SUNAN: AKIN İNCECİK

SOFTWARE **PROJECT** *Portfolio*

SUNAN: AKIN İNCECİK

Presentation Content

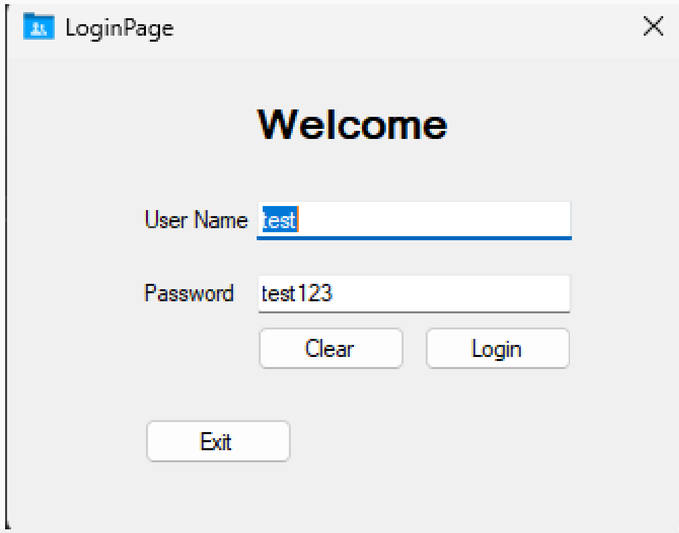
1	Crm Demo Project
2	Database Listener Service (RabbitMQ)
3	RabbitMQ Listener Service
4	MVC With AdminLTE (IIS)
5	Angular Web Project

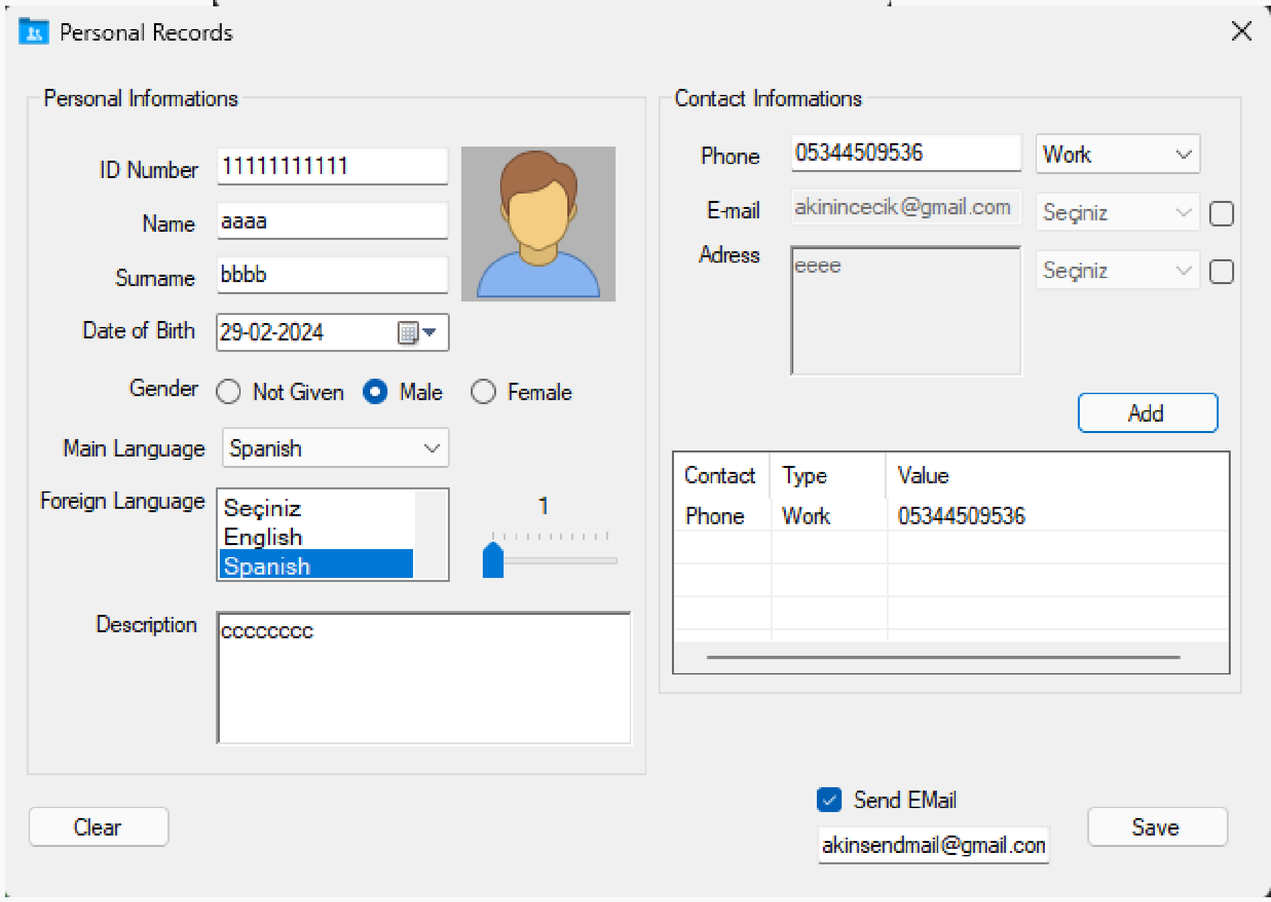
6	React Web Project
7	Vue Web Project
8	Wordpress Project
9	Android Notepad App
10	Python Web Project

SUNAN: AKIN İNCECİK

Crm Demo Project

The project I developed for managing student registrations is designed to collect not only basic user information but also the languages the student knows and wants to learn. After registration, email addresses are saved in the database and used for automated email delivery. The user interface and database integration work seamlessly together.





SUNAN: AKIN İNCECİK

Crm Demo Codes

CRMDemo is a dynamic CRM demo application that manages student registrations and email delivery processes. After logging in, users can register student information; validation rules are applied, and the data can be stored in either MSSQL or PostgreSQL databases.

Project Features:

- **Secure Data Connection:** Secure data communication is ensured through encrypted connection strings.
- **Asynchronous Email Sending:** Emails are queued and sent asynchronously using RabbitMQ and a background service.
- **Email Content:** Email bodies can be stored in both JSON and HTML formats.
- **Modular Structure:** The modular SQLUtils structure simplifies and separates database operations.
- **Flexible CRM Environment:** This architecture provides a flexible and extensible CRM demo environment.

```
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.");
}
```

```
public class SQLUtils
{

    10 references
    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) }
            };

        1 reference
        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}");
            }
        }

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

            return connectionFactory(GetConnectionString(databaseType));
        }

        5 references
        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}");
            }
        }
    }
```

