Таким образом, благодаря наглядности в преподавании дисциплины «Математика» у обучающихся формируются конкретные образы воспринятого объекта, повышается мотивация к познавательной деятельности.

Реализация наглядности на основе использования информационных технологий способствует активизации познавательной деятельности учащихся. Использование таких средств обучения на лекционных и практических занятиях облегчает образовательный процесс за счет более понятного, яркого и наглядного представления материала.

УДК 004.42

АВТОМАТИЗИРОВАННОЕ ФОРМИРОВАНИЕ WEB-ПРИЛОЖЕНИЙ ДЛЯ ЗАДАННОЙ ПРЕДМЕТНОЙ ОБЛАСТИ

О. Д. Асенчик, В. Ю. Леоненко

Учреждение образования «Гомельский государственный технический университет имени П. О. Сухого», Республика Беларусь

Описана структура и алгоритмы работы программного комплекса для автоматизированного формирования серверных ASP .NET Core приложений с задаваемой архитектурой на базе настраиваемых шаблонов.

Ключевые слова: Web-приложения, архитектура программного обеспечения, ASP .NET Core, RESTful API, low code.

AUTOMATED FORMATION OF THE WEB-APPLICATIONS FOR A GIVEN SUBJECT AREA

O. D. Asenchik, V. Yu. Leonenko

Sukhoi State Technical University of Gomel, the Republic of Belarus

The structure and algorithms of software for automatic formation of server ASP.NET Core applications with a specified architecture based on adjustable templates are described.

Keywords: Web applications, software architecture, ASP .NET Core, RESTful API, low code.

Развитие экономики постоянно вызывает рост спроса на сложные программные системы, обеспечивающие надежную и эффективную работу приложений в различных предметных областях. В настоящее время значительная часть используемых в реальном секторе приложений – это Web-приложения. Их разработка, как правило, осуществляется с использованием языков программирования высокого уровня и может быть значительно ускорена с помощью различных фреймворков и библиотек.

Универсальные фреймворки и библиотеки для ускоренного создания web-приложений, предназначенные для использования профессиональными IT-специалистами, можно разделить в зависимости от их назначения на три большие категории. Приведем примеры основных из них с указанием соответствующих языков программирования: фреймворки для web-разработки широко профиля: Spring Boot (язык Java), ASP.NET Core (язык C#, платформа .NET), Ruby on Rails (язык Ruby), Django (язык Python), Laravel (язык PHP); фреймворки для разработки графических интерфейсов пользователя: React (язык JavaScript/TypeScript), Angular (язык TypeScript), Vue.js (язык JavaScript), Blazor (язык C#, платформа .NET); инструменты для работы с базами данных и API: Entity Framework Core (язык C#, платформа .NET), Hasura (платформа GraphQL), Strapi (Node.js).

Существует также отдельная категория продуктов, применение которых не предполагает постоянное участие IT-специалистов. Они помогают осуществлять визуальное проектирование и автоматическую генерацию приложений. Это так называемые no-code и low-code платформы, позволяющие создавать приложения и автоматизировать бизнес-процессы в заданной предметной области с минимальным или нулевым программированием. Примерами таких платформ являются OutSystems, Mendix, Bubble и др.

Несмотря на преимущества, у no-code и low-code платформ есть ряд проблем и ограничений, таких как техническая и экономическая зависимость от платформы; ограниченная функциональность, производительность и масштабируемость; отсутствие контроля над кодом; ограниченные возможности по безопасности и соответствию ее требованиям; сложность интеграции с более сложными и нестандартными системами, особенно, если они имеют уникальные API или устаревшие технологии.

Поэтому необходимо привлечение IT-специалистов для разработки ряда сложных Web-приложений. Однако многие разработчики, использующие универсальные фреймворки, сталкиваются с проблемами, связанными с созданием сложных серверных приложений на начальных этапах реализации выбранной программной архитектуры и создания базовой оснастки. Создание и настройка приложений, как правило, сопровождаются трудоемкими процессами и многократно повторяющимися однотипными задачами.

Разработка специальных инструментов для более полной автоматизации формирования приложений с заданной архитектурой и заданной предметной области позволит существенно ускорить процесс разработки программного обеспечения (ПО) и повысить его качество.

Цель данной работы – разработка программного комплекса для автоматизированного формирования серверных ASP .NET Core приложений в заданной предметной области и с задаваемой архитектурой. стек технологий ASP .NET Core в настоящее время благодаря кроссплатформенности, высокой производительности и гибкости стал одним из лидирующих решений для разработки серверных web-приложений и web-сервисов.

Опишем структурные и алгоритмические основы функционирования разработанного программного комплекса (ПК).

Для настройки шаблонов генераторов программных архитектур используется система конфигурационных файлов в формате JSON. При этом осуществлена реализация следующих шаблонов: 3-слойной, «чистой» (clean) и гибридной программных архитектур [1].

Конфигурационные файлы (КФ) для каждой из архитектур содержат настройки и параметры создаваемого Web-приложения [2], что позволяет разработчикам изменять параметры приложения без изменения исходного кода и последующей повторной сборки, компиляции и развертывания приложения. Настройки конфигурации решения располагается в трех секциях: директории решения (solutionFolders); проекты (projects); название проекта, хранящего миграции в БД (migrationProjectName).

Секция solutionFolders КФ сохраняет информацию о директориях, находящихся на уровне решения и хранящих в себе проекты. Данная секция является необязательной. Каждая сущность директории проекта содержит идентификатор (id) и название (name). Идентификатор используется для установления принадлежности определенного проекта к заданной директории.

