Name: Chebrolu Rukmini

Q1.1 (D) @RequestMapping
Q1.2 (B False
Q1.3 (D) class
Q1.4 (A) @PathVariable
Q1.5 (C) front Controller
Q1.6 (B) False
Q1.7 (A) getBean method
Q1.8 (B) scope
Q1.9 (C) @Controller
Q1.10 (B) Prototype

Q2.
Person.java
—----------
```
package com.dxc.person.bean;

public class Person {
   private int id;
   private String firstName;
   private String lastName;
   private String emailId;
   private String city;
   public int getId() {
         return id;
   }
   public void setId(int id) {
         this.id = id;
   }
   public String getFirstName() {
         return firstName;
   }
   public void setFirstName(String firstName) {
         this.firstName = firstName;
   }
   public String getLastName() {
         return lastName;
   }
   public void setLastName(String lastName) {
         this.lastName = lastName;
   }
   public String getEmailId() {
         return emailId;
```

```
    }
    public void setEmailId(String emailId) {
        this.emailId = emailId;
    }
    public String getCity() {
        return city;
    }
    public void setCity(String city) {
        this.city = city;
    }
    public Person() {
        super();
        // TODO Auto-generated constructor stub
    }
    public Person(int id, String firstName, String lastName, String emailId, String city) {
        super();
        this.id = id;
        this.firstName = firstName;
        this.lastName = lastName;
        this.emailId = emailId;
        this.city = city;
    }
    @Override
    public String toString() {
        return "\n" + id + ", firstName=" + firstName + ", lastName=" + lastName + ", emailId=" +
emailId
                + ", city=" + city ;
    }
}
```

—----------

PersonMapper.java

—----------

```
package com.dxc.person.bean;

import java.sql.ResultSet;
import java.sql.SQLException;
import org.springframework.jdbc.core.RowMapper;
import com.dxc.person.bean.Person;

public class PersonMapper implements RowMapper<Person> {

    public Person mapRow(ResultSet rs, int rowNum) throws SQLException {
        // TODO Auto-generated method stub
```

```
            Person person=new Person();
            person.setId(rs.getInt("id"));
            person.setFirstName(rs.getString("firstName"));
            person.setLastName(rs.getString("lastName"));
            person.setEmailId(rs.getString("emailId"));
            person.setCity(rs.getString("city"));
            return person;
    }

}
```

—----------------
PersonDao.java
—----------------

```
package com.dxc.person.dao;

import java.util.List;

import javax.sql.DataSource;

import com.dxc.person.bean.Person;

public interface PersonDao {

            public void setDataSource(DataSource ds);

            // Adding new person details
            public void createNewPerson(String firstName,String lastName, String emailId,String
city);

            // For displaying  all records
            public List<Person> listPersons();

            // Show person info by id
            public Person getPersonById(int id);

            // To update person info
            public void updatePerson(int id,String firstName,String lastName, String emailId,String
city);

            //for remove info info by id
            public void deletePersonById(int id);   }
```

PersonDaoImpl.java

—----------
```
package com.dxc.person.dao;

import java.util.List;

import javax.sql.DataSource;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.jdbc.core.JdbcTemplate;

import com.dxc.person.bean.Person;
import com.dxc.person.bean.PersonMapper;



public class PersonDaoImpl implements PersonDao{
   private DataSource dataSource;
   private JdbcTemplate jdbcTamplate;

   public void setDataSource(DataSource ds) {
         // TODO Auto-generated method stub
         this.dataSource=ds;
         this.jdbcTamplate=new JdbcTemplate(this.dataSource);
   }

   public void createNewPerson(String firstName, String lastName, String emailId, String city) {
         String query="insert into person(firstName, lastName,emailId,city) values(?,?,?,?)";
         this.jdbcTamplate.update(query, firstName ,lastName,emailId,city);
         System.out.print("\nStudent record has been created...");

   }

   public List<Person> listPersons() {
         String query="select * from person";
         return this.jdbcTamplate.query(query, new PersonMapper());
   }

   public Person getPersonById(int id) {
         String query="select * from person where id=?";
         Person person=null;
         try {
         person=this.jdbcTamplate.queryForObject(query,new Object[] {id},new
PersonMapper());
         }catch(Exception e) {
               System.err.print("error found:"+e.getMessage());
```

```
        }
        return person;
    }

    public void updatePerson(int id,String firstName, String lastName, String emailId, String city) {
        // TODO Auto-generated method stub
String query="update person set =?, avgmarks=? where rollno=?";

        this.jdbcTamplate.update(query, firstName,lastName,emailId,city,id);
        System.out.print("\nPerson record has been modified...");

    }

    public void deletePersonById(int id) {
        String query="delete from person where id=?";
        this.jdbcTamplate.update(query,id);
        System.out.print("\nStudent record has been deleted...");

    }


}
```

