Department of Information Technology
Web Programming and Development Lab
Internal lab Exam
2022-23
III/IV B.Tech. IT

1	Create a Table with fields Employeeid, Firstname, Lastname, Email, Address, City. Implement crud operations on it
2	Create placement registration form to read basic details from a student. Read all the information in a servlet. Create cookies for student name and roll number. Retrieve the information in another servlet
3	Read the marks of a student in six subjects and compute the CGPA in a sevlet. Use session to store information and retrieve them in another servlet
4	Create web pages to registration and login page. Once the login is validated print the student information in a servlet
5	Use the table with attributes : Customerid,Name ,Address, Phone, Mailid ,Age.
6	Create a Servlet to read Library information. The html page should contain atleast five entries. Create Cookies and retrieve them
7	Create a login page from html. Read the credentials from html page and validate the data from relational database. Once it is validated, print the information in another servlet. Otherwise redirect to the same html page
8	Create a table with the fields: Custid Number, Name, Address,Phone No. Apply CRUD Operations on this table and display the results
9	Create placement registration form to read basic details from a student. Read all the information in a servlet. Create cookies for student name and roll number. Retrieve the information in another servlet
10	Implement Crud operations on Product table in a servlet
11	Create a user input form - Name, pasword, gender, age, languages known, certificatins. Read the information in a servlet and print the information. Use session handling for this
12	Read the marks of a student in six subjects and compute the CGPA in a sevlet. Use session to store information and retrieve them in another JSP
13	Create angular template driven form to read information suitable to placement activities
14	Create a Login page and registration page similar to Moodle and retrieve the data in a servlet
15	Develop a web application to simulate the following:Create a login, logout and profile pages. Create 3 links: login, logout and profile. User can't go to profile page until he/she is logged in. If user is logged out, he need to login again to visit profile. In this application, generate the following files.
	1. index.html
	2. link.html
	3. login.html
	4. LoginServlet.java

	5. LogoutServlet.java
	ProfileServlet.java
16	Create a web page that reads the placement details and do the following: Read the data in a servlet, set Session and print the information in the same servlet Also, print the information in another servlet
17	Web application in which a web form allows the end user to perform online transfer of funds from saving account to current account (SQL database to be connected)
18	1. Create a web page of your choice to read the information from the user (Atleast 5 controls in html page). Read the data in a servlet and do the following:
	a. Print the information in the servlet
	b. Read the data in one servlet and print the information in another servlet
	Use RequestDispatcher
19	Write a servlet program for Creating a form with username and password and validate the details UsingRequestDispatcher method
20	Create a web page that reads the placement details and do the following: Read the data in a servlet, set Session and print the information in the same servletAlso, print the information in another servlet
21	Write a Servlet program for Creating a form with username and password and validate the details UsingRequestDispatcher method
22	Create a html page that reads student information such as sno,sname,marks. Read the data in a Servlet and set cookies and retrived the information
23	Create a html page that reads vehicle information such as vno, vname, price, model name, year. Read the data in a Servlet and set session and retrived the information
24	Create a Servlet to find the internal marks obtained by a student in vrsec by reading all the exam marks. Read the data from html
25	Create a Course class with three fields id, name and author. Use parameterised constructor to set the values and write the getters to retrieve the data. Override the toString( ) method too. Create another CourseController to return list of all objects as an array(the end point is /courses). Test the classes using Spring Boot
26	Create a Student class with three fields sno, sname. Use parameterised constructor to set the values and write the getters to retrieve the data. Override the toString( ) method too. Create another StudentController to return list of all objects as an array(the end point is /students). Test the classes using Spring Boot
27	3)Create a Library with three fields lno, lname and lauthor. Use parameterised constructor to set the values and write the getters to retrieve the data. Override the toString( ) method

