Компьютерные инструменты в образовании, 2024
№ -: 1–15
УДК: 004.4
http://cte.eltech.ru

ВЕБ-ОРИЕНТИРОВАННАЯ СИСТЕМА ДЛЯ ОБРАБОТКИ ДАННЫХ МЕДИЦИНСКИХ ИССЛЕДОВАНИЙ

Топузов Э. Р.[1], аспирант, ✉ edem.topuzov@gmail.com, orcid.org/0009-0003-9208-7931
Ампилова Н. Б.[1], к.ф.-м.н., доцент, доцент кафедры информатики, n.ampilova@spbu.ru, orcid.org/0000-0002-2154-9399

[1]СПбГУ, Санкт-Петербургский государственный университет, Университетский проспект, д. 28, Санкт-Петербург, Петергоф, 198504, Россия

Аннотация

В статье рассмотрены проблемы, связанные с обработкой большого объема медицинских данных. Перечислены ограничения использования существующих медицинских информационных систем в индивидуальных исследованиях. Рассматривается веб-ориентированная система, разработанная для упрощения и автоматизации процессов медицинских исследований. Система имеет ряд преимуществ по сравнению с существующими медицинскими информационными системами, а именно позволяет создавать дизайн исследований, обеспечивает совместную работу, ввод данных, управление доступом, аудит изменений базы данных, регулярное резервное копирование, анализ и экспорт данных в виде таблиц и графиков. Разработанная система внедрена в эксплуатацию и получила положительные отзывы от пользователей.

Ключевые слова: *медицинские информационные системы, обработка медицинских данных, веб-ориентированная система*

Цитирование: Топузов Э. Р., Ампилова Н. Б. Веб-ориентированная система для обработки данных медицинских исследований // Компьютерные инструменты в образовании. 2024. № -. С. 1–15.

1. ВВЕДЕНИЕ

Применение современных технологий в области хранения и обработки больших объемов информации позволяет использовать технологические достижения в области медицины для некоторой оптимизации медицинских исследований. Для подтверждения научно-медицинских гипотез выполняются обширные рандомизированные исследования, в ходе которых осуществляется сбор многочисленных показателей, подлежащих статистической обработке. [1] Объемы медицинских данных постоянно растут, а процесс извлечения полезной информации из них является достаточно трудоемким [2–4]. Для решения подобных задач широко используются медицинские информационные системы (МИС) [5, 6].

Обзор исследований в области МИС показывает, что в настоящее время не существует систем, разработанных для выполнения индивидуальных исследований, например, диссертационных работ, грантов или государственных заданий. Анализ процессов обработки данных медицинских исследований выявил, что большинство программных реше-

ний чаще всего ориентировано на узкий круг клинических задач и, как правило, сводится к созданию электронных медицинских карт или систем, предполагающих возможный диагноз на основе клинико-лабораторных данных [5, 7–9].

В таблице 1 представлена сравнительная характеристика существующих способов обработки данных медицинских исследований: с помощью МИС (на примере qMS, АМУЛЕТ, Фобос-медицина, КМИС), а также таблиц Excel или Google Sheets.

Таблица 1. Соответствие существующих способов хранения и обработки данных требованиям проведения медицинских исследований

Критерий	МИС	Таблицы Excel, Google Sheets
Подбор пациентов для исследования из общей базы данных	+	–
Возможность прикладывать файлы с дополнительной информацией о пациентах	+	–
Возможность создания дизайна конкретного исследования	–	+
Возможность хранения данных для конкретного исследования	–	+
Возможность автоматизированной индивидуальной группировки данных по показателям согласно дизайну конкретного исследования	–	–
Возможность автоматизированного статистического анализа данных конкретного исследования по заданным группам	–	–
Возможность выгрузки данных в виде таблиц для последующей статистической обработки	–	+
Возможность автоматизированной группировки данных для последующей статистической обработки	–	–
Возможность построения графиков и диаграмм	–	+
Гибкое ограничение прав пользователей для конкретного исследования	–	–
Хранение данных и информации группы контроля (здоровые и добровольцы)	–	+

В таблице перечислены критерии, требуемые для выполнения конкретных индивидуальных медицинских исследований. Так, например, каждая медицинская научно-исследовательская работа требует создания индивидуального дизайна исследования, включающего определение необходимых показателей, их группировку и применение статистических методов обработки, что доступные МИС не позволяют сделать, так как эти системы в основном направлены на решение клинических задач, и следовательно, в полной мере не имеют необходимых для научного исследования функциональных возможностей, указанных в таблице 1, ввиду того, что на сегодняшний день они не реализованы, а поэтому не представлены в таких системах. По этой причине в настоящее время врачи-исследователи вынуждены вручную формировать множество таблиц Excel, копируя данные между ними. Указанное решение является трудозатратным и поэтому малоэффективным.

Таким образом, существует потребность в разработке более гибкой и многофункциональной МИС, способной эффективно поддерживать разнообразные научные исследования в области медицины.

Предлагаемый нами вариант решения проблемы – разработка веб-ориентированной

системы, которую можно использовать на различных этапах медицинских исследований: от ввода данных о пациентах до автоматического анализа показателей и предоставления прогностических данных.

Использование веб-подхода обеспечивает следующие свойства системы:

- совместимость с любой операционной системой;
- простой запуск при переходе по адресу веб-ресурса;
- отсутствие необходимости загрузки, установки, обновления;
- экономичность разработки и обслуживания [10].