—----------

App.java

—----------

```
package com.dxc.training.test;

import java.util.List;
import java.util.Scanner;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.dxc.person.bean.Person;
import com.dxc.person.dao.PersonDao;
import com.dxc.person.dao.PersonDaoImpl;
import com.dxc.training.bean.Student;
import com.dxc.training.dao.StudentDao;
import com.dxc.training.dao.StudentDaoImpl;

public class App
{
    public static void main( String[] args )
    {
```

```
	int choice;
	int id;
	String firstName,lastName,emailId,city;

	Scanner sc=new Scanner(System.in);

	ApplicationContext context=new
ClassPathXmlApplicationContext("ApplicationContext.xml");
	PersonDao studentdao=(PersonDaoImpl)context.getBean("personDao");

	do {
		System.out.print("\n1. Add Person");
		System.out.print("\n2. Show All Persons");
		System.out.print("\n3. Show Person By Id");
		System.out.print("\n4. Modify Person");
		System.out.print("\n5. Delete Person");
		System.out.print("\n0. Exit from Application");
		System.out.print("\nEnter your choice(0-5):");
		choice=sc.nextInt();
		Object persondao;
		switch(choice) {
		case 1:
			System.out.print("\nEnter Person First Name:");
			firstName=sc.next();
			System.out.print("\nEnter Person Last Name:");
			lastName=sc.next();
			System.out.print("\nEnter Email:");
			emailId=sc.next();
			System.out.print("\nEnter City:");
			city=sc.next();
			studentdao.createNewPerson(firstName,lastName,emailId, city);
			break;
		case 2:
			List<Person> persons=studentdao.listPersons();
			for(Person person:persons) {
				System.out.print(persons);
			}


			break;
		case 3:
			System.out.print("\nEnter ID:");
			id=sc.nextInt();
			Person person2=((Object) persondao).getStudentById(id);
```

```
                        System.out.print(person2);
                        break;
                case 4:
                        System.out.print("\nEnter Id:");
                        id=sc.nextInt();
                        System.out.print("\nEnter Person First Name:");
                        firstName=sc.next();
                        System.out.print("\nEnter Person Last Name:");
                        lastName=sc.next();
                        System.out.print("\nEnter Email:");
                        emailId=sc.next();
                        System.out.print("\nEnter City:");
                        city=sc.next();


                        persondao.updatePerson(id,firstName,lastName,emailId, city);
                        break;
                case 5:
                        System.out.print("\nEnter Id:");
                        id=sc.nextInt();
                        studentdao.deletePersonById(id);
                        break;
                case 0:
                        System.out.print("\n Application terminated..");
                        break;
                default:
                        System.out.print("\n only 0 to 5 options availble..press any key to
continue..");
                }
        }while(choice!=0);
   }
}
```

—----------

ApplicationContext.xml

—----------

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">

<bean id="personDao" class="com.dxc.person.dao.PersonDaoImpl">
<property name="dataSource" ref="dataSourceBean"/>
</bean>
```

```
<bean id="dataSourceBean"
class="org.springframework.jdbc.datasource.DriverManagerDataSource">
<property name="driverClassName" value="com.mysql.cj.jdbc.Driver" />
<property name="url" value="jdbc:mysql://localhost:3306/dxcdb" />
<property name="username" value="root" />
<property name="password" value="Mysql@2710" />
</bean>

</beans>
—-------------------------
use dxcdb;
create table person(
id int(8) primary key auto_increment,
firstName varchar(30) not null,
lastName varchar(30) not null,
emailId varchar(30) not null,
city varchar(30) not null
);
—-------------------
```

Q3.