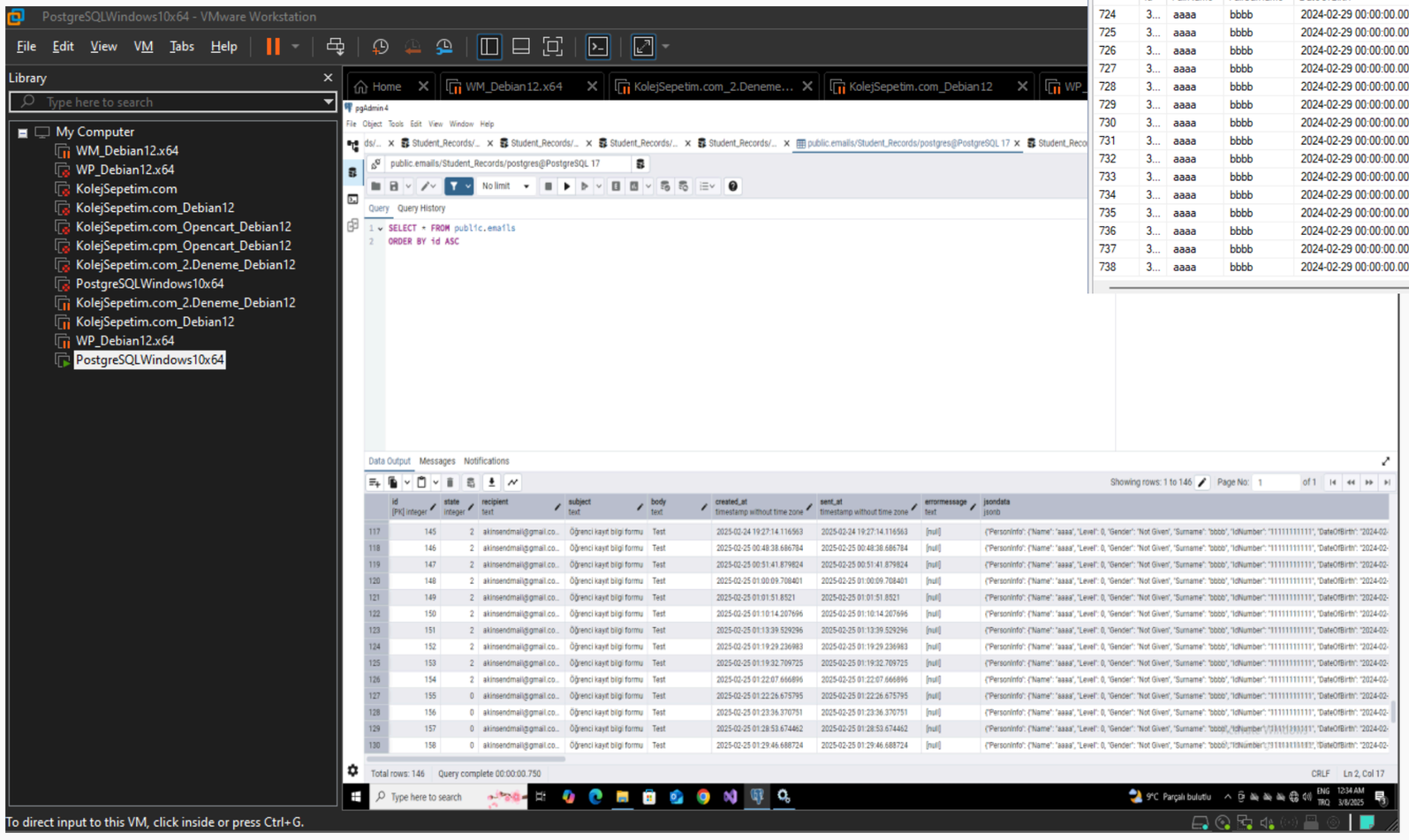
SUNAN: AKIN İNCECİK

CRM Demo Simultaneous Data Recording and Mail Preparation

Data received from the form is saved simultaneously to both MSSQL and PostgreSQL databases. This process ensures data integrity across both systems and enables synchronization between them.

```
SELECT TOP (1000) [Id]
      ,[FullName]
      ,[FullSurname]
      ,[DateOfBirth]
      ,[MainLanguage]
      ,[ForeignLanguage]
      ,[ForeignLanguageLevel]
      ,[Description]
      ,[PhoneNumber]
      ,[Email]
      ,[Adress]
      ,[PicturePath]
  FROM [Student_Records].[dbo].[StudentInformations]
```

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

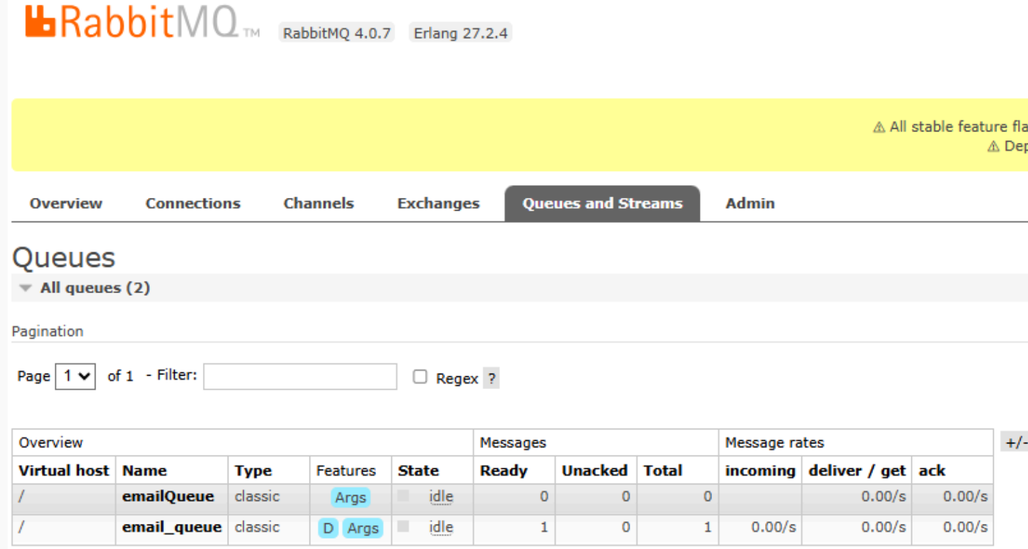


The saved data is stored with a "Queued" status, ready for email delivery. The background service picks up this data, formats it into an email, and automatically initiates the sending process.

SUNAN: AKIN İNCECİK

Database Listener Service

This service analyzes the statuses of the data saved by the CRM demo application. It identifies the records that are ready for email delivery and forwards them to the RabbitMQ queue. This way, the necessary data for the sending process is automatically transferred to the next stage.




```
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 ver
                { "@HtmlData", emailData["HtmlData"] },  // HTML formatındaki ve
                { "@EmailId", emailData["Id"] }          // EmailId parametresi e
            };

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

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

                // Başarıyla RabbitMQ'ya gönderildiyse durumu 6 (Sent) yap
                SQLUtils.UpdateSqlEmailStatus(emailId, 6);
                Console.WriteLine($"☑ Email ID {emailId} RabbitMQ'ya başarıyla
            }
            catch (Exception ex)
            {
                // RabbitMQ'ya gönderilemediyse durumu 4 (Failed) yap
                SQLUtils.UpdateSqlEmailStatus(emailId, 4);
                Console.WriteLine($"❌ Email ID {emailId}: RabbitMQ gönderimi ba
            }
        }
    }
}
```

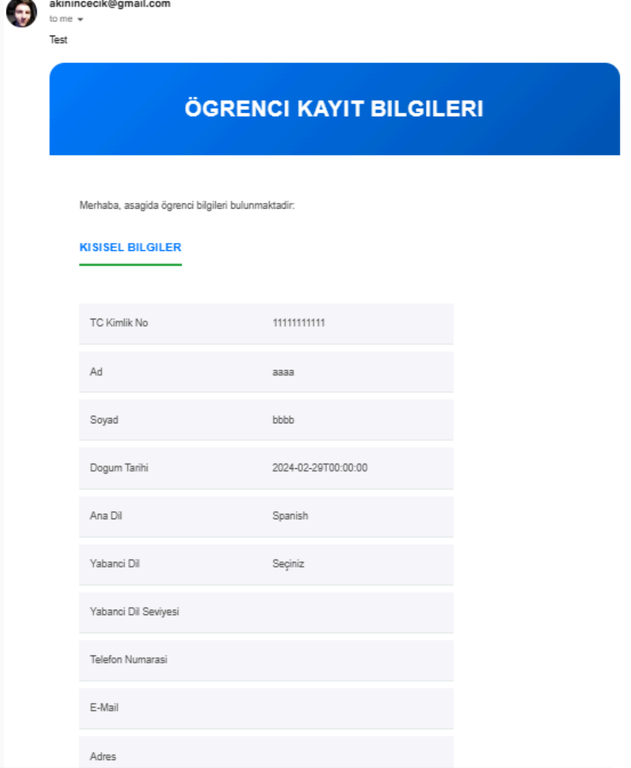
```
namespace WindowsServiceTest2
{
    1 reference
    public class Worker
    {
        private static Object _lock = new Object();
        private static DatabaseParameters _dbParams = new DatabaseParameters();  // Bağlantı bilgilerini merkezi olarak

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

            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);
            }
        }
    }
}
```

