

SUNAN: AKIN İNCECİK

YAZILIM PROJE *Portföyü*



SUNAN: AKIN İNCECİK

Sunum İçeriği

| | |
|----------|--------------------------------------|
| 1 | Crm Demo Project |
| 2 | Database Listener Service (RabbitMQ) |
| 3 | RabbitMQ Listener Service |
| 4 | MVC With AdminLTE (IIS) |
| 5 | Angular Web Project |

| | |
|-----------|---------------------|
| 6 | React Web Project |
| 7 | Vue Web Project |
| 8 | Wordpress Project |
| 9 | Android Notepad App |
| 10 | Python Web Project |

SUNAN: AKIN İNCECİK

Crm Demo Project

ÖĞRENCİ KAYITLARINI YÖNETMEK
İÇİN GELİŞTİRDİĞİM PROJE,
KULLANICIDAN TEMEL BİLGİLERİN
YANI SIRA ÖĞRENCİNİN BİLDİĞİ VE
ÖĞRENMEK İSTEDİĞİ DİLLERİ DE
ALACAK ŞEKİLDE TASARLANDI. KAYIT
SONRASI E-POSTA ADRESLERİ
VERİTABANINA KAYDEDİLİYOR VE
OTOMATİK E-POSTA GÖNDERİMİNDE
KULLANILIYOR. KULLANICI ARAYÜZÜ
VE VERİTABANI ENTEGRASYONU TAM
ENTEGRE ÇALIŞMAKTADIR.



Welcome

User Name: test

Password: test123

Personal Records

Personal Informations

| | |
|------------------|--|
| ID Number | 111111111111 |
| Name | aaaa |
| Surname | bbbb |
| Date of Birth | 29-02-2024 |
| Gender | <input type="radio"/> Not Given <input checked="" type="radio"/> Male <input type="radio"/> Female |
| Main Language | Spanish |
| Foreign Language | Seçiniz English Spanish |
| Description | cccccccc |

Contact Informations

| | | |
|--------|-----------------------|---------|
| Phone | 05344509536 | Work |
| E-mail | akinincecik@gmail.com | Seçiniz |
| Adress | eeee | Seçiniz |

Contact **Type** **Value**

| | | |
|-------|------|-------------|
| Phone | Work | 05344509536 |
|-------|------|-------------|

Send Email
akinsendmail@gmail.com

SUNAN: AKIN İNCECİK

Crm Demo Kodları

CRMDEMO, ÖĞRENCİ KAYITLARI VE E-POSTA GÖNDERİMİ İŞLEMLERİNİ YÖNETEN DİNAMİK BİR CRM DEMO UYGULAMASIDIR. KULLANICILAR GİRİŞ YAPTIKTAN SONRA ÖĞRENCİ BİLGİLERİ KAYDEDEBİLİR; DOĞRULAMA KURALLARI İLE KONTROLLER YAPILIR VE VERİLER MSSQL İLE POSTGRESQL VERİTABANLARINDA SAKLANABİLİR.

PROJEDE:

- GÜVENLİ VERİ BAĞLANTISI:** ŞİFRELENMİŞ BAĞLANTI STRING'LERİ KULLANILARAK GÜVENLİ BİR VERİ BAĞLANTISI SAĞLANIR.
- ASENKRON E-POSTA GÖNDERİMİ:** RABBITMQ İLE E-POSTA GÖNDERİMLERİ KUYRUKLANARAK SERVİSLE ASENKRON BİR ŞEKİLDE YAPILIR.
- E-POSTA İÇERİĞİ:** E-POSTA İÇERİĞİ, JSON VE HTML FORMATLARINDA SAKLANABİLİR.
- MODÜLER YAPI:** MODÜLER SQLUTILS YAPISI İLE VERİTABANI İŞLEMLERİ AYRIŞTIRILMIŞ VE SADELEŞTİRİLMİŞTİR.

ESNEK CRM ORTAMI: BU YAPI SAYESİNDE HEM ESNEK HEM DE GENİŞLETİLEBİLİR BİR CRM DEMO ORTAMI SAĞLANMAKTADIR.

```
public class SQLUtils
{
    public static class DatabaseHelper
    {
        private static readonly Dictionary<string, Func<string, IDbConnection>> _connectionFactories =
            new Dictionary<string, Func<string, IDbConnection>>
        {
            { "MsSqlConnection", connStr => new SqlConnection(connStr) },
            { "PostgreSqlConnection", connStr => new NpgsqlConnection(connStr) }
        };

        public static string GetConnectionString(string databaseType)
        {
            if (string.IsNullOrWhiteSpace(databaseType))
                throw new ArgumentException("Veritabanı tipi boş olamaz.", nameof(databaseType));

            // Bağlantı dizesini al
            string encryptedConnectionString = ConfigurationManager.ConnectionStrings[databaseType].ConnectionString;
            if (string.IsNullOrEmpty(encryptedConnectionString))
                throw new Exception($"Bağlantı dizesi bulunamadı: {databaseType}");

            // Şifre çözme işlemi
            try
            {
                return SecurityHelper.Decrypt(encryptedConnectionString);
            }
            catch (Exception ex)
            {
                throw new Exception($"Bağlantı dizesi çözülemedi: {ex.Message}");
            }
        }

        public static IDbConnection CreateConnection(string databaseType)
        {
            if (!_connectionFactories.TryGetValue(databaseType, out var connectionFactory))
                throw new ArgumentException($"Geçersiz veritabanı tipi: {databaseType}");

            return connectionFactory(GetConnectionString(databaseType));
        }

        public static void ManageConnectionState(IDbConnection connection, bool open)
        {
            if (connection == null) throw new ArgumentNullException(nameof(connection));

            try
            {
                if (open && connection.State != ConnectionState.Open)
                    connection.Open();
                else if (!open && connection.State != ConnectionState.Closed)
                    connection.Close();
            }
            catch (Exception ex)
            {
                throw new Exception($"Bağlantı yönetimi hatası: {ex.Message}");
            }
        }
    }
}
```

```
namespace CRMDemo
{
    6 references
    public class SQLUtils
    {
        10 references
        public static class DatabaseHelper
        {
            1 reference
            public static Dictionary<string, ValidationRule> GetValidationRulesFromDb(string databaseType)
        }

        2 references
        public static class QueryHelper
        {
        }

        1 reference
        public static class AccountRepository
        {
        }

        4 references
        public static class DatabaseExecutor
        {
        }

        private void SaveStudentInformation()
        {
            if (!ValidationUtils.ValidateForm(this, cb_SendMail, skipInfoMailCheck: true))
                return;

            if (!ValidateAndPrepareForm(out var studentParams))
                return;

            if (!cb_SendMail.Checked && !MessageUtils.ConfirmNoInfoMail())
            {
                cb_SendMail.Checked = true;
                tb_InfoMail.Enabled = true;
                tb_InfoMail.Focus();
                return;
            }

            List<string> databases = new List<string> { "MsSqlConnection" /*, "PostgreSqlConnection" */ };
            int lastInsertedId = -1;

            foreach (var db in databases)
            {
                // ☑ Öğrenci verisini fotoğraf ile birlikte her iki veritabanına ekle
                int insertedId = StudentRepository.InsertStudentInformation(db, studentParams);

                if (db == "MsSqlConnection")
                {
                    lastInsertedId = insertedId; // MSSQL'deki ID'yi referans al
                }

                Console.WriteLine($"* Kaydedilen StudentID ({db}): {insertedId}");
            }

            lastInsertedStudentId = lastInsertedId; // Global değişkeni güncelle
            Console.WriteLine($"* Global StudentID: {lastInsertedStudentId}");

            if (cb_SendMail.Checked)
            {
                if (!ValidateAndPrepareEmail(out var emailParams))
                    return;

                foreach (var db in databases)
                {
                    EmailRepository.InsertEmailInformation(db, emailParams);
                }
            }

            MessageBox.Show("☑ Kayıt işlemi her iki veritabanı için başarılı.");
        }
    }
}
```

SUNAN: AKIN İNCECİK

Crm Demo Eş Zamanlı Veri Kaydı ve Mail Hazırlığı

FORMDAN GELEN VERİLER, AYNI ANDA HEM MSSQL HEM DE POSTGRESQL VERİTABANLARINA KAYDEDİLİR. BU İŞLEM SAYESİNDE İKİ FARKLI SİSTEMDE VERİ BÜTÜNLÜĞÜ KORUNUR VE SENKRON ÇALIŞABİLİRLİK SAĞLANIR.

The screenshot shows the pgAdmin 4 interface with several database connections open. On the left, the 'Library' sidebar lists various hosts and databases. In the center, a query editor window displays a simple SELECT statement. Below it, a 'Data Output' window shows the results of the query, listing 146 rows of student information. The bottom of the screen shows a taskbar with various application icons.

