())) { last = c.getValue(); break; }
            }
        }
```

```
        resp.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = resp.getWriter()) {
            if (last != null) {
                long lastMillis = Long.parseLong(last);
                long diff = System.currentTimeMillis() - lastMillis;
                long seconds = diff / 1000;
                out.println("<p>Your last visit was " + seconds + " seconds ago.</p>");
            } else {
                out.println("<p>This seems to be your first visit (or cookies cleared).</p>");
            }
            Cookie newCookie = new Cookie("lastVisit", Long.toString(System.currentTimeMillis()));
            newCookie.setMaxAge(60*60*24*365); // 1 year
            resp.addCookie(newCookie);
        }
    }
}
```

---

3. JSPs

3.1 — Simple JSP message (message.jsp)

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html><body>
  <h1>Welcome to JSP page!</h1>
  <p>This is a simple JSP message.</p>
</body></html>
```

3.2 — JSP using include directive (header/footer)

header.jsp:

```
<!-- header.jsp -->
<div style="background:#eee;padding:10px"><h1>Site Header</h1></div>
```

footer.jsp:

<!-- footer.jsp -->

<small>Footer © 2025</small>

page.jsp:

<%@ include file="header.jsp" %>

Main content goes here.

<%@ include file="footer.jsp" %>

3.3 — JSP using scripting elements (expression, scriptlet, declaration)

<%@ page language="java" contentType="text/html;charset=UTF-8" %>

<%! int counter = 0; %> <!-- declaration -->

<%

  // scriptlet

  counter++;

  String name = request.getParameter("name");

%>

<html><body>

  <h2>Hello <%= (name!=null?name:"Guest") %></h2> <!-- expression -->

  Page view count (this JVM): <%= counter %>

</body></html>

3.4 — Page-composite using <jsp:include> and <jsp:forward>

main.jsp (include example):

<jsp:include page="header.jsp"/>

Page body here

<jsp:include page="footer.jsp"/>

redirect.jsp (forward example):

<%

  if (request.getParameter("action") != null && request.getParameter("action").equals("go")) {

    RequestDispatcher rd = request.getRequestDispatcher("target.jsp");

    rd.forward(request, response);

    return;

```
 }
%>
<p>No forward performed.</p>
```

3.5 — Display personal info and result info in two tables

```
<%@ page contentType="text/html;charset=UTF-8" %>
<%
 // sample data (could come from request/session/DB)
 String name="Alice"; String roll="MCA001"; String course="MCA";
 int marks = 420; String grade = "A";
%>
<html><body>
<h3>Personal Information</h3>
<table border="1">
 <tr><th>Name</th><td><%=name%></td></tr>
 <tr><th>Roll No</th><td><%=roll%></td></tr>
 <tr><th>Course</th><td><%=course%></td></tr>
</table>

<h3>Result Information</h3>
<table border="1">
 <tr><th>Total Marks</th><td><%=marks%></td></tr>
 <tr><th>Grade</th><td><%=grade%></td></tr>
</table>
</body></html>
```

3.6 — JSP to perform DB operations for Student table

SQL for Student:

```
CREATE TABLE Student (
 StudId INT PRIMARY KEY,
 Name VARCHAR(100),
```

```
  Address VARCHAR(255),
  result VARCHAR(20)
);
```

studentForm.jsp (form to add/update/delete):

```
<form action="studentAction.jsp" method="post">
  Id: <input name="studid"/><br/>
  Name: <input name="name"/><br/>
  Address: <input name="address"/><br/>
  Result: <input name="result"/><br/>
  <button name="action" value="insert">Insert</button>
  <button name="action" value="update">Update</button>
  <button name="action" value="delete">Delete</button>
</form>
```

studentAction.jsp (do DB ops — simple example)

```
<%@ page import="java.sql.*" %>
<%
String url="jdbc:mysql://localhost:3306/yourdb", user="youruser", pass="yourpass";
String action=request.getParameter("action");
int id=Integer.parseInt(request.getParameter("studid"));
try (Connection conn = DriverManager.getConnection(url,user,pass)) {
 if ("insert".equals(action)) {
   PreparedStatement ps=conn.prepareStatement("INSERT INTO
Student(StudId,Name,Address,result) VALUES(?,?,?,?)");
   ps.setInt(1,id); ps.setString(2, request.getParameter("name"));
   ps.setString(3, request.getParameter("address")); ps.setString(4, request.getParameter("result"));
   int r=ps.executeUpdate(); out.println(r+" inserted");
 } else if ("update".equals(action)) {
   PreparedStatement ps=conn.prepareStatement("UPDATE Student SET Name=?,Address=?,result=?
WHERE StudId=?");
   ps.setString(1, request.getParameter("name")); ps.setString(2, request.getParameter("address"));
   ps.setString(3, request.getParameter("result")); ps.setInt(4, id);
   out.println(ps.executeUpdate()+" updated");
```

```
 } else if ("delete".equals(action)) {
   PreparedStatement ps=conn.prepareStatement("DELETE FROM Student WHERE StudId=?");
   ps.setInt(1, id); out.println(ps.executeUpdate()+" deleted");
 }
} catch(Exception e){ out.println("Error: "+e.getMessage()); }
%>
```

(You may prefer to implement these as Servlets for separation of concerns.)

3.7 — JSP Session tracking for online shopping (simple cart)

addToCart.jsp — adds product ID to cart stored in session:

```
<%@ page import="java.util.*" %>
<%
String pid = request.getParameter("pid");
HttpSession session = request.getSession(true);
List<String> cart = (List<String>)session.getAttribute("cart");
if (cart == null) { cart = new ArrayList<>(); session.setAttribute("cart", cart); }
if (pid != null) cart.add(pid);
%>
<p>Added product <%= pid %> to cart.</p>
<a href="viewCart.jsp">View Cart</a>
```

viewCart.jsp:

```
<%@ page import="java.util.*" %>
<%
List<String> cart = (List<String>)session.getAttribute("cart");
if (cart == null || cart.isEmpty()) {
  out.println("<p>Your cart is empty</p>");
} else {
  out.println("<ul>");
  for (String p : cart) out.println("<li>Product ID: "+p+"</li>");
  out.println("</ul>");
}
```

%>

Quick notes & suggestions

- Replace DB connection strings/credentials and driver class to match your DB (MySQL, PostgreSQL, Oracle). Class.forName(...) may not be required in modern JDBC if driver is on classpath.
- For production: always hash passwords (authentication servlet uses plain text for demo only).
- Consider using MVC pattern (Servlets -> DAOs -> JSP) for clearer separation.
- Use PreparedStatement (shown) to avoid SQL injection.
- For Servlets/JSPs with forms, remember to set proper character encoding if you handle non-ASCII input: request.setCharacterEncoding("UTF-8");.
- For building and deploying: use Tomcat (or any servlet container); place JSPs in webapp folder.