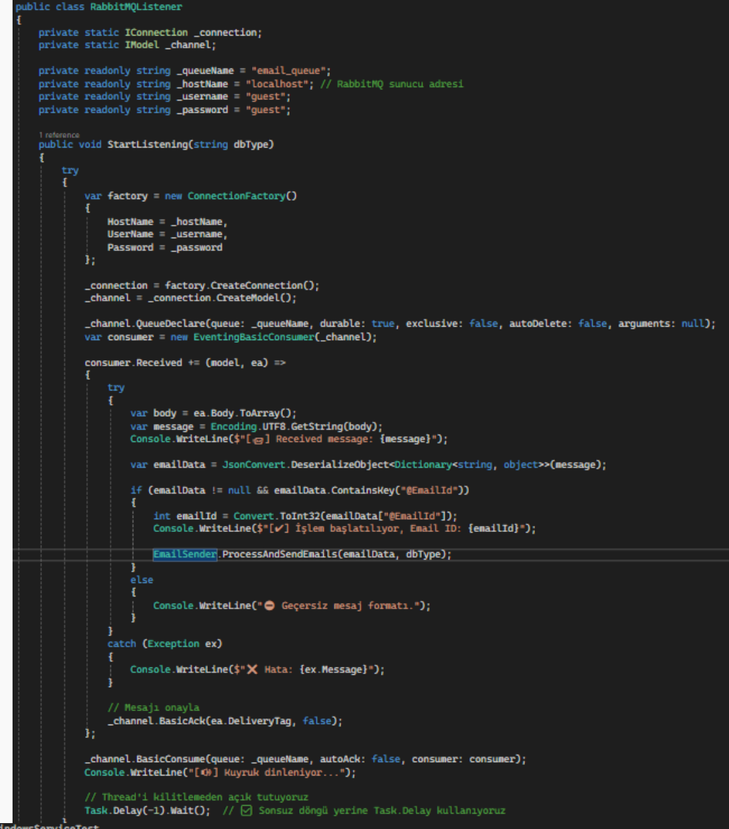
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace WindowsServiceTest2
{
    1 reference
    public class RabbitMQHelper
    {
        1 reference
        public static void SendMessage(string message)
        {
            try
            {
                var factory = new ConnectionFactory() { HostName = "localhost" };
                using (var connection = factory.CreateConnection())
                using (var channel = connection.CreateModel())
                {
                    channel.QueueDeclare(queue: "email_queue", durable: true, exclusive: false, autoDelete: false, arguments: null);

                    var body = Encoding.UTF8.GetBytes(message);
                    var properties = channel.CreateBasicProperties();
                    properties.Persistent = true;

                    channel.BasicPublish(exchange: "", routingKey: "email_queue", basicProperties: properties, body: body);
                    Console.WriteLine($"📨 Mesaj Kuyruğa Gönderildi: {message}");
                }
            }
            catch (Exception ex)
            {
                Console.WriteLine($"❌ Kuyruğa mesaj gönderilirken hata oluştu: {ex.Message}");
            }
        }
    }
}
```

SUNAN: AKIN İNCECİK

RabbitMQ Listener Service

This service continuously listens to the RabbitMQ queue to check for new data. When data is found in the queue, it retrieves the appropriate email template stored in the database, combines it with the relevant data, and executes the email delivery.





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

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

    1 reference
    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}");
                }

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

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

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

```
e WindowsServiceTest

ence
ic class Worker


private static Object _lock =new Object();
private static DatabaseParameters _dbParams = new DatabaseParameters();  // ☑ Bağlantı bilgilerini merkezi olarak çekiyoruz.

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)
                {
                    RabbitMQListener subscriber = new RabbitMQListener();
                    subscriber.StartListening(dbType);
                }
            }
        }
        catch (Exception ex)
        {
            LogError(ex);
        }

        Thread.Sleep(20);
    }
}

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);
    }
}
```

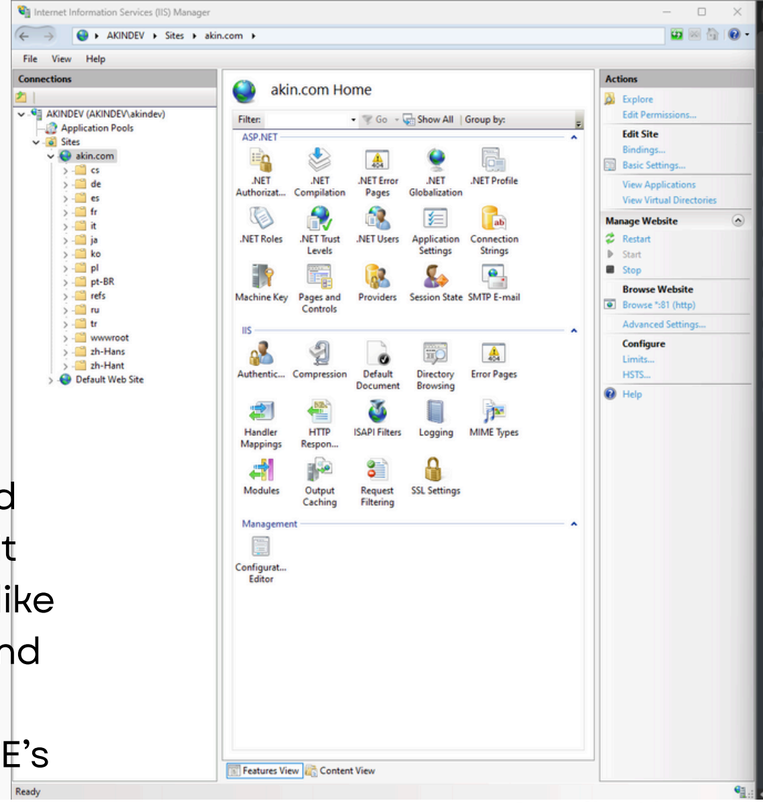
```
public static class EmailSender
{
    1 reference
    public static void ProcessAndSendEmails(Dictionary<string, object> emailData, string dbType)
    {
        try
        {
            if (!IsValidEmailData(emailData, out int emailId, out string recipient, out string subject, out string body, out 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} gönderilemedi. Retrying...");
                UpdateEmailStatus(dbType, emailId, 4); // Retrying
            }
        }
        catch (Exception ex)
        {
            Console.WriteLine($"❌ Email işlenirken hata oluştu: {ex.Message}");
        }
    }

    1 reference
    private static bool IsValidEmailData(Dictionary<string, object> emailData, out int emailId, out string recipient, out string subject, out string body, out string jsonDat
    {
        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;
    }
```

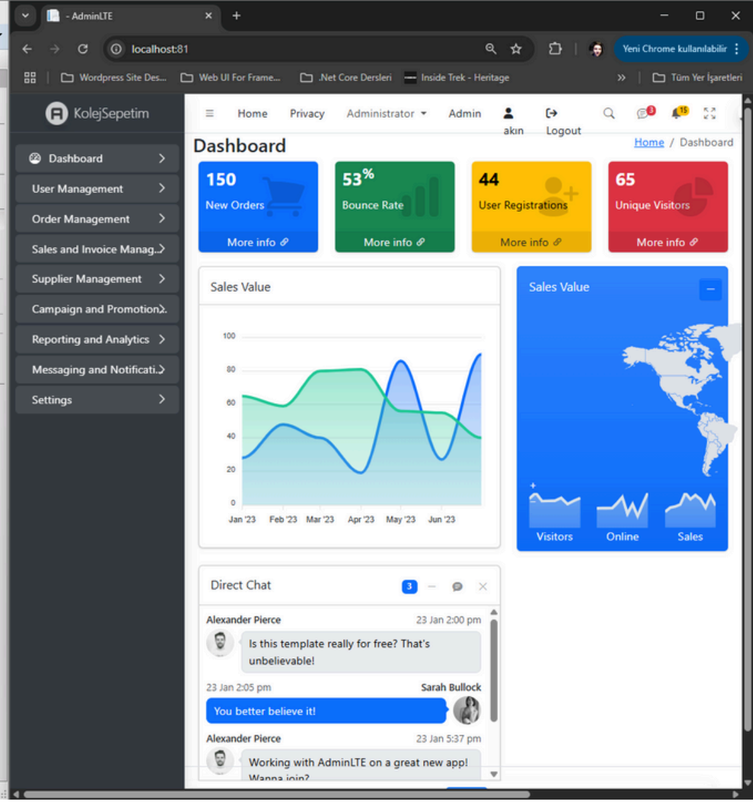
```
private static string PrepareEmailBody(string dbType, int emailId, string body, string jsonData)
{
    int studentId = GetStudentIdFromJson(jsonData);
    string picturePath = SQLUtils.GetStudentPicturePath(dbType, studentId);
    string htmlData = Utils.GenerateHtmlTable(jsonData);

    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;
}

1 reference
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;
}
```

SUNAN: AKIN İNCECİK

MVC With IIS

This project is an MVC-based admin panel application built using the AdminLTE template. It provides efficient management with a user-friendly interface and can operate in a real environment on IIS. With its modern and responsive design, it offers core functions like data management and user interaction. Additionally, thanks to AdminLTE's flexibility and customizability, it can be easily extended.






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

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

        0 references
        public IActionResult Index()
        {

            //ViewBag.CurrentPage = 1;
            ViewBag.ActiveMenu = "supplier-management"; // Açık kalmasını istediğin menü

            return View();
        }

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

        //    return PartialView("_SupplierListPartial", suppliers);
        //}

        0 references
        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);
        }
```

With its modern and responsive design, it offers core functions like data management and user interaction. Additionally, AdminLTE's flexibility and customizability allow for easy expansion.