```
package com.dxc.training;
import java.util.Scanner;
import java.util.List;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.dxc.training.bean.Student;
import com.dxc.training.dao.StudentDAOImpl;

public class App
{
   public static void main( String[] args )
   {    int rollno=0;
   String name=null,stream=null;
   float avgmarks=0.0f;
   ApplicationContext context=new ClassPathXmlApplicationContext("ApplicationContext.xml");
   StudentDAOImpl stDAO=(StudentDAOImpl)context.getBean("studentDAOImpl");
   Scanner sc=new Scanner(System.in);
   int choice=0;
   do {
         System.out.print("\n1. Adding new Records");
```

```
System.out.print("\n2. Display All Records");
System.out.print("\n3. Display Record By Rollnumber");
System.out.print("\n4. Update Record");
System.out.print("\n5. Delete Record by Rollnumber");
System.out.print("\n0. Exit From App ");
System.out.print("\nEnter your choice(0-5):");
choice=sc.nextInt();
switch(choice) {
case 1:
	System.out.print("\nEnter Student name:");
	name=sc.next();
	System.out.print("\nEnter Student Stream:");
	stream=sc.next();
	System.out.print("\nEmnter Student AvgMarks:");
	avgmarks=sc.nextFloat();
	System.out.print("\n------------Adding new Records----------------------");
	stDAO.createStudent(name, stream, avgmarks);
	break;
case 2:
	System.out.print("\n------------Show All Records----------------------");
	List <Student> students=stDAO.listStudents();
	for(Student st:students) {
		System.out.print("\n"+st);
	}
	break;
case 3:
	System.out.print("\nEnter Student Roll number:");
	rollno=sc.nextInt();
	System.out.print("\n------------Show record by rollno----------------------");
	Student student=stDAO.getStudentById(rollno);
	System.out.print("\n"+student);
	break;
case 4:
	System.out.print("\nEnter Student Roll number for updating data:");
	rollno=sc.nextInt();
	System.out.print("\nEnter Student Stream:");
	stream=sc.next();
	System.out.print("\nEmnter Student AvgMarks:");
	avgmarks=sc.nextFloat();
	System.out.print("\n------------Modifying a record----------------------");
	stDAO.updateStudent(rollno, stream, avgmarks);
	break;
case 5:
	System.out.print("\n------------Delete record by rollno----------------------");
```

```
                System.out.print("\nEnter Student Roll number for delete data:");
                rollno=sc.nextInt();
                stDAO.deleteStudentById(rollno);
                break;
        case 0:
                System.out.print("\nYou have pressed 0 so Application terminated..");
                break;
        default:
                System.out.print("\nPress any key to continue.. valid choices are 005 only...");
        }

   }while(choice!=0);
   }
}

—----------------------------------
package com.dxc.training.bean;
public class Student {
   private int rollno;
   private String name;
   private String stream;
   private float avgmarks;
   public int getRollno() {
        return rollno;
   }
   public void setRollno(int rollno) {
        this.rollno = rollno;
   }
   public String getName() {
        return name;
   }
   public void setName(String name) {
        this.name = name;
   }
   public String getStream() {
        return stream;
   }
   public void setStream(String stream) {
        this.stream = stream;
   }
   public float getAvgmarks() {
        return avgmarks;
   }
   public void setAvgmarks(float avgmarks) {
```

```
        this.avgmarks = avgmarks;
    }
    public Student() {
        super();
        // TODO Auto-generated constructor stub
    }
    public Student(int rollno, String name, String stream, float avgmarks) {
        super();
        this.rollno = rollno;
        this.name = name;
        this.stream = stream;
        this.avgmarks = avgmarks;
    }
    @Override
    public String toString() {
        return "\n" + rollno + ", " + name + ", " + stream + ", " + avgmarks;
    }
}

—---------------------------------
package com.dxc.training.dao;
import java.util.List;
import javax.sql.DataSource;
import com.dxc.training.bean.Student;

public interface StudentDAO {
    //1.set datasource for db operation
    public void setDataSource(DataSource ds);

    //2. Add new Student
    public void createStudent(String name,String stream,float avgmarks);

    //3. To show all records
    public List<Student> listStudents();

    //4. Show Student info by rollno
    public Student getStudentById(int rollno);

    //5. To update student info
    public void updateStudent(int rollno,String stream,float avgmarks);