The screenshot shows the SSMS interface with a query window containing a SELECT statement. The results grid displays the same 146 rows of student information as the pgAdmin output. The columns include Id, FullName, FullSurname, DateOfBirth, MainLanguage, ForeignLanguage, ForeignLanguageLevel, Description, PhoneNumber, Email, Adress, and PicturePath.

| Id | FullName | FullSurname | DateOfBirth | MainLanguage | ForeignLanguage | ForeignLanguageLevel | Description | PhoneNumber | Email | Adress | PicturePath | |
|-----|----------|-------------|-------------|-------------------------|-----------------|----------------------|-------------|-------------|-------------|-----------------------|-------------|----------------------|
| 724 | 3... | aaaa | bbbb | 2024-02-29 00:00:00.000 | English | Spanish | 0 | cccccccc | 05344508315 | akinincecik@gmail.com | eeee | 0xFFD8FFE000104A4649 |
| 725 | 3... | aaaa | bbbb | 2024-02-29 00:00:00.000 | English | Spanish | 0 | cccccccc | 05344508315 | akinincecik@gmail.com | eeee | 0xFFD8FFE000104A4649 |
| 726 | 3... | aaaa | bbbb | 2024-02-29 00:00:00.000 | English | Spanish | 0 | cccccccc | 05344508315 | akinincecik@gmail.com | eeee | 0xFFD8FFE000104A4649 |
| 727 | 3... | aaaa | bbbb | 2024-02-29 00:00:00.000 | English | Spanish | 0 | cccccccc | 05344508315 | akinincecik@gmail.com | eeee | 0xFFD8FFE000104A4649 |
| 728 | 3... | aaaa | bbbb | 2024-02-29 00:00:00.000 | English | Spanish | 0 | cccccccc | 05344508315 | akinincecik@gmail.com | eeee | 0xFFD8FFE000104A4649 |
| 729 | 3... | aaaa | bbbb | 2024-02-29 00:00:00.000 | English | Spanish | 0 | cccccccc | 05344508315 | akinincecik@gmail.com | eeee | 0xFFD8FFE000104A4649 |
| 730 | 3... | aaaa | bbbb | 2024-02-29 00:00:00.000 | English | Spanish | 0 | cccccccc | 05344508315 | akinincecik@gmail.com | eeee | 0xFFD8FFE000104A4649 |
| 731 | 3... | aaaa | bbbb | 2024-02-29 00:00:00.000 | English | Spanish | 0 | cccccccc | 05344508315 | akinincecik@gmail.com | eeee | 0xFFD8FFE000104A4649 |
| 732 | 3... | aaaa | bbbb | 2024-02-29 00:00:00.000 | English | Spanish | 0 | cccccccc | 05344508315 | akinincecik@gmail.com | eeee | 0xFFD8FFE000104A4649 |
| 733 | 3... | aaaa | bbbb | 2024-02-29 00:00:00.000 | English | Spanish | 0 | cccccccc | 05344508315 | akinincecik@gmail.com | eeee | 0xFFD8FFE000104A4649 |
| 734 | 3... | aaaa | bbbb | 2024-02-29 00:00:00.000 | English | Spanish | 0 | cccccccc | 05344508315 | akinincecik@gmail.com | eeee | 0xFFD8FFE000104A4649 |
| 735 | 3... | aaaa | bbbb | 2024-02-29 00:00:00.000 | English | Spanish | 0 | cccccccc | 05344508315 | akinincecik@gmail.com | eeee | 0xFFD8FFE000104A4649 |
| 736 | 3... | aaaa | bbbb | 2024-02-29 00:00:00.000 | English | Spanish | 0 | cccccccc | 05344508315 | akinincecik@gmail.com | eeee | 0xFFD8FFE000104A4649 |
| 737 | 3... | aaaa | bbbb | 2024-02-29 00:00:00.000 | English | Spanish | 0 | cccccccc | 05344508315 | akinincecik@gmail.com | eeee | 0xFFD8FFE000104A4649 |
| 738 | 3... | aaaa | bbbb | 2024-02-29 00:00:00.000 | English | Spanish | 0 | cccccccc | 05344508315 | akinincecik@gmail.com | eeee | 0xFFD8FFE000104A4649 |

KAYDEDİLEN VERİLER, E-POSTA GÖNDERİMİNE HAZIR OLACAK ŞEKLİDE “QUEUED” (KUYRUKTA) DURUMUyla SAKLANIR. ARKA PLANDA ÇALIŞAN SERVİS BU VERİLERİ ALIR, UYGUN FORMATTAA E-POSTA HALİNE GETİRİR VE GÖNDERİM SÜRECİNİ OTOMATİK OLARAK BAŞLATIR.

Database Listener Service

BU SERVİS, CRM DEMO UYGULAMASI TARAFINDAN KAYDEDİLEN VERİLERİN DURUMLARINI ANALİZ EDER. MAIL GÖNDERİMİNE HAZIR OLAN KAYITLARI TESPİT EDEREK, BU VERİLERİ RABBITMQ KUYRUĞUNA İLETİR. BÖYLECE, GÖNDERİM SÜRECİ İÇİN GEREKLİ OLAN VERİLER OTOMATİK OLARAK BİR SONRAKİ AŞAMAYA AKTARILMIŞ OLUR.

```

1 reference
public static void ProcessEmailQueue()
{
    Console.WriteLine($"⌚ Yeni kayıtlar kontrol ediliyor...");
    // StatusId = 0 olan e-mailleri al
    DataTable newEmails = SQLUtils.GetSqlQueuedEmails();

    foreach (DataRow row in newEmails.Rows)
    {
        int emailId = Convert.ToInt32(row["Id"]);

        // Her bir emailId için detaylı veriyi al
        DataTable emailDetails = SQLUtils.GetEmailFromId(emailId);

        if (emailDetails.Rows.Count > 0)
        {
            // Veritabanından gelen veriyi uygun formata getir
            var emailData = emailDetails.Rows[0];

            var emailParams = new Dictionary<string, object>
            {
                { "@Recipient", emailData["Recipient"] },
                { "@Subject", emailData["Subject"] },
                { "@Body", emailData["Body"] },
                { "@JsonData", emailData["JsonData"] }, // JSON formatındaki veri
                { "@HtmlData", emailData["HtmlData"] }, // HTML formatındaki veri
                { "@EmailId", emailData["Id"] } // EmailId parametresi
            };

            // JSON'a çevir
            string jsonData = JsonConvert.SerializeObject(emailParams);

            try
            {
                // RabbitMQ'ya gönder
                RabbitMQHelper.SendMessage(jsonData);

                // Başarıyla RabbitMQ'ya gönderildiye durumu 6 (Sent) yap
                SQLUtils.UpdateSqlEmailStatus(emailId, 6);
                Console.WriteLine($"✉️ Email ID {emailId} RabbitMQ'ya başarıyla gönderildi");
            }
            catch (Exception ex)
            {
                // RabbitMQ'ya gönderilemediye durumu 4 (Failed) yap
                SQLUtils.UpdateSqlEmailStatus(emailId, 4);
                Console.WriteLine($"✖️ Email ID {emailId}: RabbitMQ gönderimi başarısız oldu");
            }
        }
    }
}

```

```

namespace WindowsServiceTest2
{
    1 reference
    public class Worker
    {
        private static Object _lock = new Object();
        private static DatabaseParameters _dbParams = new DatabaseParameters(); // Bağlantı bilgilerini merkezi olarak tutuyoruz

        1 reference
        public static void testMethod(object arg)
        {
            ThreadParameters threadParameters = (ThreadParameters)arg;
            List<string> databases = new List<string> { "SQL/*", "PostgreSQL"/* }; // Veritabanı türlerini listeye alıyoruz

            while (true)
            {
                try
                {
                    lock (_lock)
                    {
                        foreach (var dbType in databases)
                        {
                            EmailQueueService.ProcessEmailQueue();
                            EmailQueueService.RetryFailedEmails(dbType);
                        }
                    }
                }
                catch (Exception ex)
                {
                    LogError(ex);
                }
                Thread.Sleep(1000);
            }
        }

        1 reference
        private static void LogError(Exception ex)
        {
            using (EventLog eventLog = new EventLog("Application"))
            {
                eventLog.Source = "Application";
                eventLog.WriteEntry("WindowsServiceTest Error: " + ex.StackTrace, EventLogEntryType.Error, 101, 1);
            }
        }
    }
}

```

| Overview | | Messages | | Message rates | | | | | | |
|--------------|-------------|----------|----------|---------------|-------|---------|-------|----------|---------------|--------|
| Virtual host | Name | Type | Features | State | Ready | Unacked | Total | incoming | deliver / get | ack |
| / | emailQueue | classic | Args | idle | 0 | 0 | 0 | | 0.00/s | 0.00/s |
| / | email_queue | classic | D Args | idle | 1 | 0 | 1 | 0.00/s | 0.00/s | 0.00/s |

