

УДК 004.415.2

**РЕАЛИЗАЦИЯ ВЕБ-ПРИЛОЖЕНИЯ НА C#
С ИСПОЛЬЗОВАНИЕМ ТЕХНОЛОГИЙ ASP.NET CORE MVC
ДЛЯ ИЗУЧЕНИЯ АНГЛИЙСКОГО ЯЗЫКА**

В. А. СКОЧКОВ

(Представлено: Т. М. ГЛУХОВА)

В статье рассматривается процесс разработки веб-приложения для изучения английских слов на основе метода интерактивных карточек. Подробно описываются этапы проектирования, реализации и тестирования приложения, созданного с использованием архитектурного шаблона Model-View-Controller (MVC) и технологического стека ASP.NET Core.

Введение. Сегодня в мире знание английского языка является одним из важнейших требований для успешной профессиональной реализации. Одним из наиболее эффективных методов запоминания лексики является метод флэш-карточек, где на одной стороне карточки написано слово на иностранном языке, а на другой – его перевод и определение. Автоматизация этого процесса с помощью веб-приложения позволяет значительно повысить его эффективность, сделать обучение более удобным и доступным.

Целью данной работы является проектирование и разработка веб-приложения, которое демонстрирует возможности платформы ASP.NET Core и языка C# для создания многофункциональных и безопасных образовательных сервисов. Выбор в пользу серверной архитектуры неслучаен: он позволяет обеспечить централизованное управление данными, реализовать сложную бизнес-логику на стороне сервера, повысить уровень безопасности и предоставить гибкость в дальнейшем расширении функционала. Данный проект призван показать, что технологии .NET являются мощным и конкурентоспособным инструментом для построения современных интерактивных веб-платформ.

Программная часть. Для реализации поставленной задачи была выбрана современная и производительная платформа ASP.NET Core в сочетании с языком программирования C#. Этот выбор обусловлен рядом факторов, включая высокую производительность, кроссплатформенность, развитую экосистему и долгосрочную поддержку со стороны Microsoft [1]. В основе архитектуры приложения лежит классический и проверенный временем паттерн проектирования Model-View-Controller (MVC). Этот паттерн обеспечивает строгое разделение ответственности между компонентами системы, что позволяет четко разграничить бизнес-логику, данные и их визуальное представление. Такое разделение не только упрощает процесс начальной разработки, но и значительно облегчает тестирование, отладку и дальнейшую поддержку, и модификацию программного продукта [2]. Архитектурный шаблон Model-View-Controller:

- Модель (Model): представляет собой концептуальное ядро приложения, состоящее из набора классов, которые описывают основные сущности и структуры данных. Ключевыми моделями в проекте являются класс Flashcard, определяющий структуру учебной карточки, и класс Topic, позволяющий логически группировать карточки по темам.

- Представление (View): полностью отвечает за формирование пользовательского интерфейса. В проекте используются файлы синтаксиса Razor (.cshtml), которые элегантно сочетают стандартную HTML-разметку с кодом на C# [1]. Это позволяет динамически генерировать контент на сервере перед отправкой его клиенту. Визуальное оформление и адаптивность интерфейса реализованы с помощью популярного CSS-фреймворка Bootstrap, что обеспечивает корректное отображение на устройствах с разным разрешением экрана.

- Контроллер (Controller): представленный в проекте классом HomeController, выполняет роль координатора и является связующим звеном между моделью и представлением. Его основная задача — обработка входящих HTTP-запросов от пользователя. Он взаимодействует с сервисным слоем для выполнения операций с данными, управляет состоянием пользовательской сессии и в конечном итоге выбирает и возвращает соответствующее представление.

Разделение ответственности, достигаемое с помощью MVC, является ключевым преимуществом, поскольку оно упрощает совместную разработку, тестирование и дальнейшую поддержку приложения.

Одной из важных функциональных особенностей системы является реализация простого механизма разделения прав доступа. В приложении предусмотрены две роли: "Учитель", обладающий полными правами, и "Ученик", с ограниченными возможностями. Выбор роли происходит на стартовой странице (рис.1), после чего сделанный выбор сохраняется в сессии пользователя. При каждом последующем запросе HomeController считывает это значение из сессии. На основе полученной роли контроллер

динамически формирует представление, скрывая или отображая определенные элементы интерфейса [3]. Например, кнопки редактирования и удаления карточек отображаются только в том случае, если переменная роли имеет значение "Teacher". Кроме того, для удобства обмена учебными материалами реализованы функции импорта и экспорта всей базы слов в универсальном формате JSON.

