ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ИНТЕЛЛЕКТУАЛЬНОЙ СИСТЕМЫ СЕМАНТИЧЕСКИХ РЕКОМЕНДАЦИЙ КНИГ

Жданова С.И., ст. преподаватель,
Путилин Н.И., студент,
БГТУ им. В.Г. Шухова, г. Белгород, Россия

Аннотация. В статье рассматриваются практические аспекты программной реализации интеллектуальной информационной системы управления персональной цифровой библиотекой. Описана гибридная архитектура решения, объединяющая классическую реляционную СУБД PostgreSQL, векторное расширение pgvector и локально развернутые нейросетевые модели на базе платформы Ollama. Представлены алгоритмы многофакторной семантической фильтрации текстов и интерактивной визуализации читательских предпочтений в виде направленного графа. Предложенный технологический стек обеспечивает высокую производительность генерации персонализированных рекомендаций при строгом соблюдении конфиденциальности пользовательских данных.

Ключевые слова: интеллектуальная система, семантический поиск, векторные эмбеддинги, локальные LLM, PostgreSQL, pgvector, визуализация графов, рекомендательные алгоритмы.

Как справедливо отмечает Д.А. Марка в своих трудах по методологии структурного анализа [2], переход от концептуального проектирования к физической реализации системы всегда сопряжен с риском архитектурных просчетов. Поэтому при разработке информационной системы фундаментальной задачей стало обеспечение бесшовной интеграции классических операций управления данными с ресурсоемкими алгоритмами машинного обучения при условии жесткого соблюдения конфиденциальности пользовательской информации. В качестве технологического базиса серверной

части был выбран язык программирования Java в экосистеме фреймворка Spring Boot. Данное решение продиктовано необходимостью надежной оркестрации многопоточных вычислений. Ядро безопасности построено на базе Spring Security [5] с реализацией механизмов аутентификации без сохранения состояния (stateless) через токены формата JWT. Криптографическая защита паролей обеспечивается алгоритмом BCrypt, а управление сессиями делегировано специализированному программному фильтру, который валидирует криптографические подписи на уровне каждого входящего запроса, что полностью исключает необходимость хранения состояния сессий на сервере и многократно повышает масштабируемость архитектуры.

Организация слоя данных потребовала отказа от традиционных парадигм поиска. Поскольку классические реляционные структуры, использующие операторы строкового сравнения, технически неприменимы для алгоритмов семантического поиска, в качестве системы управления базами данных была развернута PostgreSQL с интеграцией расширения pgvector. Данный программный модуль обеспечил нативную работу с многомерными математическими векторами (эмбеддингами) непосредственно на уровне стандартных SQL-запросов. Подобное архитектурное решение позволило полностью исключить из проекта сторонние ресурсоемкие векторные базы данных, тем самым существенно снизив накладные сетевые расходы и объединив текстовые метаданные с их математическими проекциями в едином защищенном хранилище.

Ключевым требованием к разрабатываемой системе являлась абсолютная изоляция пользовательских данных от внешних облачных провайдеров. Вся инфраструктура искусственного интеллекта была развернута внутри закрытого серверного контура с использованием платформы Ollama и интеграционного модуля Spring AI. Логика распределения вычислительных мощностей разделена на два независимых потока. Фоновая векторизация осуществляется легковесной моделью Gemma-Embedding, которая активируется в момент добавления новой книги в библиотеку и преобразует сжатый текст в векторную форму. В свою

очередь, ресурсоемкая модель LLaMA 3.1 подключается динамически и исключительно в режиме интерактивного чтения, отвечая за синтез связных ответов, перевод текстов и суммаризацию выделенных фрагментов. Дополнительно, для предотвращения уязвимостей типа XSS при программном парсинге исходных файлов электронных книг, был реализован санитайзер на базе библиотеки Jsoup, осуществляющий жесткую фильтрацию сгенерированного HTML-кода по строгому списку разрешенных тегов.