Supplier List

+ Add Supplier Reload

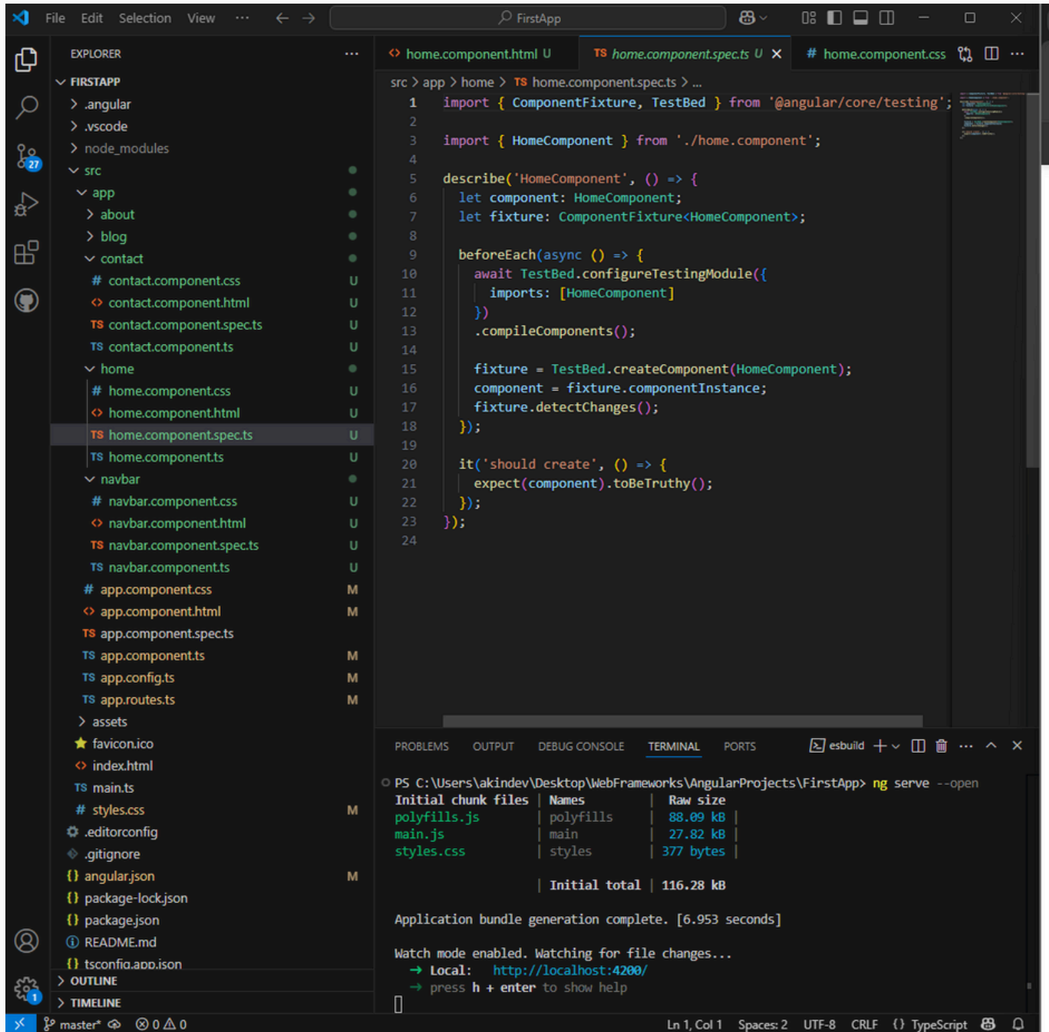
1 2 3 4 5

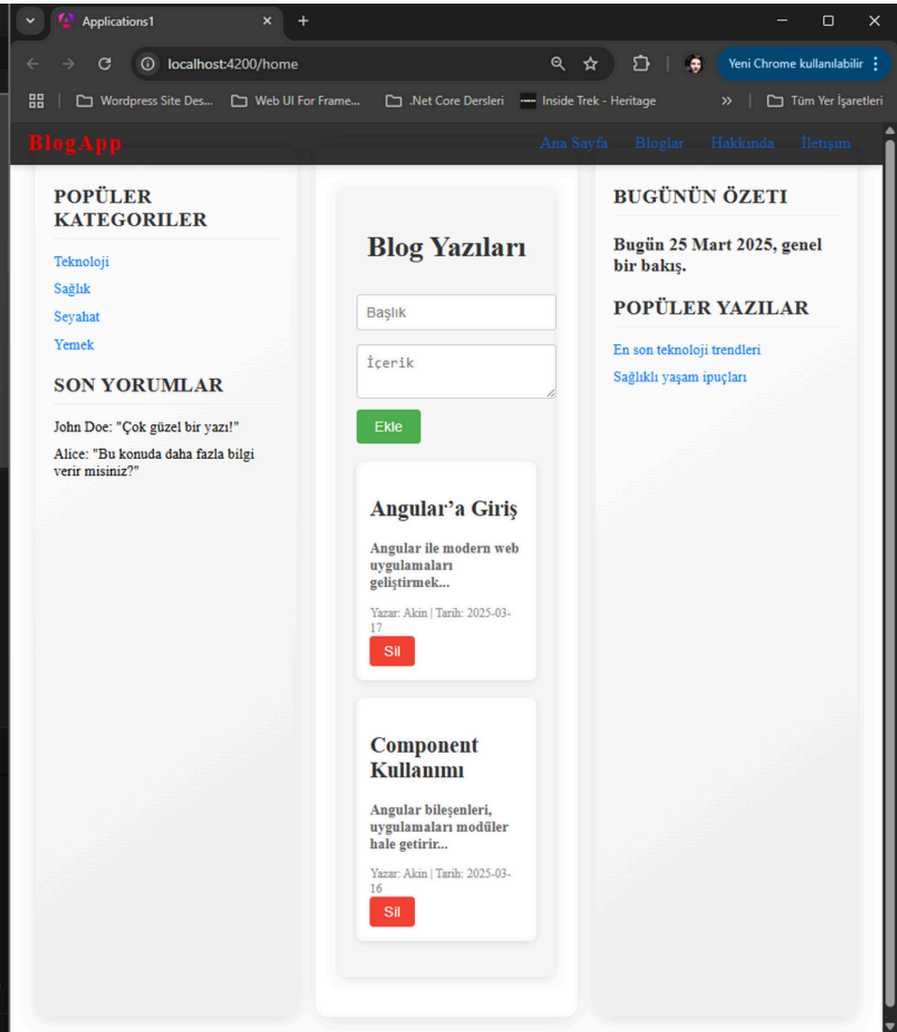
Supplier ID	Supplier Name	Address	Phone Number	Email	Created At
79	Supplier 79	Street 79, City, USA	+1-555-1079	supplier79@mail.com	2025-01-07 23:51
78	Supplier 78	Street 78, City, USA	+1-555-1078	supplier78@mail.com	2025-01-08 23:51
77	Supplier 77	Street 77, City, USA	+1-555-1077	supplier77@mail.com	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

SUNAN: AKIN İNCECİK

Angular Web Project

The Angular-First-App project is a simple blog page developed using the Angular framework. This project was created to learn and apply the fundamental features of Angular. A modular structure was built using Angular's strengths such as component-based architecture and data binding. The project is developed with TypeScript, HTML, and CSS and includes important Angular concepts like routing, user interaction, and service integration between key components.

