Описание технической архитектуры программного обеспечения «БАРС-МЭИ»

ФГБОУ ВО «НИУ «МЭИ»

Москва, 2024 г.

ОГЛАВЛЕНИЕ

1. ВВЕДЕНИЕ

Программа для ЭВМ «БАРС-МЭИ» функционирует по клиент-серверной архитектуре и предоставляет пользователем доступ через веб-интерфейс (поддерживается большинство современных веб-браузеров, в том числе на мобильных устройствах).

Система разработана на программной платформе .NET с открытым исходным кодом с использованием языка программирования C#. База данных (БД) работает под управлением свободной объектно-реляционной системы управления базами данных Postgres. Взаимодействие с БД в большинстве случаев обеспечивается за счёт применения объектно-ориентированная технологии доступа к данным ADO.NET Entity Framework.

При разработке веб-приложения использована схема разделения данных Model-View-Controller (MVC, «Модель-Представление-Контроллер»). Приложение функционирует по принципу многостраничного приложения (Multi Page Application, MPA). На клиентской части используются HTML, CSS и JavaScript. Для обеспечения интерактивности пользовательского интерфейса в ряде случаев используется AJAX (Asynchronous Javascript and XML). При этом формат данных передаваемых данных может быть, как JSON, так и HTML.

2. ОБЗОР АРХИТЕКТУРЫ ПРОГРАММННОЙ СИСТЕМЫ БАРС-МЭИ

Текущая архитектура ПО «БАРС-МЭИ» построена по клиент-серверной схеме с тонким (web) клиентом, т.е. специальное программное обеспечение храниться и выполняется на выделенном сервере. Доступ пользователей к функциям системы осуществляется через сеть, посредством прикладного программного обеспечения для просмотра веб-страниц («браузер»).

Система реализует адаптивный интерфейс пользователя, который позволяет взаимодействовать с ней как через стационарный ПК, так и через мобильное устройство (например, через смартфон или планшет).

На рисунке представлена обобщённая архитектора ПО:

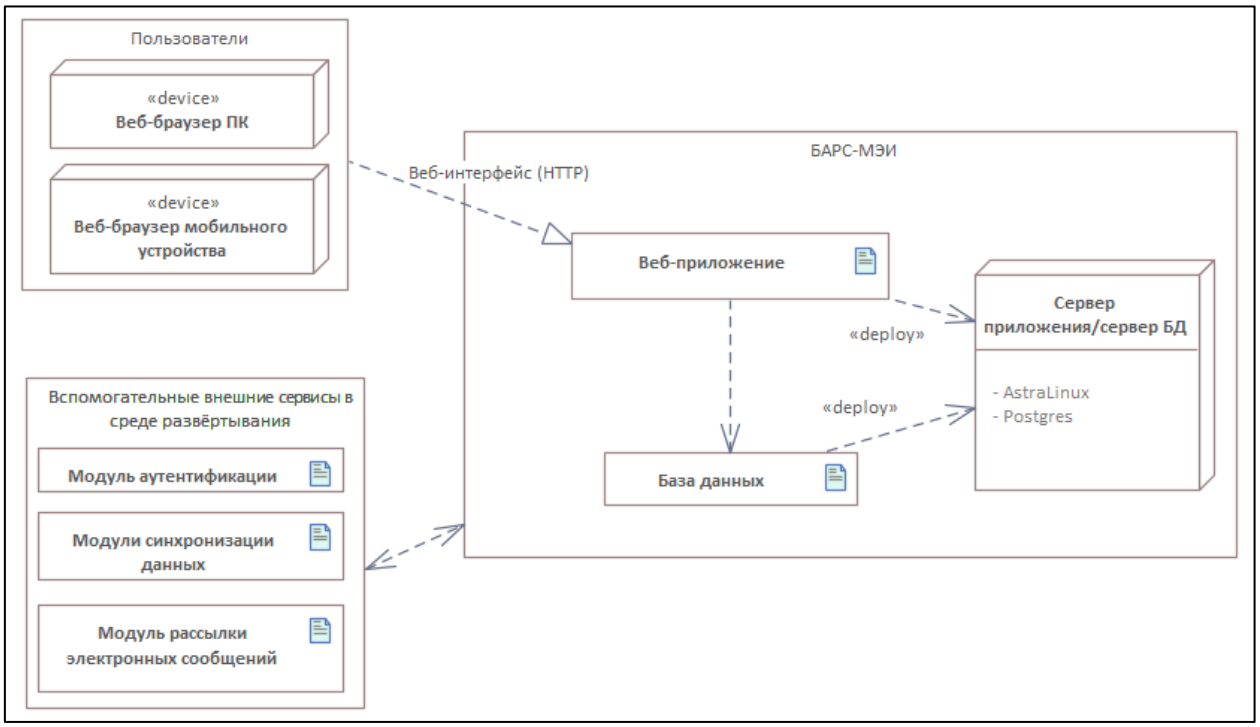

Рис. 1. Архитектура ПО

Модуль веб-приложения размещается на сервере приложений под управлением AstraLinux, база данных – на сервере БД с СУБД Postgres.

