DOI: https://doi.org/10.36910/6775-2524-0560-2024-57-32
УДК 004.415.3
**Pekh Petro**, PhD
https://orcid.org/0000-0002-6327-3319
**Yankovsky Bohdan,** bachelor
Lutsk National Technical University, Lutsk, Ukraine

# FEATURES OF CREATING WEB APPLICATIONS USING THE PLATFORM C# ASP .NET CORE MVC

**Pekh P., Yankovsky B. Features of creating web applications using C# ASP .NET CORE MVC.** The article discusses the features of using the ASP.NET Core MVC platform on the example of developing a web application in the form of a pharmacy customer information system. The strengths and weaknesses of this platform are analyzed. Pharmacy customers are primarily interested in information about the availability of the necessary drugs, new drug arrivals, current promotional offers and much more. In addition, it is important for the client to be able to obtain the necessary information online and in a short time. And here you cannot do without a reliable, effective and fast web resource. However, in many cases, a web application can be quite simple and limited to providing several functions: user registration, payment processing, report generation. This is due to the fact that most pharmacies are small enterprises with limited financial capabilities. Therefore, it will be beyond the power of such an enterprise to develop and operate its own database. The question immediately arises of how to get around this problem. That is, what should be the platform for developing its own information system. In our opinion, a good solution is the ASP.NET Core MVC platform, on the basis of which the web application proposed in the article is developed. If only because it allows you to create a fully functional customer information system without using a database, while at the same time allowing you to store information in JSON file format.
**Keywords**: C# ASP .NET CORE MVC platform, MVC technology, web application

**Пех П. А., Янковський Б. О. Особливості створення веб-додатків з використанням C# ASP.NET Core MVC.** У статті розглянуто особливості використання платформи ASP.NET Core MVC на прикладі розробки веб-додатку у вигляді системи інформування клієнтів аптеки. Проаналізовані сильні та слабкі сторони даної платформи. Клієнтів аптеки в першу чергу цікавить інформація про наявність потрібних ліків, нові надходження ліків, актуальні акційні пропозиції та багато іншого. Крім того, клієнту важливо мати можливість отримати необхідну інформацію онлайн і в короткі терміни. І тут без надійного, ефективного і швидкодіючого веб-ресурсу не обійтися. Однак у багатьох випадках веб-додаток може бути досить простим і обмежуватись забезпеченням кількох функцій: реєстрацією користувачів, обробкою платежів, генеруванням звітів. Це пов'язано з тим, що більшість аптек є невеликими підприємствами з обмеженими фінансовими можливостями. Тож розробляти та експлуатувати власну базу даних такому підприємству буде не до снаги. Зразу ж виникає питання, як обійти цю проблему. Тобто, якою має бути платформа для розробки власної інформаційної системи. На наше переконання, хорошим рішенням є платформа ASP.NET Core MVC, на базі якої розроблений запропонований у статті веб-додаток. Хоча би тому, що для створення повнофункціональної системи інформування клієнтів вона надає можливість не використовувати базу даних, водночас дозволяючи зберігати інформацію у форматі JSON-файлів.
**Ключові слова:** платформа C# ASP .NET CORE MVC, технологія MVC , веб-додаток

**Problem statement**. Automation of customer service processes in pharmacies contributes to improving the quality of service, reducing the burden on staff and informing visitors in a timely manner. In this regard, there is a need to create a modern system that allows informing customers about the availability of medicines, new arrivals, promotions and other important information in an interactive format.

**The purpose of the study** is to develop a software and hardware information system that allows pharmacy customers to receive information about the availability of medicines, their description and price, and also provides a convenient administrative panel for managing this data, using the modern ASP.NET Core MVC web framework.

**The novelty of the study** lies in creating a fully functional customer information system that does not use a database, but stores information in JSON file format, which greatly simplifies its deployment on local devices or in small pharmacies without a complex IT infrastructure. In addition, the system has a hidden administrator mode that allows staff to send customers notifications about new arrivals or promotions.

**Main part.** The system is developed using the ASP.NET Core MVC template, which implements the Model-View-Controller pattern, which allows logical separation of business logic, user interface and request processing. The client part provides viewing of the medicine catalog with an image, description, price and search function. Client registration is performed by name and phone number. The administrative panel implements the functions of adding new medicines, viewing registered clients and informing them about updates.