Преимуществом такого решения является соответствие всем требованиям, указанным в таблице 1, что обеспечивает:

- создание дизайна медицинского исследования;
- внесение данных;
- предоставление разных уровней доступа к системе;
- контроль со стороны главного исследователя изменений в базе данных (БД);
- регулярное резервное копирование БД;
- проведение анализа содержимого БД;
- выгрузку внесенных данных в таблицы Excel с проведением статистического анализа;
- построение и экспорт графиков и диаграмм.

2. МАТЕРИАЛЫ И МЕТОДЫ

2.1. Архитектура прототипа системы

В первую очередь была разработана архитектура приложения, запланирована реализация серверной и клиентских частей. Использовалась контейнеризация с помощью Docker. Это позволяет изолировать приложение от основной операционной системы, что с одной стороны повышает безопасность приложения, с другой позволяет легко настроить окружение, системные утилиты, настройки, не влияя на инфраструктуру узла, на котором эксплуатируется система. Кроме того, это ускоряет разворачивание приложения, позволяет масштабировать его. Для организации взаимодействия контейнеров между собой используется утилита docker-compose. В дальнейшем при существенном расширении проекта возможен переход к средствам оркестрации Docker Swarm или Kubernetes.

В качестве базового концепта использовалась схема «модель — представление — контроллер» для разделения данных приложения на отдельные компоненты, взаимодействующие между собой. Для обеспечения многопоточности и быстроты работы использован веб-сервер Puma, в качестве обратного прокси-сервера используется nginx. Связь этих двух компонентов обеспечивает взаимодействие клиента с основной частью системы, а именно отправку запросов для дальнейшей обработки маршрутизатором и получения ответа. Для проверки концепции прототипа и с целью ускорения разработки и внедрения системы, компонент, отвечающий за статистическую обработку данных, является элементом основной части приложения, и таким образом получает данные из системы управления БД (СУБД) через слой модели. Были подобраны инструменты, обеспечивающие расчеты требуемых показателей и экспорт полученных данных, а также связанные с безопасностью: аутентификацией, авторизацией и управлением ролями

пользователей, шифрованием данных, предотвращением и обнаружением попыток несанкционированного доступа и повреждения данных, созданием и восстановлением резервных копий данных, механизмом уведомлений о появлении уязвимостей программных решений. Схема архитектуры прототипа указана на рисунке 1 . Технологический стек и основные компоненты представлены в таблице 2.

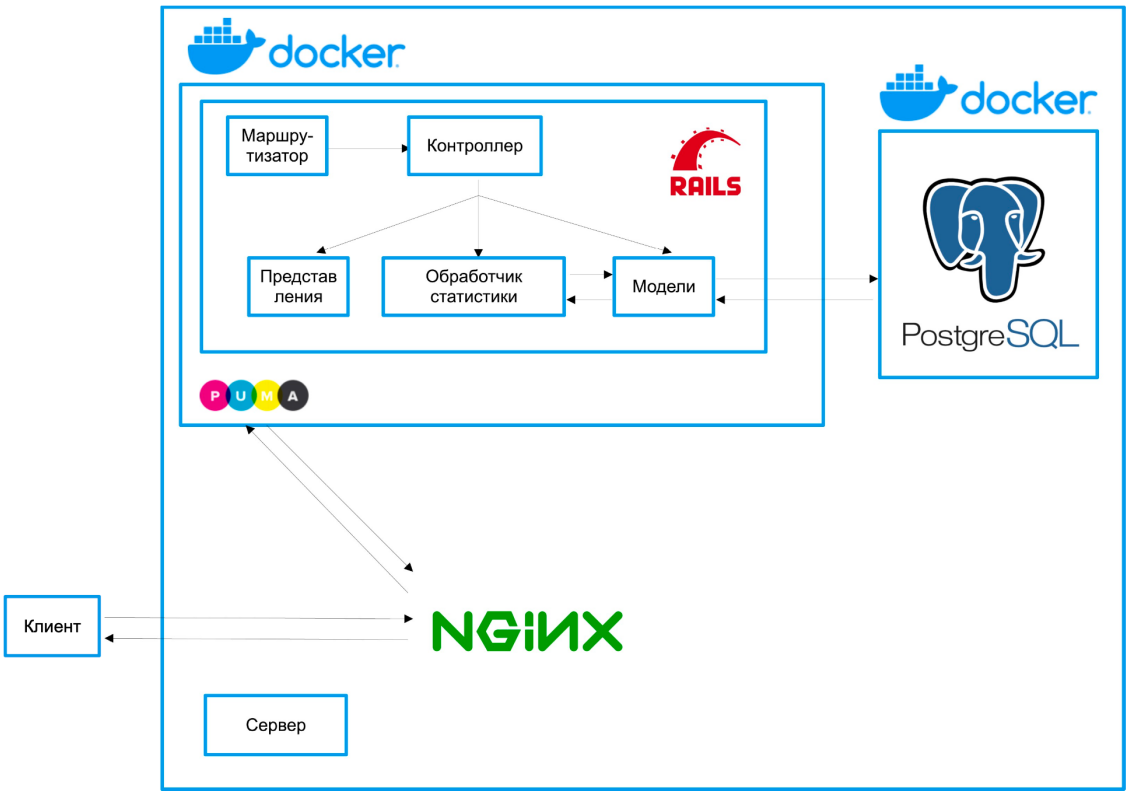

Рис. 1. Архитектура прототипа

Таблица 2. Технологический стек разрабатываемой системы

Параметр	Значение
Веб-фреймворк	Ruby on Rails
СУБД	PostgreSQL, MongoDB
Управление зависимостями библиотек	Bundler
Аутентификация	Devise
Авторизация	ActionPolicy, RoleModel
Контроль изменений в БД	PaperTrail
Работа со статистикой	ruby-statistics
Импорт и экспорт Excel	RubyXL
Средство контейнеризации	Docker
Веб-сервер	Puma
Обратный прокси-сервер	Nginx
Средства тестирования	RSpec, Simplecov
Линтер	Rubocop
Средство отслеживания уязвимостей	Bundle Audit
Сборка клиентской части	Webpack
Шаблонизатор	Slim
Система контроля версий (удаленный репозиторий)	Git (GitHub)

2.2. Компоненты системы

2.2.1. Основная среда

Важным шагом разработки является выбор подходящей среды разработки (фреймворка) – динамичной библиотеки языка программирования, обладающей базовыми модулями для упрощения работы [11].