Добро пожаловать в приложение для изучения слов!

Пожалуйста, выберите вашу роль, чтобы продолжить.

Я Учитель

Вы получаете полный доступ ко всем функциям приложения:

- Создание новых карточек и тем.
- Редактирование и удаление существующих карточек.
- Загрузка и скачивание базы слов.
- Доступ к игровому режиму.

Войти как Учитель

Я Ученик

Ваша основная задача - учиться! Вам доступны следующие функции:

- Доступ к игровому режиму для тренировки.
- Просмотр всех существующих карточек.
- Загрузка и скачивание готовой базы слов.
- (Редактирование и создание отключены)

Войти как Ученик

Рисунок 1. – Стартовая страница выбора роли

С целью повышения вовлеченности пользователей и геймификации образовательного процесса в приложение был интегрирован интерактивный игровой режим (рис.2). Процесс его работы является наглядным примером эффективного взаимодействия серверной и клиентской частей. Логика подготовки данных для игры целиком выполняется на сервере: метод Game в HomeController обращается к сервису для получения необходимого набора карточек, затем случайным образом отбирает из них не более десяти, перемешивает их и передает в представление. Непосредственное взаимодействие пользователя с игровым полем, такое как выбор слов, отслеживание пар и динамическая отрисовка соединительных линий, реализуется уже на стороне клиента с использованием нативного JavaScript. Такой гибридный подход, где серверная часть на .NET отвечает за бизнес-логику и управление данными, а клиентская — за интерактивность и визуальное представление, является современной и эффективной практикой в веб-разработке.

Соедините слово с его переводом

Appetite

Cuisine

Recipe

Flavor

Ingredient

вкус
(Уникальное сочетание ощущений, которое мы чувствуем, когда пробуем еду.)

аппетит
(Желание поесть, возникающее при виде или запахе вкусного блюда.)

кухня (национальная)
(Совокупность традиционных блюд и способов их приготовления в определённой культуре.)

ингредиент
(То, из чего складывается блюдо, как кирпичики в постройке.)

рецепт
(Подробное руководство, которое подсказывает, что и в каком порядке смешать, чтобы получилось вкусно.)

Проверить

Выйти

Рисунок 2. – Страница игрового режима

Заключение. В ходе выполнения данной работы было успешно спроектировано и разработано полнофункциональное веб-приложение для изучения английских слов. Выбранный технологический стек, в частности платформа ASP.NET Core и архитектурный паттерн MVC, позволил создать надежную,

поддерживаемую и масштабируемую систему с четко определенной структурой. Реализованный функционал, который включает в себя разделение ролей пользователей, полный набор операций по управлению учебным контентом и интерактивный игровой режим, полностью соответствует изначально поставленным задачам. Данный проект демонстрирует, что серверные технологии .NET являются мощным и современным инструментом для создания сложных интерактивных образовательных платформ, не уступающих по своим возможностям клиентским приложениям.

В качестве потенциальных направлений для дальнейшего развития проекта можно выделить несколько ключевых векторов. Во-первых, это переход от файлового хранилища к полноценной реляционной базе, что позволит повысить производительность, обеспечить транзакционность и масштабируемость. Во-вторых, внедрение полноценной системы аутентификации и авторизации на базе ASP.NET Core Identity, что даст возможность пользователям создавать личные аккаунты, отслеживать свой прогресс и формировать индивидуальные учебные планы. Наконец, создание Web API на основе существующей бизнес-логики позволит отделить бэкенд от фронтенда и в будущем разработать нативные мобильные клиенты для iOS и Android, использующие единую кодовую базу на сервере.

ЛИТЕРАТУРА

1. ASP.NET Core Documentation. [Электронный ресурс] – Режим доступа: <https://learn.microsoft.com/en-us/aspnet/core/?view=aspnetcore-9.0>. – Дата доступа: 06.08.2025.
2. C# language documentation. [Электронный ресурс] – Режим доступа: <https://docs.microsoft.com/en-us/dotnet/csharp/>. – Дата доступа: 13.08.2025.
3. Круг, С. Веб-дизайн: не заставляйте меня думать! Подход к юзабилити на основе здравого смысла / С. Круг. – 3-е изд. – СПб., Питер, 2021. – 216 с.