В основе формирования умных рекомендаций лежит трансформация семантики текста в многомерное математическое пространство. Как отмечает Е.Б. Солонин, интеллектуальные технологии анализа данных работают как «автоматизированный процесс исследования больших массивов данных, производимый с целью обнаружения скрытых закономерностей (знаний)» [3]. Опытным путем в ходе разработки было установлено, что векторизация полного текста литературного произведения генерирует избыточный семантический шум за счет обилия диалогов и служебных конструкций, что приводит к резкому снижению точности работы алгоритмов машинного обучения. В связи с этим был реализован программный механизм, который изолирует и конкатенирует исключительно семантическое ядро произведения: название, массив жанровых тегов и предварительно очищенную от разметки аннотацию. Сформированный таким образом сжатый контекст передается локальной эмбеддинг-модели, возвращающей вектор высокой размерности. Критически важно, что ресурсоемкие математические вычисления, в частности расчет косинусного сходства, не выгружаются в оперативную память сервера, а делегируются непосредственно ядру базы данных PostgreSQL. Подобная гибридная архитектура позволяет системе сохранять высокую скорость отклика: объемные текстовые массивы хранятся в классических реляционных таблицах, тогда как их легковесные математические проекции индексируются в специализированном векторном пространстве.

Опираясь на реализованный механизм поиска ближайших соседей, в серверной логике были разработаны три независимых алгоритмических

сценария персонализации выдачи. Первый сценарий направлен на решение проблемы так называемого информационного пузыря и базируется на подходе мульти-якорного разнообразия. Алгоритм программно извлекает из пользовательской истории несколько высокооцененных изданий, выступающих в роли независимых математических ориентиров. Система выполняет параллельные векторные запросы для каждого из якорей, после чего синтезирует агрегированную подборку с жестким отсевом дубликатов, обеспечивая высокую вариативность предложений. Для решения задачи по расширению читательского кругозора был внедрен алгоритм поиска неочевидных связей. На практике он работает за счет инверсии стандартного поискового запроса. Сначала база данных выдает максимально широкую выборку книг, полностью игнорируя пороги отсечения. Только после получения этого избыточного массива данных включается программная фильтрация. Главная цель такого подхода, убрать две крайности. Алгоритм жестко отсекает слишком очевидные совпадения (с высоким косинусным коэффициентом) и одновременно удаляет полный смысловой «мусор». В результате читателю предлагается литература, которая пересекается по скрытым философским мотивам или атмосфере, даже если жанры произведений совершенно разные. Если же пользователю нужен точный контроль, предусмотрен гибридный поиск. В этом случае нейросеть обрабатывает свободный текстовый запрос, но выдача сразу ограничивается классическими фильтрами базы данных (например, по году издания). Это помогает компенсировать излишнюю абстрактность искусственного интеллекта.

Разработка клиентской части велась в формате Single Page Application. В качестве основы была выбрана библиотека React.js [4] вместе со сборщиком Vite. С архитектурной точки зрения этот подход означает, что «веб-приложение загружает единственный HTML-документ и динамически обновляет его содержимое» по мере взаимодействия с пользователем. Архитектура фронтенда построена на строгом разделении логики. Визуальные компоненты полностью

изолированы от сервисного слоя, который забирает на себя всю работу с токенами авторизации и маршрутизацией запросов к REST API. Чтобы сделать управление библиотекой более дружелюбным, было решено отказаться от скучных табличных списков, заменив их метафорой физического книжного шкафа.