В качестве такой среды выбран Ruby on Rails. Она позволяет вести разработку, быстро выводя продукт в рабочий прототип, что дает возможность осуществить проверку концепции и протестировать минимально жизнеспособный продукт [12, 13]. Кроме того, фреймворк хорошо масштабируется и успешно используется в таких высоконагруженных международных сервисах как GitHub, GitLab, AirBnB [14], а также в российских платформах Сбермаркет, Учи.ру, Aviasales и многих других.

Архитектура приложений Ruby on Rails определяется как полнофункциональная программная платформа (fullstack framework), поскольку она может работать со всеми возможными частями веб-приложения. К таким частям можно отнести: веб-сервисы, управление БД, генерация шаблонов на языке гипертекстовой разметки (HyperText Markup Language, HTML), сборка каскадных таблиц стилей (cascading style sheets, CSS) и JavaScript файлов, маршрутизация, кеширование, аутентификация, авторизация, локализация и многое другое [12–14].

Одним из преимуществ данной среды, увеличивающим скорость разработки приложения является принцип соглашения по конфигурации (convention over configuration). Данный принцип позволяет избегать большого количества повторяющегося кода, а также конфигурировать программный комплекс только тех случаях, когда какие-то его части не удовлетворяют принятой в фреймворке спецификации [15, 16]. Данная концепция также увеличивает скорость адаптации персонала для развития и поддержки разработанного программного комплекса, что также является важным аспектом.

Среда Ruby on Rails, уделяет большое внимание безопасности. Многие механизмы встроены – это экранирование запросов, выполняемых на языке структурированных за-

просов (structured query language, SQL) для защиты от SQL-инъекций, защита от межсайтовой подделки запроса (Cross-Site Request Forgery, CSRF), защита от межсайтового скриптинга (cross-site scripting, XSS), фильтрация чувствительных данных при логировании, очистка данных из БД при выводе страницы в браузер, встроенное кодирование данных при сохранении в БД и многое другое [17].

Благодаря высокой популярности среды Ruby on Rails в разработке веб-приложений, в ее экосистеме существует огромное количество библиотек, предназначенных для различных целей и расширения функционала. В качестве менеджера пакетов Ruby был использован сборщик bundler, который скачивает библиотеки, выстраивает дерево зависимостей между ними, не допуская конфликтов [18].

2.2.2. СУБД

В качестве СУБД выбрана PostgreSQL в связи с тем, что в приложении достаточно много связанных между собой сущностей со строгой системой атрибутов. PostgreSQL позволяет осуществлять многопользовательский доступ к БД, обладает всеми необходимыми свойствами: атомарностью, согласованностью, изолированностью, устойчивостью (atomicity, consistency, isolation, durability, ACID). Эта система при необходимости позволяет бесперебойно масштабировать приложение с помощью распределенных СУБД, совместимых с PostgreSQL, таких как CockroachDB, YugabyteDB и других. В связи с тем, что личные данные пациентов (фамилия, имя, отчество, контактные данные) являются конфиденциальными, использовано расширение pgcrypto. Шифрование и дешифрование осуществляются с помощью симметричного ключа Pretty Good Privacy (PGP) по стандарту OpenPGP (RFC 4880) [19].

Принимая во внимание предъявляемые требования к разрабатываемой системе, была спроектирована схема БД, которая включает в себя таблицы для хранения данных о пользователях, ролях системы, пациентах, исследованиях, группах и подгруппах исследований и пациентов, результатах исследований, а также таблицы для хранения метаданных и связей между данными.

Во время разработки были созданы две отдельные БД – непосредственно для разработки и для тестирования, что позволило изолировать данные, избежать конфликты и ошибки, связанные со случайным изменением или удалением данных во время указанных процессов, делая их более гибкими.

Изменения в схеме БД поддерживаются с помощью миграций, которые позволяют обеспечивать актуальное состояние схемы БД, гибко добавлять, изменять, удалять таблицы, индексы, триггеры, функции и прочие объекты СУБД, согласовывать текущее состояние с помощью системы контроля версий Git [20].

Распространенной причиной снижения производительности веб-приложений, работающих с БД, является так называемая проблема N+1. Ее суть заключается в том, что при запросе к БД для получения списка объектов с последующим доступом к связанным объектам, для каждого из них, может возникнуть N дополнительных запросов к БД, где N — количество связанных объектов для каждого из исходных объектов. В качестве решения проблемы, разработчики используют технику оптимизации запросов к БД — нетерпеливую загрузку. Она позволяет предварительно загружать связанные данные вместе с основными данными. Однако необдуманное использование этой техники также снижает производительности работы приложения. Для предотвращения проблемы N+1 и неоправданного использования нетерпеливой загрузки во время разработки и тестирования использовалась библиотека bullet [14].

Для сбора данных о выгружаемых исследователями отчетах используется СУБД MongoDB. Эти данные представляют интерес для дальнейшего исследования и разработки. Необходимо накопить достаточно большой объем данных, чтобы впоследствии можно было реализовать функцию подсказки или предсказания выгружаемых параметров. Данная СУБД является документоориентированной, не требует описания схемы таблиц. Получаемые данные о выгружаемых отчетах разнородны, слабо структурированы, поэтому хорошо вписываются в концепцию БД без схемы [20].