Для логического разделения элементов в БД используются различные схемы, среди которых можно выделить следующие:

- «sec» – таблицы, представления и процедуры, связанные с разграничением прав.

- «reference» – хранимые процедуры для генерации документов офисных пакетов.
- «repl» – хранимые процедуры для синхронизации данных и ряда ежедневных внутренних системных процессов.
- «report» – хранимые процедуры применяемые для формирования отчётности.
- «dbo» – основные таблицы, представления, процедуры и функции, связанные с пользовательскими задачами.

При интеграции с использованием технологии представлений СУБД для каждой системы-приёмника создаётся отдельная схема (примеры схем в эталонной версии НИУ МЭИ – lk, portal, edu_prog).

В качестве вспомогательных внешних сервисов среды развёртывания предполагаются:

- Модуль аутентификации, который должен поддерживать взаимодействие с компонентами аутентификации приложения по протоколу Lightweight Directory Access Protocol (LDAP).
- Модули синхронизации данных, которые должны поддерживать обмен данными (передачу данных и, при необходимости, приём) с технологиями доступа через средства СУБД и/или веб-сервисы в согласованном формате.
- Модуль рассылки электронных сообщений, который обеспечивает уведомление пользователей о возникновении «важных» событиях зарегистрированных в системе (например, проверке отчёта о выполненной работе, записи на приём в подразделение).

3. МОДУЛЬ ВЕБ-ПРИЛОЖЕНИЯ

Основная задача модуля – обеспечить интерфейс для доступа пользователей к ресурсам всего приложения.

Модуль реализован по классической архитектуре Model-View-Controller (MVC) или Модель-Представление-Контроллер. Модели (Model) представляют собой классы DTO (data transfer object) для представления данных из БД. Представления отвечают за построение пользовательского веб-интерфейса. Контроллеры служат для обработки пользовательских запросов: принимают параметры запроса от клиента, формируют запрос к БД, генерируют необходимые фрагменты пользовательского интерфейса (HTML, CSS, JavaScript, JSON) и возвращают его в качестве ответа на вопрос.

В ряде случаев для реализации контроллеров используется наследование классов. Для построение типовых интерфейсных решений в представлениях широко задействован механизм макетов (layout). Для взаимодействия с БД используется библиотека EntityFrameworkCore. Для проецирования записей БД на объекты классов веб-приложения используется библиотека AutoMapper. Логгирование осуществляется средствами библиотеки NLog. Также в приложении используются различные вспомогательные Javascript-библиотеки.

Для генерации документов для пакетов офисных приложений используется формат OpenXML. Вспомогательная библиотека DocumentGenerator реализованная в составе БАРС-МЭИ обеспечивает функционал экспорта документов, для работы с которыми далее можно использовать программы типа текстового (например, LibreOffice Writer, OpenOffice Writer) и табличного (например, Excel, LibreOffice Calc, OpenOffice Calc, Excel) процессоров.

В файловой структуре приложения используется группировка элементов MVC по областям (Area). Каждая область содержит классы, отвечающие за определённое функциональное направление. Ниже перечислены основные Area и основные задачи, которые они позволяют решить:

Наименование области (Area)	Комментарий
Admin	Административные возможности: редактирование пользователей, групп пользователей, прав доступа и т.п.
Guide	Административный возможности: ведение справочников.
Open	Данные доступные без аутентификации.
StudyGroup	Возможности в рамках личного кабинета учебной группы (ЛКГ).
Timetable	Просмотр расписания.
User	Профиль пользователя.
Области, связанные с личным кабинетом подразделения (ЛКПод).	
AE_INST	ЛКПод. Возможности для подразделения типа «Институт».
AE_LK	ЛКПод. Общие возможности для подразделений различных типов.
AE_ORK	ЛКПод. Возможности для Отдела развития и карьеры.
AE_SF	ЛКПод. Возможности для подразделений типа «Кафедра».
AEF_Q	ЛКПод. Возможности для проведения анкетирования.
AEF_Bypass	ЛКПод. Возможности по согласованию обходного листа.
AEF_Cont	ЛКПод. Возможности по учёту контингента учебных групп, студентов, преподавателей и т.п.
AEF_Event	ЛКПод. Возможности по учёту данных относящихся к общественному и культурно-творческому рейтингам.
AEF_EventN3	
AEF_EventN4	
AEF_EventN4_Archive	