JSON files are used to store data, which are read and written using .NET tools. This allows you to reduce the complexity of project setup and increase its portability.

A brief description of the web application development technology is given in Table 1.

Table 1 – Web application development technologies and tools

| Component | Опис |
|---|---|
| Programming Language | C# (.NET 9.0) |
| Platform | ASP.NET Core MVC |
| Data Storage | JSON-файли (clients.json, medicines.json) |
| Interface | Razor Pages (.cshtml) |
| Development Environment | Visual Studio 2022 |

**The project structure** is shown in Figure 1, and its elements are described in Table 2.

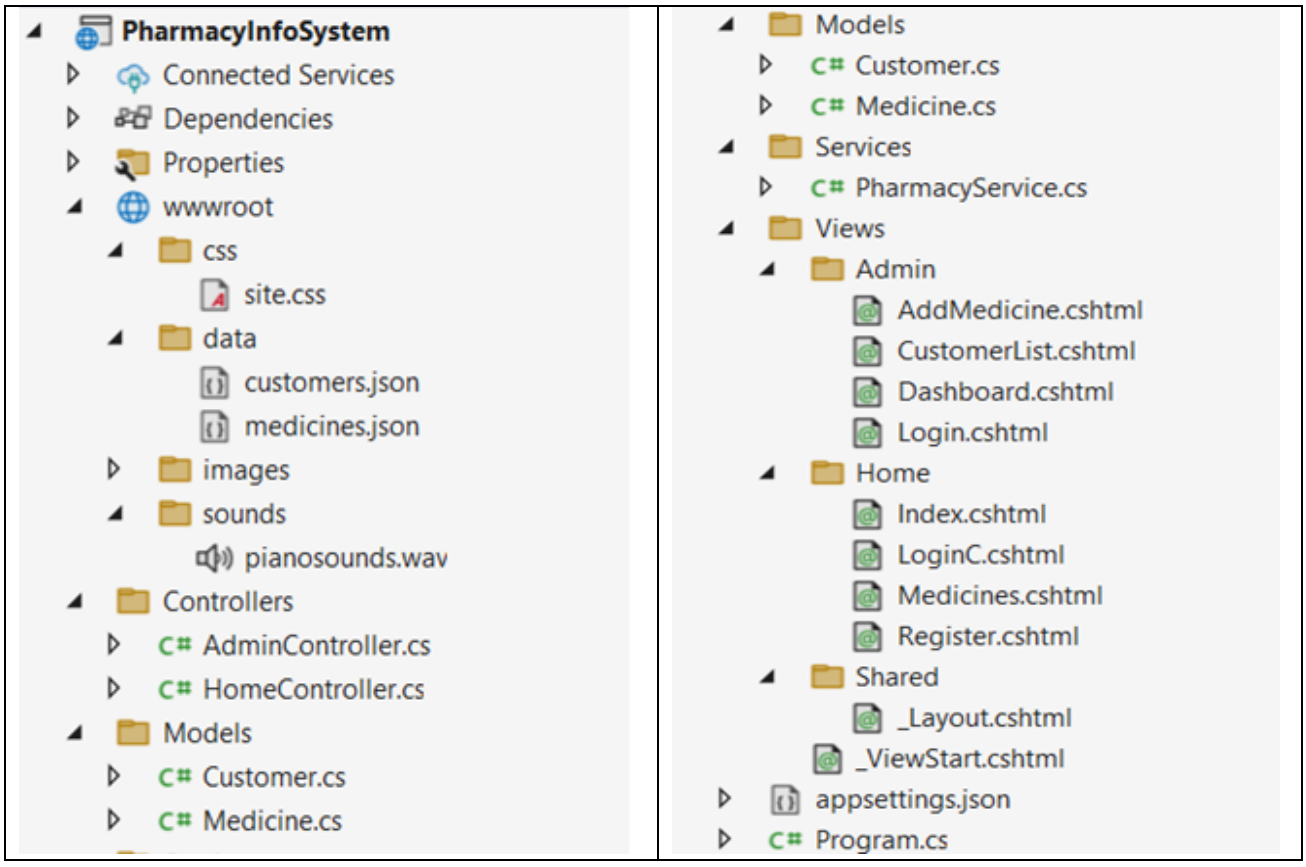


Figure 1 – Project structure based on the MVC pattern

The information system is implemented according to the MVC principles based on three blocks:

– Model: Client, Medicine classes, which describe data (name, phone, name of medicine, price).