2.2.3. Аутентификация и авторизация

Для аутентификации использована библиотека Devise [16, 18]. Для выделений ролей и авторизации было использовано сочетание библиотек RoleModel и ActionPolicy. Последняя обладает гибкостью для управления правами на различные действия в веб-приложении, в том числе ограничение на создание, чтение, обновление, удаление сущностей, ограничение списка сущностей для просмотра и т.д.

Одно из требований к системе – контроль произведенных изменений (создание, изменение, удаление записей) в БД, сделанных разными пользователями. Данный функционал реализован с помощью библиотеки PaperTrail, которая позволяет проводить аудит изменений, сделанных разными пользователями. Инструментарий позволяет отслеживать какие изменения каким пользователем были осуществлены, кроме того, при необходимости производить откат и восстановление данных [16].

2.2.4. Клиентская часть

В качестве шаблонизатора был использован slim [18] – быстрый легковесный движок, который по производительности быстрее используемого по умолчанию встроенного в Ruby on Rails ERB (встроенный Ruby, embedded Ruby). Другим преимуществом является отсутствие открывающих и закрывающих HTML-тегов, что с одной стороны ускоряет разработку, с другой – уменьшает вероятность ошибок.

Для динамической отрисовки элементов, например, добавления, изменения, скрытия полей ввода в веб-формах и т.д., в зависимости от действий пользователей, был использован JavaScript.

Для ввода дат и диапазона дат используется отечественный легкий, быстрый календарь с открытым исходным кодом, написанный на JavaScript без сторонних зависимостей – Air Datepicker.

В качестве менеджера JavaScript библиотек использовался yarn, который скачивает их, выстраивает зависимости между ними, предотвращая конфликты несовместимых версий.

Для ускорения разработки использован CSS-фреймворк Bootstrap, как наиболее популярный (по количеству звезд Github и скачиваний в составе npm-пакетов), в его основе лежит компонентный подход [21].

Сборка и компоновка CSS и JavaScript файлов осуществлялась с помощью библиотеки webpack. Для удобства использовалась Ruby-обертка – webpacker, которая позволяет удобно организовать код и имеет встроенные Ruby-методы для доступа к стилям, скриптам, изображениям, шрифтам и прочим объектам. Во время разработки сервер webpack-dev-server позволял вносить правки в стили и скрипты таким образом, чтобы эти изменения подхватывались на лету без перезагрузки сервера, что также ускоряло процесс разработки [22].

2.2.5. Тестирование

Для тестирования был использован фреймворк RSpec – наиболее популярное средство для тестирования приложений, написанных на Ruby. RSpec позволяет наглядно с помощью предметно-ориентированного языка (domain specific language, DSL) писать тесты так, что их можно читать, как документацию (spec – спецификация), при этом существует опция, которая в стандартном выводе на терминал в структурированном виде указывает что делает тестируемый класс и метод. Тестирование с помощью RSpec с одной стороны позволит в дальнейшем развивать приложение, масштабируя его, с другой — благодаря самодокументированности тестов увеличит скорость адаптации к проекту новых разработчиков [18, 23]. Для вычисления степени покрытия кодовой базы тестами была использована библиотека SimpleCov [23].

2.2.6. Статический анализ кода

Для статического анализа кода по заданным правилам используются линтеры (linter) - инструменты, позволяющие выявлять потенциальные проблемы или стилевые несоответствия в исходном коде программы [24]. Чтобы сделать код единообразным, удобным для восприятия, в том числе для дальнейшей поддержки и развития приложения, используется линтер Rubocop [25], который автоматически вызывается перед каждым сохранением кода в системе управления версиями Git с помощью утилиты lefthook.

Перед отправкой изменений во внешний репозиторий Git эта же утилита вызывает задачу bundle-audit, которая проверяет имеющиеся в проекте библиотеки на уязвимости, используя БД общеизвестных уязвимостей (Common Vulnerabilities and Exposures, CVE). Это позволяет поддерживать безопасность проекта, вовремя реагируя на выявляемые угрозы, обновляя библиотеки и их зависимости.

2.2.7. Изменение архитектуры после испытаний прототипа

В ходе разработки после внедрения в эксплуатацию рабочего прототипа, получения положительной обратной связи от пользователей системы, а также запросов на расширение функционала, архитектура была модифицирована. В связи с тем, что количество и уровень поддержки библиотек, отвечающих за статистическую обработку на языке Ruby ниже по сравнению с Python, для дальнейшего развития системы был добавлен отдельный микросервис, отвечающий за статистическую обработку данных, с использованием Python в качестве основного языка [26]. Кроме того, это позволило осуществить построение графиков непосредственно в самой системе с возможностью их экспорта.

В роли менеджера зависимостей Python была применена библиотека Poetry [27], которая кроме построения и управления деревом связанности пакетов, позволяет конфигурировать их настройки с помощью нотаций очевидного минимального языка Тома (Tom's Obvious, Minimal Language, TOML).

Для статистической обработки данных использованы как встроенная в Python библиотека Statistics, так и внешние пакеты Scipy [28], Numpy.

Построение графиков осуществляется посредством пакетов Matplotlib [27] и Seaborn.

Для тестирования использованы Pytest [27] и Hypothesis. Для вычисления степени покрытия кодовой базы тестами была использована библиотека Pytest-cov [27].

Соответствие стандарту предложения по усовершенствованию Python (Python Enhancement Proposal #8, PEP 8) проверялось при помощи статического анализатора кода

Flake8 [29].

