

**ISTANBUL 29 MAYIS ÜNİVERSİTESİ**

**İKTİSADİ VE İDARİ BİLİMLER FAKÜLTESİ**

**YÖNETİM BİLİŞİM SİSTEMLERİ BÖLÜMÜ**

2024-2025 GÜZ DÖNEMİ

YBS203 NESNEYE YÖNELİK PROGRAMLAMA

ÖDEV KONUSU:WEB GELİŞTİRME

RAFİYA ZHORABEK 030423057

**Rapor: Framework, Tasarım Deseni ve Veritabanı Mimarileri**

**Giriş: Bu raporda, web geliştirme sürecinde sık kullanılan bir framework olan Google Web Toolkit (GWT), yazılım dünyasında nesneler arası ilişki yönetiminde kritik rol oynayan Observer Tasarım Deseni ve Google Cloud Platform’un bir hizmeti olan Google Cloud SQL mimarileri detaylı bir şekilde ele alınacaktır. Her biri için avantajlar, dezavantajlar, kullanım alanları ve kod örnekleri paylaşılacak, ardından bu mimarilerin birlikte kullanıldığı bir masaüstü uygulaması incelenecektir.**

**1. Google Web Toolkit (GWT)**

**Nedir?**

**Google Web Toolkit (GWT), Java tabanlı bir framework olup geliştiricilere dinamik ve etkileşimli web uygulamaları oluşturma imkânı tanır. Java kodları GWT Compiler tarafından tarayıcıda çalışabilen optimize edilmiş JavaScript kodlarına dönüştürülür.**

**Avantajları:**

**• Java bilgisiyle web geliştirme: GWT, yalnızca Java bilgisiyle, HTML, CSS ve JavaScript’i doğrudan öğrenmeye gerek kalmadan web uygulamaları oluşturmayı sağlar.**

**• Optimize edilmiş performans: GWT, performanslı ve sıkıştırılmış JavaScript kodu üretir.**

**• Hata ayıklama kolaylığı: Java IDE’lerinde kolayca hata ayıklama yapılabilir.**

**Dezavantajları:**

**• Yüksek öğrenme eğrisi: İlk başta framework’ü öğrenmek zaman alabilir.**

**• Modern kütüphane desteği: Angular, React gibi modern frameworklere göre kütüphane desteği sınırlıdır.**

**Kullanım Alanları:**

**GWT genellikle büyük ölçekli kurumsal web projelerinde, yüksek performans ve güvenlik gerektiren uygulamalarda tercih edilir.**

**Kod Örneği:**

**Aşağıdaki örnekte basit bir “Merhaba Dünya” web uygulaması geliştirilmiştir:**

**import com.google.gwt.core.client.EntryPoint;**

**import com.google.gwt.user.client.ui.Button;**

**import com.google.gwt.user.client.ui.Label;**

**import com.google.gwt.user.client.ui.RootPanel;**

**public class HelloWorld implements EntryPoint {**

**public void onModuleLoad() {**

**Button button = new Button("Tıkla!");**

**Label label = new Label();**

**button.addClickHandler(event -> label.setText("Merhaba Dünya!"));**

**RootPanel.get().add(button);**

**RootPanel.get().add(label);**

**}**

**}**

**Açıklama: Bu kod, bir butona tıklanıldığında ekranda “Merhaba Dünya!” mesajını görüntüler. RootPanel, GWT’de tarayıcıya HTML içeriği eklemek için kullanılan ana kapsayıcıdır.**

**2. Observer Tasarım Deseni**

**Nedir?**

**Observer, bir nesnede meydana gelen değişiklikleri diğer nesnelere bildiren bir tasarım desenidir. Publisher-Subscriber (yayıncı-abone) modeline benzer şekilde çalışır.**

**Avantajları:**

**• Gevşek bağlılık: Nesneler arasında bağımsız bir ilişki sağlar.**

**• Dinamik yapı: Değişikliklere kolayca adapte edilebilir.**

**Dezavantajları:**

**• Performans sorunları: Çok sayıda gözlemci olduğunda performans kayıpları yaşanabilir.**

**• Yönetim zorluğu: Karmaşık projelerde takip edilmesi zor olabilir.**

**Kullanım Alanları:**

**• Kullanıcı arayüzü güncellemeleri.**

**• Gerçek zamanlı veri akışı.**

**• Gözlemlenen verilerin sürekli değiştiği oyun projeleri.**

**Kod Örneği:**

**Aşağıdaki örnekte Observer Deseni kullanılarak bir konuya (subject) abone olan gözlemcilerin (observer) güncellemeleri alması sağlanmıştır:**