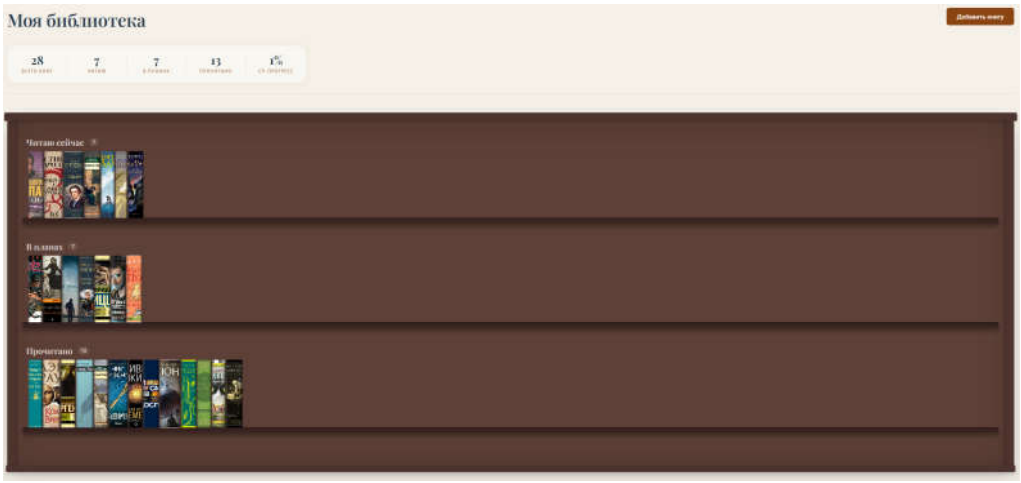

Рис. 1. Пользовательский интерфейс книжной полки

Сам процесс распределения книг по статусам прочтения опирается на механику Drag-and-Drop. Важный инженерный нюанс здесь заключается в использовании паттерна оптимистичного обновления (Optimistic UI). Когда пользователь перетаскивает книгу, клиентское приложение сразу же перерисовывает экран, не дожидаясь ответа от сервера. Параллельно с этим в фоновом режиме идет асинхронная отправка данных для синхронизации с базой PostgreSQL. Благодаря такому подходу возникает эффект абсолютно бесшовного взаимодействия, и интерфейс не зависает даже при работе с массивными каталогами. В свою очередь, встроенный модуль интерактивного чтения технически опирается на динамический расчет свойств колоночной верстки, адаптирующей текстовый массив под габариты видоискателя устройства с синхронным пересчетом процента освоения материала при регистрации событий прокрутки. Интеграция генеративного искусственного интеллекта в процесс чтения базируется на программном перехвате браузерного интерфейса выделения текста, что позволяет позиционировать слой контекстного меню непосредственно над областью фокуса пользователя.

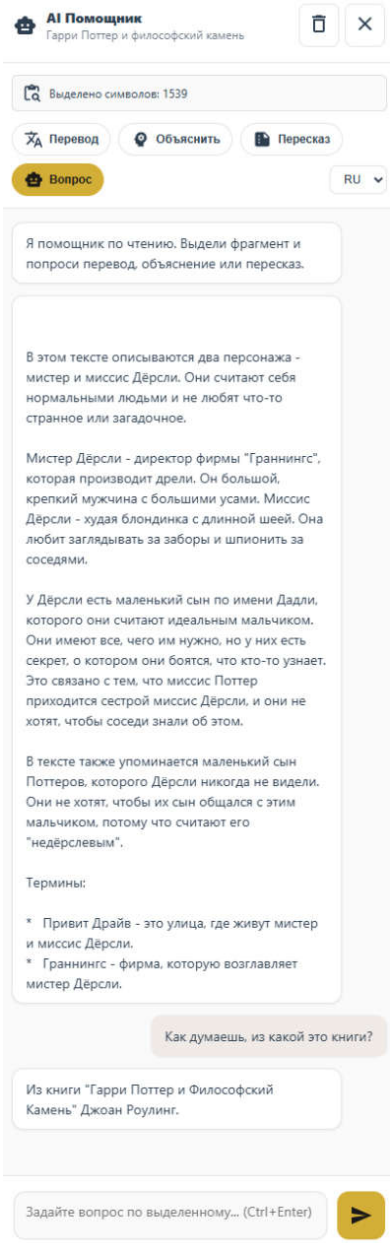

Рис. 2. Взаимодействие с AI-ассистентом в интерфейсе ридера

Критически важным инженерным решением на этапе реализации стала оптимизация канала передачи данных от локальной языковой модели. Для предотвращения блокировки интерфейса в периоды длительной алгоритмической генерации ответов была применена технология трансляции серверных событий (Server-Sent Events) на базе реактивного стека Spring WebFlux. Сервер не аккумулирует итоговый ответ, а передает сгенерированные токены в виде непрерывного асинхронного потока, что позволяет клиентской части осуществлять посимвольный рендеринг текста в режиме реального времени. Дополнительно, для глубокого аналитического исследования скрытых

паттернов читательских предпочтений, был разработан специализированный графовый модуль визуализации семантических связей, функционирующий на базе физического движка d3-force.