Взаимодействие внутри сервиса, отвечающего за статистику, а также обмен данными с основным приложением в формате нотации объектов JavaScript (JavaScript Object Notation, JSON) налажено с использованием легкого веб-фреймворка Flask [30].

Таким образом, в результате добавления нового микросервиса, технологический стек существенно расширился (таблица 3).

Таблица 3. Расширение технологического стека системы

Параметр	Значение
Управление зависимостями библиотек	Poetry
Работа со статистикой	Statistics, Scipy, Numpy
Построение графиков	Matplotlib, Seaborn
Веб-сервер	Flask
Средства тестирования	Pytest, Hypothesis, Pytest-cov
Линтер	Flake8

С учетом указанных изменений, архитектура приложения претерпела изменения, обновленная версия приведена на рисунке 2.

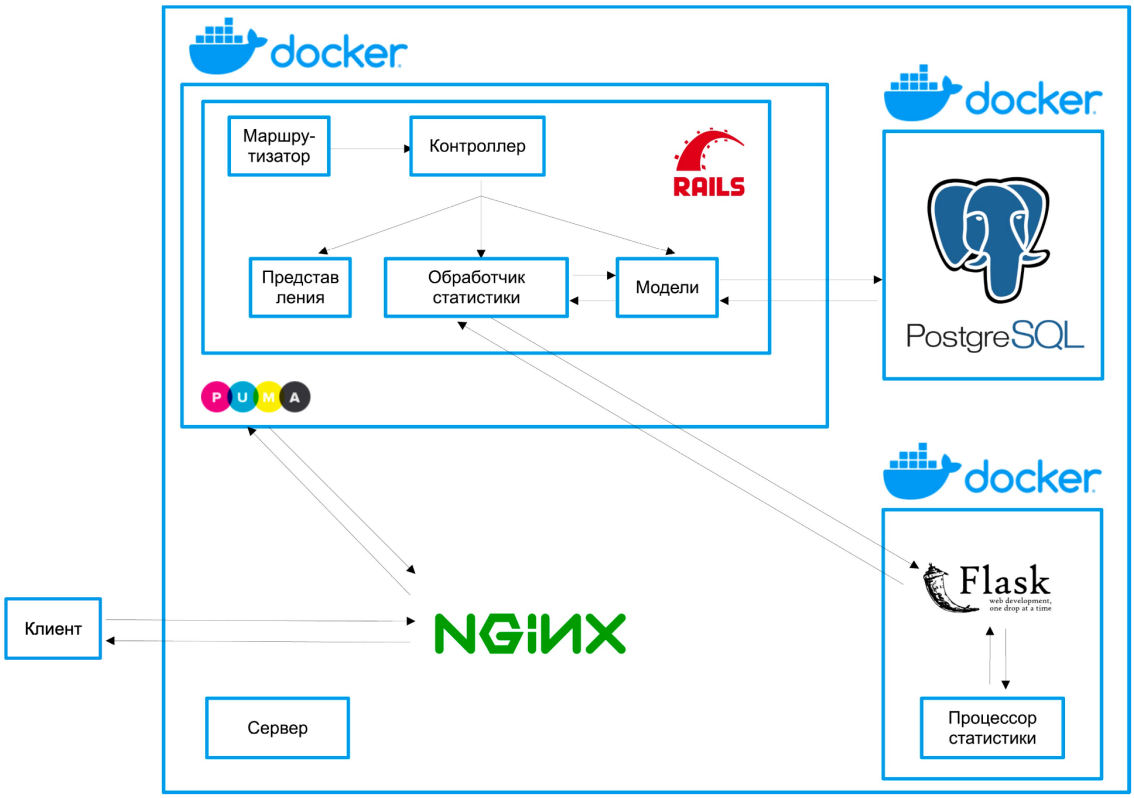

Рис. 2. Обновленная архитектура системы

3. РЕЗУЛЬТАТЫ

Веб-ориентированная система для обработки данных медицинских исследований разработана в соответствии со спроектированной архитектурой, приложение развернуто на виртуальном выделенном сервере.

В таблице 4 представлен обзор функций, обеспечиваемых разработанной системой.

Таблица 4. Функциональные возможности системы

Функции
Создание дизайна медицинского исследования
Внесение данных, в т.ч. согласно группам и подгруппам, выделенным в дизайне
Предоставление разных уровней доступа к системе (полные права, ограниченные права) для обеспечения безопасности данных
Контроль со стороны главного исследователя изменений в базе данных, сделанными разными пользователями
Регулярное резервное копирование и восстановление (при необходимости) копий БД
Проведение анализа содержимого БД, внесенных в систему, путем поиска и просмотра данных по заданным параметрам
Выгрузка внесенных данных в таблицы Excel
Кодирование данных, внесенных в систему с присвоением кодов для регистрации БД и подготовки к использованию в программах статистической обработки
Группировка данных с выгрузкой в таблицы Excel для проведения статистической обработки, основанное на выделении определенных группирующих показателей (как одиночных, так и множественных)
Проведение статистического анализа
Построение и экспорт графиков

Разработанная система покрыта тестами. Покрытие тестами веб-сервиса, написанного на Ruby – 99.3%, покрытие тестами сервиса статистики, написанного на Python – 98%.

Система внедрена в эксплуатацию в Национальном медицинском исследовательском центре имени В. А. Алмазова. За время эксплуатации система показала свою работу без сбоев, от пользователей получена положительная обратная связь, а также запросы для расширения функционала.