– View: Razor pages (.cshtml) for displaying a list of medicines, customer registration form, administrative panel.

– Controller: logic of interaction with the user, routes Home/Index, Home/Client, Home/Admin, Admin/AddMedicine.

Table 2 – Web application elements

| Path | Element | Description |
|---|---|---|
| /wwwroot | css | CSS style directory. Contains files responsible for the appearance of the site (fonts, colors, padding, etc.). |

| | | |
|---|---|---|
| /wwwroot | data | Directory for storing data in the form of JSON files, medicines.json, and clients.json. Analogue of a database, implemented as local files. |
| /wwwroot | images | Contains images used on the site |
| /wwwroot | sounds | Directory for audio files used on the site |
| /Controllers | AdminController.cs | Controller responsible for the administrator's functionality |
| /Controllers | HomeController.cs | Main controller for clients. Processes requests to the main page, customer registration, viewing medicines, etc. |
| /Models | Customer.cs | Model representing the client: name, phone number. |
| /Models | Medicine.cs | Medicine model: name, description, price, path to the image. |
| /Services | PharmacyService.cs | Class that implements the logic of interaction with JSON files. Reads and stores data about clients and medicines. |
| /Views | _ViewStart.cshtml | File that sets the basic settings for all Razor pages. |
| /Views/Admin | AddMedicine.cshtml | Page with a form for adding a new medicine in the admin panel. |
| /Views/Admin | CustomerList.cshtml | Page that displays a list of registered clients for the administrator. |
| /Views/Admin | Dashboard.cshtml | Administrator main page |
| /Views/Admin | Login.cshtml | Administrator login form (login/password) |
| /Views/Home | Index.cshtml | Site main page. |
| /Views/Home | LoginC.cshtml | Client login form |
| /Views/Home | Medicines.cshtml | Page that displays the drug catalog. Includes search, sorting, filtering. |
| /Views/Home | Register.cshtml | New client registration form (name, phone). |
| /Views/Shared | _Layout.cshtml | General site template. Includes HTML page structure, header, footer, menu, etc. |
| | appsettings.json | Configuration file. |
| | Program.cs | The main application startup file. Creates a web server, configures routing, adds services, middleware (e.g., static file support, MVC, etc.). |

The web application, developed on the basis of the MVC platform (Fig. 2), consists of two parts: administrative and user. It differs in the following:

– for users, the following were implemented: the ability to register and authorize; viewing the list of medications with the ability to search and sort;

– for the administrator, the following were implemented: the ability to authorize; viewing the list of medications; adding new medications; deleting medications; viewing the list of clients;

– JSON files were used as an alternative to a database for storing information about medications and clients;

– styles and background images were added to improve the visual design;

– audio accompaniment was implemented in the form of a background melody;

– a header with a logo and a pop-up menu was created for ease of navigation;

– a footer with dynamic elements was added.