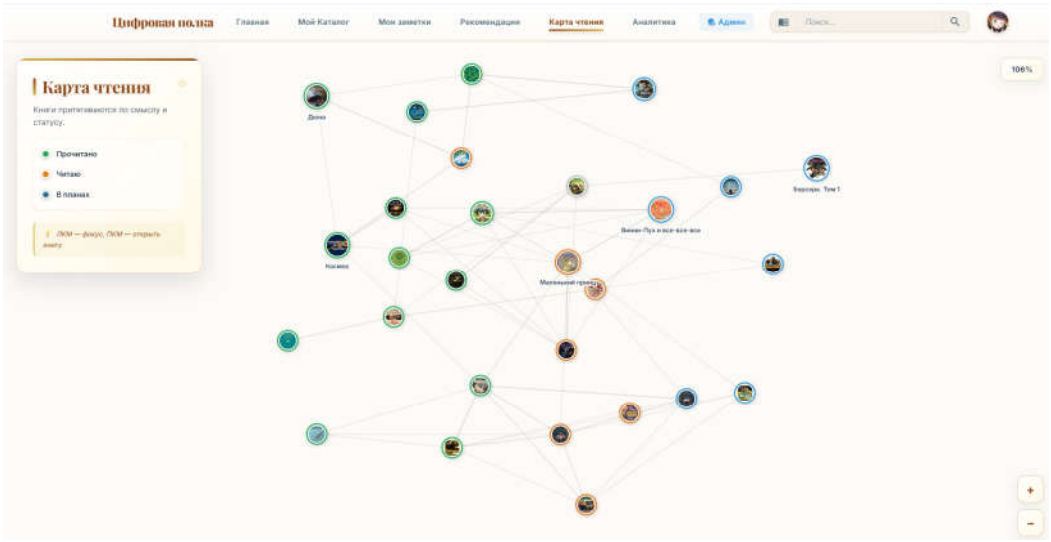

Рис. 3. Визуализация семантических связей в графовом модуле

Программный алгоритм визуализации производит квадратичный расчет косинусного сходства между многомерными векторами всех прочитанных пользователем книг. При превышении заданного порога семантической близости между информационными узлами формируются математические связи, подверженные воздействию симулированных физических сил притяжения и отталкивания. В результате сложных математических преобразований неструктурированный массив данных автономно самоорганизуется в наглядные смысловые кластеры. Интеграция графового представления с панелями агрегированных метрик читательской активности завершает формирование целостной информационной среды. Разработанная архитектура программного комплекса доказывает высокую техническую эффективность синергии классических реляционных структур данных, аппарата многомерной векторной алгебры и локально развернутых нейросетевых моделей в задачах проектирования безопасных систем управления персональными знаниями [1].

Литература

1. Жданова С.И. Цифровая трансформация в формировании образовательных траекторий личности // Наукоемкие технологии и инновации (XXV научные чтения): сб. докл. Междунар. науч.-практ. конф. Белгород: БГТУ им. В.Г. Шухова, 2023. С. 713-716.

2. Марка Д.А. Методология структурного анализа и проектирования: SADT (Structured Analysis & Design Technique) / Д.А. Марка, К.Л. МакГоуэн. М.: Мета-технология, 1993. 240 с.

3. Солонин Е.Б. Интеллектуальные технологии поиска и анализа данных [Электронный ресурс] / Урал. федер. ун-т. Екатеринбург: УрФУ, 2015. URL: https://study.urfu.ru/Aid/Publication/13334/1/Solonin.pdf (дата обращения: 03.05.2026).

4. Чиннатамби К. Изучаем React. СПб.: Питер, 2019. 368 с.

5. Spring Security Reference [Электронный ресурс]. URL: https://docs.spring.io/spring-security/reference/index.html (дата обращения: 27.04.2026).