

Система автоматизированной обработки информационных потоков медицинского учреждения

Е.И. Сукач, К.А. Поленок, В.А. ШКАРУБО

Представлен опыт разработки медицинской информационной системы, направленной на автоматизацию документооборота в медицинских учреждениях. Описывается архитектура веб-приложения, реализованного с использованием Java, Spring Boot и базы данных MySQL, а также интеграция с Telegram-ботом «Aesculapius Lab Reports» для оперативного уведомления пациентов о результатах лабораторных исследований. Предлагается программное решение для голосового ввода типовых медицинских данных.

Ключевые слова: веб-приложение, информационная система, цифровой медицинский документооборот, Spring Framework, JWT-токен, клиент-серверная архитектура, распознавание речи.

The experience of developing a medical information system aimed at automating document flow in medical institutions is presented. It describes the architecture of a web application implemented using Java, Spring Boot and the MySQL database, as well as integration with the Telegram bot «Aesculapius Lab Reports» to promptly notify patients about laboratory results. A software solution for voice input of typical medical data is offered.

Keywords: web application, information system, digital medical document flow, Spring Framework, JWT-token, client-server architecture, speech recognition.

Введение. Решающую роль в принятии решений, планировании, прогнозировании и проведении исследований играет точность и доступность информации. Автоматизированная обработка информационных потоков на уровне отдельных медицинских учреждений в сочетании с интеграцией полученных данных на региональном и республиканском уровнях позволяют избежать ошибок при формировании медицинской статистики и её анализе с использованием современных методов и информационных технологий [1].

С другой стороны, одним из ключевых элементов обеспечения качества и надежности оперативности предоставления услуг медицинскими учреждениями является эффективная обработка информационных потоков, которые включают данные первичного осмотра пациентов, данные лабораторных исследований, информацию для формирования электронных медицинских карт, листов нетрудоспособности и других медицинских документов [2]. На текущем этапе значительная часть этих процедур осуществляется вручную, что создаёт риски потери и искажения информации, увеличивает нагрузку на медицинский персонал и затрудняет оперативный обмен данными. В связи с этим разработка гибкой, защищённой и масштабируемой системы управления медицинскими информационными потоками позволит оптимизировать процессы обработки данных, улучшить доступность информации для медицинского персонала и пациентов, а также минимизировать ошибки при работе с документами [3].

В статье рассматривается программное обеспечение, автоматизирующее функции обработки информации в медицинских учреждениях и поддерживающее ведение электронных медицинских карт, формирование консультативных заключений, листов нетрудоспособности, рецептов врачей и других документов. Программные средства позволяют интегрировать фармакологический сектор в процесс лечения пациентов и использовать Telegram-бот «Aesculapius Lab Reports», предоставляющий пациентам возможность подписки на рассылку результатов лабораторных исследований.

Архитектура веб-приложения для автоматизации обработки медицинских информационных потоков. В результате анализа требований, предъявляемых к информационной системе медицинского учреждения, были выделены взаимосвязанные функциональные блоки, определившие основу веб-приложения (рисунок 1).

Блок веб-клиента содержит программную логику по взаимодействию с блоком веб-сервера, которая выполняется браузером конечного пользователя.

Блок аутентификации и авторизации предназначен для защиты данных, которыми оперирует приложение при установлении личности и прав пользователя, пытающегося получить доступ к информации. В случае установления личности запрашивающего и наличия у него необходимых прав доступа, его требование удовлетворяется, в противном случае – отклоняется.

Блок валидации данных, введённых пользователем, служит для проверки вводимых потенциальными пользователями данных на соответствие принятым ограничениям.

Блок веб-сервера отвечает за приём, декапсуляцию HTTP-запросов и их маршрутизацию к соответствующим обработчикам с последующей инкапсуляцией ответа обработчика.