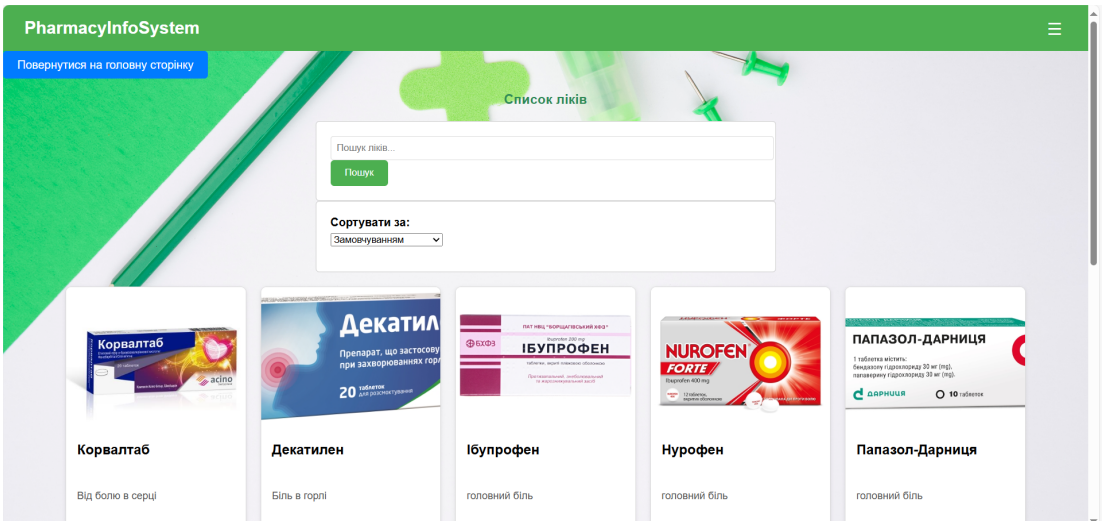


Figure 2 – Main window of the web application after client authorization

**Features of the web application implementation:**

- created without using databases: data is stored in JSON files (clients.json, medicines.json);
- client registration: the user enters the name and phone number, the data is stored in clients.json;
- search for medicines: implemented filter by name and sorting by price/availability;
- administrative panel: available at URL /Home/Admin, login and password – admin/admin.

**The results of testing the web application** are given in Table 3.

**Advantages of using ASP.NET Core MVC:**

- separation of logic, presentation and data;
- flexible routing configuration;
- built-in support for forms, validation and authorization;
- ability to quickly deploy without additional services.

Table 3 – Results of testing the web application

| Function | Status | Comment |
|---|---|---|
| Client registration | successful | Data is stored in JSON |
| Search and view medications | successful | Search works by partial name |
| Add medications | successful | Added medications are immediately visible to the client |
| Work without a database | successful | No SQL connection |
| Client authorization | successful | Previously registered client successfully authenticated |
| View client list | successful | Administrator can view registered clients |

**Security and reliability.** Although the system does not use a database, it allows you to store data in a secure format. Access to the administrative panel is limited by authorization, which prevents unauthorized editing of information. Additionally, the platform structure allows you to implement data encryption in JSON files, restrict access to certain routes and integrate backup mechanisms.

**Practical significance of the development.** The proposed system is of practical importance for pharmacies that seek to digitize the customer service process without significant financial investments. Due to the rejection of the server database and the use of JSON files, the system can be quickly configured and implemented even on computers with minimal resources. This makes it attractive for small pharmacies or temporary retail outlets in rural areas where there is no constant Internet connection.
Comparison with alternative solutions. Unlike solutions based on PHP or JavaScript frameworks such as Node.js, the ASP.NET Core MVC system has better integration with Windows tools, support for multi-

level routing, and a convenient access control model. In addition, development in C# allows you to reduce the number of errors due to strict typing and effective work with an object-oriented structure.

**System development prospects.** Thanks to the use of ASP.NET Core MVC, the developed system has a flexible architecture that allows you to easily expand its functionality. In the future, it can be supplemented with integration with third-party API services to check the availability of medicines in other pharmacies, support for push notifications for customers, expansion of the administrative panel with statistical modules, or integration with mobile applications. It is also possible to implement a voice control system or support for a multilingual interface to improve accessibility.

**Conclusions.**

The work developed a web application for informing pharmacy customers, which has an intuitive interface, implemented on the basis of ASP.NET Core MVC. The system does not require the use of a database and can run on any server with .NET support.

The effectiveness of using the MVC platform for building software and hardware information systems for small enterprises has been demonstrated. In the future, the system can be upgraded by ensuring interaction with the hardware part (for example, LED screens in the pharmacy, smart displays, voice assistants).

**References**

1. Overview of ASP.NET Core MVC. *Microsoft Learn: Build skills that open doors in your career*. URL: https://learn.microsoft.com/uk-ua/aspnet/core/mvc/overview?view=aspnetcore-9.0 (date of access: 02.05.2025).
2. Freeman A. Pro ASP.NET Core MVC 2. Berkeley, CA : Apress, 2017. URL: https://doi.org/10.1007/978-1-4842-3150-0 (date of access: 02.05.2025).
3. Peleshenko S. Web programming: tutorial. K.: KNU, 2020. 256 p.
4. JSON. *JSON*. URL: https://www.json.org/json-en.html (date of access: 02.05.2025).
5. Shapovalov O. Technologies for creating web applications in .NET. Lviv: LNU, 2021. 178 p.