ТЕХНОЛОГИЯ .NET CORE ДЛЯ РАЗРАБОТКИ ВЕБ-ПРИЛОЖЕНИЙ

Д. А. Истомин

Белорусский государственный университет, г. Минск;
zagamant5311@gmail.com;
науч. рук. – О. В. Дубровина, ст. преподаватель

Модульная платформа .NET Core для разработки веб-приложений является одной из наиболее активно развивающихся в современном цифровом мире. По версии Github более полутора миллионов разработчиков за 5 лет и вход в пятерку самых используемых языков по [1].

Развитие этого фреймворка началось с 2016 года на основе .Net Framework и активно продвигается компанией Microsoft, которая параллельно развивает и новые базирующиеся на ней технологии.

Рассмотрим стек технологий .NET Core, позволяющий создавать насыщенные веб-приложения:

1. Entity Framework Core (EF Core), которая выступает в качестве объектно-реляционного отображения (ORM) и позволяет работать с базой данных с использованием объектов [2]. ORM фреймворк генерирует объекты, которые виртуально отображают таблицы в базе данных. Затем эти таблицы используются для взаимодействия с самой базе. EF Core поддерживает множество баз данных, таких как: Microsoft SQL Server, Postgresql, OracleDB, MySQL, MongoDB и т. п. Для работы с ORM пользователю не требуются знания специального языка программирования для работы с базами данных.

2. ASP.NET Core – новый кроссплатформенный фреймворк с открытым исходным кодом для создания современных веб-приложений и мобильных серверных частей [3]. Основная его концепция строится на основе протокола HTTP взаимодействия браузера и сервера. Основными преимуществами ASP.NET Core являются кроссплатформенность, открытый исходный код, полная поддержка консольных инструментов и высокое быстродействие по сравнению со старыми версиями.

3. Для реализации клиентской части веб-приложения может использоваться фреймворк Blazor. Это кроссплатформенный фреймворк пользовательского интерфейса. Он позволяет создавать браузерные приложения, используя, помимо HTML и CSS, язык C# вместо JavaScript [4].

В связи с тем, что эта технология новая и находится в процессе развития, из нее можно выделить два компонента: Blazor Server и Blazor WebAssembly, которые уже активно используются в разработке, и еще три, находящиеся на этапе проектирования и/или тестирования – Blazor PWA, Blazor Hybrid и Blazor Native (рисунок 1).

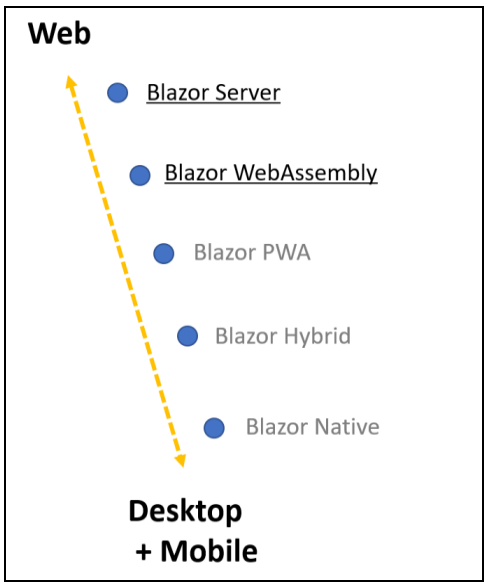

Рис. 1. Разновидности технологий Blazor

Blazor WebAssembly позволяет запускать программный код .NET непосредственно в браузере, без привязки к языку программной реализации. По аналогии с языком программирования веб-приложений JavaScript, приложения, реализованные с помощью WebAssembly работают на устройстве пользователя из изолированной программной среды безопасности браузера. Основным аргументом для развития технологии служит то, что WebAssembly стандартизирован ассоциацией W3C и поддерживается всеми распространенными браузерами, исключая Internet Explorer 11 [4].

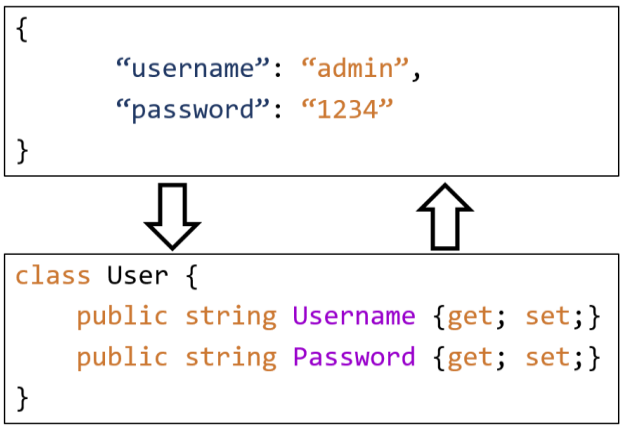

Рис. 2. Пример трансформации JSON в виртуальный объект

Следующей задачей при разработке приложений является обмен текстовой информацией между сервером и клиентской частью. В частности, в веб-разработке используется формат JavaScript Object Notation (JSON) – текстовый формат обмена данными, основанный на языке JavaScript. При этом он независим от JavaScript и может использоваться

в любом языке программирования [5]. На рисунке 2 изображен пример трансформации JSON объекта в виртуальный и наоборот.