    //6. To remove student info by rollno
    public void deleteStudentById(int rollno);
}
```

—----------
```
package com.dxc.training.dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import java.util.List;

import javax.sql.DataSource;

import com.dxc.training.bean.Student;

public class StudentDAOImpl implements StudentDAO{
   private DataSource dataSource;

   public void setDataSource(DataSource ds) {
         this.dataSource=ds;

   }

   public void createStudent(String name, String stream, float avgmarks) {
         String query="insert into student(name,stream,avgmarks) values(?,?,?)";
         Connection con=null;
         PreparedStatement pstmt=null;
         try {
                  con=dataSource.getConnection();
                  pstmt=con.prepareStatement(query);
                  pstmt.setString(1, name);
                  pstmt.setString(2, stream);
                  pstmt.setFloat(3, avgmarks);
                  int i=pstmt.executeUpdate();
                  if(i>0) {
                           System.out.print("\n Record inserted..");
                  }else {
                           System.err.print("\n Record could not inserte..");
                  }
         }catch(Exception e) {
                  e.printStackTrace();
         }finally {
                  try {
                           pstmt.close();
                           con.close();
```

```
		}catch(Exception e) {
			System.out.print(e.getMessage());
		}

	}

}

public List<Student> listStudents() {
	String sql="select * from student";
	Connection con=null;
	PreparedStatement pstmt=null;
	List <Student> studentList=new ArrayList<Student>();
	ResultSet rs=null;
	try {
		con=dataSource.getConnection();
		pstmt=con.prepareStatement(sql);
		rs=pstmt.executeQuery();
		Student st;
		while(rs.next()) {
			st=new Student();
			st.setRollno(rs.getInt(1));
			st.setName(rs.getString(2));
			st.setStream(rs.getString(3));
			st.setAvgmarks(rs.getFloat(4));
			studentList.add(st);
			}
		}catch(Exception e) {
			System.out.print(e.getMessage());
		}
	return studentList;
}

public Student getStudentById(int rollno) {
	String sql="select * from student where rollno=?";
	Connection con=null;
	PreparedStatement pstmt=null;
	ResultSet rs=null;
	Student st=null;
	try {
		con=dataSource.getConnection();
		pstmt=con.prepareStatement(sql);
		pstmt.setInt(1, rollno);
		rs=pstmt.executeQuery();
```

```
                while(rs.next()) {
                        st=new Student();
                        st.setRollno(rs.getInt(1));
                        st.setName(rs.getString(2));
                        st.setStream(rs.getString(3));
                        st.setAvgmarks(rs.getFloat(4));
                        }
                }catch(Exception e) {
                        System.out.print(e.getMessage());
                }
        return st;
}



public void updateStudent(int rollno, String stream,float avgmarks) {
        String sql="update student set stream=?, avgmarks=? where rollno=?";
        Connection con=null;
        PreparedStatement pstmt=null;
        try {
                con=dataSource.getConnection();
                pstmt=con.prepareStatement(sql);
                pstmt.setString(1, stream);
                pstmt.setFloat(2, avgmarks);
                pstmt.setInt(3, rollno);
                int i=pstmt.executeUpdate();
                if(i>0) {
                        System.out.print("\n Record modified for rollno:"+rollno);
                }else {
                        System.out.print("\n Record modification failed for rollno:"+rollno);
                        }

                }catch(Exception e) {
                        System.out.print(e.getMessage());
                }

}



public void deleteStudentById(int rollno) {
        String sql="delete from student where rollno=?";
        Connection con=null;
```

```
        PreparedStatement pstmt=null;
        try {
                con=dataSource.getConnection();
                pstmt=con.prepareStatement(sql);
                pstmt.setInt(1, rollno);
                int i=pstmt.executeUpdate();
                if(i>0) {
                        System.out.print("\n Record deleted for rollno:"+rollno);
                }else {
                        System.out.print("\n Record deletion failed for rollno:"+rollno);
                        }

                }catch(Exception e) {
                        System.out.print(e.getMessage());
                }
   }

}
```

—----------

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">

<bean id="studentDAOImpl" class="com.dxc.training.dao.StudentDAOImpl">
<property name="dataSource" ref="dataSourceBean"/>
</bean>

<bean id="dataSourceBean"
class="org.springframework.jdbc.datasource.DriverManagerDataSource">
<property name="driverClassName" value="com.mysql.cj.jdbc.Driver" />
<property name="url" value="jdbc:mysql://localhost:3306/dxcdb" />
<property name="username" value="root" />
<property name="password" value="Mysql@2710" />
</bean>

</beans>
```

—----------

Q4.