SUNAN: AKIN İNCECİK

RabbitMQ Listener Service

BU SERVİS, RABBİTMQ KUYRUĞUNU SÜREKLİ DİNLEYEREK YENİ BİR VERİ OLUP OLMADIĞINI KONTROL EDER. KUYRUKTA BİR VERİ BULUNDUĞUNDA, VERİTABANINDA KAYITLI OLAN UYGUN E-POSTA ŞABLONUNU ALIR VE İLGİLİ VERİLERLE BİRLEŞTİREREK MAIL GÖNDERİMİNİ GERÇEKLEŞTİRİR.

```
public static class EmailSender
{
    public static void ProcessAndSendEmails(Dictionary<string, object> emailData, string dbType)
    {
        try
        {
            if (!IsValidEmailData(emailData, out int emailId, out string recipient, out string subject, out string body, out string jsonData))
            {
                UpdateEmailStatus(dbType, emailId, 5); // JSON Hatalı (Failed)
                return;
            }

            string finalBody = PrepareEmailBody(dbType, emailId, body, jsonData);

            if (string.IsNullOrWhiteSpace(finalBody))
            {
                Console.WriteLine($"⚠️ Email ID {emailId}: HTML tablo oluşturulamadı.");
                UpdateEmailStatus(dbType, emailId, 5);
                return;
            }

            if (SendEmailWithRetry(recipient, subject, finalBody))
            {
                Console.WriteLine($"✉️ Email ID {emailId} başarıyla gönderildi.");
                UpdateEmailStatus(dbType, emailId, 2); // Sent
            }
            else
            {
                Console.WriteLine($"✖️ Email ID {emailId} gönderilmedi. Retrying...");
                UpdateEmailStatus(dbType, emailId, 4); // Retrying
            }
        }
        catch (Exception ex)
        {
            Console.WriteLine($"✖️ Email işlenirken hata oluştu: {ex.Message}");
        }
    }

    private static bool IsValidEmailData(Dictionary<string, object> emailData, out int emailId, out string recipient, out string subject, out string body, out string jsonData)
    {
        emailId = emailData.ContainsKey("@EmailId") ? Convert.ToInt32(emailData["@EmailId"]) : 0;

        recipient = emailData.TryGetValue("@Recipient", out var recipientObj) ? recipientObj?.ToString() ?? "" : "";
        subject = emailData.TryGetValue("@Subject", out var subjectObj) ? subjectObj?.ToString() ?? "" : "";
        body = emailData.TryGetValue("@Body", out var bodyObj) ? bodyObj?.ToString() ?? "" : "";
        jsonData = emailData.TryGetValue("@jsonData", out var jsonObj) ? jsonObj?.ToString() ?? "" : "";

        if (string.IsNullOrWhiteSpace(recipient) || string.IsNullOrWhiteSpace(subject) || string.IsNullOrWhiteSpace(body) || string.IsNullOrWhiteSpace(jsonData))
        {
            Console.WriteLine($"⚠️ Email ID {emailId}: Eksik email alanları tespit edildi.");
            return false;
        }

        if (!Utils.IsValidJson(jsonData))
        {
            Console.WriteLine($"⚠️ Email ID {emailId}: JSON verisi hatalı.");
            return false;
        }

        return true;
    }

    private static string PrepareEmailBody(string dbType, int emailId, string body, string jsonData)
    {
        int studentId = GetStudentIdFromJson(jsonData);
        string picturePath = SQLUtils.GetStudentPicturePath(dbType, studentId);
        string htmlData = Utils.GenerateHTMLTable(jsonData);

        if (!string.IsNullOrWhiteSpace(picturePath))
        {
            body += $"{<br><br><img src='{picturePath}' width='200' height='200' />}";
        }

        if (string.IsNullOrWhiteSpace(htmlData))
        {
            Console.WriteLine($"⚠️ Email ID {emailId}: HTML tablo oluşturulamadı.");
            return null;
        }

        return body + "<br><br>" + htmlData;
    }

    private static bool SendEmailWithRetry(string recipient, string subject, string finalBody, int retryCount = 3)
    {
        for (int i = 0; i < retryCount; i++)
        {
            string result = Utils.SendEmail(recipient, subject, finalBody);

            if (result == "OK")
                return true;

            Console.WriteLine($"⚠️ Email gönderme denemesi başarısız (Deneme {i + 1}/{retryCount})");
            Thread.Sleep(2000); // 2 saniye bekleme
        }

        return false;
    }
}
```

akincicek@gmail.com
to me
Test

ÖĞRENCİ KAYIT BİLGİLERİ

Merhaba, aşağıda öğrenci bilgileri bulunmaktadır.

| KİŞİSEL BİLGİLER | |
|----------------------|---------------------|
| TC Kimlik No | 1111111111 |
| Ad | aaaa |
| Soyad | bbbb |
| Dogum Tarihi | 2024-02-29T00:00:00 |
| Ana Dil | Spanish |
| Yabancı Dil | Seçiniz |
| Yabancı Dil Seviyesi | |
| Telefon Numarası | |
| E-Mail | |
| Adres | |

```
public class RabbitMQListener
{
    private static IConnection _connection;
    private static IModel _channel;

    private readonly string _queueName = "email_queue";
    private readonly string _hostName = "localhost"; // RabbitMQ sunucusu adresi
    private readonly string _username = "guest";
    private readonly string _password = "guest";

    public void StartListening(string dbType)
    {
        try
        {
            var factory = new ConnectionFactory()
            {
                HostName = _hostName,
                UserName = _username,
                Password = _password
            };

            _connection = factory.CreateConnection();
            _channel = _connection.CreateModel();

            _channel.QueueDeclare(queue: _queueName, durable: true, exclusive: false, autoDelete: false, arguments: null);
            var consumer = new EventingBasicConsumer(_channel);

            consumer.Received += (model, ea) =>
            {
                try
                {
                    var body = ea.Body.ToArray();
                    var message = Encoding.UTF8.GetString(body);
                    Console.WriteLine($"✉️ [+] Received message: {message}");

                    var emailData = JsonConvert.DeserializeObject<Dictionary<string, object>>(message);

                    if (emailData != null && emailData.ContainsKey("@EmailId"))
                    {
                        int emailId = Convert.ToInt32(emailData["@EmailId"]);
                        Console.WriteLine($"✉️ [+] İşlem başlatılıyor, Email ID: {emailId}");

                        EmailSender.ProcessAndSendEmails(emailData, dbType);
                    }
                    else
                    {
                        Console.WriteLine("⚠️ Geçersiz mesaj formatı.");
                    }
                }
                catch (Exception ex)
                {
                    Console.WriteLine($"✖️ Hata: {ex.Message}");
                }
            };
        }
        catch (Exception ex)
        {
            Console.WriteLine($"✖️ Hata: {ex.Message}");
        }
    }

    // Mesajı onayla
    _channel.BasicAck(ea.DeliveryTag, false);

    _channel.BasicConsume(queue: _queueName, autoAck: false, consumer: consumer);
    Console.WriteLine("✉️ [+] Kuyruk dinleniyor...");

    // Thread'i kilitleden açık tutuyoruz
    Task.Delay(-1).Wait(); // ✅ Sonsuz döngü yerine Task.Delay kullanıyoruz
}

class Worker
{
    private static Object _lock = new Object();
    private static DatabaseParameters _dbParams = new DatabaseParameters(); // ✅ Bağlantı bilgilerini merkezi olarak çekiyoruz.

    public static void testMethod(object arg)
    {
        ThreadParameters threadParameters = (ThreadParameters)arg;
        List<string> databases = new List<string> { "SQL/*", "PostgreSQL/*" }; // ✅ Veritabanı türlerini listeye alıyoruz.

        while (true)
        {
            try
            {
                lock (_lock)
                {
                    foreach (var dbType in databases)
                    {
                        RabbitMQListener subscriber = new RabbitMQListener();
                        subscriber.StartListening(dbType);
                    }
                }
            }
            catch (Exception ex)
            {
                LogError(ex);
            }
            Thread.Sleep(20);
        }
    }

    private static void LogError(Exception ex)
    {
        using (EventLog eventLog = new EventLog("Application"))
        {
            eventLog.Source = "Application";
            eventLog.WriteEntry($"WindowsServiceTest Error: {ex.StackTrace}, EventLogEntryType.Error, 101, 1");
        }
    }
}
```

