МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»
Кафедра математической кибернетики и компьютерных наук

РАЗРАБОТКА ВЕБ-СЕРВИСА ПО ПОДБОРУ ИГРОКОВ НА СПОРТИВНЫЕ ПЛОЩАДКИ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 411 группы
направления 02.03.02 — Фундаментальная информатика и информационные технологии
факультета КНиИТ
Ведутина Ильи Сергеевича

Научный руководитель
к. ф.-м. н., доцент _____ М. И. Сафрончик

Заведующий кафедрой
доцент, к. ф.-м. н. _____ С. В. Миронов

Саратов 2026

ВВЕДЕНИЕ

Актуальность темы. В настоящее время наблюдается устойчивый рост интереса населения к любительским занятиям спортом, при этом организация совместных тренировок, матчей и турниров всё чаще реализуется через веб-сервисы. Поиск партнёров, бронирование спортивных площадок и формирование команд исторически опираются на ручную координацию через мессенджеры и социальные сети, что повышает временные затраты и снижает регулярность занятий. Рынок специализированного программного обеспечения для любительских микрокоммьюнити на территории Российской Федерации остаётся слабо насыщенным, тогда как применяемый в работе современный технологический стек (Java 21, Spring Boot 3.5, PostgreSQL, gRPC, Spring Security) соответствует актуальным требованиям ИТ-индустрии. Дополнительную значимость работе придаёт исследование инженерных приёмов интеграции отечественной большой языковой модели GigaChat в бизнес-приложение, что особенно важно в условиях курса на технологический суверенитет.

Объектом исследования являются процессы организации совместных любительских спортивных мероприятий в условиях городской среды. **Предметом исследования** выступают методы и средства программной реализации серверной части веб-сервиса для подбора игроков на спортивные площадки, в том числе подходы к проектированию реляционной модели данных, защите доступа и интеграции внешних сервисов искусственного интеллекта.

Цель работы заключается в проектировании и программной реализации серверной части веб-приложения SportBuddy, которое поддерживает поиск спортивных площадок, бронирование временных слотов, систему отзывов и рейтинговых матчей, а также взаимодействие пользователя с большой языковой моделью.

Задачи работы: провести аналитический обзор предметной области, существующих аналогов и технологий разработки веб-сервисов на платформе Spring Boot; сформулировать функциональные и нефункциональные требования к серверной части системы; спроектировать архитектуру приложения, реляционную модель данных и REST API; реализовать модуль авторизации и аутентификации с несколькими сценариями входа; разработать бизнес-логику бронирования с защитой от пересечения по времени, систему отзывов и рейтинговых матчей; интегрировать большую языковую модель GigaChat через прото-

кол gRPC с механизмом OAuth-авторизации и поддержкой российских корневых сертификатов; выполнить тестирование, локальное развёртывание и анализ полученных результатов.

Материалы исследования. Основой работы выступает собственная программная реализация серверной части сервиса SportBuddy на языке Java 21 с использованием Spring Boot 3.5, Hibernate ORM и СУБД PostgreSQL 15. При выполнении работы использовалась официальная документация Spring Framework, Spring Security, Hibernate ORM, PostgreSQL и API GigaChat, а также научная и техническая литература по разработке серверных приложений.

Структура работы. Выпускная квалификационная работа состоит из введения, двух глав, заключения, списка использованных источников и приложений. Первая глава «Обзор предметной области и технологий разработки» посвящена анализу предметной области, существующих аналогов и применяемых технологий. Вторая глава «Разработка серверной части веб-сервиса SportBuddy» содержит описание процесса проектирования и реализации сервиса. Работа дополнена приложениями с листингами ключевых классов и описанием электронного носителя.

ОСНОВНОЕ СОДЕРЖАНИЕ РАБОТЫ

Анализ предметной области. Под предметной областью разрабатываемого сервиса понимается процесс организации любительских спортивных мероприятий, в котором участвуют несколько типов пользователей. В системе выделяются четыре роли: игрок-любитель (`USER`), пользователь с подпиской (`PREMIUM`), владелец спортивной площадки (`COURT`) и администратор сервиса (`ADMIN`). Основной сценарий выглядит следующим образом: пользователь регистрируется, указывая фамилию, имя, контактный телефон и адрес электронной почты, после авторизации выбирает вид спорта и город, просматривает список одобренных модератором площадок и доступные часовые слоты на ближайшие трое суток. Записавшись на слот, пользователь автоматически добавляется в список участников, после чего любой из них может инициировать создание группового чата и поделиться ссылкой с остальными. После проведения игры пользователь может оставить отзыв о площадке и принять участие в рейтинговых матчах, влияющих на персональный игровой рейтинг. Параллельно владелец площадки подаёт заявки на добавление площадок и управляет их расписанием, а администратор одобряет или отклоняет заявки и формирует календарь рейтинговых матчей.