4. ЗАКЛЮЧЕНИЕ

Разработанная система предусматривает дальнейшее развитие: предсказание выгружаемых отчетов, расширение списка методов статистической обработки и диаграмм, необходимых для конкретных исследований. Также предполагается расширение функциональных возможностей для поддержки научных исследований. Все это в конечном итоге позволит полностью отказаться от использования дополнительных программных комплексов для статистического анализа.

Таким образом, в ходе проделанной работы спроектирована архитектура, подобраны подходящие для решаемой задачи инструменты, разработан, протестирован и внедрен программный продукт. Получено свидетельство о государственной регистрации программы для ЭВМ №2023688104 от 20 декабря 2023 года. Также были продуманы и представлены дальнейшие варианты развития системы. В связи с поступающими запросами для расширения функционала системы, работа над ее разработкой будет продолжена.

Список литературы

1. *Evidence-Based Medicine Working Group* “Evidence-based medicine. A new approach to teaching the practice of medicine,” JAMA, vol. 268, no. 17, pp. 2420-2425, 1992, doi: 10.1001/jama.1992.03490170092032.
2. *A.V. Gusev* “Prospects for the use of big data in Russian healthcare,” Moscow Medicine, vol. 1, no. 47, pp. 26-30, 2022 (In Russian).
3. *K. Moutselos, D. Kyriazis, V. Diamantopoulou, et al.* “Trustworthy data processing for health analytics tasks,” IEEE International Conference on Big Data (Big Data), pp. 3774-3779, 2018, doi: 10.1109/BigData.2018.8622449.
4. *C. Auffray, R. Balling, I. Barroso, et al.* “Making sense of big data in health research: towards an EU action plan,” Genome medicine, vol. 8, no. 1, pp. 1-13, Jun. 2016, doi: 10.1186/s13073-016-0323-y.
5. *A.S. Shakhanov, E.V. Ushakova* “Use of modern information technologies in public administration,” Transformation of business and public institutions in the context of digitalization of the economy, pp. 199-211, 2020 (In Russian).
6. *E.V. Lutsenko* “Development of medical information technologies in the Russian Federation,” Medical newsletter of Vyatka, vol. 2, no. 54, pp. 73-77, 2017 (In Russian).
7. *T. Magrupov, S. Yusupov, Y. Talatov, et al.* “Intelligent Medical System of Designing Medical Technics and Technology,” 2020 International Conference on Information Science and Communications Technologies (ICISCT), pp. 1-4, 2020, doi: 10.1109/ICISCT50599.2020.9351375.
8. *P.S. Pugachev, A.V. Gusev, O.S. Kobyakova, et al.* “Global trends in the digital transformation of the healthcare industry,” National Health Care (Russia), vol. 2, no. 2, pp. 5-12, Nov. 2021, doi: 10.47093/2713-069x.2021.2.2.5-12 (In Russian).
9. *D. Singh, S. Verma, J. Singla.* “A comprehensive review of intelligent medical diagnostic systems,” 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184), pp. 977-981, 2020, doi: 10.1109/ICOEI48184.2020.9143043.
10. *Y. Duan, J. Edwards, M. Xu* “Web-based expert systems: Benefits and challenges,” Information & Management, vol. 42, no. 6, pp. 799–811, Sep. 2005, doi: 10.1016/j.im.2004.08.005.
11. *A.A. Baidybekov, R.G. Gilvanov, I.A. Molodkin* “Modern Frameworks for Web Development,” Intellectual Technologies on Transport, no. 4, pp. 23-29, 2020 (In Russian).
12. *D. Klochkov, J. Mulawka* “Improving Ruby on Rails-Based Web Application Performance,” Information, vol. 12, no. 8, p. 319, Aug. 2021, doi: 10.3390/info12080319.
13. *P. Łuczak, A. Poniszewska-Maranda, V. Karovič* “The Process of Creating Web Applications in Ruby on Rails,” in Developments in Information & Knowledge Management for Business Applications. Studies in Systems, Decision and Control, vol. 330. Cham: Springer, 2021, pp. 371-401, doi: 10.1007/978-3-030-62151-3_9.
14. *J. Yang, P. Subramaniam, S. Lu, et al.* “How not to structure your database-backed web applications,” 2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE), pp. 800-810, 2018, doi: 10.1145/3180155.3180194.
15. *C. Chunling* “Construction of the Individualized College English Learning Management System Using Ruby on Rails,” 2015 International Conference on Service Science (ICSS), pp. 160-163, 2015, doi: 10.1109/ICSS.2015.22.
16. *A. Tapiador, J. Salvachúa* “Content Management in Ruby on Rails,” Proceedings of the IADIS International Conference on Collaborative Technologies 2011, 2011, doi: 10.48550/arxiv.1209.3878.
17. *A. Aborujilah, J. Adamu, A.M. Shariff, et al.* “Descriptive Analysis of Built-in Security Features in Web Development Frameworks,” 2022 16th International Conference on Ubiquitous Information Management and Communication (IMCOM), pp. 1-8, 2022, doi: 10.1109/IMCOM53663.2022.9721750.
18. *G. Donald* Hands-on Test-Driven Development Using Ruby, Ruby on Rails, and RSpec. Berkeley, Apress, 2024.
19. *M. Costa, M. Rodrigues, P. Baptista, et al.* “Database Encryption Performance Impact on PostgreSQL and MongoDB,” in Marketing and Smart Technologies. Smart Innovation, Systems and Technologies, vol. 279, Singapore: Springer, 2022, pp. 121-127, doi: 10.1007/978-981-16-9268-0_10.
20. *D. Yedilkhan, A. Mukasheva, D. Bissengaliyeva, et al.* “Performance Analysis of Scaling NoSQL vs