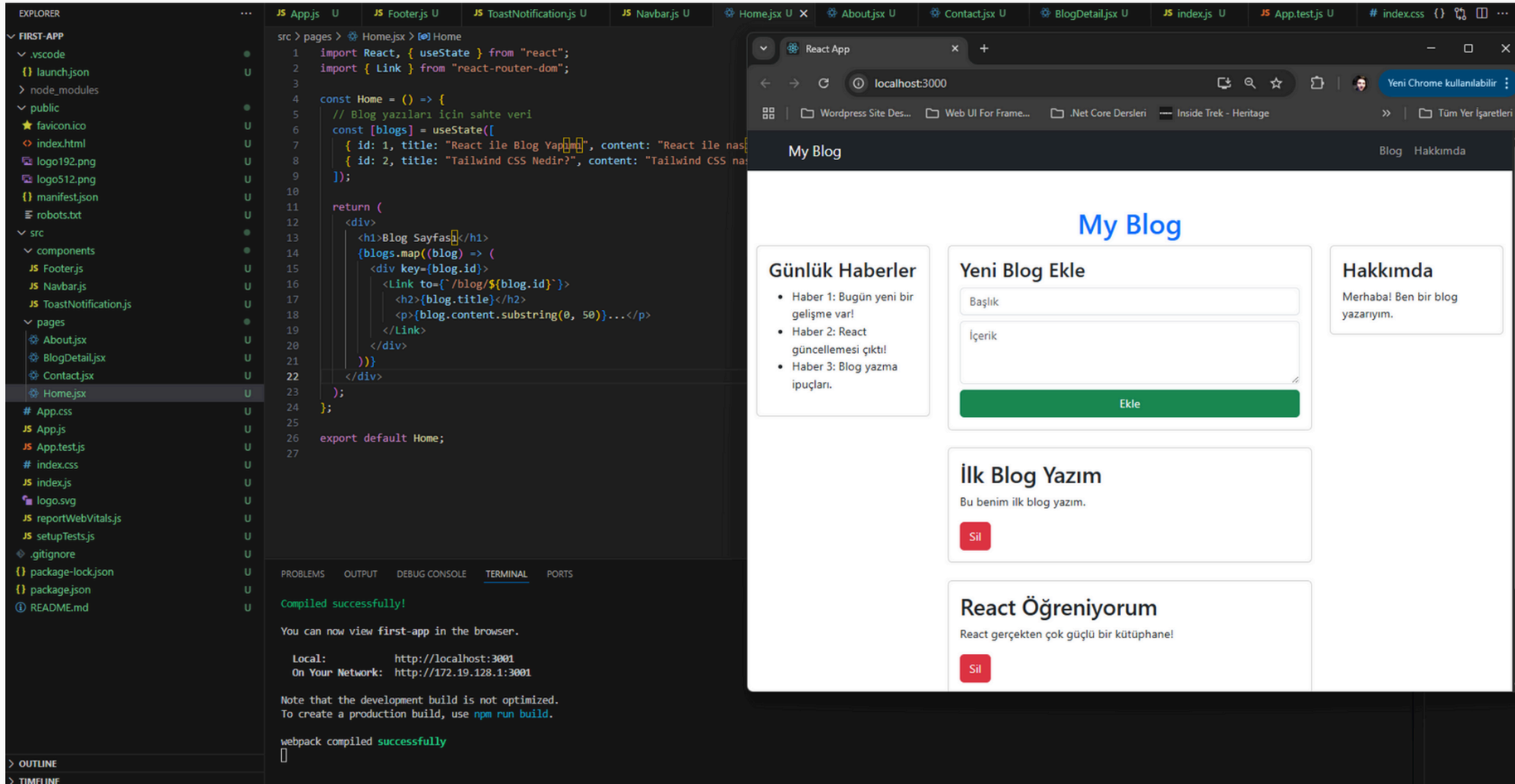




SUNAN: AKIN İNCECİK

React Web Project

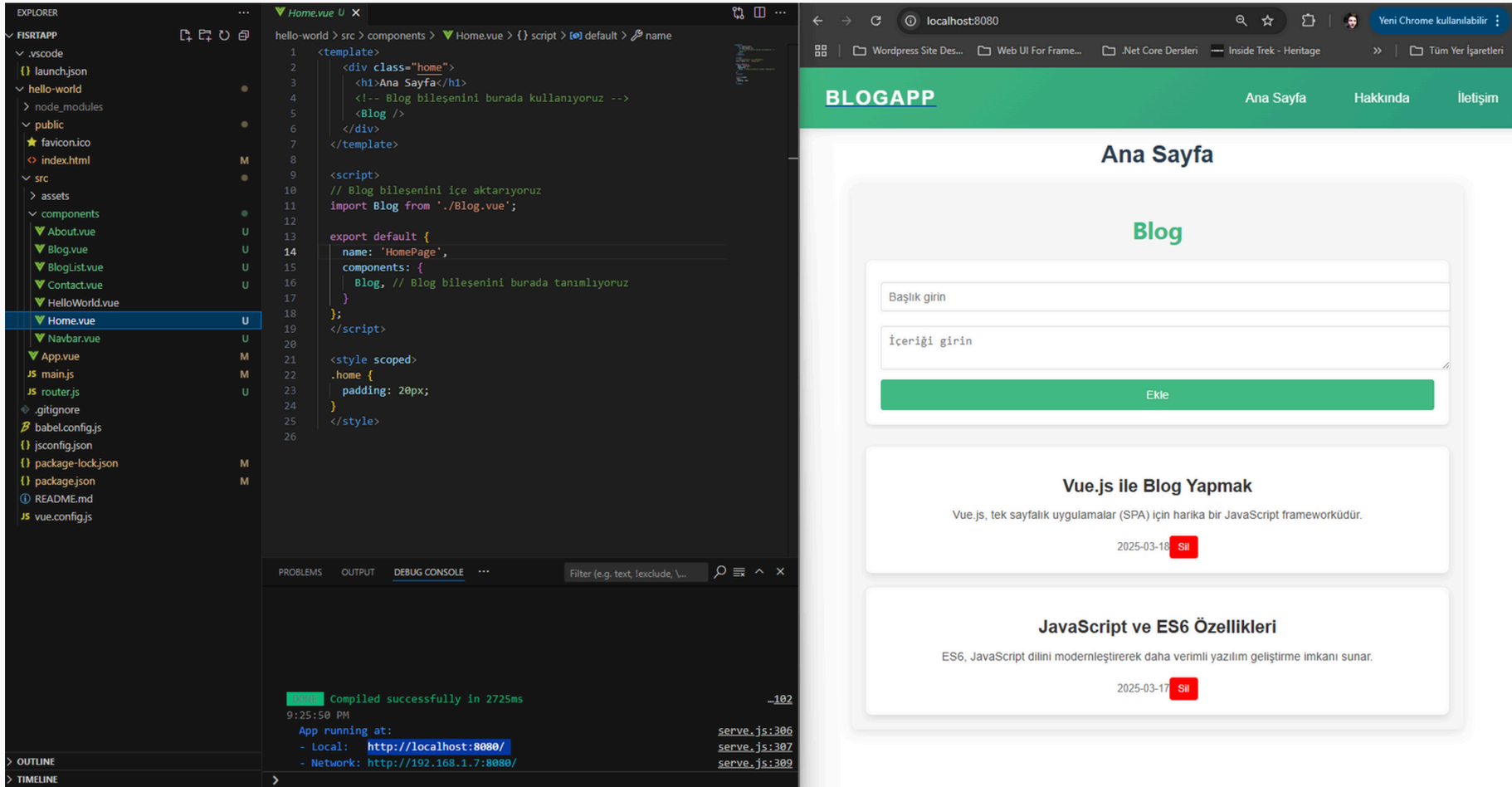
The React-first-app project is a simple web application developed using the React framework. This project was created to learn and apply the core concepts of React. The app offers a modular design using component-based architecture and efficiently manages state and user interactions with React's powerful features. The project serves as an example to understand key functions of React like useState, useEffect, and event handling. Additionally, a dynamic and interactive user interface is created using modern JavaScript and JSX.