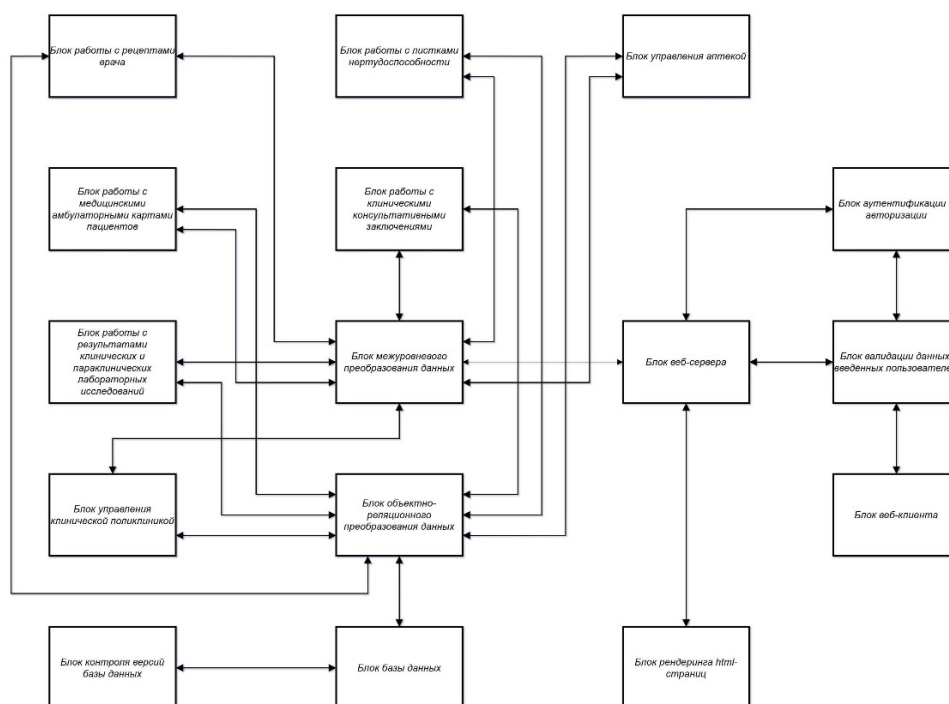


Рисунок 1 – Схема веб-приложения для автоматизации обработки медицинских информационных потоков

Блок рендеринга html-страниц позволяет динамически наполнять html-страницы содержимым согласно существующим шаблонам.

Блок управления поликлиникой предоставляет необходимый набор функций для управления работой младшего и старшего медперсонала в программном решении.

Блок управления аптекой предоставляет необходимый набор функций для управления работой фармацевтов.

Блок работы с консультативными заключениями предоставляет возможность врачам заполнять и сохранять заключения, просматривать заключения, выданные пациенту ранее.

Блок работы с листками нетрудоспособности даёт возможность врачам выдавать листки нетрудоспособности пациентам, а также просматривать их в свободном доступе по идентификационному номеру лица, которому листок был выдан.

Блок работы с рецептами предоставляет врачам набор функций для оформления рецептов, а фармацевтам – реализации лекарственных препаратов по ним.

Блок работы с результатами лабораторных исследований позволяет врачам-лаборантам составлять заключения об анализе биологических жидкостей и продуктов жизнедеятельности пациента, а также прикреплять их к его амбулаторной карте.

Блок работы с медицинскими и амбулаторными картами пациентов отвечает за работу с соответствующими документами, предоставляя возможность создавать карты, редактировать их и прикреплять к ним отчёты различных специалистов. В данном блоке реализована возможность речевого ввода однотипных данных первичного осмотра пациентов с их последующей обработкой и хранением в базе данных [4].

Блок межуровневого преобразования данных необходим для обеспечения транзита данных в виде объектов между функциональными уровнями приложения.

Блок объектно-реляционного преобразования данных отвечает за перенос реляционной модели базы данных на объектную модель, которой оперирует приложение.

Блок контроля версий базы данных отвечает за поддержание схемы базы данных в актуальном состоянии.

Блок базы данных отвечает за установление соединения между программным решением и базой данных.

