# VIRTUAL EVENT MANAGEMENT SYSTEM

## SE 410

Berkay Arda - 20200602004
Gözde Kahraman - 20210602032
Özge Özer - 20210602047

Supervisor
Dr. Serhat Uzunbayır

# Contents

# 1. Introduction

Efficiently organizing and managing events is very important for everyone in today's world. The Virtual Event Scheduler is a system that is designed for easy event scheduling for companies, organization and groups, and designed for both admins, staff and attendees.

Virtual Event offers two interfaces: a web application for attendees, and a desktop application for admins and staff. From the desktop application, events and users can be managed. From the web application, the attendees can view the events and can register to the events. The key features of the Virtual Event are role based access control, real time updates and notifications, conflict detection for overlapping events, filtering, and cross platform compatiblity.

The system uses RESTful APIs and a shared SQL database for communication between desktop application and the web application. Virtual Event is very user friendly for admins, staff and attendees and efficient for event scheduling.

# 2. Requirements

## 2.1. Non-Functional Requirements

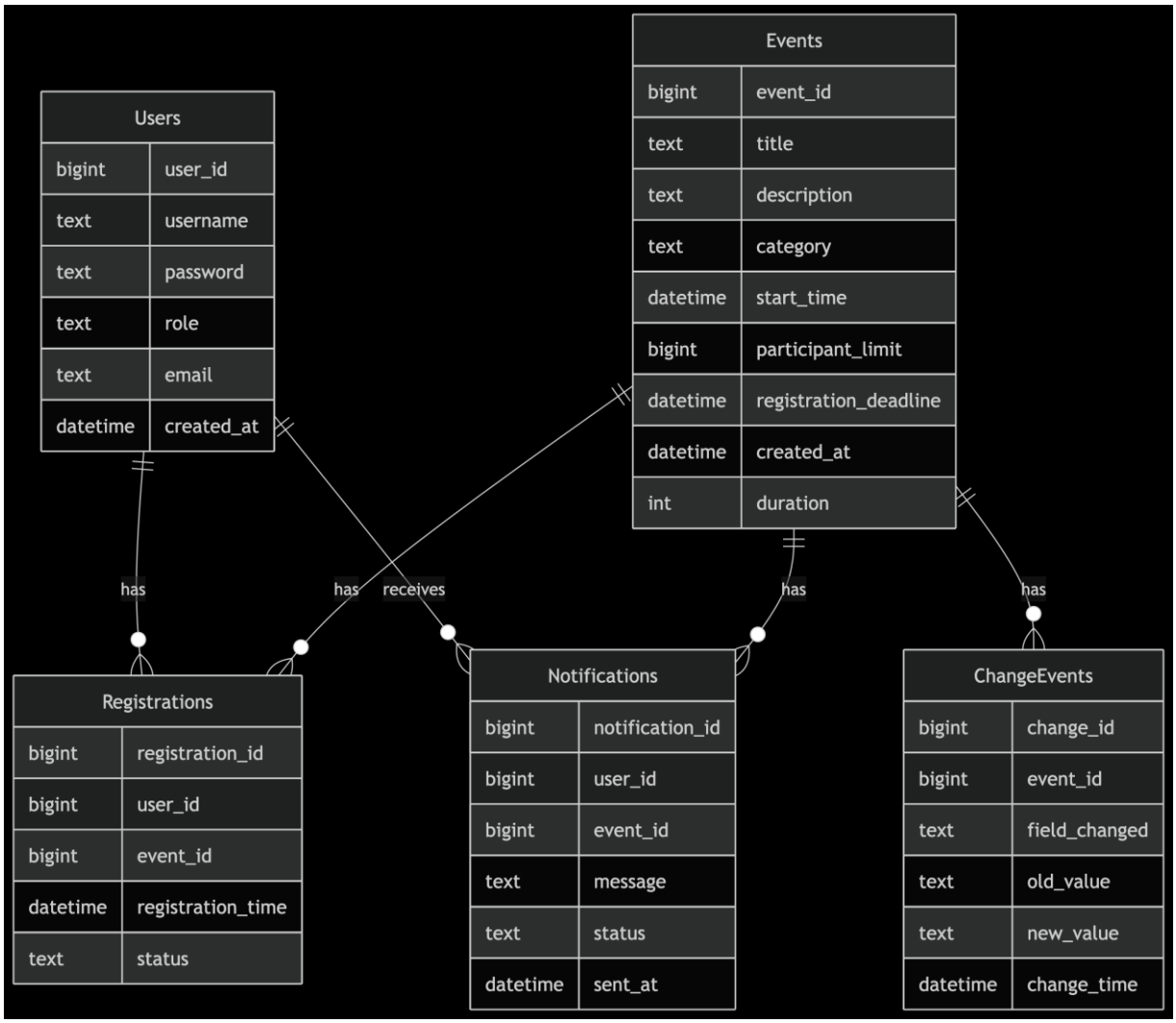
- The system should handle at least 200 concurrent users.
- The desktop app should launch in under 5 seconds.
- Desktop App must support Windows 10 and newer.
- The Web app must be compatible with browsers such as Google Chrome, Mozilla Firefox, Microsoft Edge.
- RESTful APIs will be used for desktop-web communication.
- RESTful APIs will handle CRUD operations (Create, Read, Update, Delete) for events and users.
- The system uses a shared SQL database.
- Data should be updated in real-time across desktop and web applications.

## 2.2. Functional Requirements

● Staff, admins and attendees must log in to access features by using a username and password.
● Role-based access control will be implemented so only admins can edit or delete events.
● Staff and admins must use the desktop application if they want to manage events.
● Admins and staff can create, edit, or delete events, assign event details like title, description, category, start/end time, participant limit, and deadline.
● All users can filter events (e.g., by date, status, number of attendees).
● Admins and staff can view a list of registered participants for each event.
● The system will send meeting links, event updates and notifications to attendees via application.
● Users can register to access event listings.
● All users can view upcoming and past events.
● All users can browse a list of attended events.
● All users can filter past and upcoming events.
● An attendee can register for only one event within the same timeline.
● Attendees can unregister from an event as long as at least one day remains before it starts.
● Attendees receive notifications for event updates (e.g., changes in time, venue, or cancellations).

# 3. Database Schema

## 3.1. Database Structure



## 3.2. The Scripts for Creating the Tables

```
-- Users Table
CREATE TABLE Users (
    user_id BIGINT NOT NULL,
    username TEXT NOT NULL,
    password TEXT NOT NULL,
    role TEXT NOT NULL,
    email TEXT NOT NULL,
    created_at DATETIME,
    PRIMARY KEY (user_id)
);

-- Events Table
CREATE TABLE Events (
    event_id BIGINT NOT NULL,
    title TEXT NOT NULL,
    description TEXT,
    category TEXT,
    start_time DATETIME NOT NULL,
    participant_limit BIGINT,
    registration_deadline DATETIME,
    created_at DATETIME,
    duration INTEGER NOT NULL,
    PRIMARY KEY (event_id)
);

-- ChangeEvents Table
CREATE TABLE ChangeEvents (
    change_id BIGINT NOT NULL,
    event_id BIGINT NOT NULL,
    field_changed TEXT NOT NULL,
    old_value TEXT,
    new_value TEXT,
    change_time DATETIME,
    PRIMARY KEY (change_id),
    FOREIGN KEY (event_id) REFERENCES
Events(event_id)
);

-- Registrations Table
CREATE TABLE Registrations (
    registration_id BIGINT NOT NULL,
    user_id BIGINT NOT NULL,
    event_id BIGINT NOT NULL,
    registration_time DATETIME,
    status TEXT,
    PRIMARY KEY (registration_id),
    FOREIGN KEY (user_id) REFERENCES
Users(user_id),
    FOREIGN KEY (event_id) REFERENCES
Events(event_id)
);

-- Notifications Table
CREATE TABLE Notifications (
    notification_id BIGINT NOT NULL,
    user_id BIGINT NOT NULL,
    event_id BIGINT NOT NULL,
    message TEXT NOT NULL,
    status TEXT,
    sent_at DATETIME,
    PRIMARY KEY (notification_id),
    FOREIGN KEY (user_id) REFERENCES
Users(user_id),
    FOREIGN KEY (event_id) REFERENCES
Events(event_id)
);
```