SUNAN: AKIN İNCECİK

MVC With IIS

BU PROJE, ADMINLTE ŞABLONUNU KULLANARAK OLUŞTURULMUŞ MVC TABANLI BİR ADMIN PANELİ UYGULAMASIDIR. KULLANICI DOSTU ARAYÜZÜ İLE VERİMLİ YÖNETİM SAĞLAR VE IIS ÜZERİNDE GERÇEK ORTAMDA ÇALIŞABİLİR. MODERN VE RESPONSİVE TASARIMIYLA VERİ YÖNETİMİ, KULLANICI ETKİLEŞİMİ GİBİ TEMEL İŞLEVLERİ SUNAR. AYRICA, ADMINLTE'NİN ESNEKLİK VE ÖZELLEŞTİRİLEBİLİRLİK ÖZELLİKLERİ SAYESİNDE KOLAYCA GENİŞLETİLEBİLİR.

```
namespace AdminLTE.Controllers
{
    [Authorize(Roles = "admin", AuthenticationSchemes = CookieAuthenticationDefaults.AuthenticationScheme)]
    public class SupplierController : Controller
    {
        private readonly DataBaseContext _dataBaseContext;
        private readonly IMapper _mapper;
        private readonly IHasher _hasher;

        public SupplierController(DataBaseContext dataBaseContext, IMapper mapper, IHasher hasher)
        {
            _dataBaseContext = dataBaseContext;
            _mapper = mapper;
            _hasher = hasher;
        }

        public IActionResult Index()
        {
            ViewBag.CurrentPage = 1;
            ViewBag.ActiveMenu = "supplier-management"; // Açık kalmasını istediğiniz menü

            return View();
        }

        public IActionResult SupplierListPartial()
        {
            List<SupplierViewModel> suppliers =
                _dataBaseContext.Suppliers.ToList()
                .Select(x => _mapper.Map<SupplierViewModel>(x)).ToList();

            return PartialView("_SupplierListPartial", suppliers);
        }

        public IActionResult SupplierListPartial(int page = 1)
        {
            Console.WriteLine($"Gelen Sayfa: {page}");

            ViewBag.CurrentPage = page;

            const int pageSize = 25;
            int skipCount = (page - 1) * pageSize;
            Console.WriteLine($"Şu anki sayfa: {page}, Atlanacak veri: {skipCount}");

            var suppliers = _dataBaseContext.Suppliers
                .OrderByDescending(s => s.Id)
                .Skip(skipCount)
                .Take(pageSize)
                .Select(s => _mapper.Map<SupplierViewModel>(s))
                .ToList();

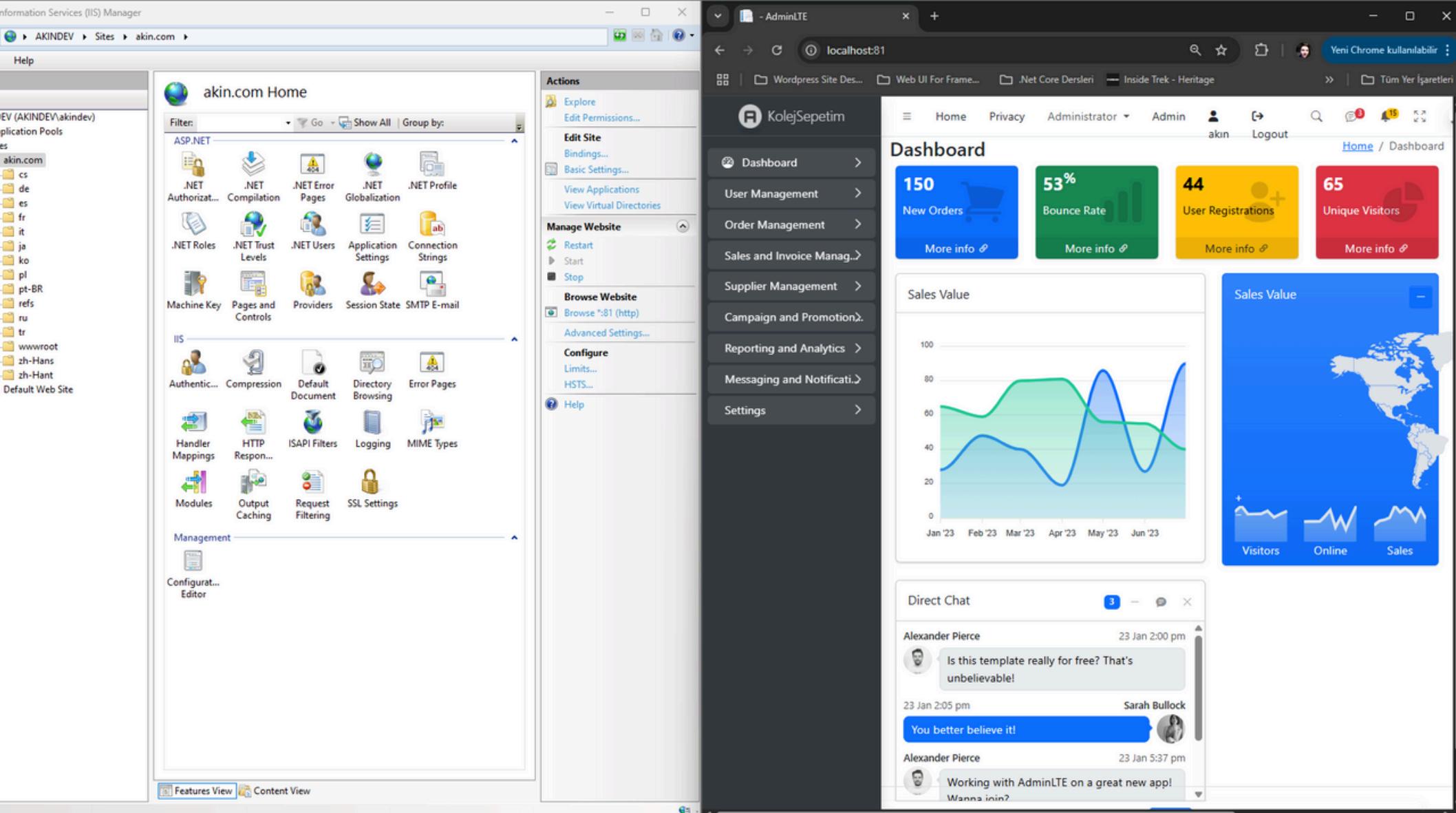
            Console.WriteLine($"Sayfa: {page}, Skip: {skipCount}, Toplam Çekilen: {suppliers.Count}");

            var totalSuppliers = _dataBaseContext.Suppliers.Count();
            var totalPages = (int)Math.Ceiling(totalSuppliers / (double)pageSize);

            Console.WriteLine($"Toplam: {totalSuppliers}, Toplam Sayfa: {totalPages}");

            var model = new SupplierListViewModel
            {
                Suppliers = suppliers,
                TotalPages = totalPages
            };
            return PartialView("_SupplierListPartial", model);
        }
    }
}
```

MODERN VE RESPONSİVE TASARIMIYLA VERİ YÖNETİMİ, KULLANICI ETKİLEŞİMİ GİBİ TEMEL İŞLEVLERİ SUNAR. AYRICA, ADMINLTE'NİN ESNEKLİK VE ÖZELLEŞTİRİLEBİLİRLİK ÖZELLİKLERİ SAYESİNDE KOLAYCA GENİŞLETİLEBİLİR.



| Supplier ID | Supplier Name | Address | Phone Number | Email | Created At |
|-------------|---------------|----------------------|--------------|---------------------|------------------|
| 79 | Supplier 79 | Street 79, City, USA | +1-555-1079 | supplier79@mail.com | 2025-01-07 23:51 |
| 78 | Supplier 78 | Street 78, City, USA | +1-555-1078 | supplier78@mail.com | 2025-01-08 23:51 |
| 77 | Supplier 77 | Street 77, City, USA | +1-555-1077 | supplier77@mail.com | 2025-01-09 23:51 |
| 76 | Supplier 76 | Street 76, City, USA | +1-555-1076 | supplier76@mail.com | 2025-01-10 23:51 |
| 75 | Supplier 75 | Street 75, City, USA | +1-555-1075 | supplier75@mail.com | 2025-01-11 23:51 |
| 74 | Supplier 74 | Street 74, City, USA | +1-555-1074 | supplier74@mail.com | 2025-01-12 23:51 |
| 73 | Supplier 73 | Street 73, City, USA | +1-555-1073 | supplier73@mail.com | 2025-01-13 23:51 |
| 72 | Supplier 72 | Street 72, City, USA | +1-555-1072 | supplier72@mail.com | 2025-01-14 23:51 |
| 71 | Supplier 71 | Street 71, City, USA | +1-555-1071 | supplier71@mail.com | 2025-01-15 23:51 |

Angular Web Project

ANGULAR-FIRST-APP PROJESİ, ANGULAR FRAMEWORK'Ü KULLANARAK GELİŞTİRİLMİŞ BASIT BİR BLOG SAYFASIDIR. BU PROJE, ANGULAR'IN TEMEL ÖZELLİKLERİNİ ÖĞRENMEK VE UYGULAMAK AMACIYLA OLUŞTURULMUŞTUR. BİLEŞEN TABANLI MİMARİ VE VERİ BAĞLAMA (DATA BINDİNG) GİBİ ANGULAR'IN GÜÇLÜ YÖNLERİ KULLANARAK MODÜLER BİR YAPI OLUŞTURULMUŞTUR. PROJE, TYPESCRIPT, HTML VE CSS İLE GELİŞTİRİLMİŞ OLUP, TEMEL BİLEŞENLER ARASINDA YÖNLENDİRME, KULLANICI ETKİLEŞİMİ VE SERVİS ENTEGRASYONU GİBİ ÖNEMLİ ANGULAR KAVRAMLARINI İÇERMEKTEDİR.