Встроенные библиотеки Newtonsoft.Json и System.Text.Json предназначены для обработки информации, хранящейся в формате JSON. Стоит отметить, в предыдущих версиях .NET Core основным инструментом для работы между JSON документами и .NET объектами являлась внешняя библиотека Newtonsoft.Json. Однако на текущий момент библиотека System.Texts.Json стала полноценной заменой, при этом значительно повысилась производительность при сериализации и десериализации коллекций, небольших типов и длинных строк JSON. Таким образом, преимущество в скорости работы составляет до 400% по сравнению с предыдущими версиями [6].

Прикладным примером использования рассмотренных выше технологии является реализованное веб-приложение, предназначенное для организации контроля учебного процесса. Этот программный продукт будет методически сопровождать обучение в детей и молодежи ИТ-клуба. Минска «It_Club», разработанный на основе правил организации.

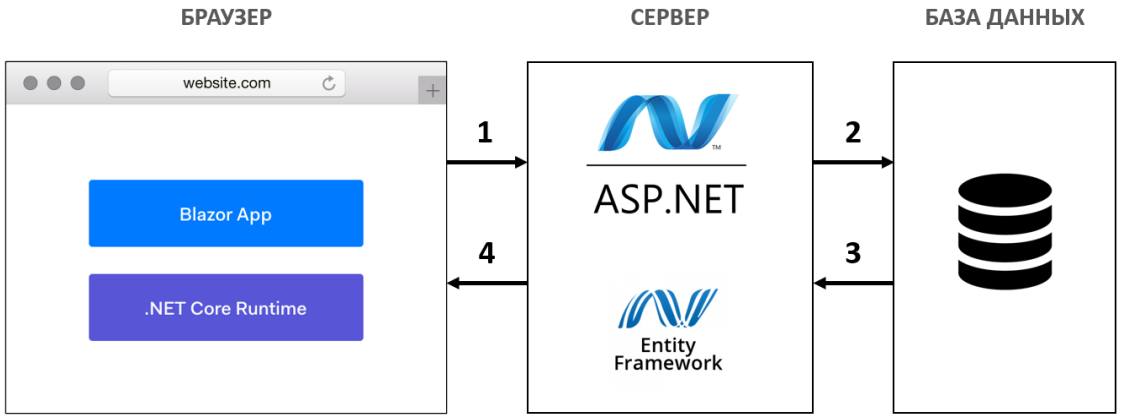

Рис. 3. Архитектура приложения

Технология Blazor (WebAssembly) выступает в роли клиентской части приложения в среде браузера, в котором пользователь производит свои действия. Серверная часть реализована с помощью технологий ASP.NET Core и Entity FrameWork Core, отвечающий за связь с базой данных. Дополнительно подключены библиотеки Newtonsoft.Json и System.Text.Json, которые позволяют быстро обработать текстовую информацию внутри приложения.

Использование данной системы, разработанной согласно требованиям «It_Club», позволит повысить общее качество учебного процесса, а также упростит контроль за группами учащихся и их преподавателями.

Библиографические ссылки

1. The State of the Octoverse [Электронный ресурс]. Режим доступа: https://octoverse.github.com/ – Дата доступа: 15.05.2021.
2. . Обзор Entity Framework Core — EF Core | Microsoft Docs [Электронный ресурс]. Режим доступа: https://docs.microsoft.com/ru-ru/ef/core/ – Дата доступа: 15.05.2021.
3. What is ASP.NET Core? | .NET Docs [Электронный ресурс]. Режим доступа: https://dotnet.microsoft.com/learn/aspnet/what-is-aspnet-core – Дата доступа: 15.05.2021.
4. Blazor | Build client web apps with C# | .NET Docs [Электронный ресурс]. Режим доступа: https://dotnet.microsoft.com/apps/aspnet/web-apps/blazor – Дата доступа: 15.05.2021.
5. JSON [Электронный ресурс]. Режим доступа: https://www.json.org/json-ru.html – Дата доступа: 16.05.2021.
6. Try the new System.Text. Json APIs | .NET Blog [Электронный ресурс]. Режим доступа: https://devblogs.microsoft.com/dotnet/try-the-new-system-text-json-apis/ – Дата доступа: 16.05.2021.