SUNAN: AKIN İNCECİK

Vue Web Project

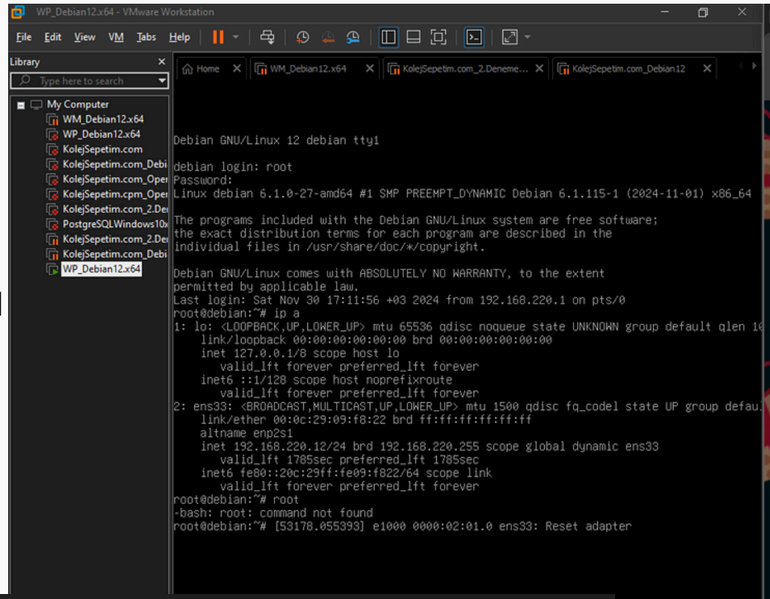
The Vue-first-app project is a simple web application developed using the Vue.js framework. This project was created to learn and apply the fundamental concepts of Vue.js. The app uses component-based architecture to offer a modular design, efficiently managing data binding, state management, and user interactions. The project demonstrates Vue.js directives like v-bind, v-for, v-if, and event handling. Additionally, the project, initiated with Vue CLI, creates a dynamic and interactive user interface using modern JavaScript and Vue features.

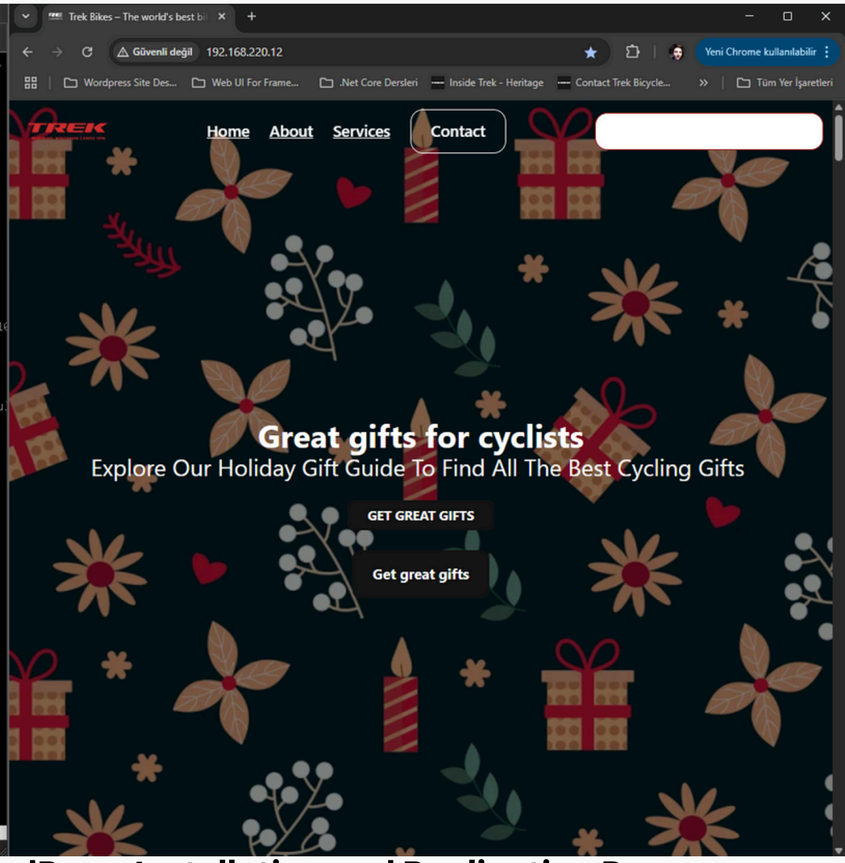


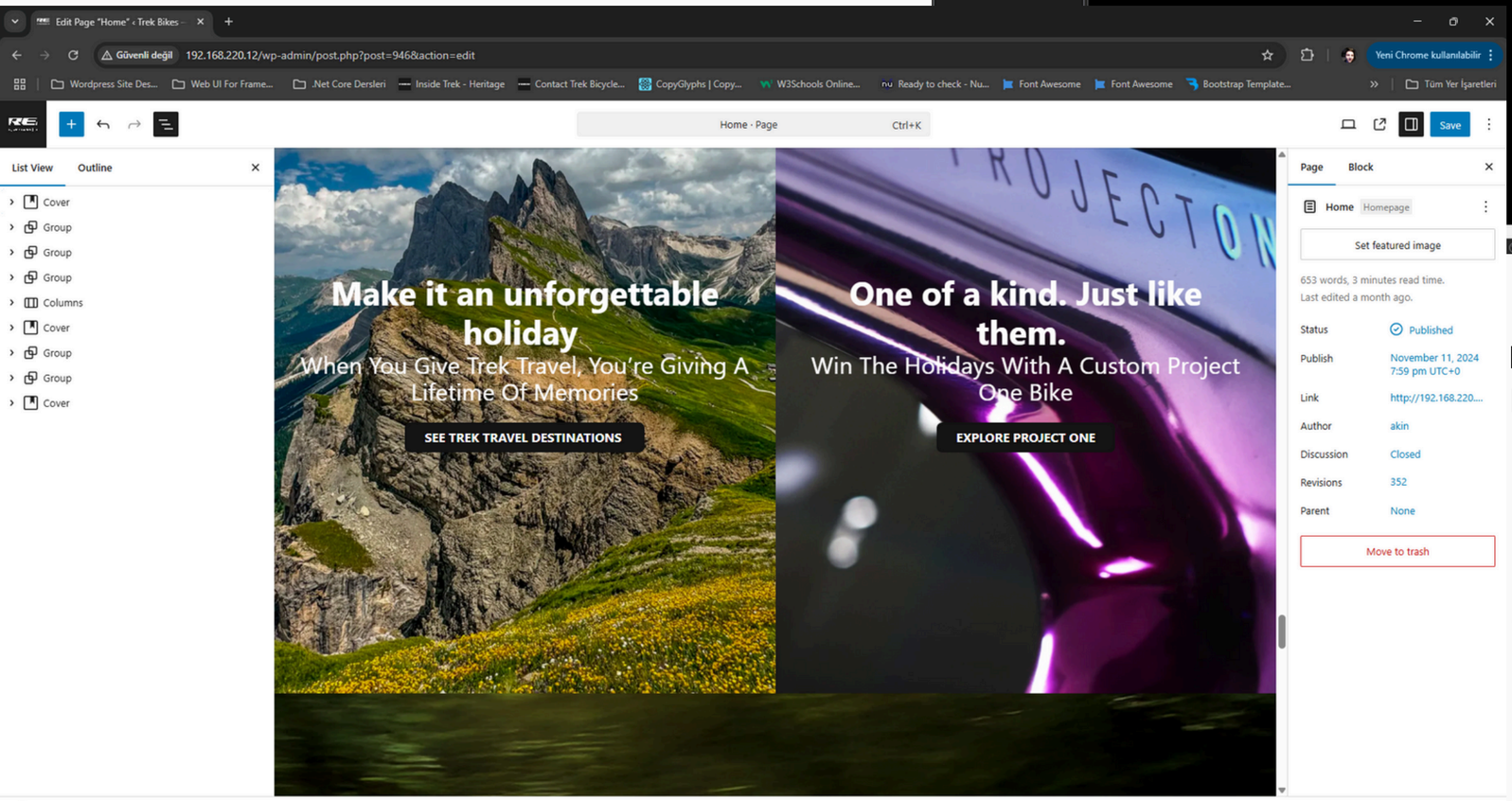
SUNAN: AKIN İNCECİK

Wordpress Project

During the project, I set up a virtual Linux machine and configured a web server and database management system. I installed the Nginx web server on a Debian-based OS in VMware and routed web traffic. Additionally, I installed MongoDB to provide flexible database management. By working with package managers and system files in the Linux command line, I created a compatible environment.