	too. Create another LibraryController to return list of all objects as an array(the end point is /libraries). Test the classes using Spring Boot
28	4)Create a Bank with three fields bno, bname and bbranch. Use parameterised constructor to set the values and write the getters to retrieve the data. Override the toString( ) method too. Create another BankController to return list of all objects as an array(the end point is /banks). Test the classes using Spring Boot
29	5)Create a Hospital with two fields hname,haddress. Use parameterised constructor to set the values and write the getters to retrieve the data. Override the toString( ) method too. Create another HospitalController to return list of all objects as an array(the end point is /hospitals). Test the classes using Spring Boot
30	6)Create a Class School with three fields sname,saddress,scode. Use parameterised constructor to set the values and write the getters to retrieve the data. Override the toString( ) method too. Create another SchoolController to return list of all objects as an array(the end point is /schools). Test the classes using Spring Boot
31	Create a Class City with two fields cname and ccode. Use parameterised constructor to set the values and write the getters to retrieve the data. Override the toString( ) method too. Create another CityController to return list of all objects as an array(the end point is /citys). Test the classes using Spring Boot
32	Create a Class Bank with three fields bno, bname and bbranch. Use parameterised constructor to set the values and write the getters to retrieve the data. Override the toString( ) method too. Create another BankController to return list of all objects as an array(the end point is /bankls). Test the classes using Spring Boot
33	Create a Class Bank with three fields bno, bname and bbranch. Create a constructor, setters and getters. Create a BankService class to insert the Bank details into the in-memory database and retrieve Bank information. Test the application as a Spring Boot Application.
34	Create a Class City with two fields cname and ccode. Create a constructor, setters and getters. Create a CityService class to insert the City details into the in-memory database and retrieve City information. Test the application as a Spring Boot Application.
35	Create a Class Subject with four fields scode, sname,smarks and sfaculty. Create a constructor, setters and getters. Create a SubjectService class to insert the Subject details into the in-memory database and retrieve Subject information. Test the application as a Spring Boot Application.
36	10)Create a Class Department with three fields dname, dhod and dcode. Create a constructor, setters and getters. Create a DepartmentService class to insert the Department details into the in-memory database and retrieve Department information. Test the application as a Spring Boot Application.

37	1Create a Class Book with three fields bno, bname and bprice. Create a constructor, setters and getters. Create a BookService class to insert the Book details into the in-memory database and retrieve Book information. Test the application as a Spring Boot Application.
38	1Create a Class Book with three fields bno, bname and bprice. Create a constructor, setters and getters. Create a BookService class to insert the Book details into the in-memory database and retrieve Book information. Test the application as a Spring Boot Application.
39	a Class Order with three fields oid, oname,odate. Create a 13)Create constructor, setters and getters. Create a OrderService class to insert the Order details into the in-memory database and retrieve Order information. Test the application as a Spring Boot Application.
40	1Create a Class Customer with four fields cno, cname,cmail and caddress. Create a constructor, setters and getters. Create a CustomerService class to insert the Customer details into the in-memory database and retrieve Customer information. Test the application as a Spring Boot Application.
41	Create a Class Costume with two fields ccolor, cname. Create a constructor, setters and getters. Create a CostumeService class to insert the Costume details into the in-memory database and retrieve Costume information. Test the application as a Spring Boot Application.
42	Create a Class Employee with three fields eno, ename and department. Create a constructor, setters and getters. Create a EmployeeService class to insert the employee details into the in-memory database and retrieve employee information. Test the application as a Spring Boot Application.
43	Create a Class Student with three fields sno, sname and smarks. Create a constructor, setters and getters. Create a StudentService class to insert the Student details into the in-memory database and retrieve Student information. Test the application as a Spring Boot Application.
44	`CREATE TABLE customer (` `id INT AUTO_INCREMENT PRIMARY KEY,`

	postalCode VARCHAR(15) default NULL,)
45	CREATE TABLE product (id INT AUTO_INCREMENT PRIMARY KEY, product_name VARCHAR(50) NOT NULL, price VARCHAR(NOT NULL, qty VARCHAR(4) NOT NULL)
46	Create a Class Library with three fields lno, lname and lauthor. Create a constructor, setters and getters. Create a LibraryService class to insert the Library details into the in-memory database and retrieve Library information. n.
47	Create a Class School with three fields sname,saddress,scode. Create a constructor, setters and getters. Create a SchoolService class to insert the School details into the in-memory database and retrieve School information. Test the application as a Spring Boot Application.
48	Create a Class Bank with three fields bno, bname and bbranch. Create a constructor, setters and getters. Create a BankService class to insert the Bank details into the in-memory database and retrieve Bank information. Test the application as a Spring Boot Application.
49	Create a Class Hospital with two fields hname,haddress Create a constructor, setters and getters. Create a HospitalService class to insert the Hospital details into the in-memory database and retrieve Hospital information. Test the application as a Spring Boot Application.
50	Create a Class Subject with four fields scode, sname,smarks and sfaculty. Create a constructor, setters and getters. Create a SubjectService class to insert the Subject details into the in-memory database and retrieve Subject information. Test the application as a Spring Boot Application.
51	Create a Class Department with three fields dname, dhod and dcode. Create a constructor, setters and getters. Create a DepartmentService class to insert the Department details into the in-memory database and retrieve Department information. Test the application as a Spring Boot Application.
52	Create a Class Watch with three fields wid, wbrand and wprice. Create a constructor, setters and getters. Create a WatchService class to insert the Watch details into the in-memory database and retrieve Watch information. Test the application as a Spring Boot Application.