# 4. LINQ Queries

## 4.1. Desktop Application LINQ

**4.1.1. Event Filtering (ManageScreen.cs) :** Dynamically filters events in the DataGridView based on user input (title, category, date, etc.).

```
// Filter events by title (case-insensitive)
var query = eventsTable.AsEnumerable()
    .Where(row =>
        row["title"] != DBNull.Value &&
        row.Field<string>("title").ToLower().Contains(titleFilter));

// Filter events by category (case-insensitive)
.Where(row =>
    row["category"] != DBNull.Value &&
    row.Field<string>("category").ToLower().Contains(categoryFilter));

// Filter events by start date
.Where(row =>
    row.Field<DateTime>("start_time").Date == filterDate);

// Filter events by duration
.Where(row =>
    row.Field<int>("duration") == duration);

// Filter events by participant slots
.Where(row =>
    row["participant_limit"] != DBNull.Value &&
    row.Field<long>("participant_limit") == slots);

// Filter events by registration deadline
.Where(row =>
    row["registration_deadline"] != DBNull.Value &&
    row.Field<DateTime>("registration_deadline").Date == filterDate);
```

**4.1.2. Total Slot Calculator (ManageScreen.cs) :** Sums up participant_limit for all events in a given category.

```
var matchingRows = eventsTable.AsEnumerable()
    .Where(row =>
        row["category"] != DBNull.Value &&
        row["participant_limit"] != DBNull.Value &&
        row.Field<string>("category").ToLower() == category);

var totalSlots = matchingRows.Sum(row => Convert.ToInt64(row["participant_limit"]));
```

**4.1.3. Event Selection for Editing (ManageScreen.cs) :** Retrieves the selected event's data for editing.

```
// Get selected event ID from DataGridView
long selectedEventId = (from DataGridViewRow row in dataGridView1.SelectedRows
                        select Convert.ToInt64(row.Cells["event_id"].Value))
                        .FirstOrDefault();

// Find the corresponding DataRow
var rowToEdit = (from DataRow row in eventsTable.Rows
                 where Convert.ToInt64(row["event_id"]) == selectedEventId
                 select row).FirstOrDefault();
```

**4.1.4. Sorting Events by Date (ManageScreen.cs) :** Sorts events chronologically after updates.

```
var sortedEvents = from DataRow row in eventsTable.Rows
                   orderby Convert.ToDateTime(row["start_time"])
                   select row;
```

## 4. 2. Web Application LINQ

**4.2.1. Checking if Events Exist (Events.aspx.cs) :** Checks if _allEvents collection has any elements before binding to the grid.

```
if (_allEvents != null && _allEvents.Any())
```

**4.2.2. Filtering Events by Title (Events.aspx.cs) :** Filters events where the title contains the search text (case-insensitive).

```
filteredEvents = filteredEvents.Where(ev =>
    ev.Title != null &&
    ev.Title.ToLower().Contains(titleFilter));
```

**4.2.3. Filtering Events by Category (Events.aspx.cs) :** Filters events where the category matches the search text.

```
filteredEvents = filteredEvents.Where(ev =>
    ev.Category != null &&
    ev.Category.ToLower().Contains(categoryFilter));
```

**4.2.4. Filtering Events by Start Date (Events.aspx.cs) :** Filters events occurring on a specific date.

```
filteredEvents = filteredEvents.Where(ev =>
    ev.StartTime.Date == startDate.Date);
```

**4.2.5. Filtering Events by Duration (Events.aspx.cs) :** Filters events with an exact duration match.

```
filteredEvents = filteredEvents.Where(ev => ev.Duration == duration);
```

**4.2.6. Filtering Events by Participant Slots (Events.aspx.cs) :** Filters events with a specific participant limit.

```
filteredEvents = filteredEvents.Where(ev => ev.ParticipantLimit == slots);
```

**4.2.7. Checking Filter Results (Events.aspx.cs) :** Checks if any events survived the filters and updates UI messaging.

```
var result = filteredEvents.ToList();
lblMessage.Text = result.Any()
    ? $"Showing {result.Count} events"
    : "No events matched your search criteria.";
```

## 5. Delegates and Events

AddEventForm class utilizes delegates and events to implement a publisher-subscriber duo, enabling loosely coupled communication between components when a new event is created and added to the database.

- public delegate void EventAddedHandler(Event newEvent);
- public static event EventAddedHandler OnEventAdded;
- OnEventAdded += AddEventForm_OnEventAdded;
- OnEventAdded?.Invoke(newEvent);

The EventAddedHandler delegate defines a method that takes an Event object and returns void. It sets the signature for methods that use the OnEventAdded event. The OnEventAdded event is static, so it's shared across all instances of the form. This allows other forms or classes to subscribe globally. It acts as a multicast delegate, so multiple methods can listen to it. In the constructor, the method AddEventForm_OnEventAdded is subscribed to the event. The handler method shows a message box confirming the event was added.