Разработанное веб-приложение представляет собой централизованную SaaS-систему, доступную через браузер, с чёткой модульной архитектурой. В основе – фреймворк Spring Boot, обеспечивающий стабильную работу, высокую расширяемость и интеграцию с различными компонентами. СУБД MySQL обеспечивает хранение всей медицинской информации, включая персональные данные пациентов, истории болезней, рецепты и заключения. Для обеспечения безопасности используется модуль Spring Security и механизм аутентификации с помощью JWT (JSON Web Token). Это исключает необходимость хранения сессий на сервере и позволяет каждому пользователю получить уникальный зашифрованный токен после входа в систему. Разграничение ролей (администратор, врач, лаборант, фармацевт и регистратор) позволяет каждому сотруднику работать только с теми данными, к которым у него есть доступ.

Клиентская часть приложения реализована с использованием HTML и JavaScript. Каждый пользователь после авторизации попадает на интерфейс, адаптированный под его роль. Например, врачу доступна вкладка оформления рецептов и заключений, в то время как лаборант может формировать результаты лабораторных исследований.

В приложении была реализована функциональность, связанная с REST, применён подход серверной генерации страниц, заложена гибкая и расширяемая архитектура для последующего увеличения вычислительных мощностей, добавления нового функционала.

Функции автоматизации формирования медицинских документов. К основным функциям, автоматизирующим заполнение медицинских документов, можно отнести формирование консультативных заключений, листов нетрудоспособности и рецептов врачей. Все эти функции доступны для зарегистрированных пользователей с назначенной ролью «врач».

Для оформления консультативного заключения необходимо убедиться, что пациент, которому заключение выдаётся, выбран, после чего перейти на вкладку «Оформление консультативного заключения» и заполнить все поля, которые помечены символом «*», остальные поля заполняются на усмотрение пользователя. Для просмотра консультативных заключений необходимо выбрать вкладку «Консультативные заключения», на которой представлен весь список заключений, которые когда-либо были выданы пациенту. Пример сформированного заключения представлен на рисунке 2. Аналогичные действия выполняются и при формировании листа нетрудоспособности, и при оформлении рецепта врача (рисунок 3).



Рисунок 2 – Пример консультативного заключения

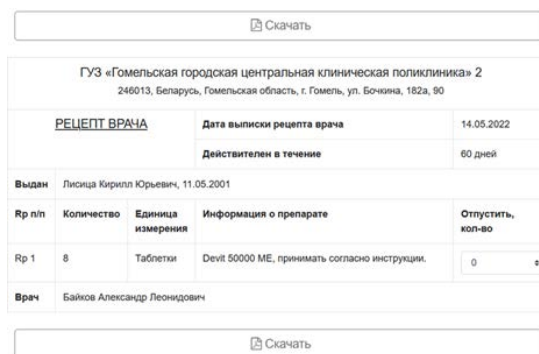


Рисунок 3 – Оформленный рецепт врача

Осуществлять выдачу препаратов, согласно рецептам врача, может только пользователь с ролью «фармацевт». Для выдачи препаратов необходимо убедиться, что пациент, которому препарат отпускается, определен, после чего выбрать один из рецептов из выпадаю-

щего списка. Выбрав необходимый рецепт врача, необходимо перейти в графу «Отпустить, количество» для отпускаемого препарата и выбрать из выпадающего списка отпускаемое количество в указанной единице измерения.

Речевой ввод информации о пациентах. В медицинских учреждениях оперативное внесение данных о пациентах критично для своевременного принятия решений. Поэтому в системе в рамках блока работы с амбулаторными картами пациентов возможен переход на речевой ввод однотипной информации.

Потенциальный пользователь может вводить данные через голосовые команды, используя встроенную систему распознавания речи, реализованную на базе библиотеки Speech Recognition. В программной реализации используется встроенный микрофон и система распознавания речи для ввода информации о пациенте. Голосовые команды преобразуются в текст с использованием библиотеки, а затем автоматически заполняют поля формы. Полученный текст в реальном времени преобразуется в значения для таких полей, как фамилия, имя, пол, возраст, рост, вес и т. д. Предложенное решение позволяет значительно упростить и ускорить процесс ввода информации, обеспечить её надёжность и минимизировать ошибки, связанные с ручным вводом.