**import java.util.ArrayList;**

**import java.util.List;**

**interface Observer {**

**void update(String message);**

**}class ConcreteObserver implements Observer {**

**private String name;**

**public ConcreteObserver(String name) {**

**this.name = name }**

**@Override**

**public void update(String message) {**

**System.out.println(name + " mesaj aldı: " + mes}**

**}**

**class Subject {**

**private List<Observer> observers = new ArrayList<>();**

**public void addObserver(Observer observer) {**

**observers.add(observer);**

**}public void notifyObservers(String message) {**

**for (Observer observer : observers) {**

**observer.update(message); }**

**}**

**}public class Main {**

**public static void main(String[] args) {**

**Subject subject = new Subject();**

**Observer obs1 = new ConcreteObserver("Observer1");**

**Observer obs2 = new ConcreteObserver("Observer2");**

**subject.addObserver(obs1);**

**subject.**

**addObserver(obs2);**

**subject.notifyObservers("Herkese merhaba!");**

**}**

**}**

**Açıklama:**

**• Subject: Mesajları gözlemcilere ileten ana nesnedir.**

**• Observer: Mesajları dinleyen ve yanıt veren nesnelerdir.**

**• Bu kodda “Subject”, iki gözlemciye mesaj göndermiştir.**

**3. Google Cloud SQL**

**Nedir?**

**Google Cloud SQL, Google Cloud Platform’da (GCP) çalışan, yönetilen bir ilişkisel veritabanı hizmetidir. MySQL, PostgreSQL ve SQL Server desteği sunar.**

**Avantajları:**

**• Kolay yönetim: Otomatik yedekleme, ölçeklenebilirlik ve güvenlik sağlar.**

**• Güçlü erişim kontrolü: IP bazlı izinler ve kimlik doğrulama mekanizmaları sunar.**

**Dezavantajları:**

**• GCP bağımlılığı: Sadece Google Cloud üzerinde çalışır.**

**• Ücret: Yüksek ölçekli projelerde maliyet artabilir.**

**Kullanım Alanları:**

**Web, mobil ve IoT uygulamaları, veri analitiği projeleri.**

**Kod Örneği:**

**Aşağıdaki kod, Google Cloud SQL’e bağlanarak bir veritabanından veri çekmeyi gösterir:**

**import java.sql.Connection;**

**import java.sql.DriverManager;**

**import java.sql.ResultSet;**

**import java.sql.Statement;**

**public class CloudSqlExample {**

**public static void main(String[] args) {**

**String jdbcUrl = "jdbc:mysql://<INSTANCE\_CONNECTION\_NAME>/<DB\_NAME>";**

**String user = "<USER>";**

**String password = "<PASSWORD>";**

**try (Connection connection = DriverManager.getConnection(jdbcUrl, user, password)) {**

**Statement statement = connection.createStatement();**

**ResultSet resultSet = statement.executeQuery("SELECT \* FROM users");**

**while (resultSet.next()) {**

**System.out.println("Kullanıcı: " + resultSet.getString("name"));**

**}**

**} catch (Exception e) {**

**e.printStackTrace();**

**}**

**}**

**}**

**Açıklama:**

**• JDBC kullanılarak Google Cloud SQL’e bağlanılmıştır.**

**• Kullanıcı tablosundan veri çekilmiştir.**

**4. Seçilen Proje: Kütüphane Yönetim Sistemi**

**Proje Açıklaması:**

**Kütüphane Yönetim Sistemi, kullanıcıların kitap araması, ödünç kitap alması ve kitap bilgilerini görüntülemesi için tasarlanmıştır. Bu projede:**

**• GWT: Web arayüzü için kullanılmıştır. Örneğin, kitap listelerini dinamik olarak yüklemek için.**

**• Observer: Kitap bilgilerinde yapılan değişikliklerin kullanıcıya anında yansıtılması için kullanılmıştır.**

**• Google Cloud SQL: Kitap ve kullanıcı verilerinin saklanması için kullanılmıştır.**

**Mimarilerin Katkıları:**

**• GWT: Hızlı ve etkileşimli bir kullanıcı deneyimi sunar.**

**• Observer: Gerçek zamanlı veri senkronizasyonu sağlar.**

**• Google Cloud SQL: Verilerin güvenilir bir şekilde saklanmasını ve yönetilmesini sağlar.**