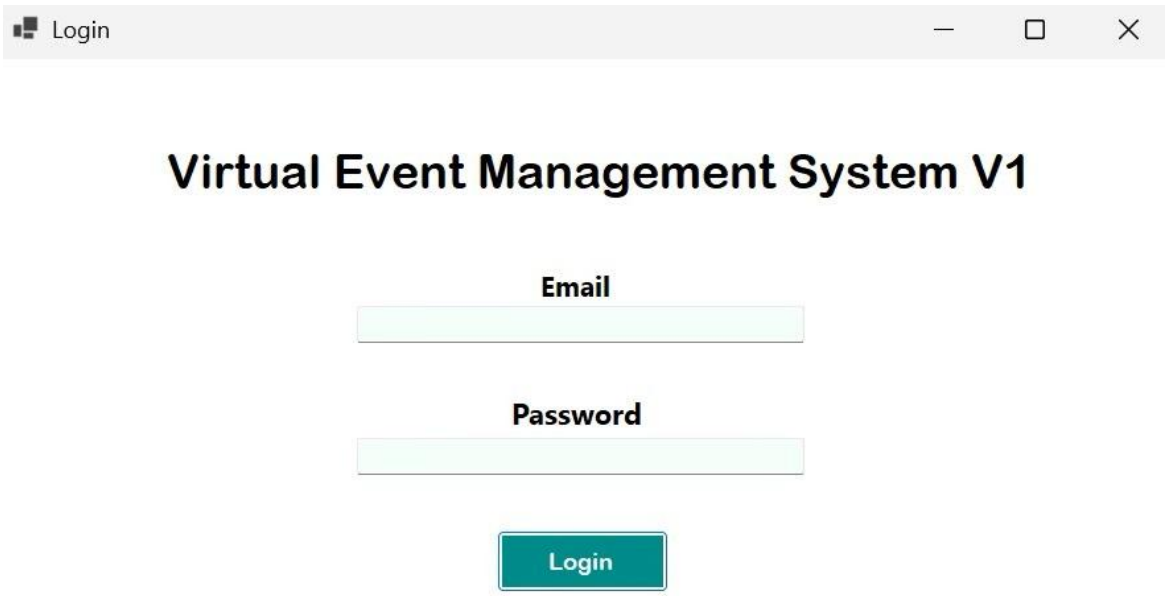
## 6. MVC Structure

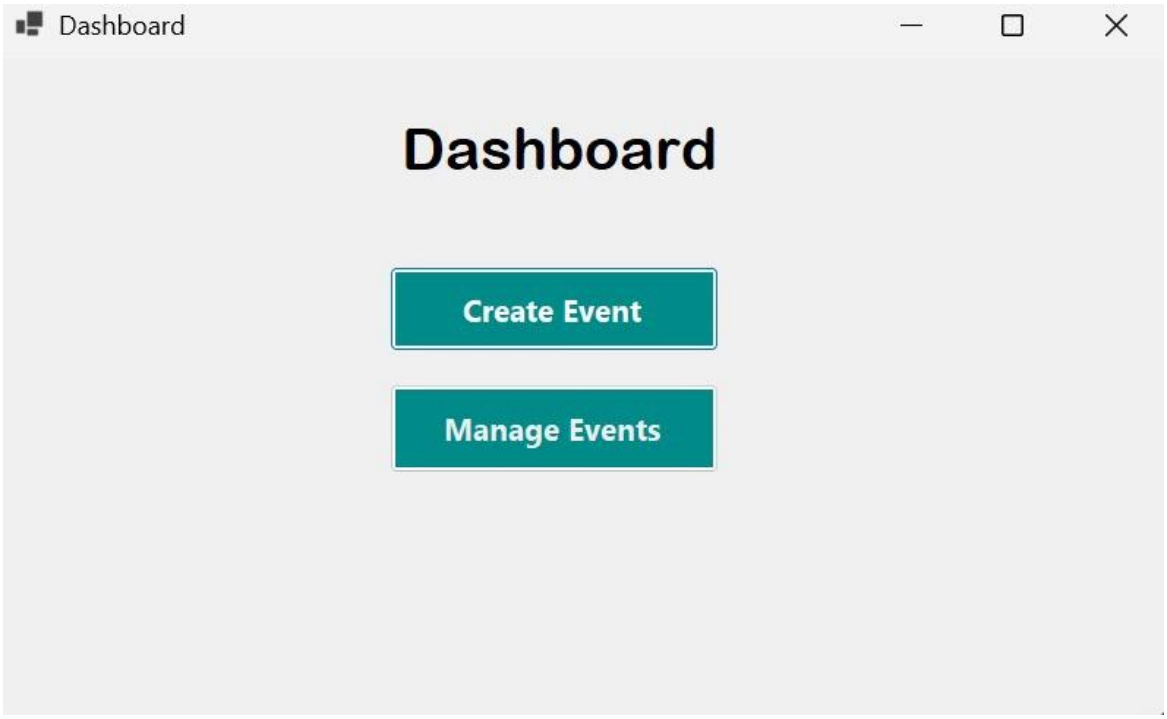
The Virtual Event project follows the MVC architecture, which helps organize the system by separating the data layer, user interface, and application logic. ASP.NET Web API layers are combined for backend operations and Web Forms for the front-end and this creates a structured and maintainable platform.The model layer defines the structure of the data through C# classes like User, LoginDto, and EventModel. Some models are placed in shared namespaces, while others are defined directly inside controllers for simplicity. Each model represents either a database entity or a data transfer object. Data annotations ensure proper mapping to the SQLite database and keeps the model definitions in sync with the schema. Controllers has the core logic of the application. Each one is responsible for a specific area and handles logins, managing events, processing registrations, or sending notifications. They manually manage SQLite connections, run SQL queries, and return HTTP responses. This direct approach keeps things straightforward and efficient. The user interface is built using ASP.NET Web Forms. Pages like Login, Events, and Notifications use HttpClient to fetch data from the API and update the UI dynamically. User sessions maintain identity across all pages, and UI elements like GridView and TextBox display the data. This MVC setup keeps the system clean and modular. Models manage the data structure, controllers handle logic and communication with the database, and views present the content to users. This approach supports maintainability and allows easier expansions to the project.

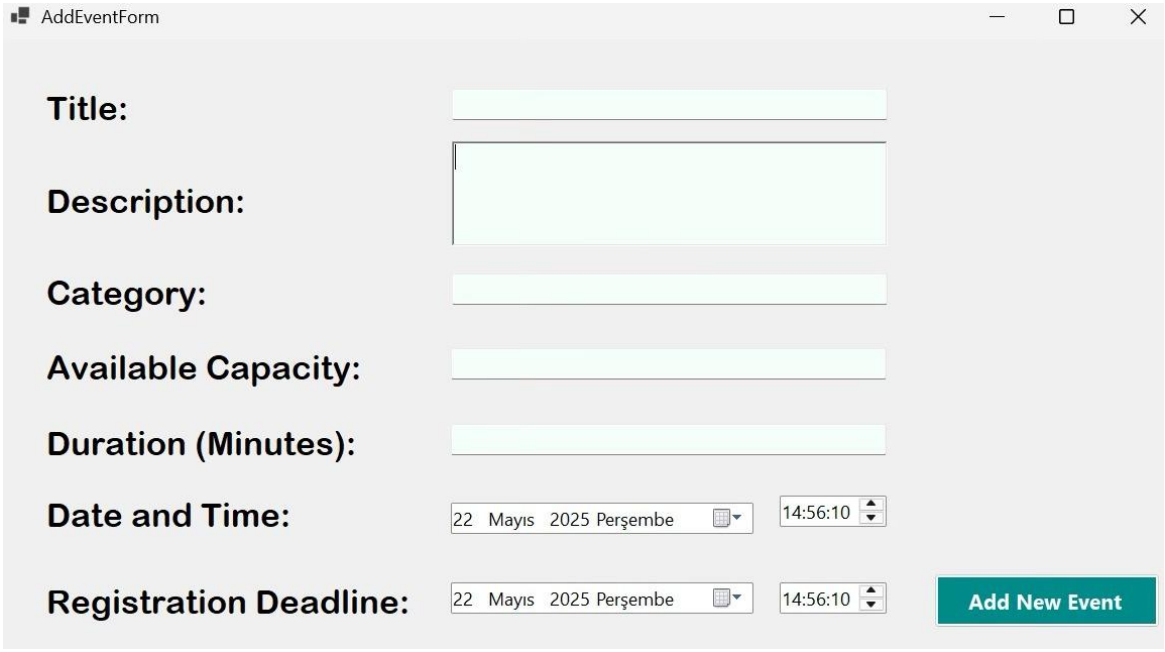
## 7. Sources

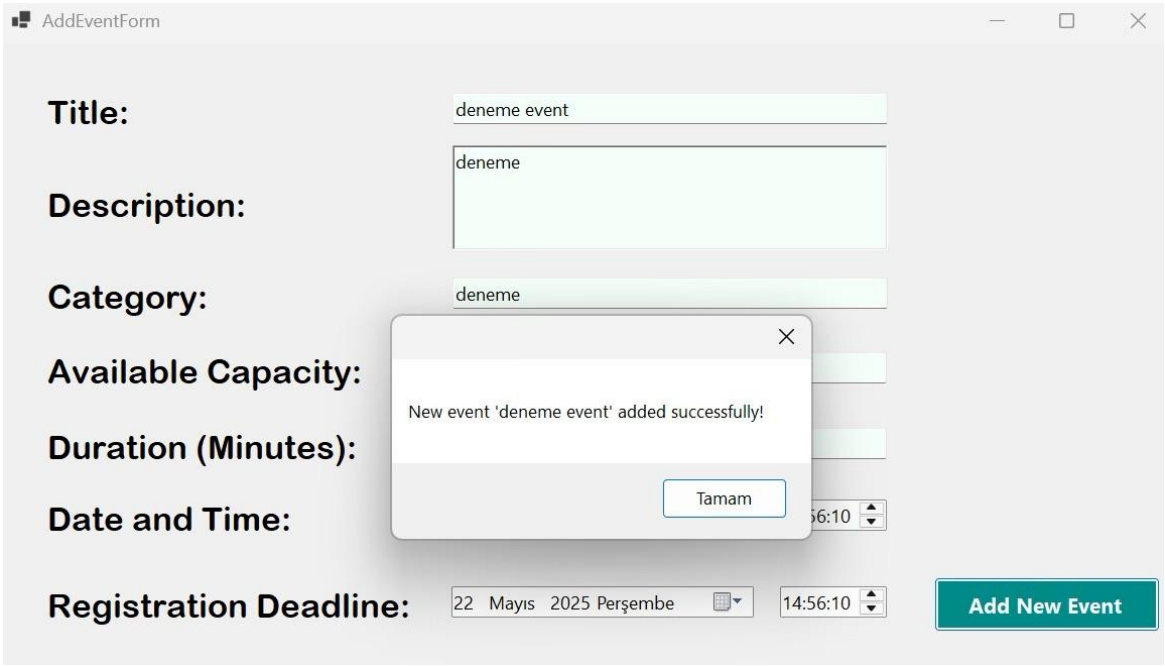
**GitHub repository link:** https://github.com/serhatuzunbayir/VirtualEvent
**How to run the project:** https://github.com/serhatuzunbayir/VirtualEvent/blob/main/README.md
**Video Link for V1:** https://youtu.be/2Yr5S9TTKfg?si=wAedOQLmUVG_n1Gc
**Video Link for V2:** https://youtu.be/7n4ZIePe5co