Реализована возможность добавления новых записей, редактирования существующих, удаления и чтения информации. Блок синхронизируется с базой данных при каждом запуске веб-приложения и после выполнения операций над записями. В случае отсутствия подключения к интернету модуль работает в офлайн-режиме, сохраняя данные во временный локальный CSV-файл. При восстановлении соединения данные могут быть синхронизированы с базой данных. Поддерживается ручной импорт и экспорт данных в CSV-формате.

Реализована система валидации данных, предупреждающая о некорректных значениях. Все действия сопровождаются сообщениями о результатах, а ошибки подключения к базе, проблемы с распознаванием речи и другие сбои перехватываются и корректно обрабатываются. Механизмы обработки ошибок и уведомлений помогают пользователю своевременно реагировать на проблемы с сетью или доступом к данным.

Интеграция с Telegram-ботом. В рамках расширения функциональности системы был разработан Telegram-бот «Aesculapius Lab Reports», задачей которого является автоматическая отправка результатов лабораторных исследований пациентам. Бот реализован на языке Java с использованием Spring Boot. Для обмена сообщениями между ботом и основной системой использовался Apache ActiveMQ [5]. После того как лаборант вносит данные об исследовании в базу данных, бот получает сообщение, извлекает результат в виде pdf-файла и отправляет его пользователю Telegram, который ранее подписался на рассылку.

Команды /subscribe и /unsubscribe позволяют пользователю самостоятельно контролировать подписку на уведомления. Также доступна команда /help, предоставляющая справочную информацию.

Заключение. Система прошла серию модульных и интеграционных тестов. Для модульного тестирования использовались Groovy и фреймворк Spock, позволяющие детально проверить логику каждого компонента. Проведено внешнее тестирование: моделировались типичные действия пользователей различных ролей, включая регистрацию пациентов, создание рецептов и лабораторных отчётов, проверку работы Telegram-бота. Результаты показали стабильную работу системы, корректную обработку ошибок и соответствие установленным требованиям.

Разработанное программное обеспечение представляет собой эффективный инструмент автоматизации документооборота в медицинских учреждениях. Благодаря модульной структуре, высокой степени безопасности и удобству в эксплуатации система способна значительно повысить скорость и качество работы медучреждения. Речевой ввод данных при регистрации пациентов может быть востребован в условиях ограниченного времени или недоступности ручного ввода. Интеграция с Telegram-ботом позволит расширить каналы взаимодействия с пациентами, повысить уровень информирования и доверия.

Внедрение подобной системы будет способствовать обоснованному планированию и предвидению возможных направлений будущего развития организации. Предложенное решение может быть масштабировано на региональном уровне и стать частью единой цифровой платформы здравоохранения Республики Беларусь.

Литература

1. Сукач, Е. И. Вероятностно-алгебраический метод анализа медицинской статистики / Е. И. Сукач, А. П. Кончиц // Проблемы физики, математики и техники. – 2024. – № 4 (61). – С. 82–88.
2. Телпцерищев, Р. А. Медико-социальные основы здоровья : учеб. пособ. / Р. А. Телпцерищев – Ростов н/Д. : Феникс 2000. – 623 с.
3. Поленок, К. А. Оптимизация процесса управления медицинской документацией в учреждениях здравоохранения / К. А. Поленок // Творчество Молодых – 2024 : сб. науч. работ студентов и аспирантов / М-во образования РБ, ГГУ им. Ф. Скорины. – Гомель : ГГУ им. Ф. Скорины, 2024. – С. 225–228.
4. Шкарубо, В. А. Использование речевого интерфейса для оперативного сохранения цифровой информации / В. А. Шкарубо // Творчество Молодых – 2024 : сб. науч. работ студентов и аспирантов / М-во образования РБ, ГГУ им. Ф. Скорины. – Гомель : ГГУ им. Ф. Скорины, 2024. – С. 247–250.
5. Облачные архитектуры : разработка устойчивых и экономичных облачных приложений : рук. разработчика / Т. Лащевски [и др.]. – СПб.: Питер, 2022. – 320 с.

Гомельский государственный
университет имени Франциска Скорины

Поступила в редакцию 17.06.2025