Case Study 1: Online Food Ordering System

Java-Based Spring Configuration

1. Customer.java

```
public class Customer {
    private String name;
    private String contact;
    private String preferredCuisine;

    public Customer(String name, String contact, String preferredCuisine) {
        this.name = name;
        this.contact = contact;
        this.preferredCuisine = preferredCuisine;
    }

    public String getPreferredCuisine() {
        return preferredCuisine;
    }

    public String getName() {
        return name;
    }
}
```

2. Restaurant.java

```
public class Restaurant {
    private String name;
    private String location;
    private String[] availableCuisines;

    public Restaurant(String name, String location, String[] availableCuisines) {
        this.name = name;
        this.location = location;
        this.availableCuisines = availableCuisines;
    }

    public boolean servesCuisine(String cuisine) {
        for (String c : availableCuisines) {
            if (c.equalsIgnoreCase(cuisine)) return true;
        }
        return false;
    }

    public String getName() {
        return name;
    }
}
```

3. FoodOrderService.java

```
public class FoodOrderService {
    private Customer customer;
    private Restaurant restaurant;

    public FoodOrderService(Customer customer, Restaurant restaurant) {
        this.customer = customer;
        this.restaurant = restaurant;
    }

    public void placeOrder() {
        if (restaurant.servesCuisine(customer.getPreferredCuisine())) {
            System.out.println("Order placed successfully at " + restaurant.getName() + " for custome
        } else {
            System.out.println("No matching cuisine available for " + customer.getName());
        }
    }
}
```

4. AppConfig.java

```
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;

@Configuration
public class AppConfig {

    @Bean
    public Customer customer() {
        return new Customer("Alice", "1234567890", "Italian");
    }

    @Bean
    public Restaurant restaurant() {
        return new Restaurant("Foodie Palace", "Downtown", new String[]{"Italian", "Chinese"});
    }

    @Bean
    public FoodOrderService foodOrderService() {
        return new FoodOrderService(customer(), restaurant());
    }
}
```

5. MainApp.java

```
import org.springframework.context.ApplicationContext;
import org.springframework.context.annotation.AnnotationConfigApplicationContext;

public class MainApp {
    public static void main(String[] args) {
        ApplicationContext context = new AnnotationConfigApplicationContext(AppConfig.class);
        FoodOrderService service = context.getBean(FoodOrderService.class);
        service.placeOrder();
    }
}
```

Case Study 2: Smart Home Automation System

Annotation-Based Spring Configuration

1. User.java

```
import org.springframework.stereotype.Component;

@Component
public class User {
    private String name = "John";
    private String homeId = "HOME123";

    public String getName() {
        return name;
    }

    public String getHomeId() {
        return homeId;
    }
}
```

2. Device.java

```
import org.springframework.stereotype.Component;

@Component
public class Device {
```

```
    private String deviceType = "Light";
    private String status = "OFF";

    public void turnOn() {
        status = "ON";
        System.out.println(deviceType + " is turned ON.");
    }

    public void turnOff() {
        status = "OFF";
        System.out.println(deviceType + " is turned OFF.");
    }

    public String getStatus() {
        return status;
    }
}
```

3. AutomationService.java

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
public class AutomationService {

    @Autowired
    private User user;

    @Autowired
    private Device device;

    public void controlDevice() {
        System.out.println("Controlling device for user: " + user.getName());
        device.turnOn();
        System.out.println("Device status: " + device.getStatus());
    }
}
```

4. AppConfig.java

```
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;

@Configuration
@ComponentScan(basePackages = "com.example")
public class AppConfig {
}
```

5. MainApp.java

```
import org.springframework.context.ApplicationContext;
import org.springframework.context.annotation.AnnotationConfigApplicationContext;

public class MainApp {
    public static void main(String[] args) {
        ApplicationContext context = new AnnotationConfigApplicationContext(AppConfig.class);
        AutomationService service = context.getBean(AutomationService.class);
        service.controlDevice();
    }
}
```