AEF_Sched	ЛКПод. Возможности по работе с графиком приёма подразделения.
AEF_Sec	ЛКПод. Возможности по работе с правами доступа в ЛКПод для руководителя подразделения.
AEF_SOD	ЛКПод. Возможности по работе с данными о высшем образовании из системы оформления документов (СОД).
AEF_Study	ЛКПод. Возможности по работе с аттестационными документами и ведомостями преподавателей.
Области, связанные с личным кабинетом преподавателя (сотрудника) (ЛКП)	
EMP_LK	ЛКП. Просмотр информации о сотруднике.
EMP_Study	ЛКП. Возможности, связанные с учебной деятельностью преподавателя (за исключением ведения электронного журнала)
Employee	ЛКП. Список сотрудников, отображение данных ЛКП в виде иерархии, работа с правами помощников.
Journal	ЛКП. Возможности, связанные с учебной деятельностью преподавателя (ведение электронного журнала)
Области, связанные с личным кабинетом студента (ЛКС)	
ST_Q	Возможности для проведения анкетирования
ST_LK	ЛКС. Работа с общей информации о студенте.
ST_Study	ЛКС. Возможности в рамках учебной деятельности студента.
Области «ST_Part» - учёт данных относящихся к рейтингу студента	
ST_PartN1	Работа с данными по учебному рейтингу.
ST_PartN2	Работа с данными по научно-инновационному рейтингу.
ST_PartN3	Работа с данными по общественному рейтингу.
ST_PartN4	Работа с данными по культурно-творческому рейтингу.
ST_PartN5	Работа с данными по спортивному рейтингу.

ST_PartN4_Archive	Просмотр архивных записей.
ST_Part2	
ST_Part3	

Из элементов проекта, обеспечивающих инфраструктурные вопросы, работы веб-приложения можно выделить:

- Program.cs – класс, реализующий точку входа в приложение и отвечающий за регистрацию и настройку всех используемых приложением сервисов, включая ведение журнала, подготовку инфраструктуры MVC и настройку маршрутов.
- BARSRepository.cs – отвечает за предоставления доступа контроллерам к сведениям из БД. Содержит сведения о текущем пользователе и его полномочия по работе в системе.
- bundleconfig.json – файл настройки формирования пакетов JavaScript и CSS.
- Mvc.sitemap – файл настройки карты сайта в формате библиотеки MvcSiteMapProvider.MVC5.Core.
- appsettings.json – файл с настройками приложения. Среди прочего содержит параметры подключения к БД, пути к директориям в файловой системе, параметры подключения к внешним сервисам.
- nlog.config – файл с настройками журналирования. Журналирование событий реализовано на основе библиотеки NLog. Настройки хранятся в соответствующем разделе файла конфигурации appsettings.json и в файле nlog.config. Запись в журнал в коде доступна через StaticContext.CurrentLogger через методы Error/Trace/Debug/Info в соответствии с необходимым уровнем логгирования.

Ниже представлена обобщённая схема обработки запроса пользователя в БАРС-МЭИ:

Рис. 2. Обработка запроса пользователя в БАРС-МЭИ.

Получив запрос от пользователя система «пропускает» его через промежуточные блоки конвейерной обработки (в англоязычной литературе используется термин «Middleware»). Каждый блок этого слоя пытается обработать запрос и получить ответ. Если блоку Middleware получается обработать запрос, то он возвращается пользователю. Если не получается – то запрос передаётся следующему блоку.

Блок маршрутизации отвечает за разбор URL, выделении всех параметров запроса и поиск подходящего обработчика (контроллера).

Блок инициализации контроллера служит для создания объекта класса контроллера. Далее обработка запроса передаётся контроллеру.

Блок выполнения метода контроллера обслуживает жизненный цикл метода. При этом могут отрабатывать дополнительные пред- и пост-фильтры метода. Например, фильтры авторизации и фильтры. В ходе обработки запроса программный код метода может обращаться к различным источникам данных (например, БД, веб-сервисы) запрашивая необходимую информацию и/или инициируя внесение изменений.

Метод контроллера возвращает объект результата, который затем преобразуется в необходимый пользователю формат (например, HTML, JSON, файл, статус). Если в качестве ответа предполагается HTML, то для формирования ответа объект результата передаётся блоку построения HTML («движку представлений»).

Сформированный ответ возвращается пользователю.