بِسْمِ ٱللَّٰهِ ٱلرَّحْمٰنِ ٱلرَّحِيم

Uml project:

ایجاد یک پروژه UML (زبان مدل‌سازی یکپارچه) شامل تعریف ساختار و رفتار یک سیستم با استفاده از نمودارهای مختلف UML است. در ادامه، یک رویکرد عمومی برای شروع یک پروژه UML ارائه می‌شود که شما را در این فرآیند راهنمایی می‌کند.

۱. تعریف هدف پروژه

- سیستم مورد نظر خود را که می‌خواهید مدل‌سازی کنید، تعیین کنید (به عنوان مثال، یک برنامه وب، یک سیستم مدیریت کتابخانه، یک پلتفرم خرید آنلاین، و غیره).

- ذینفعان را شناسایی کرده و نیازها را جمع‌آوری کنید.

۲. شناسایی موارد استفاده

- موارد استفاده را تعریف کنید تا نیازهای عملکردی سیستم را نشان دهند.

- هر مورد استفاده باید یک تعامل خاص بین کاربر و سیستم را توصیف کند.

۳. ایجاد نمودارهای مورد استفاده

نمودارهای مورد استفاده، تعاملات بین بازیگران (کاربران یا سیستم‌های دیگر) و موارد استفاده را به صورت بصری نمایش می‌دهند.

\*\*مثال:\*\*

```plaintext

+-------------------------+

| سیستم |

+-------------------------+

| مورد استفاده ۱ |

| مورد استفاده ۲ |

| مورد استفاده ۳ |

+-------------------------+

/ \

بازیگر ۱ بازیگر ۲

```

۴. تعریف کلاس‌ها و روابط

- کلاس‌های اصلی را که نمایانگر داده‌ها و قابلیت‌های سیستم هستند شناسایی کنید.

- ویژگی‌ها (داده‌ها) و متدها (توابع) آنها را تعیین کرده و روابط آنها (هم‌پوشانی، وابستگی‌ها، وراثت) را تعریف کنید.

۵. ایجاد نمودار کلاس

نمودارهای کلاس کلاس‌ها، ویژگی‌ها، متدها و روابط آنها را نشان می‌دهند.

\*\*مثال:\*\*

```plaintext

+-----------------+

| کلاس A |

+-----------------+

| - ویژگی ۱ |

| - ویژگی ۲ |

+-----------------+

| + متد ۱() |

| + متد ۲() |

+-----------------+

|

| ۱

|

| ۰..\*

+-----------------+

| کلاس B |

+-----------------+

| - ویژگی ۱ |

+-----------------+

| + متد ۱() |

+-----------------+

```

۶. تعریف تعاملات اشیاء

- مدل‌سازی کنید که اشیاء چگونه درون سیستم از طریق متدها تعامل می‌کنند.

- پیام‌هایی که بین اشیاء ارسال می‌شوند را شناسایی کنید.

۷. ایجاد نمودارهای توالی

نمودارهای توالی نحوه تعامل اشیاء را در یک توالی خاص نمایش می‌دهند.

\*\*مثال:\*\*

```plaintext

بازیگر کلاس A کلاس B

|  |  |
| --- | --- |
|  |  |

| --- پیام ۱ -🡪 |

| | --- پیام ۲ 🡪 |

| | 🡨- پاسخ --- |

```

. مدل‌سازی حالات سیستم

* حالات سیستم یا اشیاء آن را تعریف کنید.

۹. ایجاد نمودارهای حالت

نمودارهای حالت، حالات یک شی و انتقالات آنها را به طور بصری نشان می‌دهند.

\*\*مثال:\*\*

```plaintext

[حالت ۱] – رویداد ۱ 🡪 [حالت ۲]

[حالت ۲] – رویداد ۲ 🡪 [حالت ۱]

```

۱۰. ایجاد نمودارهای فعالیت

نمودارهای فعالیت جریان کنترل یا داده‌ها در یک سیستم را نمایش می‌دهند.

\*\*مثال:\*\*

```plaintext

[شروع] -🡪 [فعالیت ۱] -🡪 [تصمیم] -🡪 [فعالیت ۲]

|  |
| --- |
|  |

[فعالیت ۳] [فعالیت ۴]

|  |
| --- |
|  |

[پایان] [پایان]

```

۱۱. بازبینی و تأیید

- نمودارها را با ذینفعان تأیید کنید تا مطمئن شوید که به درستی نمایانگر سیستم هستند.

- بر اساس بازخورد، تغییرات لازم را انجام دهید.

ابزارهای UML

در اینجا برخی از ابزارهای محبوب برای ایجاد نمودارهای UML آمده است:

- \*\*Lucidchart\*\*: نرم‌افزار آنلاین برای ایجاد نمودارهای UML.

- \*\*Draw.io\*\*: ابزار رایگان آنلاین برای ایجاد انواع نمودارها، شامل UML.

- \*\*PlantUML\*\*: ابزاری که به شما امکان می‌دهد با استفاده از توصیف متنی ساده نمودارهای UML ایجاد کنید.

- \*\*StarUML\*\*: یک برنامه دسکتاپ برای ایجاد نمودارهای UML.

مثال پروژه: سیستم مدیریت کتابخانه

برای ارائه یک مثال ملموس‌تر، بیایید سیستم مدیریت کتابخانه را به عنوان یک پروژه نمونه در نظر بگیریم.