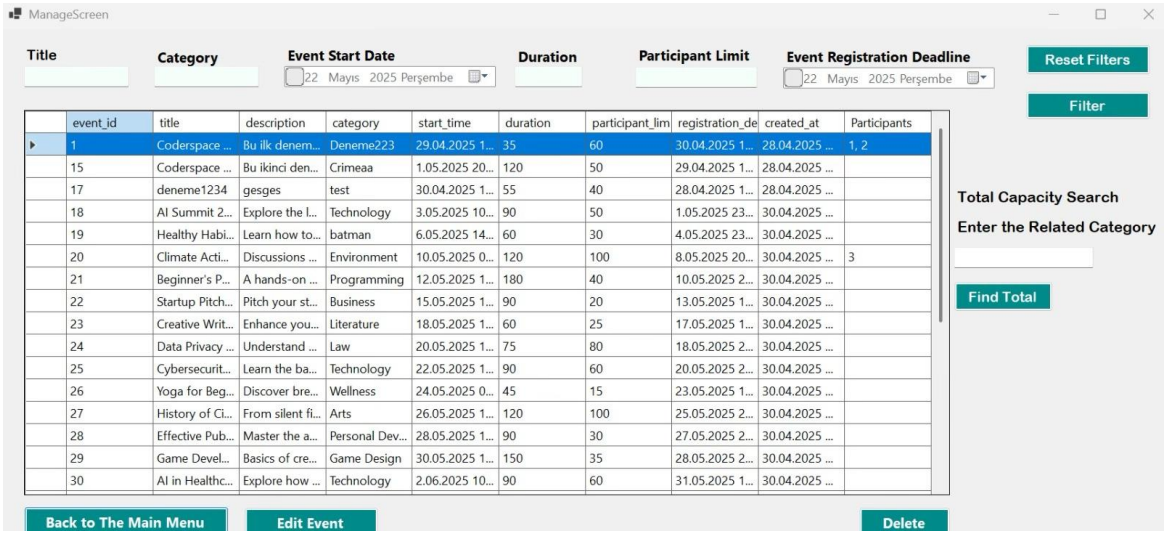
## 8. User Interface Design

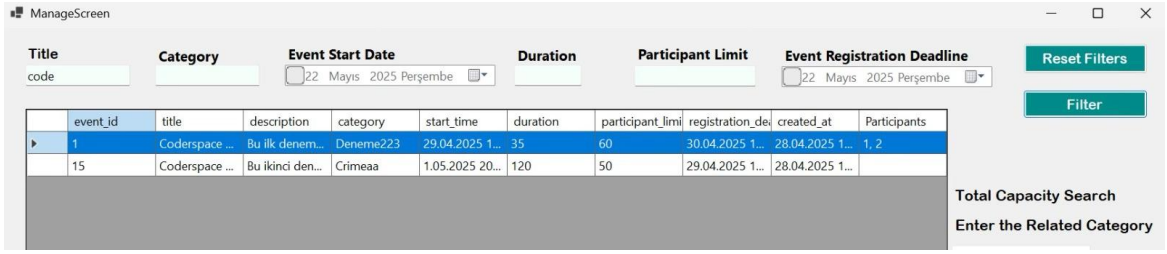


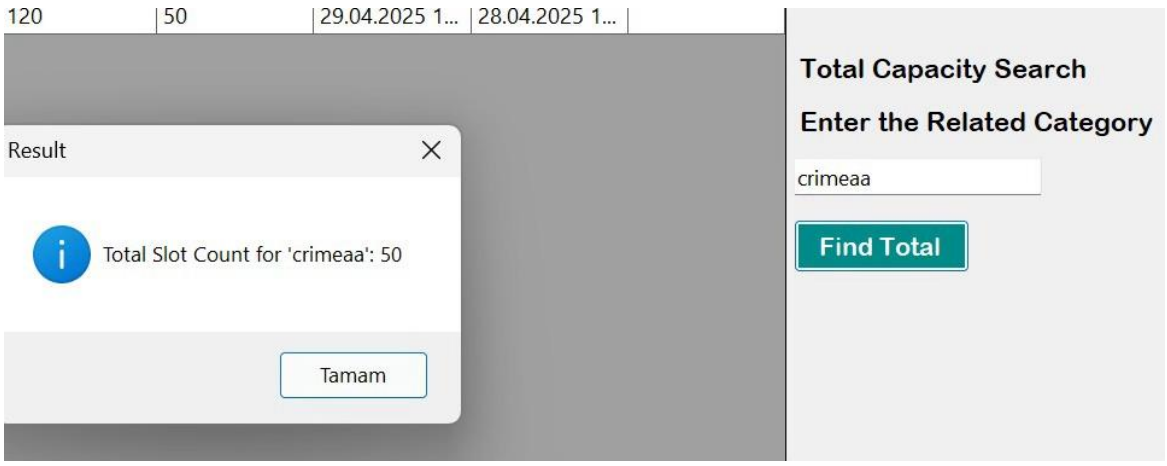



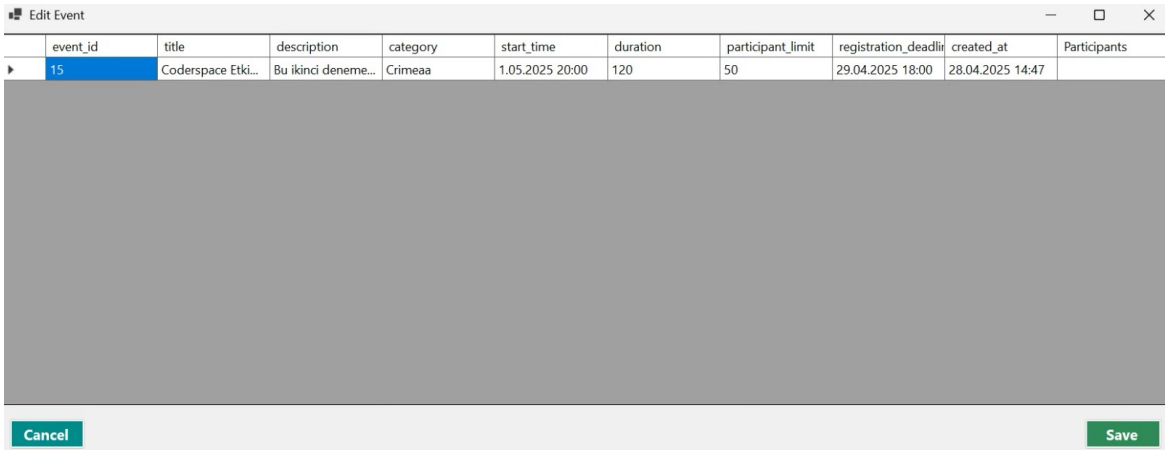

AddEventForm
Title:
Description:
Category:
Available Capacity:
Duration (Minutes):
Date and Time:
22 Mayıs 2025 Perşembe
14:56:10
Registration Deadline:
22 Mayıs 2025 Perşembe
14:56:10
Add New Event

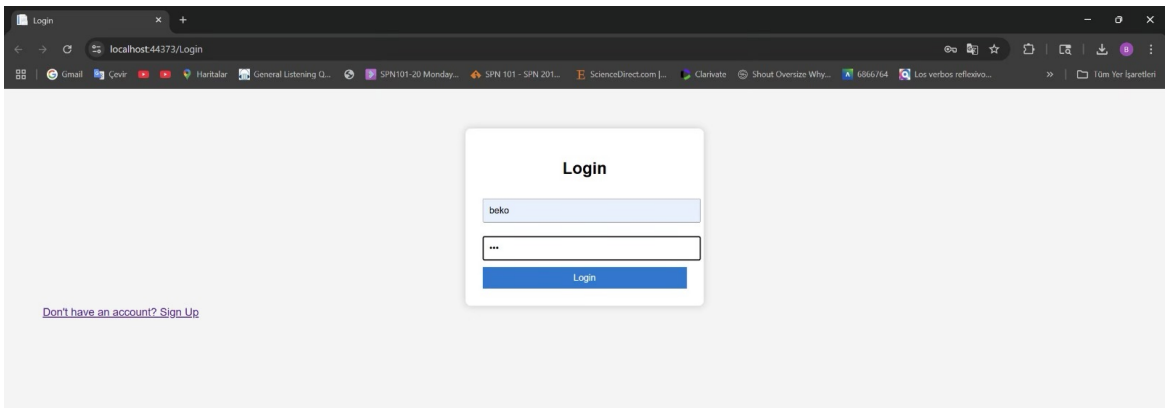

AddEventForm
Title:
deneme event
Description:
deneme
Category:
deneme
Available Capacity:
Duration (Minutes):
Date and Time:
New event 'deneme event' added successfully!
Tamam
Registration Deadline:
22 Mayıs 2025 Perşembe
14:56:10
Add New Event