SQL: A Comparative Study of MongoDB, Cassandra, and PostgreSQL," 2023 IEEE International Conference on Smart Information Systems and Technologies (SIST), pp. 479-483, 2023, doi: 10.1109/sist58284.2023.10223568.
21. *T.K. Mohd, J. Thompson, A. Carmine, et al.* "Comparative Analysis on Various CSS and JavaScript Frameworks," Journal of Software, pp. 282–291, Nov. 2022, doi: 10.17706/jsw.17.6.282-291.
22. *F. Zammetti* Modern Full-Stack Development Using TypeScript, React, Node.js, Webpack, and Docker. Berkeley, Apress, 2022.
23. *C. Carneiro, T. Schmelmer* "Development Environment and Workflow," in Microservices From Day One, Berkeley: Apress, 2016, pp. 105-126, doi: 10.1007/978-1-4842-1937-9_8.
24. *S. Rakovsky, S. Magomedov* "Ensuring the Security of Open Python Projects: The Challenge of Assessing Potentially Destructive Functionality," International Journal of Open Information Technologies, vol. 11, no. 10, pp. 113-118, 2023 (In Russian).
25. *M. Beller, R. Bholanath, S. McIntosh, et al.* "Analyzing the State of Static Analysis: A Large-Scale Evaluation in Open Source Software," 2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER), pp. 470-481, 2016, doi: 10.1109/SANER.2016.105.
26. *S. Chandran, K. Abraham* "A Correlative Scrutiny on two Programming Dialects: RUBY Vs PYTHON," International Journal of Engineering and Advanced Technology (IJEAT), vol. 9, no. 3, pp. 4395–4404, Feb. 2020, doi: 10.35940/ijeat.c6435.029320.
27. *T. Beuzen, T. Timbers* Python Packages. Boca Raton, CRC Press, 2022.
28. *P. Virtanen, R. Gommers, T.E. Oliphant, et al.* "SciPy 1.0: fundamental algorithms for scientific computing in Python," Nature Methods, vol. 17, no. 3, pp. 261–272, Feb. 2020, doi: 10.1038/s41592-019-0686-2.
29. *Y. Mohialden, N. Mahmood, E. Baker, et al.* "A Comparative Analysis of Python Code-Line Bug-Finding Methods," Scientific Research Journal of Engineering and Computer Science, vol. 3, no. 4, pp. 35-41, 2023.
30. *M. Grinberg* Flask Web Development: Developing Web Applications with Python. Cambridge, O'Reilly, 2018.

Топузов Эдем Русланович, Санкт-Петербургский государственный университет, ✉ **edem.topuzov@gmail.com**

Ампилова Наталья Борисовна, Санкт-Петербургский государственный университет, **n.ampilova@spbu.ru**

Computer tools in education, 2024
№ -: 1–15
http://cte.eltech.ru

Web-based system for processing medical research data

Topuzov E. R.[1], PhD student, ✉ edem.topuzov@gmail.com, orcid.org/0009-0003-9208-7931
Ampilova N. B.[1], Associate Professor, n.ampilova@spbu.ru, orcid.org/0000-0002-2154-9399

[1]SPbU, Saint Petersburg State University

Abstract

The issues associated with processing of large amount of medical data are discussed in the article. The limitations of using existing medical information systems in individual research are noted. To simplify medical research processes, web-based system has been developed. The system introduces such features as research design, collaboration, data input, access control, audit of database changes, regular backup, data analysis, and exporting data as sheets and plots. The designed system was deployed to production and received positive feedback from users. The certificate of state registration of a computer program was obtained.

Keywords: *medical information systems, processing medical research data, web-based system.*

Citation: Topuzov E. R., Ampilova N. B.. Web-based system for processing medical research data. Computer tools in education, 2024. № -. P. 1–15. .

References

1. *Evidence-Based Medicine Working Group* “Evidence-based medicine. A new approach to teaching the practice of medicine,” JAMA, vol. 268, no. 17, pp. 2420-2425, 1992, doi: 10.1001/jama.1992.03490170092032.
2. *A.V. Gusev* “Prospects for the use of big data in Russian healthcare,” Moscow Medicine, vol. 1, no. 47, pp. 26-30, 2022 (In Russian).
3. *K. Moutselos, D. Kyriazis, V. Diamantopoulou, et al.* “Trustworthy data processing for health analytics tasks,” IEEE International Conference on Big Data (Big Data), pp. 3774-3779, 2018, doi: 10.1109/BigData.2018.8622449.
4. *C. Auffray, R. Balling, I. Barroso, et al.* “Making sense of big data in health research: towards an EU action plan,” Genome medicine, vol. 8, no. 1, pp. 1-13, Jun. 2016, doi: 10.1186/s13073-016-0323-y.
5. *A.S. Shakhanov, E.V. Ushakova* “Use of modern information technologies in public administration,” Transformation of business and public institutions in the context of digitalization of the economy, pp. 199-211, 2020 (In Russian).
6. *E.V. Lutsenko* “Development of medical information technologies in the Russian Federation,” Medical newsletter of Vyatka, vol. 2, no. 54, pp. 73-77, 2017 (In Russian).
7. *T. Magrupov, S. Yusupov, Y. Talatov, et al.* “Intelligent Medical System of Designing Medical Technics and Technology,” 2020 International Conference on Information Science and Communications Technologies (ICISCT), pp. 1-4, 2020, doi: 10.1109/ICISCT50599.2020.9351375.
8. *P.S. Pugachev, A.V. Gusev, O.S. Kobyakova, et al.* “Global trends in the digital transformation of the healthcare industry,” National Health Care (Russia), vol. 2, no. 2, pp. 5-12, Nov. 2021, doi: 10.47093/2713-069x.2021.2.2.5-12 (In Russian).