The screenshot displays a developer's environment with two main windows. On the left, the Visual Studio Code (VS Code) interface is shown, featuring an Explorer sidebar with project files like 'FIRSTAPP', 'src/app/home/home.component.spec.ts', and 'src/app/app.component.ts'. The main editor area contains the 'home.component.spec.ts' file, which includes code for testing the HomeComponent. Below the editor is the Terminal pane, showing the command 'ng serve --open' being run, and the resulting output of the build process, including the generation of polyfills.js, main.js, and styles.css files. On the right, a web browser window titled 'Applications1' shows the 'BlogApp' application. The page has a header with 'BlogApp' and navigation links. It features a sidebar with 'POPÜLER KATEGORİLER' (Technology, Health, Travel, Food) and 'SON YORUMLAR' (John Doe: "Çok güzel bir yazı!", Alice: "Bu konuda daha fazla bilgi verir misiniz?"). The main content area is titled 'Blog Yazılıları' and includes a search form for 'Başlık' and 'İçerik'.

React Web Project

REACT-FIRST-APP PROJESİ, REACT FRAMEWORK'Ü KULLANILARAK GELİŞTİRİLMİŞ BASIT BİR WEB UYGULAMASIDIR. BU PROJE, REACT'İN TEMEL KAVRAMLARINI ÖĞRENMEK VE UYGULAMAK AMACIYLA OLUŞTURULMUŞTUR. UYGULAMA, BİLEŞEN TABANLI YAPIMI KULLANARAK MODÜLER BİR TASARIM SUNMAKTA, REACT'İN GÜÇLÜ ÖZELLİKLERİ OLAN DURUM YÖNETİMİ VE KULLANICI ETKİLEŞİMLERİNİ VERİMLİ BİR ŞEKİLDE YÖNETMEKTEDİR. PROJE, REACT'İN TEMEL ÖZELLİKLERİ OLAN USESTATE, USEEFFECT VE EVENT HANDLING GİBİ TEMEL İŞLEVLERİN NASIL ÇALIŞTIĞINI ANLAMAYA YÖNELİK BİR ÖRNEK TEŞKİL ETMEKTEDİR. AYRICA, MODERN JAVASCRIPT VE JSX KULLANILARAK DİNAMİK VE ETKİLEŞİMLİ BİR KULLANICI ARAYÜZÜ SAĞLANMIŞTIR.