ProductController.java

—----------

```
package com.dxc.training.rest.controller;
```

```
import java.util.List;
import java.util.Optional;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RestController;

import com.dxc.training.rest.dao.ProductRepository;
import com.dxc.training.rest.model.Product;


@RestController
@RequestMapping("/api")
public class ProductController {
   @Autowired
   private ProductRepository productRepo;

   //1.add new product
   @RequestMapping(value="/products",method=RequestMethod.POST)
   public Product addProduct(@RequestBody Product product) {
         return productRepo.save(product);
   }

   //2.remove existing product
         @RequestMapping(value="/products/{productID}",method=RequestMethod.DELETE)
         public String delProduct(@PathVariable Integer productID) {
               productRepo.deleteById(productID);
               return "Record with productID:"+productID+" has been deleted..";
         }

   //3. To check user by productid
         @RequestMapping(value="/products/{productID}",method=RequestMethod.GET)
         public String checkUser(@PathVariable Integer productID) {
               if(productRepo.existsById(productID)) {
                     return "Record with product id:"+productID+" is existing..";
               }else {
                     return "Record with product id:"+productID+" does not exist..";
               }
         }
   //4. Count number of products
```

```
	@RequestMapping(value="/products/count",method=RequestMethod.GET)
	public String countUser() {
		long count=productRepo.count();
		if(count>0) {
			return "Number of records found:"+count;
		}else {
			return "No record found..";
		}

	}

	//5.to get user details by user id
	@RequestMapping(value="/product/{productID}", method=RequestMethod.GET)
	public Optional <Product> getUser(@PathVariable Integer productID){
		Optional<Product> user=productRepo.findById(productID);
		return user;
	}

		//6.get all products details
	@RequestMapping(value="/users",method=RequestMethod.GET)
	public List<Product> getAllUsers(){
		List<Product> list=(List<Product>) productRepo.findAll();
		return list;
	}

}
```

—----------

ProductRepository.java

—----------

```
package com.dxc.training.rest.dao;

import org.springframework.data.repository.CrudRepository;

import com.dxc.training.rest.model.Product;

public interface ProductRepository extends CrudRepository<Product,Integer> {

}
```

—----------

Product.java

—----------

```
package com.dxc.training.rest.model;
```

```
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name = "product")
public class Product {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Column(name = "productID")
    private Integer productID;

    @Column(name = "productName")
    private String productName;

    @Column(name = "totalStock")
    private Integer totalStock;

    @Column(name = "pricePerUnit")
    private Integer pricePerUnit;

    public Integer getProductID() {
        return productID;
    }

    public void setProductID(Integer productID) {
        this.productID = productID;
    }

    public String getProductName() {
        return productName;
    }

    public void setProductName(String productName) {
        this.productName = productName;
    }

    public Integer getTotalStock() {
        return totalStock;
```

```
	}

	public void setTotalStock(Integer totalStock) {
		this.totalStock = totalStock;
	}

	public Integer getPricePerUnit() {
		return pricePerUnit;
	}

	public void setPricePerUnit(Integer pricePerUnit) {
		this.pricePerUnit = pricePerUnit;
	}

	public Product() {
		super();
		// TODO Auto-generated constructor stub
	}

	public Product(Integer productID, String productName, Integer totalStock, Integer
pricePerUnit) {
		super();
		this.productID = productID;
		this.productName = productName;
		this.totalStock = totalStock;
		this.pricePerUnit = pricePerUnit;
	}

	@Override
	public String toString() {
		return "productID=" + productID + ", productName=" + productName + ", totalStock=" +
totalStock
				+ ", pricePerUnit=" + pricePerUnit;
	}



}
```

—----------

Pom.xml

—----------

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
```

```
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.6.7</version>
        <relativePath /> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.example</groupId>
    <artifactId>007.SpringBootWebRestApiWithDbAcess</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>004.Question4</name>
    <description>Demo project for Spring Boot</description>
    <properties>
        <java.version>11</java.version>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-data-jdbc</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-data-jpa</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-jdbc</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>

        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>
    </dependencies>

    <build>
```

```
        <plugins>
                <plugin>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-maven-plugin</artifactId>
                </plugin>
        </plugins>
    </build>

</project>

—----------------------------
```