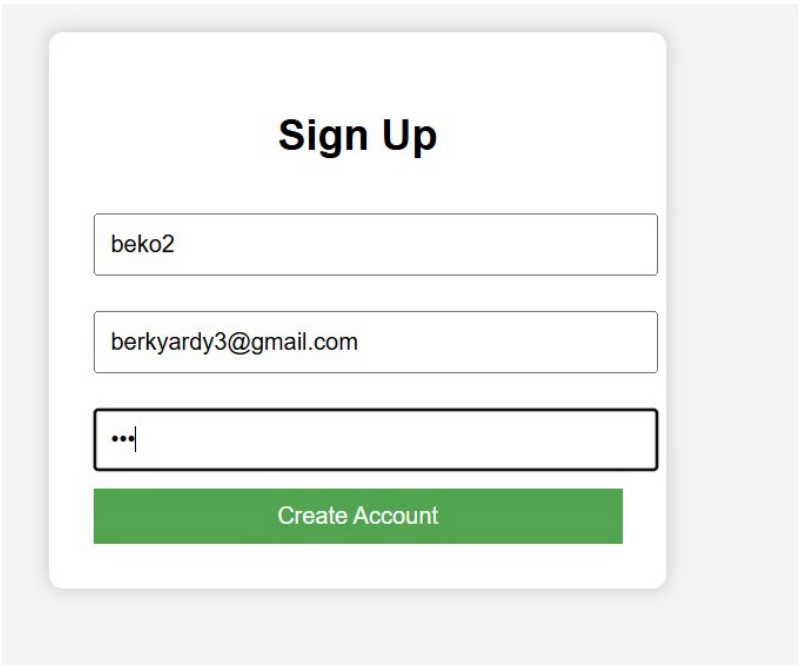

ManageScreen
Title
Category
Event Start Date
22 Mayıs 2025 Perşembe
Duration
Participant Limit
Event Registration Deadline
22 Mayıs 2025 Perşembe
Reset Filters
Filter
event_id
title
description
category
start_time
duration
participant_lim
registration_de
created_at
Participants
Total Capacity Search
Enter the Related Category
Find Total
Back to The Main Menu
Edit Event
Delete

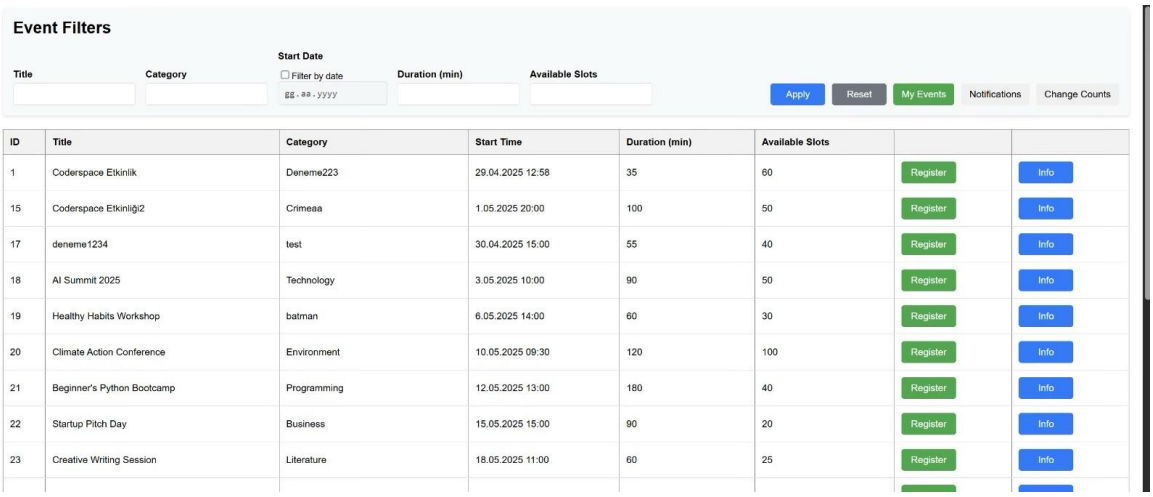

ManageScreen
Title
code
Category
Event Start Date
22 Mayıs 2025 Perşembe
Duration
Participant Limit
Event Registration Deadline
22 Mayıs 2025 Perşembe
Reset Filters
Filter
event_id
title
description
category
start_time
duration
participant_limi
registration_de
created_at
Participants
1
Coderspace ...
Bu ilk denem...
Deneme223
29.04.2025 1...
35
60
30.04.2025 1...
28.04.2025 1...
1, 2
15
Coderspace ...
Bu ikinci den...
Crimeaa
1.05.2025 20...
120
50
29.04.2025 1...
28.04.2025 1...
Total Capacity Search
Enter the Related Category

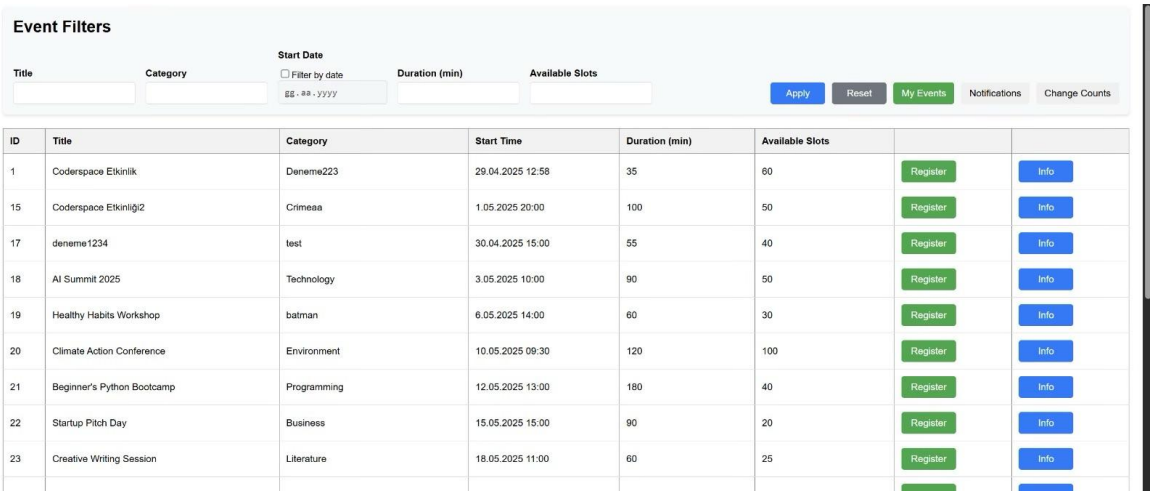

120
50
29.04.2025 1...
28.04.2025 1...
Result
Total Slot Count for 'crimeaa': 50
Tamam
Total Capacity Search
Enter the Related Category
crimeaa
Find Total


Edit Event
event_id
title
description
category
start_time
duration
participant_limit
registration_deadlir
created_at
Participants
15
Coderspace Etki...
Bu ikinci deneme...
Crimeaa
1.05.2025 20:00
120
50
29.04.2025 18:00
28.04.2025 14:47
Cancel
Save


Login
localhost:44373/Login
Login
beko
Login
Don't have an account? Sign Up


Sign Up
beko2
berkyardy3@gmail.com
Create Account


Event Filters
Title
Category
Start Date
Filter by date
gg.aa.yyyy
Duration (min)
Available Slots
Apply
Reset
My Events
Notifications
Change Counts
ID
Title
Category
Start Time
Duration (min)
Available Slots
1 Coderspace Etkinlik Deneme223 29.04.2025 12:58 35 60 Register Info
15 Coderspace Etkinliği2 Crimeaa 1.05.2025 20:00 100 50 Register Info
17 deneme1234 test 30.04.2025 15:00 55 40 Register Info
18 AI Summit 2025 Technology 3.05.2025 10:00 90 50 Register Info
19 Healthy Habits Workshop batman 6.05.2025 14:00 60 30 Register Info
20 Climate Action Conference Environment 10.05.2025 09:30 120 100 Register Info
21 Beginner's Python Bootcamp Programming 12.05.2025 13:00 180 40 Register Info
22 Startup Pitch Day Business 15.05.2025 15:00 90 20 Register Info
23 Creative Writing Session Literature 18.05.2025 11:00 60 25 Register Info


Event Filters
Title
Category
Start Date
Filter by date
gg.aa.yyyy
Duration (min)
Available Slots
Apply
Reset
My Events
Notifications
Change Counts
ID
Title
Category
Start Time
Duration (min)
Available Slots
1 Coderspace Etkinlik Deneme223 29.04.2025 12:58 35 60 Register Info
15 Coderspace Etkinliği2 Crimeaa 1.05.2025 20:00 100 50 Register Info
17 deneme1234 test 30.04.2025 15:00 55 40 Register Info
18 AI Summit 2025 Technology 3.05.2025 10:00 90 50 Register Info
19 Healthy Habits Workshop batman 6.05.2025 14:00 60 30 Register Info
20 Climate Action Conference Environment 10.05.2025 09:30 120 100 Register Info
21 Beginner's Python Bootcamp Programming 12.05.2025 13:00 180 40 Register Info
22 Startup Pitch Day Business 15.05.2025 15:00 90 20 Register Info
23 Creative Writing Session Literature 18.05.2025 11:00 60 25 Register Info