The screenshot illustrates a development environment for a React web project. On the left, a code editor shows the `Home.js` file from the `src/pages` directory. The code uses `useState` to manage blog posts and `useEffect` to fetch them. The browser window on the right displays the application at `localhost:3000`, titled "My Blog". It features a sidebar with sections for "Günlük Haberler" (with a list of three bullet points), "Yeni Blog Ekle" (with input fields for "Başlık" and "İçerik", and a green "Ekle" button), "Hakkında" (with a message "Merhaba! Ben bir blog yazarıyım."), and two cards: "İlk Blog Yazım" (with a "Sil" button) and "React Öğreniyorum" (with a "Sil" button). The bottom of the screen shows a terminal window with build logs, indicating a successful compilation.

```
src > pages > Home.js > Home
1 import React, { useState } from "react";
2 import { Link } from "react-router-dom";
3
4 const Home = () => {
5   // Blog yazıları için sahte veri
6   const [blogs] = useState([
7     { id: 1, title: "React ile Blog Yapımı", content: "React ile nas..."},
8     { id: 2, title: "Tailwind CSS Nedir?", content: "Tailwind CSS na..."},
9   ]);
10
11   return (
12     <div>
13       <h1>Blog Sayfası</h1>
14       {blogs.map((blog) => (
15         <div key={blog.id}>
16           <Link to={`/blog/${blog.id}`}>
17             <h2>{blog.title}</h2>
18             <p>{blog.content.substring(0, 50)}...</p>
19           </Link>
20         </div>
21       )));
22     </div>
23   );
24 }
25
26 export default Home;
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Compiled successfully!

You can now view first-app in the browser.

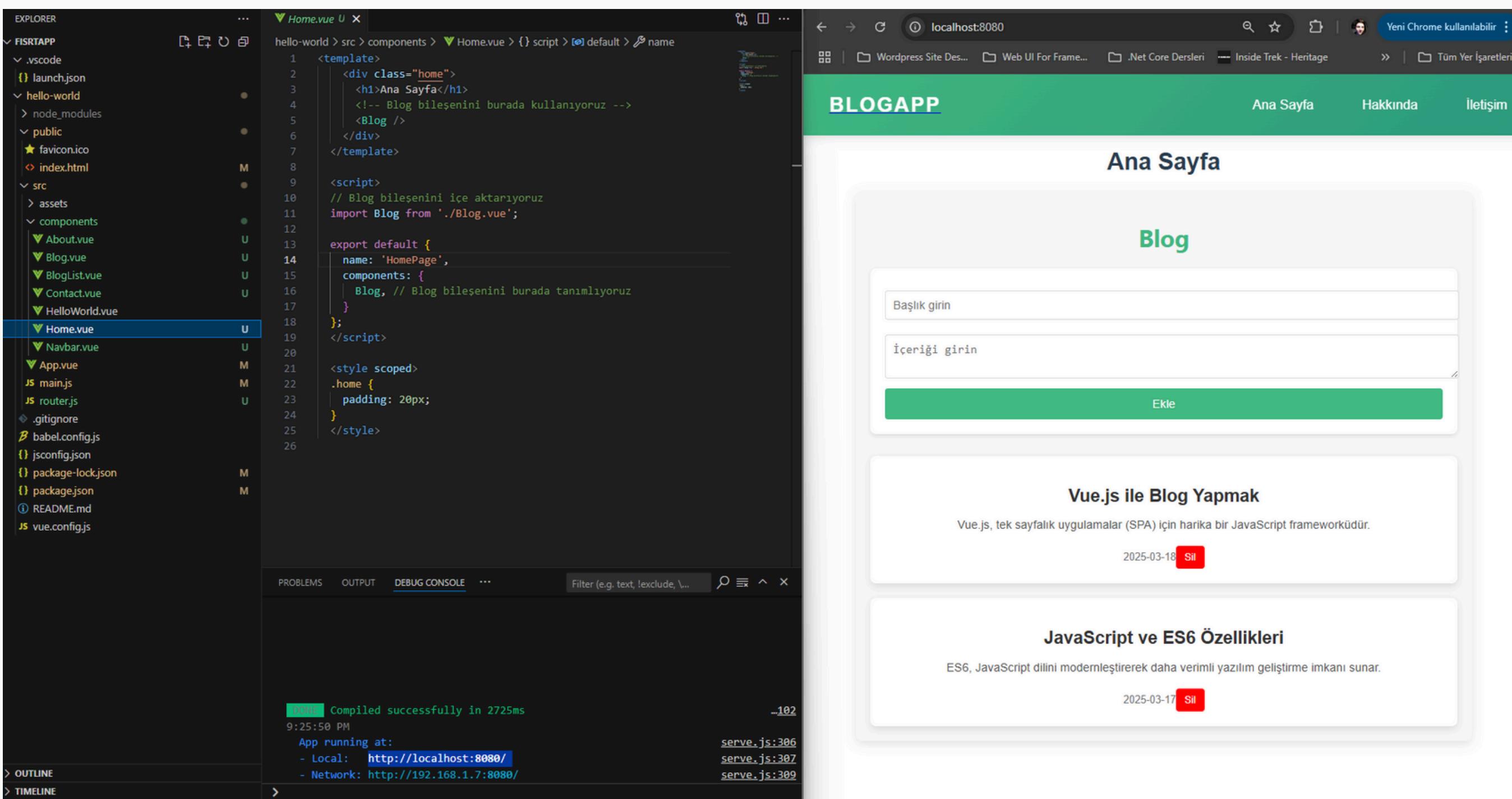
Local: http://localhost:3001
On Your Network: http://172.19.128.1:3001

Note that the development build is not optimized.
To create a production build, use `npm run build`.

webpack compiled successfully

Vue Web Project

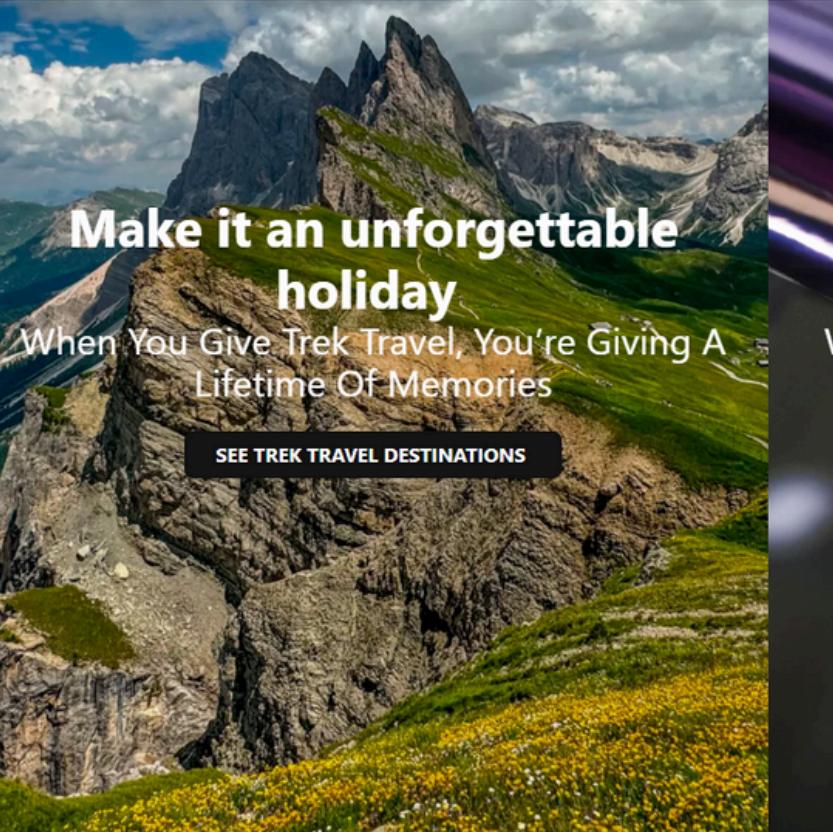
VUE-FIRST-APP PROJESİ, VUE.JS FRAMEWORK'Ü KULLANILARAK GELİŞTİRİLMİŞ BASIT BİR WEB UYGULAMASIDIR. BU PROJE, VUE.JS'İN TEMEL KAVRAMLARINI ÖĞRENMEK VE UYGULAMAK AMACIYLA OLUŞTURULMUŞTUR. UYGULAMA, BİLEŞEN TABANLI YAPIYI KULLANARAK MODÜLER BİR TASARIM SUNMAKTA, VUE.JS'İN GÜÇLÜ ÖZELLİKLERİ OLAN VERİ BAĞLAMA (DATA BINDİNG), DURUM YÖNETİMİ VE KULLANICI ETKİLEŞİMLERİNİ VERİMLİ BİR ŞEKİLDE YÖNETMEKTEDİR. PROJE, VUE.JS'İN TEMEL ÖZELLİKLERİ OLAN V-BIND, V-FOR, V-IF GİBİ DİREKTİFLERİ VE EVENT HANDLING'İ UYGULAMALI BİR ŞEKİLDE GöSTERMEKTEDİR. AYRICA, VUE CLI İLE BAŞLATILAN PROJE SAYESİNDE, MODERN JAVASCRIPT VE VUE ÖZELLİKLERİ KULLANILARAK DİNAMİK VE ETKİLEŞİMLİ BİR KULLANICI ARAYÜZÜ OLUŞTURULMUŞTUR.