53	Create a Class Customer with four fields cno, cname,cmail and caddress. Create a constructor, setters and getters. Create a CustomerService class to insert the Customer details into the in-memory database and retrieve Customer information. Test the application as a Spring Boot Application.
54	Create a Class Department with three fields dname, dhod and dcode. Create a constructor, setters and getters. Create a DepartmentService class to insert the Department details into the in-memory database and retrieve Department information. Test the application as a Spring Boot Application.
55	Create a Class Movie with three fields mname, mgenre, mlanguage. Create a constructor, setters and getters. Create a MovieService class to insert the Movie details into the in-memory database and retrieve Movie information. Test the application as a Spring Boot Application.
56	Create a Class Project with three fields pname, pdomain, pstatus. Create a constructor, setters and getters. Create a ProjectService class to insert the Movie details into the in-memory database and retrieve Project information. Test the application as a Spring Boot Application
57	Create a Class Product with three fields pid, pname, pprice. Create a constructor, setters and getters. Create a ProductService class to insert the Product details into the in-memory database and retrieve Product information. Test the application as a Spring Boot Application.
58	Create a Class Event with four fields eid, ename, edate, elocation. Create a constructor, setters and getters. Create a EventService class to insert the Event details into the in-memory database and retrieve Event information. Test the application as a Spring Boot Application.
59	Create a Class Inventory with four fields ino, iname, icategory, icost. Create a constructor, setters and getters. Create a InventoryService class to insert the Inventory details into the in-memory database and retrieve Inventory information. Test the application as a Spring Boot Application.
60	Create a Class Hostel with three fields hno, hname and hrent. Use parameterised constructor to set the values and write the getters to retrieve the data. Override the toString( ) method too. Create another Hostel Controller to return list of all objects as an array(the end point is / hostels). Test the classes using Spring Boot
61	Create a Class Shoe with four fields sid, sbrand, scost and scolor. Use parameterised constructor to set the values and write the getters to retrieve the data. Override the toString( ) method too. Create another ShoeController to return list of all objects as an

	array(the end point is /shoes). Test the classes using Spring Boot
62	Create a Class Train with four fields tno, tsource, tdestination and tname. Use parameterised constructor to set the values and write the getters to retrieve the data. Override the toString( ) method too. Create another TrainController to return list of all objects as an array(the end point is /trains). Test the classes using Spring Boot
63	Create a Class Event with four fields eid, ename, edate, elocation. Use parameterised constructor to set the values and write the getters to retrieve the data. Override the toString( ) method too. Create another EventController to return list of all objects as an array(the end point is /events). Test the classes using Spring Boot
64	Create a Class Inventory with four fields ino, iname, icategory, icost. Use parameterised constructor to set the values and write the getters to retrieve the data. Override the toString( ) method too. Create another InventoryController to return list of all objects as an array(the end point is /inventories). Test the classes using Spring
65	Create a Class Event with four fields eid, ename, edate, elocation. Use parameterised constructor to set the values and write the getters to retrieve the data. Override the toString( ) method too. Create another EventController to return list of all objects as an array(the end point is /events). Test the classes using Spring Boot
66	Create a Class Project with three fields pname, pdomain, pstatus. Use parameterised constructor to set the values and write the getters to retrieve the data. Override the toString( ) method too. Create another ProjectController to return list of all objects as an array(the end point is /projects). Test the classes using Spring Boot
67	Create a Class Movie with three fields mname, mgenre, mlanguage. Use parameterised constructor to set the values and write the getters to retrieve the data. Override the toString( ) method too. Create another MovieController to return list of all objects as an array(the end point is /movies). Test the classes using Spring Boot
68	Create a Class Faculty with three fields fid, fname and fsalary. Use parameterised constructor to set the values and write the getters to retrieve the data. Override the toString( ) method too. Create another FacultyController to return list of all objects as an array(the end point is /faculties). Test the classes using Spring Boot
69	Create a Class Laptop with four fields lid, lram, lcost and lcolor. Use parameterised constructor to set the values and write the getters to retrieve the data. Override the toString( ) method too. Create another LaptopController to return list of all objects as an array(the end point is /laptops). Test the classes using Spring Boot
70	Create a Class College with three fields ccode, cname and caddress. Use parameterised constructor to set the values and write the getters to retrieve the data. Override the toString( ) method too. Create another CollegeController to return list of all objects as an array(the end point is /colleges). Test the classes using Spring Boot

.