9. *D. Singh, S. Verma, J. Singla.* “A comprehensive review of intelligent medical diagnostic systems,” 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184), pp. 977-981, 2020, doi: 10.1109/ICOEI48184.2020.9143043.
10. *Y. Duan, J. Edwards, M. Xu* “Web-based expert systems: Benefits and challenges,” Information & Management, vol. 42, no. 6, pp. 799–811, Sep. 2005, doi: 10.1016/j.im.2004.08.005.
11. *A.A. Baidybekov, R.G. Gilvanov, I.A. Molodkin* “Modern Frameworks for Web Development,” Intellectual Technologies on Transport, no. 4, pp. 23-29, 2020 (In Russian).
12. *D. Klochkov, J. Mulawka* “Improving Ruby on Rails-Based Web Application Performance,” Information, vol. 12, no. 8, p. 319, Aug. 2021, doi: 10.3390/info12080319.
13. *P. Łuczak, A. Poniszewska-Maranda, V. Karovič* “The Process of Creating Web Applications in Ruby on Rails,” in Developments in Information & Knowledge Management for Business Applications. Studies in Systems, Decision and Control, vol. 330. Cham: Springer, 2021, pp. 371-401, doi: 10.1007/978-3-030-62151-3_9.
14. *J. Yang, P. Subramaniam, S. Lu, et al.* “How not to structure your database-backed web applications,” 2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE), pp. 800-810, 2018, doi: 10.1145/3180155.3180194.
15. *C. Chunling* “Construction of the Individualized College English Learning Management System Using Ruby on Rails,” 2015 International Conference on Service Science (ICSS), pp. 160-163, 2015, doi: 10.1109/ICSS.2015.22.
16. *A. Tapiador, J. Salvachúa* “Content Management in Ruby on Rails,” Proceedings of the IADIS International Conference on Collaborative Technologies 2011, 2011, doi: 10.48550/arxiv.1209.3878.
17. *A. Aborujilah, J. Adamu, A.M. Shariff, et al.* “Descriptive Analysis of Built-in Security Features in Web Development Frameworks,” 2022 16th International Conference on Ubiquitous Information Management and Communication (IMCOM), pp. 1-8, 2022, doi: 10.1109/IMCOM53663.2022.9721750.
18. *G. Donald* Hands-on Test-Driven Development Using Ruby, Ruby on Rails, and RSpec. Berkeley, Apress, 2024.
19. *M. Costa, M. Rodrigues, P. Baptista, et al.* “Database Encryption Performance Impact on PostgreSQL and MongoDB,” in Marketing and Smart Technologies. Smart Innovation, Systems and Technologies, vol. 279, Singapore: Springer, 2022, pp. 121-127, doi: 10.1007/978-981-16-9268-0_10.
20. *D. Yedilkhan, A. Mukasheva, D. Bissengaliyeva, et al.* “Performance Analysis of Scaling NoSQL vs SQL: A Comparative Study of MongoDB, Cassandra, and PostgreSQL,” 2023 IEEE International Conference on Smart Information Systems and Technologies (SIST), pp. 479-483, 2023, doi: 10.1109/sist58284.2023.10223568.
21. *T.K. Mohd, J. Thompson, A. Carmine, et al.* “Comparative Analysis on Various CSS and JavaScript Frameworks,” Journal of Software, pp. 282–291, Nov. 2022, doi: 10.17706/jsw.17.6.282-291.
22. *F. Zammetti* Modern Full-Stack Development Using TypeScript, React, Node.js, Webpack, and Docker. Berkeley, Apress, 2022.
23. *C. Carneiro, T. Schmelmer* “Development Environment and Workflow,” in Microservices From Day One, Berkeley: Apress, 2016, pp. 105-126, doi: 10.1007/978-1-4842-1937-9_8.
24. *S. Rakovsky, S. Magomedov* “Ensuring the Security of Open Python Projects: The Challenge of Assessing Potentially Destructive Functionality,” International Journal of Open Information Technologies, vol. 11, no. 10, pp. 113-118, 2023 (In Russian).
25. *M. Beller, R. Bholanath, S. McIntosh, et al.* “Analyzing the State of Static Analysis: A Large-Scale Evaluation in Open Source Software,” 2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER), pp. 470-481, 2016, doi: 10.1109/SANER.2016.105.
26. *S. Chandran, K. Abraham* “A Correlative Scrutiny on two Programming Dialects: RUBY Vs PYTHON,” International Journal of Engineering and Advanced Technology (IJEAT), vol. 9, no. 3, pp. 4395–4404, Feb. 2020, doi: 10.35940/ijeat.c6435.029320.
27. *T. Beuzen, T. Timbers* Python Packages. Boca Raton, CRC Press, 2022.
28. *P. Virtanen, R. Gommers, T.E. Oliphant, et al.* “SciPy 1.0: fundamental algorithms for scientific computing in Python,” Nature Methods, vol. 17, no. 3, pp. 261–272, Feb. 2020, doi: 10.1038/s41592-019-0686-2.
29. *Y. Mohialden, N. Mahmood, E. Baker, et al.* “A Comparative Analysis of Python Code-Line Bug-Finding Methods,” Scientific Research Journal of Engineering and Computer Science, vol. 3, no. 4, pp. 35-41,

2023.
30. *M. Grinberg* Flask Web Development: Developing Web Applications with Python. Cambridge, O'Reilly, 2018.

Topuzov Edem Ruslanovich, Saint Petersburg State University, ✉ **edem.topuzov@gmail.com**

Ampilova Natalia Borisovna, Saint Petersburg State University, n.ampilova@spbu.ru