# Event Description

Bu ilk deneme eventidir katılım zorunlu değildir

**Available Slots: 58**

Back to Events

**Event Filters**

Title: coder | Category | Start Date: Filter by date, gg.aa.yyyy | Duration (min) | Available Slots

Apply | Reset | My Events | Notifications | Change Counts

| ID | Title | Category | Start Time | Duration (min) | Available Slots | | |
|---|---|---|---|---|---|---|---|
| 1 | Coderspace Etkinlik | Deneme223 | 29.04.2025 12:58 | 35 | 60 | Register | Info |
| 15 | Coderspace Etkinliği2 | Crimeaa | 1.05.2025 20:00 | 100 | 50 | Register | Info |

Showing 2 events

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 40 | kontrol 1 | a | 21.05.2025 15:00 | 1 | 1 | Register | Info |
| 41 | Kontrol 2 | a | 21.05.2025 15:00 | 1 | 1 | Register | Info |
| 42 | aaa | a | 23.05.2025 15:00 | 1 | 2 | Register | Info |
| 43 | özge | 1 | 21.05.2025 18:49 | 1 | 1 | Register | Info |
| 44 | deneme event | deneme | 24.05.2025 14:56 | 1 | 1 | Register | Info |

Etkinliğe başarıyla kayıt olundu.

← Geri Dön

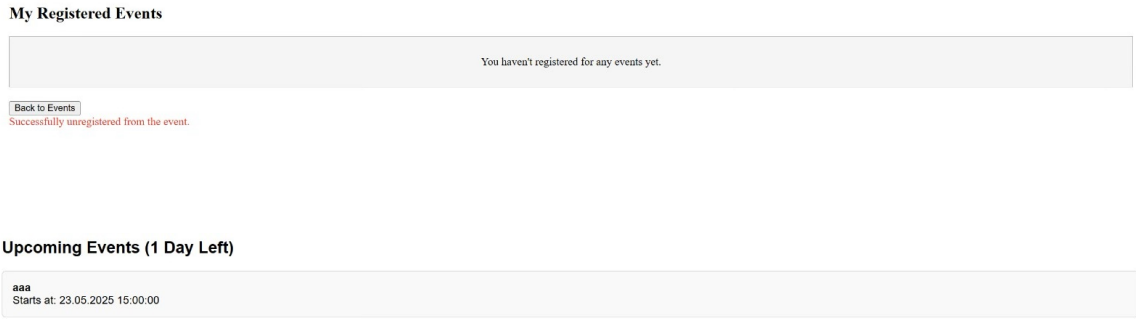
**Event Change History**

| Event ID | Title | Field Changed | Old Value | New Value | Change Time |
|---|---|---|---|---|---|
| 15 | Coderspace Etkinliği2 | duration | 120 | 100 | 22.05.2025 11:58 |
| 42 | aaa | start_time | 21.05.2025 15:00:00 | 23.05.2025 15:00:00 | 22.05.2025 10:01 |
| 42 | aaa | participant_limit | 1 | 2 | 22.05.2025 10:01 |
| 37 | Photography Essentials | start_time | 16.06.2025 17:00:00 | 23.06.2025 17:00:00 | 22.05.2025 09:59 |
| 19 | Healthy Habits Workshop | category | adana | batman | 20.05.2025 18:27 |
| 15 | Coderspace Etkinliği2 | category | Crime | Crimeaa | 20.05.2025 18:17 |
| 19 | Healthy Habits Workshop | category | Health | adana | 20.05.2025 18:15 |
| 1 | Coderspace Etkinlik | category | Deneme22 | Deneme223 | 20.05.2025 18:11 |
| 1 | Coderspace Etkinlik | category | Deneme | Deneme22 | 20.05.2025 18:01 |
| 1 | Coderspace Etkinlik | duration | 0 | 35 | 20.05.2025 17:48 |

**My Registered Events**

| Action | Event Title | Category | Start Time |
|---|---|---|---|
| Unregister | Photography Essentials | Visual Arts | 23.06.2025 17:00 |

Back to Events




# 9. Tests

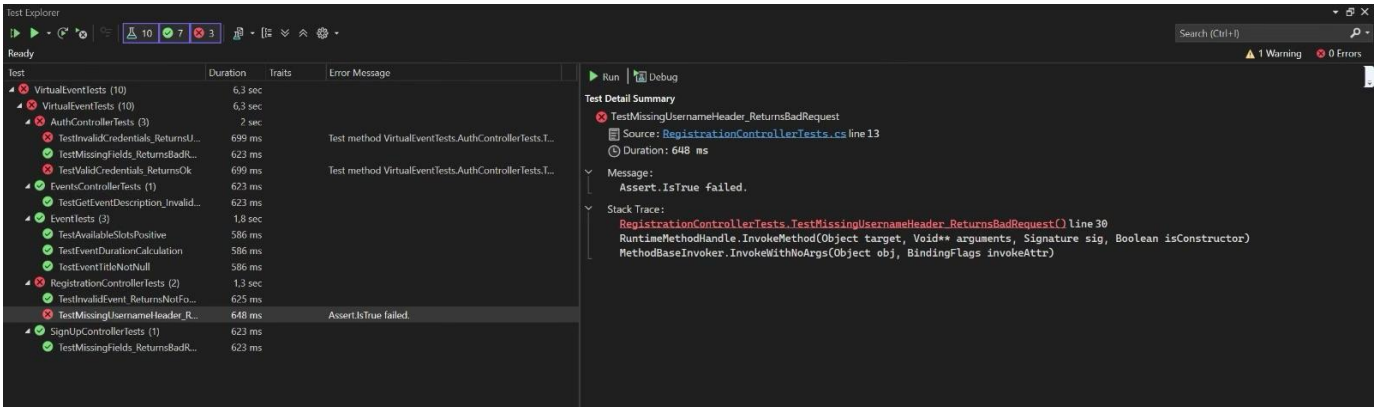
```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using Microsoft.AspNetCore.Mvc;
using System.Threading.Tasks;
using WEBAPI.Controllers;

namespace VirtualEventTests
{
    [TestClass]
    public class AuthControllerTests
    {
        [TestMethod]
        public async Task TestMissingFields_ReturnsBadRequest()
        {
            var controller = new AuthController();
            var request = new AuthController.LoginDto
            {
                Username = "",
                Password = ""
            };

            var result = await controller.Login(request);
            Assert.IsInstanceOfType(result, typeof(BadRequestObjectResult));
        }

        [TestMethod]
        public async Task TestInvalidCredentials_ReturnsUnauthorized()
        {
            var controller = new AuthController();
            var request = new AuthController.LoginDto
            {
                Username = "nonexistent_user",
                Password = "wrong_password"
            };
```