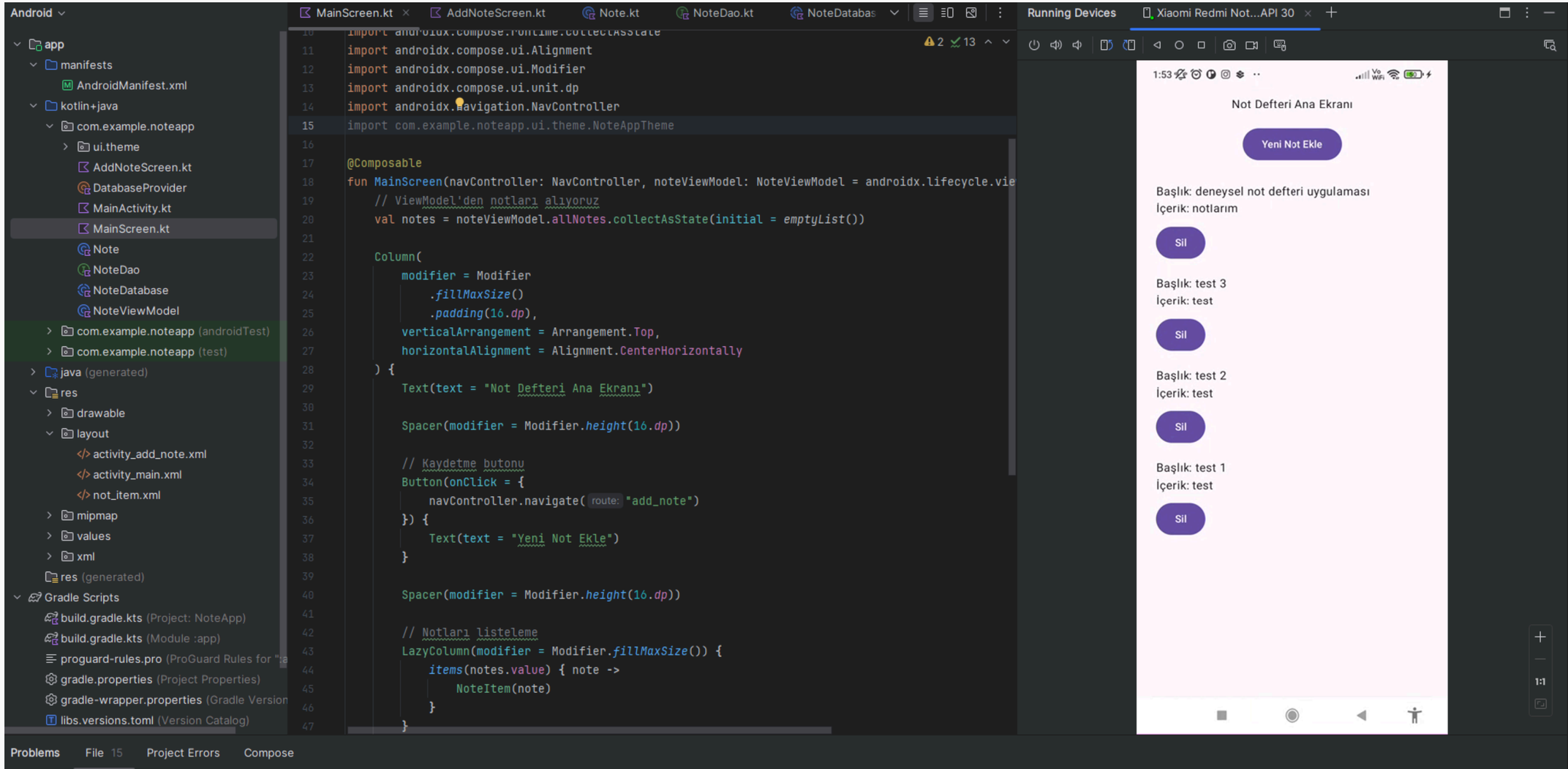
WordPress Installation and Replication Process:

After successfully installing Nginx and MongoDB on the Linux machine, I proceeded with the WordPress installation by setting up the required PHP version and dependencies. I configured Nginx to work seamlessly with WordPress. While creating the replica of the WordPress site, I properly configured the theme and plugins, resulting in an exact copy of the original site. This process included file transfer, database backup, and configuration settings. Through this project, I gained valuable experience in both web server configuration and WordPress management while optimizing system performance for a successful replication.

SUNAN: AKIN İNCECİK

Android Notepad App

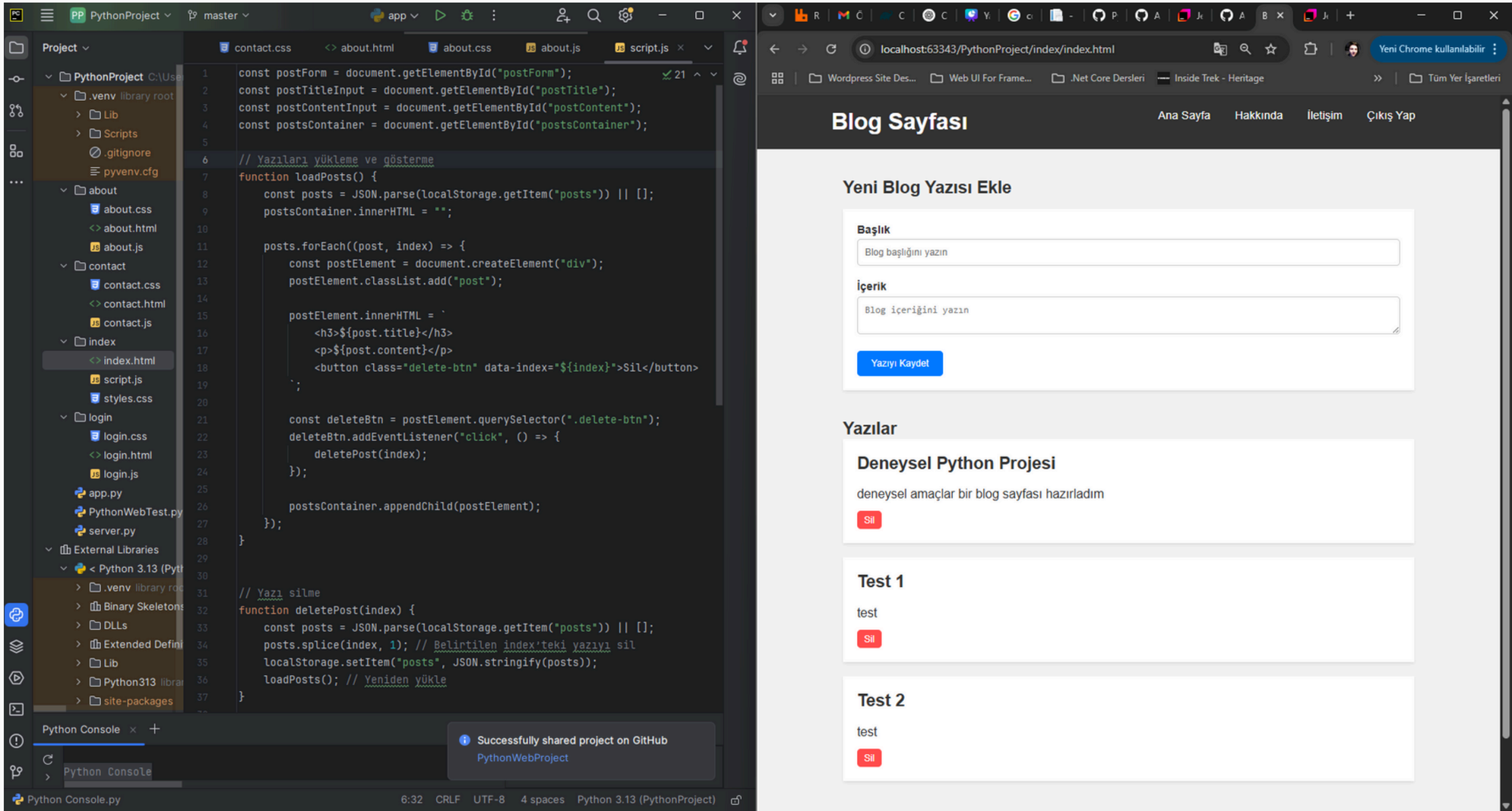
AndroidNoteApp is a user-friendly application designed to simplify note-taking and management. Users can quickly create, edit, and delete notes. The app stores data locally using Activity, RecyclerView, and SQLite technologies, offering an interface in line with Material Design principles. This project provides an opportunity for beginners in Android development to learn fundamental topics such as UI design, database operations, and RecyclerView, effectively managing basic CRUD operations.