موارد استفاده

1. امانت‌گیری کتاب

2. بازگشت کتاب

3. جستجوی کتاب

4. ثبت‌نام عضو

نمودار کلاس

```plaintext

+--------------------------+

| کتابخانه |

+--------------------------+

| - کتاب‌ها: لیست<کتاب> |

| - اعضا: لیست<عضو> |

+--------------------------+

| + امانت‌گیری کتاب() |

| + بازگشت کتاب() |

| + جستجوی کتاب() |

+--------------------------+

+--------------------------+

| کتاب |

+--------------------------+

| - عنوان |

| - نویسنده |

| - در دسترس |

+--------------------------+

| + بررسی دسترسی() |

+--------------------------+

+--------------------------+

| عضو |

+--------------------------+

| - نام |

| - شناسه عضو |

+--------------------------+

| + ثبت‌نام() |

+--------------------------+

```

**ACP.NETCore**:

ASP.NET Core یک فریم‌ورک متن باز و چند سکویی برای ساخت برنامه‌های وب و خدمات (API) است. این فریم‌ورک توسط مایکروسافت توسعه یافته و نسخه‌ای بهینه و مدرن از ASP.NET است.

ویژگی‌های اصلی ASP.NET Core

1. \*\*چند سکویی بودن\*\*:

- ASP.NET Core به‌صورت native بر روی Windows، macOS و Linux اجرا می‌شود، که این امکان را به توسعه‌دهندگان می‌دهد که برنامه‌های خود را در هر سسیستمی اجرا کنند.

2. \*\*عملکرد بالا\*\*:

- این فریم‌ورک برای بهبود عملکرد بهینه‌سازی شده است. بر اساس نتایج، ASP.NET Core یکی از سریع‌ترین فریم‌ورک‌های وب موجود است.

3. \*\*زیرساخت سبک\*\*:

- ASP.NET Core یک مدل برنامه‌نویسی مبتنی بر Middlewareها (میان‌افزارها) دارد که به توسعه‌دهندگان امکان کنترل دقیقی بر روی درخواست‌ها و پاسخ‌ها می‌دهد.

4. \*\*پشتیبانی از Dependency Injection\*\*:

- مدیریت وابستگی‌ها به‌صورت داخلی در ASP.NET Core قرار دارد و این ویژگی به ایجاد کدهای قابل تست و نگهداری کمک می‌کند.

5. \*\*APIهای Web\*\*:

- طراحی و پیاده‌سازی APIهای RESTful به راحتی امکان‌پذیر است.

6. \*\*مدیریت هویت و اعتبارسنجی\*\*:

- ASP.NET Core دارای امکانات داخلی برای مدیریت کاربران، احراز هویت و مجوزها است.

7. \*\*پیکربندی آسان\*\*:

- پیکربندی برنامه‌ها به سادگی از طریق فایل‌های JSON، متغیرهای محیطی و خط فرمان انجام می‌شود.

8. \*\*پشتیبانی از Razor Pages\*\*:

- فریم‌ورک Razor Pages به‌صورت ساده‌تری برای توسعه صفحات وب و MVVM طراحی شده است.

شروع کار با ASP.NET Core

۱. نصب ASP.NET Core

برای شروع توسعه با ASP.NET Core، باید نرم‌افزار .NET SDK را نصب کنید. می‌توانید آن را از [سایت رسمی مایکروسافت](https://dotnet.microsoft.com/download) دانلود کنید.

. ایجاد یک پروژه جدید

پس از نصب SDK، می‌توانید یک پروژه جدید ASP.NET Core را با استفاده از خط فرمان یا ترمینال ایجاد کنید:

```bash

dotnet new webapp -n MyWebApp

```

فرمان بالا یک پروژه جدید ASP.NET Core به نام `MyWebApp` ایجاد می‌کند.

. اجرای پروژه

برای اجرا کردن پروژه، به دایرکتوری پروژه بروید و از فرمان زیر استفاده کنید:

```bash

cd MyWebApp

dotnet run

```

پروژه شما به‌طور پیش‌فرض روی `http://localhost:5000` اجرا خواهد شد.

. ساخت یک API ساده

برای ساختن یک API ساده، می‌توانید کد زیر را درون فایل `Controllers/WeatherForecastController.cs` اضافه کنید:

```csharp

using Microsoft.AspNetCore.Mvc;

namespace MyWebApp.Controllers

{

[ApiController]

[Route("[controller]")]

public class WeatherForecastController : ControllerBase

{

private static readonly string[] Summaries = new[]

{

"Freezing", "Bracing", "Chilly", "Cool", "Mild", "Warm", "Balmy", "Hot", "Sweltering", "Scorching"

};

[HttpGet]

public IEnumerable<WeatherForecast> Get()

{

var rng = new Random();

return Enumerable.Range(1, 5).Select(index => new WeatherForecast

{

Date = DateTime.Now.AddDays(index),

TemperatureC = rng.Next(-20, 55),

Summary = Summaries[rng.Next(Summaries.Length)]

})

.ToArray();

}

}

public class WeatherForecast

{

public DateTime Date { get; set; }

public int TemperatureC { get; set; }

public string Summary { get; set; }

}

}

```

هم گروه فرشته رنجبری و عالیه خرمی سرریگانی

استاد محمد احمد زاده