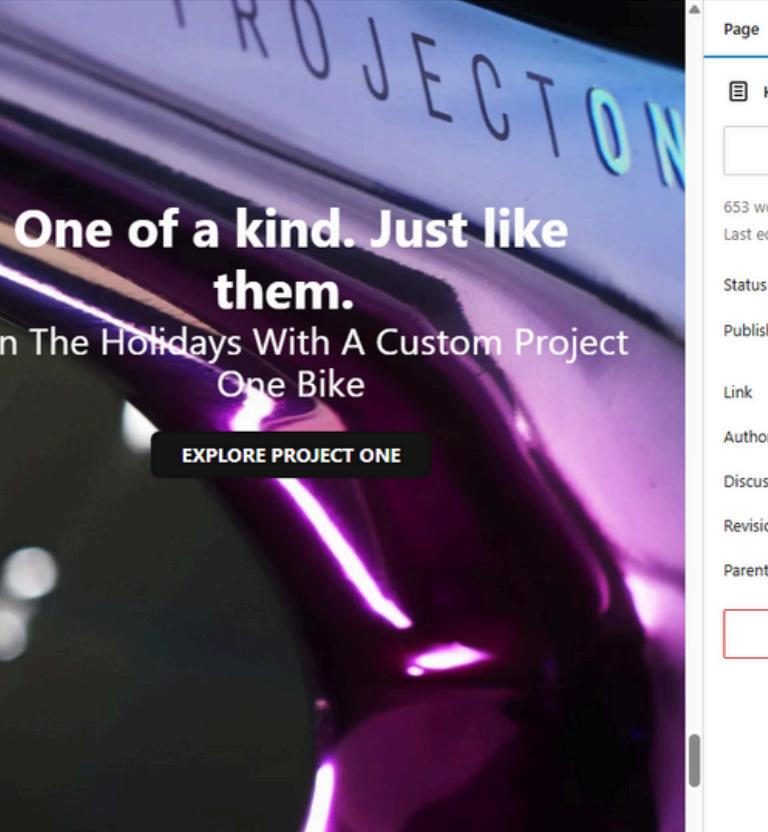
SUNAN: AKIN İNCECİK

Wordpress Project

PROJE SÜRECİNDE SANAL BİR LINUX MAKİNESİ KURARAK, BU MAKİNEDE WEB SUNUCUSU VE VERİTABANI YÖNETİM SİSTEMİ YAPILANDIRDİM. VMWARE ÜZERİNDE DEBIAN TABANLI İŞLETİM SİSTEMİ İLE NGINX WEB SUNUCUSunu KURDUM VE WEB TRAFİĞİNİ YÖNLENDİRDİM. AYRICA, MONGODB'YI KURARAK ESNEK BİR VERİTABANI YÖNETİMİ SAĞLADIM. LINUX KOMUT SATIRINDA PAKET YÖNETİCİLERİ VE SİSTEM DOSYALARI İLE ÇALIŞARAK UYUMLU BİR ORTAM OLUŞTURDUM.



Make it an unforgettable holiday
When You Give Trek Travel, You're Giving A Lifetime Of Memories
SEE TREK TRAVEL DESTINATIONS



One of a kind. Just like them.
Win The Holidays With A Custom Project One Bike
EXPLORE PROJECT ONE

WP_Debian12x64 - VMware Workstation

File Edit View VM Tabs Help

Library Type here to search

My Computer WM_Debian12x64 WP_Debian12x64 KolejSepetim.com KolejSepetim.com_Debian KolejSepetim.com_Open KolejSepetim.com_cpm_Open KolejSepetim.com_2Debian PostgreSQLWindows10x2 KolejSepetim.com_2Debian KolejSepetim.com_Debian WP_Debian12x64

Debian GNU/Linux 12 debian tty1

debian login: root

Password:

Linux debian 6.1.0-27-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.115-1 (2024-11-01) x86_64

The programs included with the Debian GNU/Linux system are free software; the exact distribution terms for each program are described in the individual files in /usr/share/doc/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent permitted by applicable law.

Last login: Sat Nov 30 17:11:56 +03 2024 from 192.168.220.1 on pts/0

root@debian:~# ip a

1: lo <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000 link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00

inet 127.0.0.1/8 brd 127.255.255.255 scope host lo

valid_lft forever preferred_lft forever

inet6 ::1/128 brd :: scope host noprefixroute

valid_lft forever preferred_lft forever

2: ens33 <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000 link/ether 00:0c:29:09:f8:22 brd ff:ff:ff:ff:ff:ff

inet 192.168.220.12/24 brd 192.168.220.255 scope global dynamic ens33

valid_lft 1785sec preferred_lft 1785sec

inet6 fe80::20c:29ff:fe09:f822/64 scope link

valid_lft forever preferred_lft forever

root@debian:~# root

-bash: root: command not found

root@debian:~# [53178.055393] e1000 0000:02:01.0 ens33: Reset adapter

Güvenli değil 192.168.220.12

Wordpress Site Des... Web UI For Frame... .Net Core Dersleri Inside Trek - Heritage Contact Trek Bicycle... Tüm Yer İşaretleri

Trek Bikes - the world's best bi...

Güvenli değil 192.168.220.12

Yeni Chrome kullanılabilir

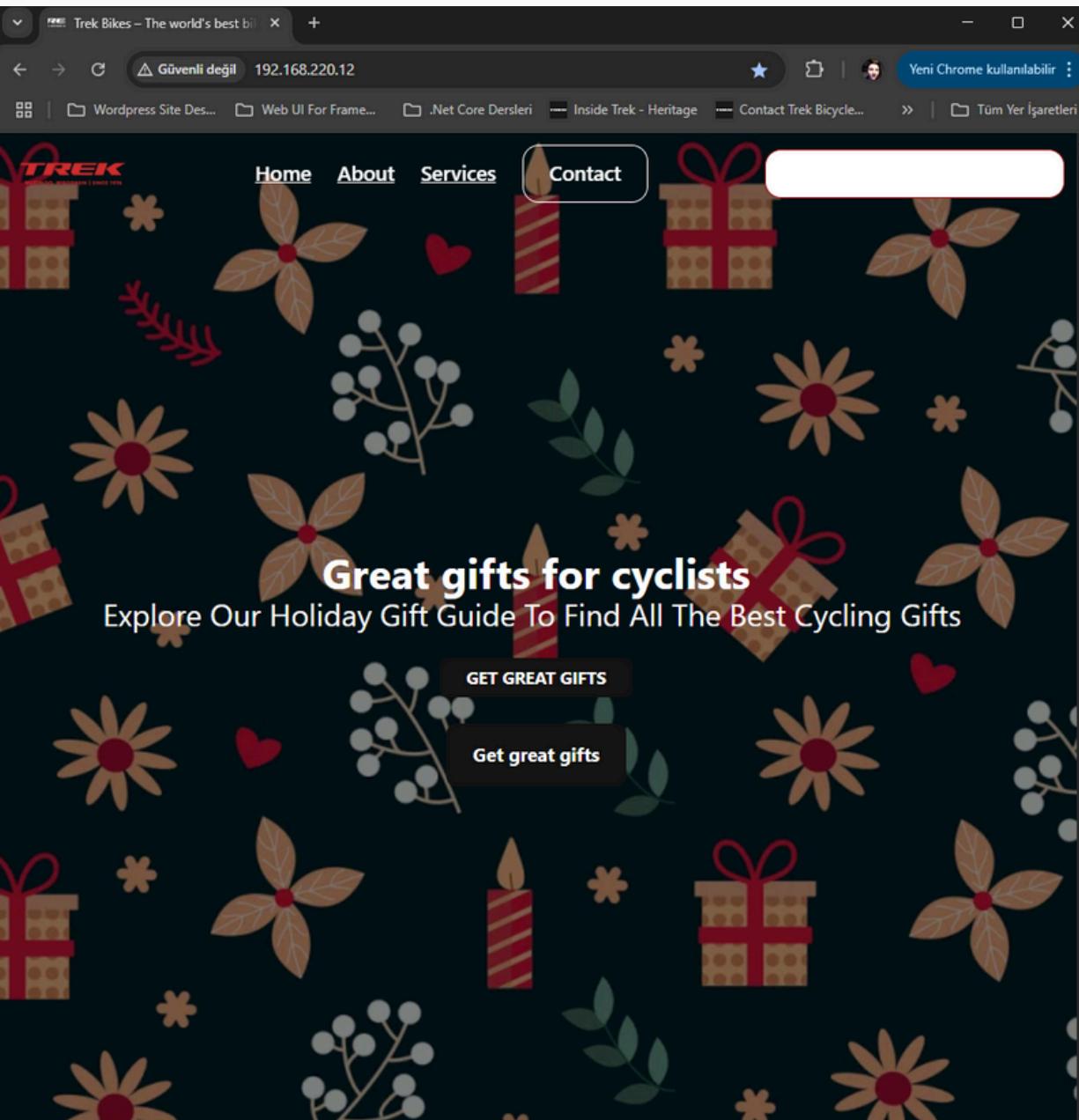