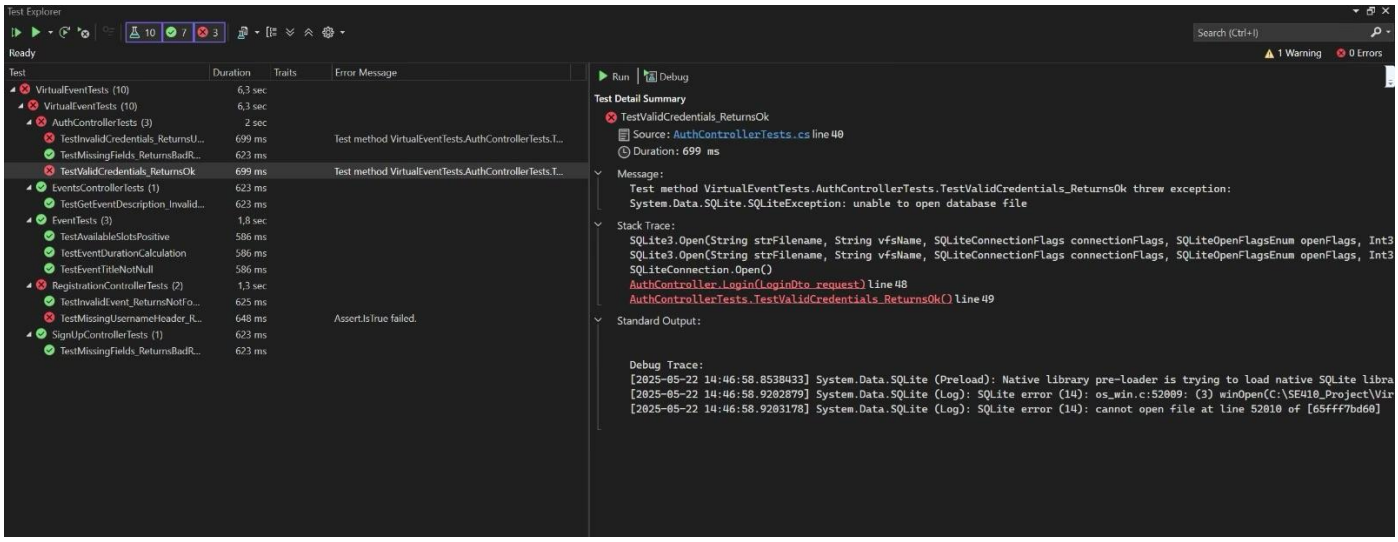
```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using System;
using VirtualEvent_SE410;

namespace VirtualEventTests
{
    [TestClass]
    public class EventTests
    {
        [TestMethod]
        public void TestEventDurationCalculation()
        {
            // Arrange
            var ev = new Event
            {
                StartTime = new DateTime(2025, 5, 21, 14, 0, 0),
                Duration = 120
            };

            // Act
            DateTime expectedEndTime = ev.StartTime.AddMinutes(ev.Duration);

            // Assert
            Assert.AreEqual(new DateTime(2025, 5, 21, 16, 0, 0), expectedEndTime);
        }

        [TestMethod]
        public void TestEventTitleNotNull()
        {
            // Arrange
            var ev = new Event
            {
                Title = "Tech Meetup"
            };

            // Assert
            Assert.IsFalse(string.IsNullOrEmpty(ev.Title), "Event title should not be null or empty.");
        }

        [TestMethod]
        public void TestAvailableSlotsPositive()
        {
            // Arrange
            var ev = new Event
            {
                AvailableSlots = 50
            };

            // Assert
            Assert.IsTrue(ev.AvailableSlots > 0, "AvailableSlots must be greater than zero.");
        }
    }
}
```

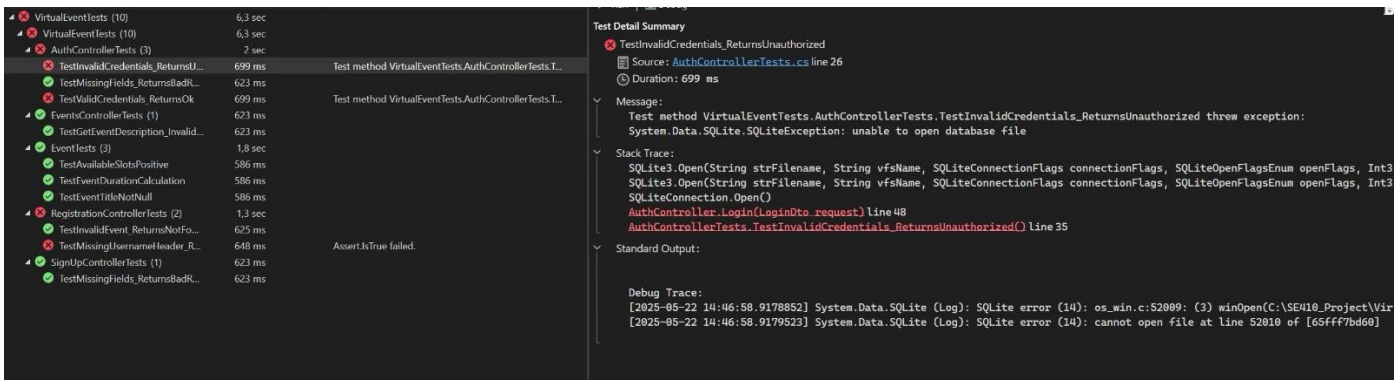
```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using Microsoft.AspNetCore.Mvc;
using WEBAPI.Controllers;
using VirtualEventAPI.Controllers;

namespace VirtualEventTests
{
    [TestClass]
    public class EventsControllerTests
    {

        [TestMethod]
        public void TestGetEventDescription_InvalidId_ReturnsNotFound()
        {
            var controller = new EventsController();
            var result = controller.GetEventDescription(999999); // Geçersiz ID
            Assert.IsInstanceOfType(result, typeof(NotFoundObjectResult));
        }
    }
}
```

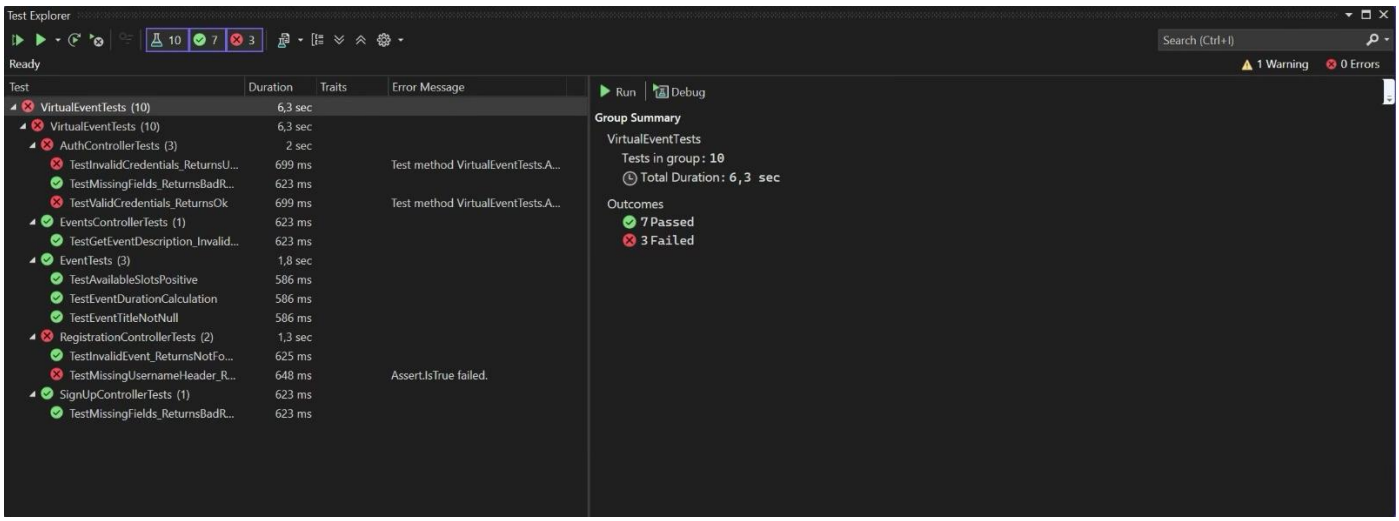
```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using VirtualEventAPI.Controllers;


namespace VirtualEventTests
{
    [TestClass]
    public class RegistrationControllerTests
    {
        [TestMethod]
        public void TestMissingUsernameHeader_ReturnsBadRequest()
        {
            // Arrange
            var controller = new RegistrationController();

            controller.ControllerContext = new ControllerContext
            {
                HttpContext = new DefaultHttpContext()
            };

            // NOT: Username hiç eklenmiyor burada
            var request = new VirtualEventAPI.Controllers.RegistrationController.RegisterRequest { EventId = 1 };

            // Act
            var result = controller.Register(request);

            // Assert
            Assert.IsTrue(result is BadRequestResult || result is BadRequestObjectResult);
        }


        [TestMethod]
        public void TestInvalidEvent_ReturnsNotFound()
        {
            // Arrange
            var controller = new RegistrationController();

            controller.ControllerContext = new ControllerContext
            {
                HttpContext = new DefaultHttpContext()
            };

            controller.ControllerContext.HttpContext.Request.Headers["Username"] = "validuser";

            var request = new RegistrationController.RegisterRequest
            {
                EventId = 999999 // büyük ihtimalle olmayan bir event id
            };

            // Act
            var result = controller.Register(request);

            // Assert
            Assert.IsInstanceOfType(result, typeof(NotFoundObjectResult));
        }
    }
}
```

```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using Microsoft.AspNetCore.Mvc;
using WEBAPI.Controllers;
using VirtualEventAPI.Controllers;

namespace VirtualEventTests
{
    [TestClass]
    public class SignUpControllerTests
    {
        [TestMethod]
        public void TestMissingFields_ReturnsBadRequest()
        {
            // Arrange
            var controller = new SignUpController();
            var request = new SignUpController.SignUpRequest
            {
                Username = "",
                Password = "",
                Email = ""
            };

            // Act
            var result = controller.Register(request);

            // Assert
            Assert.IsInstanceOfType(result, typeof(BadRequestObjectResult));
        }
    }
}

[assembly: Parallelize(Scope = ExecutionScope.MethodLevel)]
```