Секция projects КФ включает список проектов приложения. Каждая сущность проекта состоит из следующих свойств: название (name); тип (type); директории

проекта (directories); зависимости (dependencies); тип корневого контента (rootContentType); идентификатор директории решения (solutionFolderName). Поддерживаются два типа проектов: webapi – серверное ASP .NET Core приложение и classlib – .NET библиотека классов.

Сущность директории проекта включает в себя название (name), родительский каталог (parentPath) и тип контента директории (contentType). В каждой директории могут находиться разные типы контента. Поддерживаются следующие типы контента директорий: класс сущности (EntityClass); класс репозитория (RepositoryClass); интерфейс репозитория (RepositoryInterface); контекст базы данных (DbContext); класс объекта передачи данных (DtoClass); класс конфигурации сопоставления сущностей (MappingProfile); класс сервиса (ServiceClass); интерфейс сервиса (ServiceInterface); Program класс (ProgramClass, ProgramClassWithMediatr); контроллер (Controller, ControllerWithMediatr); расширения сервисов (ServiceExtensions, ServiceExtensionsWithServices); запрос на получение всех сущностей определенной модели (GetAllQuery); запрос на получение сущности по идентификатору (GetByIdQuery); команда создания сущности (CreateCommand); команда обновления сущности (UpdateCommand); команда удаления сущности (DeleteCommand); обработчики запросов (GetAllQueryHandler, GetByIdQueryHandler); обработчики команд (CreateCommandHandler, UpdateCommandHandler, DeleteCommandHandler).

Сущность зависимостей проекта разделяется на пакеты (packages) и ссылки на проекты (projectReferences).

С учетом данных, имеющихся в конфигурационных файлах, ПК запускает и выполняет следующие процедуры: создание проектов и директорий, формирование необходимых проектных зависимостей, формирование классов, добавление строки подключения к СУБД и автоматическое формирование и применение миграций для создания соответствующей базы данных на сервере выбранной СУБД.

Специальная процедура создает настройки шаблона класса и возвращает отформатированную строку, представляющую собой код класса. Для различных типов контента, таких как DTO, репозитории, сервисы и контроллеры, алгоритм использует специфические шаблоны. Например, для классов сущностей вызывается метод, который формирует строку с определениями свойств сущности, основываясь на шаблоне и параметрах сущности. В завершение процесса сгенерированные строки записываются в соответствующие файлы.

Создан таким образом и на тестовых примерах апробирован программный комплекс для автоматизированного формирования серверных приложений на платформе ASP .NET Core с возможностью задания архитектуры. Для разработки ПК были использованы язык программирования C# и платформа ASP .NET.

Разработанный ПК предназначен для настраиваемой генерации проектов (решений) среды разработки Microsoft Visual Studio, реализующих различные современные программные архитектуры, для создания серверных Web-приложений на платформе ASP .NET Core. Он обеспечивает возможность задания описания базовых сущностей (классов) предметной области в визуальном режиме, реализует возможность хранения информации об экземплярах сущностей в база данных Microsoft SQL Server или PostgreSQL, а также генерацию классов для реализации бизнес-логики в виде выполнения типовых операций выборки, вставки, удаления и обновления данных об этих экземплярах. ПК генерирует соответствующие прикладные интерфейсы RESTful API и клиентские средства с использованием инструментов Swagger для доступа к реализации этих операций.

Таким образом, разработанный инструмент автоматизации разработки позволяет специалистам сосредоточиться на решении специфических задач проекта, минимизируя рутинные операции. Кроме того, он может послужить ядром low-code платформы, которая будет генерировать приложения с открытым исходным кодом.

Литература

1. Документация Microsoft: Common web application architectures / Microsoft.com. – URL: <https://learn.microsoft.com/en-us/dotnet/architecture/modern-web-apps-azure/common-web-application-architectures> (дата обращения: 16.09.2024).
2. Репозиторий ApiBuilder на GitHub / GitHub.com. – URL: <https://github.com/Olgasn/ApiBuilder> (дата обращения: 16.09.2024).

УДК 004.942

КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ ПРОЦЕССА СУШКИ ПЕСКА В ПОТОКЕ ОТХОДЯЩИХ ГАЗОВ

М. С. Половцев, В. И. Токочаков

Учреждение образования «Гомельский государственный технический университет имени П. О. Сухого», Республика Беларусь

Обсуждена важность сушки песка в различных отраслях промышленности и предложено использование компьютерного моделирования для оптимизации этого процесса и повышения его эффективности. Описаны различные этапы создания модели, начиная от определения входных параметров и заканчивая разработкой числовых и вычислительных моделей. Отмечено, что разработанная компьютерная программа позволяет моделировать сушку песка с использованием различных видов топлива. Доступный графический интерфейс дает возможность пользователю управлять моделью и получать статистические результаты для исследования.

Ключевые слова: сушка песка, компьютерное моделирование, оптимизация, математическая модель, компьютерная модель, графический интерфейс.

COMPUTER SIMULATION OF THE SAND DRYING PROCESS IN THE EXHAUST GAS STREAM

M. S. Polovtsev, V. I. Tokochakov

Sukhoi State Technical University of Gomel, the Republic of Belarus

The report discusses the importance of sand drying in various industries and suggests using computer modeling to optimize this process and increase its efficiency. The paper describes the various stages of model creation, starting from the definition of input parameters and ending with the development of numerical and computational models. The developed computer program allows to simulate sand drying using various types of fuels. An accessible graphical interface allows the user to control the model and obtain statistical results for the study.

Keywords: sand drying, computer modeling, optimization, mathematical model, computer model, graphical interface.

Сушка песка является необходимым этапом подготовки песка для дальнейшего его использования в таких отраслях промышленности, как строительство, нефтегазовая отрасль, металлургия. Оптимизация и повышение эффективности процесса сушки песка позволяют существенно сократить время и затраты на производственные