Home About Services Contact

Great gifts for cyclists

Explore Our Holiday Gift Guide To Find All The Best Cycling Gifts

GET GREAT GIFTS

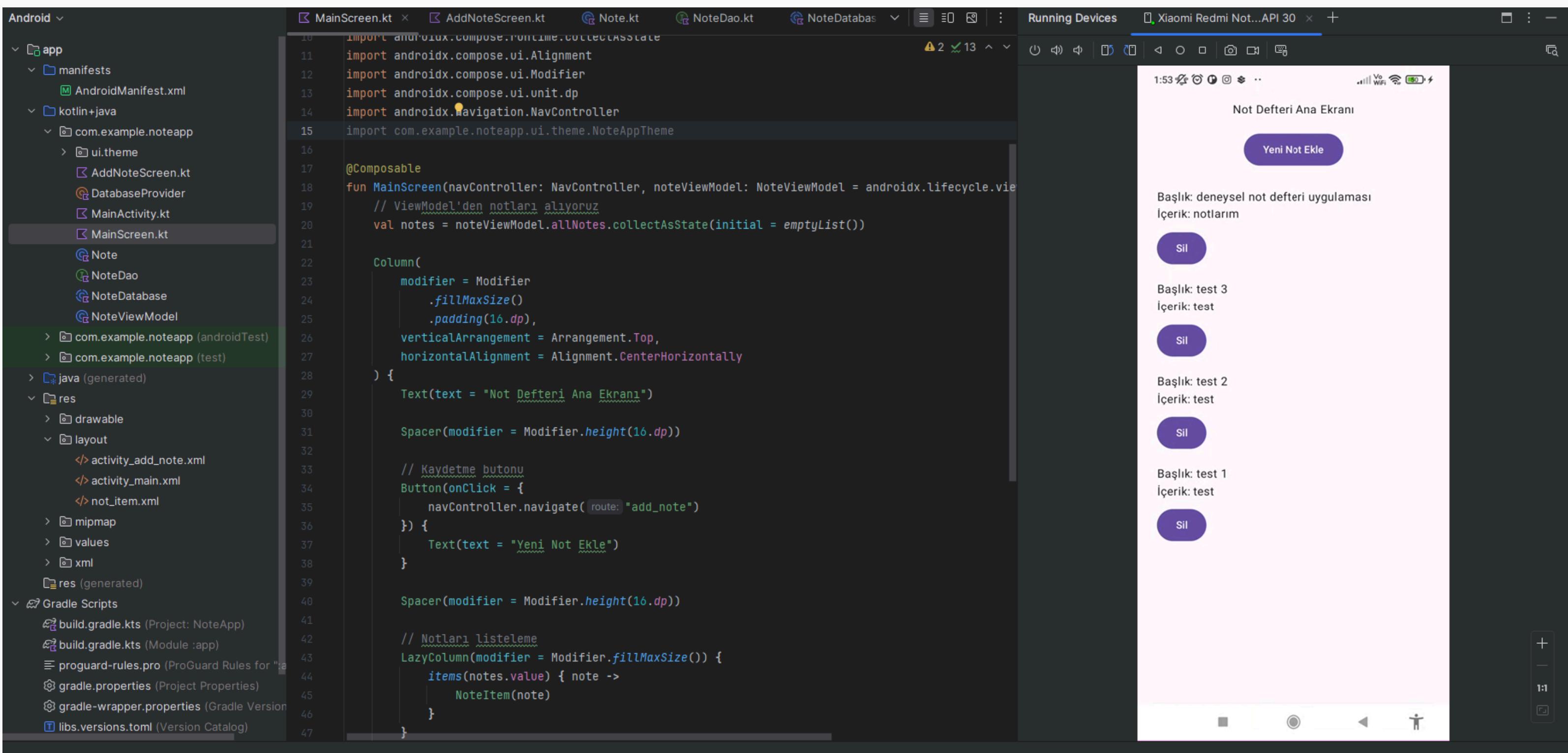
Get great gifts



WORDPRESS KURULUMU VE REPLİKASYON SÜRECİ:
LINUX MAKİNESİ ÜZERİNDE NGINX VE MONGODB BAŞARIYLA KURULDuktan sonra, WORDPRESS'İ KURARAK İÇERİK YÖNETİM SİSTEMİNİ DEVREYE ALDIM. WORDPRESS KURULUMU İÇİN GEREKLİ PHP SÜRÜMÜNÜ VE GEREKLİ BAĞIMLILIKLARI YÜKLEDİM, ARDINDAN NGINX'İ WORDPRESS İLE UYUMLU ŞEKLDE YAPILANDIRDİM. WORDPRESS SİTESİNİN REPLİKASINI OLUSTURURKEN, TEMA VE EKLİTİLERİ DOĞRU ŞEKLDE YAPILANDIRARAK, MEVCUT SİtenİN TAM BİR KOPYASINI ELDE ETTİM. BU SÜREÇ, DOSYA TRANSFERİ, VERİTABANI YEDEKLEMESİ VE YAPILANDIRMA AYARLARININ DOĞRU ŞEKLDE YAPILMASINI İÇERİYORDU. BU PROJEDe HEM WEB SUNUCUSU YAPILANDIRMASI HEM DE WORDPRESS YÖNETİMİ KONULARINDA ÖNEMLİ TEKRÜBELER EDİNDİM VE SİSTEMİN PERFORMANSINI OPTİMİZE EDEREK BAŞARILI BİR REPLİKASYON GERÇEKLEŞTİRDİM.

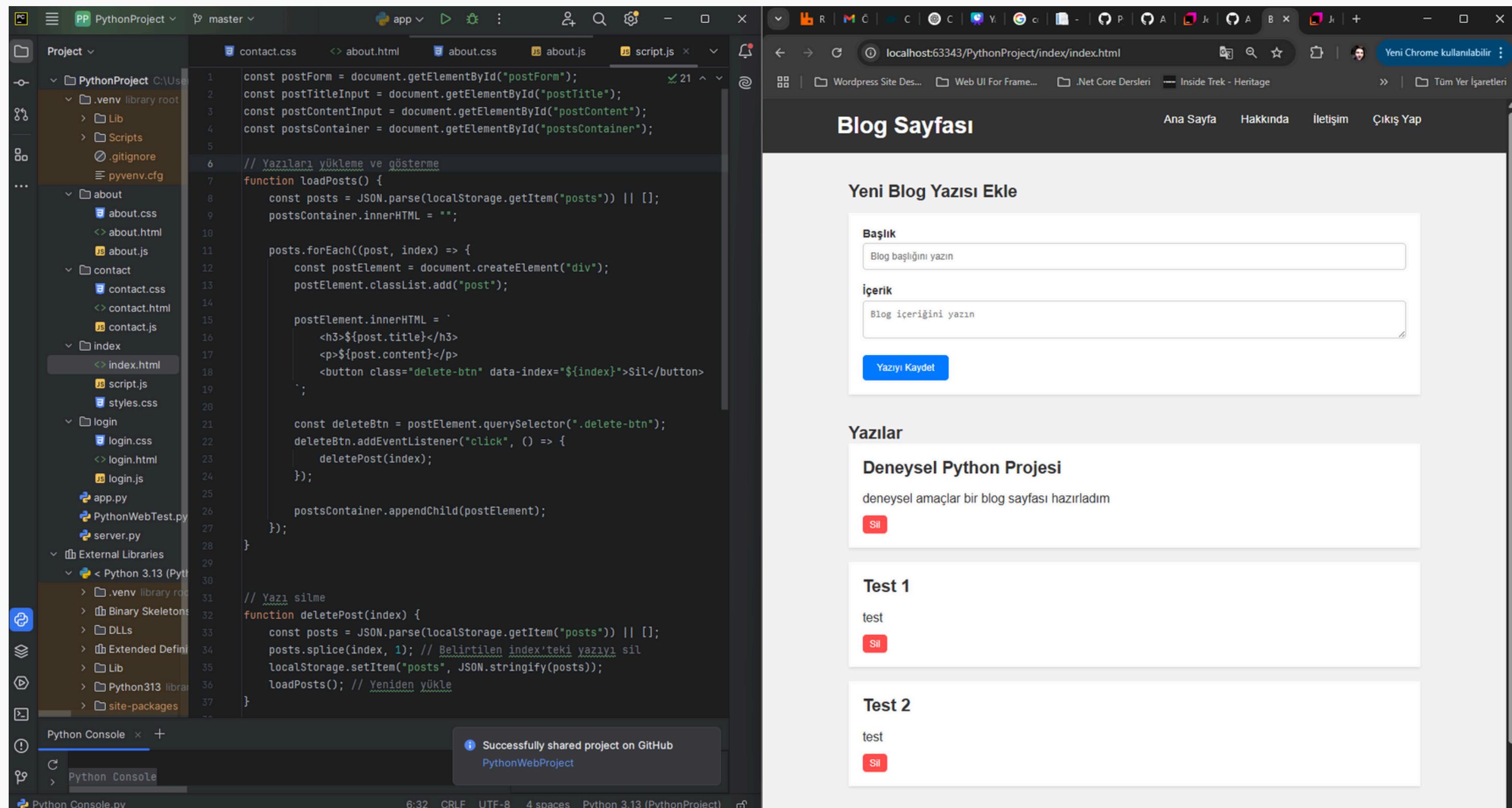
Android Notepad App

ANDROİDNOTEAPP, NOT ALMAYI VE YÖNETMEYİ KOLAYLAŞTIRAN KULLANICI DOSTU BİR UYGULAMADIR. KULLANICILAR HIZLICA NOT OLUŞTURABİLİR, DÜZENLEYEBİLİR VE SİLEBİLİR. UYGULAMA, ACTIVITY, RECYCLERVIEW VE SQLİTE TEKNOLOJİLERİNI KULLANARAK VERİLERİ YEREL OLARAK DEPOLAR VE MATERIAL DESIGN PRENSİPLERİNE UYGUN BİR ARAYÜZ SUNAR. BU PROJE, ANDROİD GELİŞTİRMEYE YENİ BAŞLAYANLAR İÇİN UI TASARIMI, VERİTABANI İŞLEMLERİ VE RECYCLERVIEW GİBİ TEMEL KONULARI ÖĞRENME FIRSATI SAĞLAR, TEMEL CRUD İŞLEMLERİNİ ETKİLİ BİR ŞEKİLDE YÖNETİR.



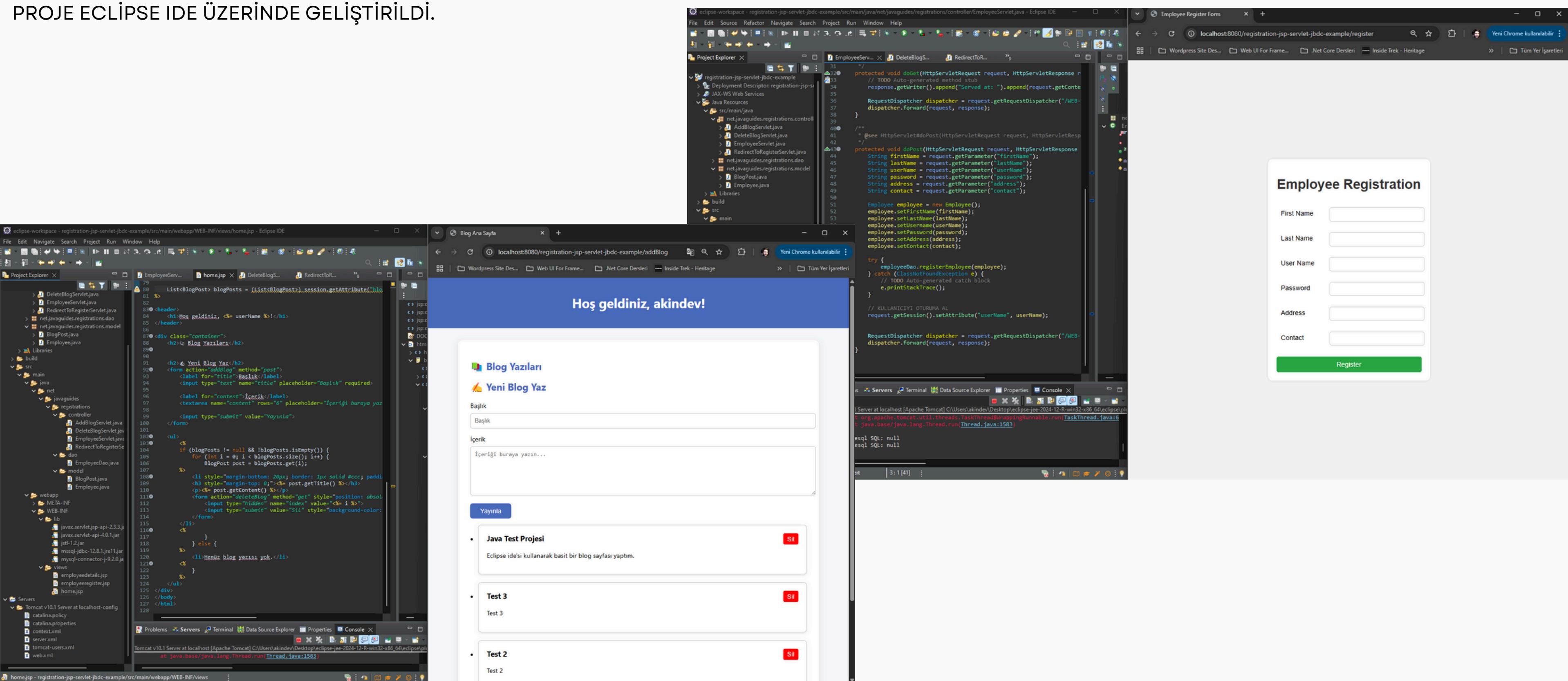
Python Web Project

PYTHONWEBPROJECT, PYTHON İLE WEB UYGULAMALARI GELİŞTİRME SÜRECİNİ TEMEL ALAN BİR PROJEDİR. FLASK WEB FRAMEWORK'Ü KULLANILARAK OLUŞTURULMUŞ OLAN BU PROJE, TEMEL VERİ TABANI İŞLEMLERİ, ROUTING (YÖNLENDİRME), HTML ŞABLONLARI VE STATİK DOSYALARIN YÖNETİMİ GİBİ WEB UYGULAMALARINDA YAYGIN KULLANILAN ÖZELLİKLERİ İÇERİR. SQLITE VERİTABANI İLE VERİ SAKLAMA İŞLEMLERİ GERÇEKLEŞTİRİLİR VE KULLANICIYA DİNAMİK WEB SAYFALARI SUNULUR. BU PROJE, PYTHON TABANLI WEB UYGULAMALARI GELİŞTİRMEK İSTEYENLER İÇİN TEMEL BİR ÖRNEK TEŞKİL ETMEKTEDİR.



Java Web Project

BU PROJEDE JAVA, JSP VE SERVLET TEKNOLOJİLERİ KULLANARAK BİR KULLANICI KAYIT VE BLOG SİSTEMİ GELİŞTİRDİM. KAYIT OLAN KULLANICILAR APACHE TOMCAT SUNUCUSU ÜZERİNDE ÇALIŞAN UYGULAMAYA YÖNLENDİRİLEREK GİRİŞ YAPABİLİYOR. ANA SAYFADA BASIT BİR BLOG EKRANI YER ALIYOR; KULLANICILAR BURADA BLOG YAZILARI OLUŞTURABİLİYOR VE DİLERSE SİLEBİLİYOR. VERİTABANI İŞLEMLERİ İÇİN JDBC KULLANILARAK MS SQL SERVER İLE BAĞLANTI SAĞLANDI. PROJE ECLIPSE IDE ÜZERİNDE GELİŞTİRİLDİ.



SUNAN: AKIN İNCECİK

Təşəkkürler!