SUNAN: AKIN İNCECİK

Python Web Project

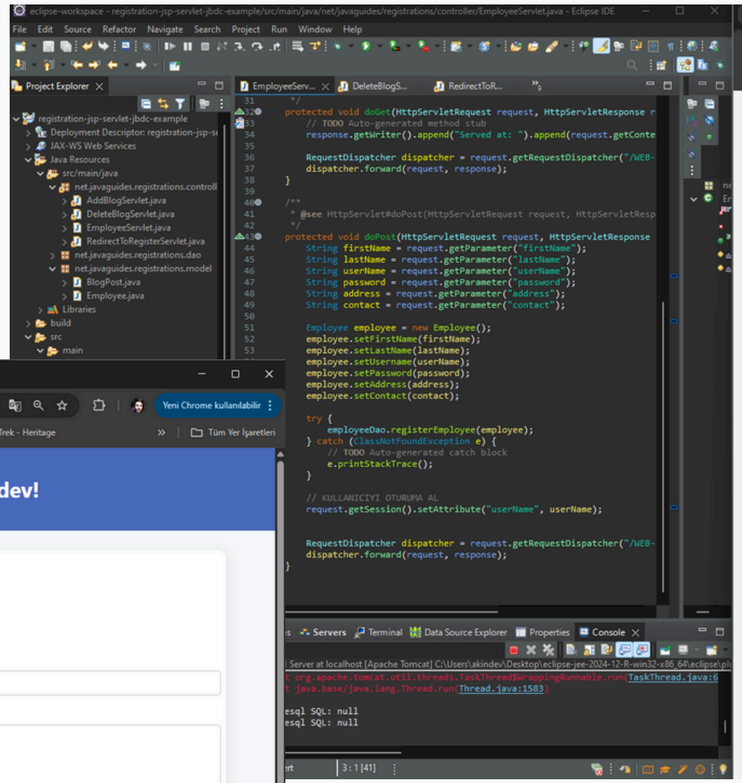
PythonWebProject is a project focused on the process of developing web applications using Python. Built with the Flask web framework, this project includes common features used in web applications, such as basic database operations, routing, HTML templates, and static file management. Data storage is performed with SQLite, and dynamic web pages are presented to the user. This project serves as a foundational example for those wanting to develop Python-based web applications.

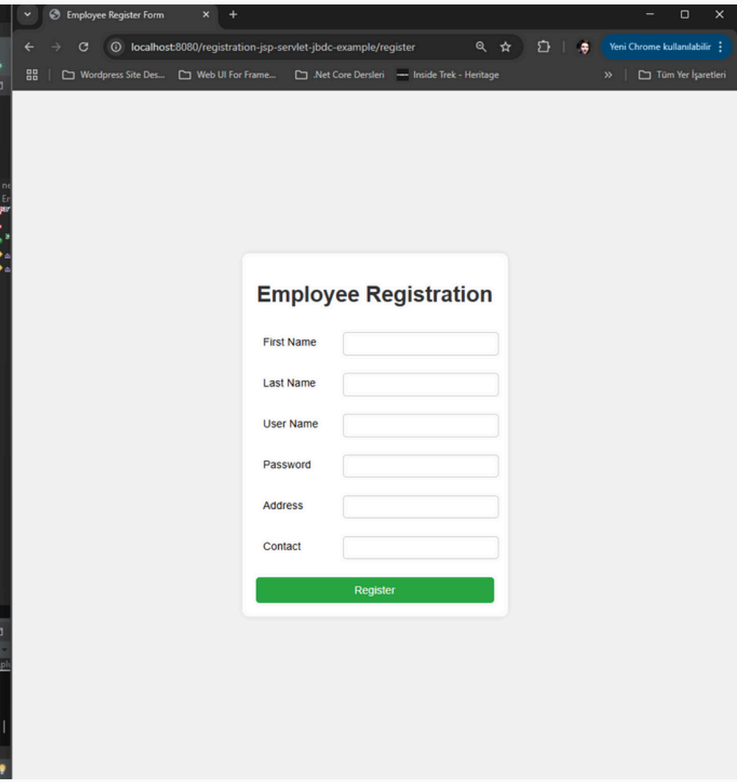


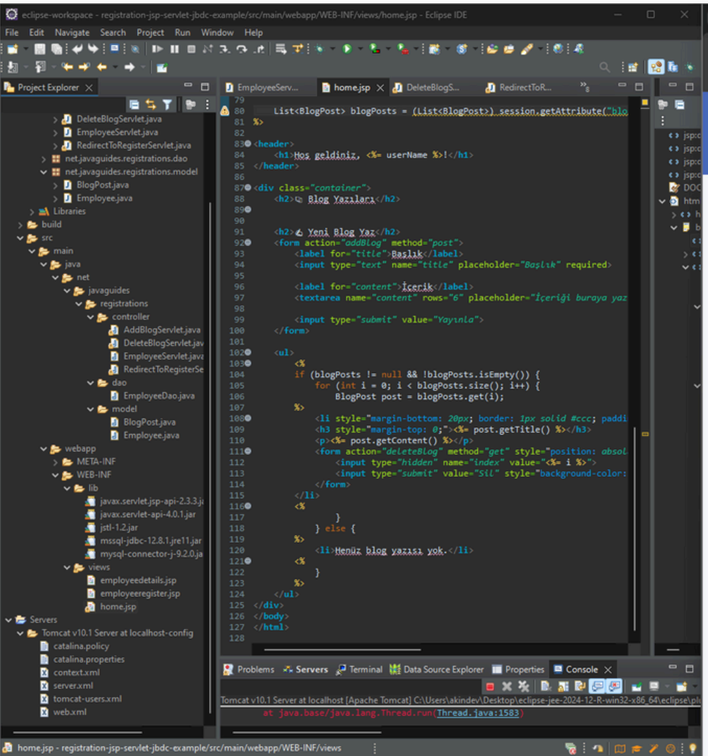
SUNAN: AKIN İNCECİK

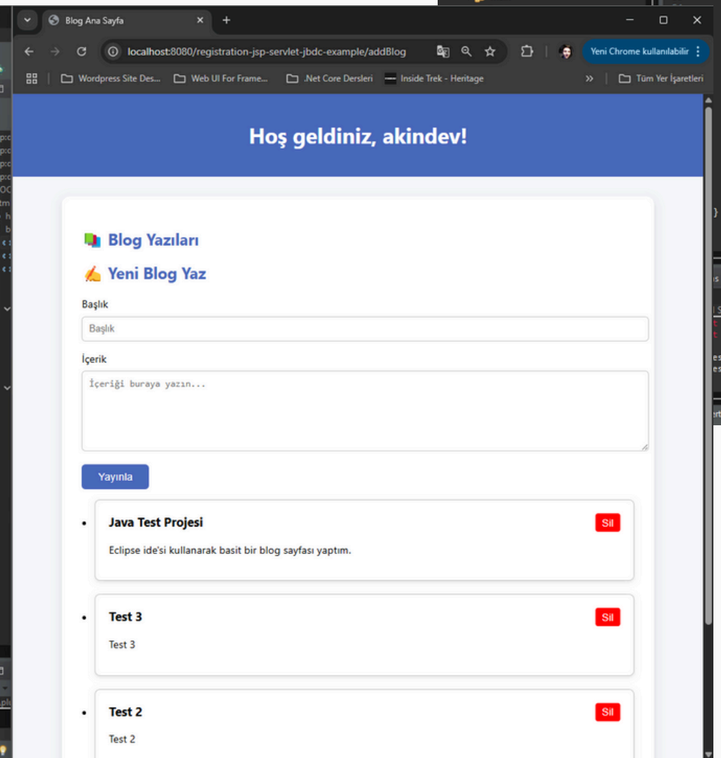
Java Web Project

IN THİS PROJECT, I DEVELOPED A USER REGİSTRATİON AND BLOG SYSTEM USİNG JAVA, JSP, AND SERVLET TECHNOLOGİES. REGİSTERED USERS CAN LOG İN TO THE APPLİCATİON RUNNİNG ON THE APACHE TOMCAT SERVER. THE HOMEPAGE FEATURES A SİMPLE BLOG İNTERFACE WHERE USERS CAN CREATE AND DELETE BLOG POSTS. DATABASE OPERATİONS ARE HANDLED USİNG JDBC, WİTH A CONNECTİON TO MS SQL SERVER. THE PROJECT WAS DEVELOPED USİNG THE ECLİPSE IDE.









SUNAN: AKIN İNCECİK

Thanks!