Test Explorer
Ready
1 Warning
0 Errors
Test
Duration
Traits
Error Message
VirtualEventTests (10) 6,3 sec
VirtualEventTests (10) 6,3 sec
AuthControllerTests (3) 2 sec
TestInvalidCredentials_ReturnsU... 699 ms Test method VirtualEventTests.AuthControllerTests.T...
TestMissingFields_ReturnsBadR... 623 ms
TestValidCredentials_ReturnsOk 699 ms Test method VirtualEventTests.AuthControllerTests.T...
EventsControllerTests (1) 623 ms
TestGetEventDescription_Invalid... 623 ms
EventTests (3) 1,8 sec
TestAvailableSlotsPositive 586 ms
TestEventDurationCalculation 586 ms
TestEventTitleNotNull 586 ms
RegistrationControllerTests (2) 1,3 sec
TestInvalidEvent_ReturnsNotFo... 625 ms
TestMissingUsernameHeader_R... 648 ms Assert.IsTrue failed.
SignUpControllerTests (1) 623 ms
TestMissingFields_ReturnsBadR... 623 ms
Run
Debug
Test Detail Summary
TestMissingUsernameHeader_ReturnsBadRequest
Source: RegistrationControllerTests.cs line 13
Duration: 648 ms
Message:
Assert.IsTrue failed.
Stack Trace:
RegistrationControllerTests.TestMissingUsernameHeader_ReturnsBadRequest() line 30
RuntimeMethodHandle.InvokeMethod(Object target, Void** arguments, Signature sig, Boolean isConstructor)
MethodBaseInvoker.InvokeWithNoArgs(Object obj, BindingFlags invokeAttr)


Test Explorer
Ready
1 Warning
0 Errors
Test
Duration
Traits
Error Message
VirtualEventTests (10) 6,3 sec
VirtualEventTests (10) 6,3 sec
AuthControllerTests (3) 2 sec
TestInvalidCredentials_ReturnsU... 699 ms Test method VirtualEventTests.AuthControllerTests.T...
TestMissingFields_ReturnsBadR... 623 ms
TestValidCredentials_ReturnsOk 699 ms Test method VirtualEventTests.AuthControllerTests.T...
EventsControllerTests (1) 623 ms
TestGetEventDescription_Invalid... 623 ms
EventTests (3) 1,8 sec
TestAvailableSlotsPositive 586 ms
TestEventDurationCalculation 586 ms
TestEventTitleNotNull 586 ms
RegistrationControllerTests (2) 1,3 sec
TestInvalidEvent_ReturnsNotFo... 625 ms
TestMissingUsernameHeader_R... 648 ms Assert.IsTrue failed.
SignUpControllerTests (1) 623 ms
TestMissingFields_ReturnsBadR... 623 ms
Run
Debug
Test Detail Summary
TestValidCredentials_ReturnsOk
Source: AuthControllerTests.cs line 40
Duration: 699 ms
Message:
Test method VirtualEventTests.AuthControllerTests.TestValidCredentials_ReturnsOk threw exception:
System.Data.SQLite.SQLiteException: unable to open database file
Stack Trace:
SQLite3.Open(String strFilename, String vfsName, SQLiteConnectionFlags connectionFlags, SQLiteOpenFlagsEnum openFlags, Int3
SQLite3.Open(String strFilename, String vfsName, SQLiteConnectionFlags connectionFlags, SQLiteOpenFlagsEnum openFlags, Int3
SQLiteConnection.Open()
AuthController.Login(LoginDto request) line 48
AuthControllerTests.TestValidCredentials_ReturnsOk() line 49
Standard Output:
Debug Trace:
[2025-05-22 14:46:58.8538433] System.Data.SQLite (Preload): Native library pre-loader is trying to load native SQLite libra
[2025-05-22 14:46:58.9202879] System.Data.SQLite (Log): SQLite error (14): os_win.c:52009: (3) winOpen(C:\SE410_Project\Vir
[2025-05-22 14:46:58.9203178] System.Data.SQLite (Log): SQLite error (14): cannot open file at line 52010 of [65fff7bd60]


VirtualEventTests (10) 6,3 sec
VirtualEventTests (10) 6,3 sec
AuthControllerTests (3) 2 sec
TestInvalidCredentials_ReturnsU... 699 ms Test method VirtualEventTests.AuthControllerTests.T...
TestMissingFields_ReturnsBadR... 623 ms
TestValidCredentials_ReturnsOk 699 ms Test method VirtualEventTests.AuthControllerTests.T...
EventsControllerTests (1) 623 ms
TestGetEventDescription_Invalid... 623 ms
EventTests (3) 1,8 sec
TestAvailableSlotsPositive 586 ms
TestEventDurationCalculation 586 ms
TestEventTitleNotNull 586 ms
RegistrationControllerTests (2) 1,3 sec
TestInvalidEvent_ReturnsNotFo... 625 ms
TestMissingUsernameHeader_R... 648 ms Assert.IsTrue failed.
SignUpControllerTests (1) 623 ms
TestMissingFields_ReturnsBadR... 623 ms
Test Detail Summary
TestInvalidCredentials_ReturnsUnauthorized
Source: AuthControllerTests.cs line 26
Duration: 699 ms
Message:
Test method VirtualEventTests.AuthControllerTests.TestInvalidCredentials_ReturnsUnauthorized threw exception:
System.Data.SQLite.SQLiteException: unable to open database file
Stack Trace:
SQLite3.Open(String strFilename, String vfsName, SQLiteConnectionFlags connectionFlags, SQLiteOpenFlagsEnum openFlags, Int3
SQLite3.Open(String strFilename, String vfsName, SQLiteConnectionFlags connectionFlags, SQLiteOpenFlagsEnum openFlags, Int3
SQLiteConnection.Open()
AuthController.Login(LoginDto request) line 48
AuthControllerTests.TestInvalidCredentials_ReturnsUnauthorized() line 35
Standard Output:
Debug Trace:
[2025-05-22 14:46:58.9178852] System.Data.SQLite (Log): SQLite error (14): os_win.c:52009: (3) winOpen(C:\SE410_Project\Vir
[2025-05-22 14:46:58.9179523] System.Data.SQLite (Log): SQLite error (14): cannot open file at line 52010 of [65fff7bd60]


Test Explorer
Ready
1 Warning
0 Errors
Test
Duration
Traits
Error Message
VirtualEventTests (10) 6,3 sec
VirtualEventTests (10) 6,3 sec
AuthControllerTests (3) 2 sec
TestInvalidCredentials_ReturnsU... 699 ms Test method VirtualEventTests.A...
TestMissingFields_ReturnsBadR... 623 ms
TestValidCredentials_ReturnsOk 699 ms Test method VirtualEventTests.A...
EventsControllerTests (1) 623 ms
TestGetEventDescription_Invalid... 623 ms
EventTests (3) 1,8 sec
TestAvailableSlotsPositive 586 ms
TestEventDurationCalculation 586 ms
TestEventTitleNotNull 586 ms
RegistrationControllerTests (2) 1,3 sec
TestInvalidEvent_ReturnsNotFo... 625 ms
TestMissingUsernameHeader_R... 648 ms Assert.IsTrue failed.
SignUpControllerTests (1) 623 ms
TestMissingFields_ReturnsBadR... 623 ms
Run
Debug
Group Summary
VirtualEventTests
Tests in group: 10
Total Duration: 6,3 sec
Outcomes
7 Passed
3 Failed

# 10. Conclusion

The Virtual Event project is a cross-platform event management system that combines a C# WinForms desktop app, an ASP.NET Web API, and a Web Forms frontend, all connected through a shared SQLite database. The desktop interface is designed for admins to manage events and participants, while the web interface allows users to browse and register for events. Built with MVC principles, the system ensures modularity and clarity across components. Key features such as user authentication, conflict detection, notifications, and change tracking make the platform both practical and scalable for real-world use.