На основе анализа сценариев были сформулированы функциональные требования: поддержка операций создания, чтения, изменения и удаления для всех сущностей, защита от пересекающихся бронирований, генерация почасовых слотов на горизонт планирования трое суток, временная блокировка функции создания группового чата, фильтрация рейтинговых матчей по дате и расчёт игровой статистики для интеграции с языковой моделью. Нефункциональные требования включают доступность по защищённому каналу, поддержку не менее ста одновременных сессий, обеспечение целостности данных при конкурентных запросах, модульность и информационную безопасность (хранение паролей в виде хэшей, ограничение доступа к административным эндпоинтам, корректную работу с российской цепочкой корневых сертификатов).

Обзор существующих решений. Проведённый сравнительный анализ показал, что существующие решения лишь частично закрывают задачу. Социальные сети позволяют находить друг друга через тематические группы и объявления, однако не обеспечивают структурированного подхода, автоматической фильтрации по времени и месту и не дают возможности сформировать список

участников на конкретную дату. Онлайн-сервисы бронирования ориентированы исключительно на бронирование услуги и не объединяют пользователей в группы. Зарубежные специализированные платформы (Spond, TeamSnap, Playtomic) не поддерживают русский язык, не интегрируются с российскими сервисами либо не учитывают специфику командных уличных видов спорта и не предоставляют функционала рейтинговых матчей. Таким образом, ни одно из решений не сочетает бронирование, формирование команд, отзывы, рейтинговые матчи и интеграцию с языковой моделью, что подтверждает актуальность разработки.

Выбор технологий. В качестве основного языка программирования выбран Java 21, относящийся к LTS-релизам платформы Java SE и предоставляющий современные возможности — pattern matching, sealed-классы, record-классы, виртуальные потоки и выражения с переключением. В качестве платформы применяется Spring Boot 3.5, обеспечивающий построение REST API на основе Spring MVC, разграничение логики на контроллеры, сервисы и репозитории, упрощённую работу с базами данных через Spring Data JPA и автоматическую конфигурацию приложения. Для долговременного хранения данных используется СУБД PostgreSQL 15 с объектно-реляционным отображением через Hibernate ORM. Безопасность обеспечивается средствами Spring Security, а интеграция с большой языковой моделью GigaChat реализована по протоколу gRPC, который использует бинарный формат сериализации Protocol Buffers и обеспечивает более высокую производительность по сравнению с REST. Выбор Java обоснован тем, что этот язык остаётся одним из наиболее распространённых для разработки серверных приложений в российском сегменте ИТ-индустрии и обладает зрелой экосистемой инструментов корпоративной разработки. Сборка проекта выполняется системой Apache Maven с применением плагинов упаковки исполняемого файла, компиляции и генерации шаблонного кода. Процесс разработки строился на элементах методологии Agile с итеративным подходом, применением системы контроля версий Git и платформы GitHub в качестве удалённого репозитория, а также среды разработки IntelliJ IDEA с проверкой эндпоинтов через инструмент Postman.

Архитектура серверного приложения. Серверная часть реализована как монолитное приложение, построенное по принципам слоистой архитектуры. Выделяются четыре основных слоя: слой представления (контроллеры пакета `controller`), слой бизнес-логики (сервисы пакета `service`), слой доступа к дан-

ным (репозитории пакета `repository`) и слой долговременного хранения (реляционная база данных). Контроллеры принимают HTTP-запросы, выполняют первичную валидацию параметров, извлекают текущего пользователя и делегируют выполнение операций сервисам; сервисы реализуют бизнес-логику в транзакционных границах, задаваемых аннотацией `@Transactional`; репозитории на основе Spring Data JPA обеспечивают доступ к данным; СУБД PostgreSQL отвечает за их хранение. Дополнительно выделены вспомогательные пакеты сущностей, объектов передачи данных и конфигураций. Такое разделение ответственности обеспечивает высокую тестируемость и упрощает дальнейшее расширение системы вплоть до перехода на микросервисную архитектуру.

Проектирование модели данных. Модель данных насчитывает девять сущностей предметной области: пользователь (`User`), роль (`Role`), спортивная площадка (`SportCourt`), временной слот (`CourtTimeSlot`), бронирование (`Booking`), отзыв (`Review`), рейтинговый матч (`RankedMatch`), город (`City`) и вид спорта (`SportType`). Связи между сущностями реализованы стандартными аннотациями JPA «один ко многим», «многие к одному» и «многие ко многим». Каждый пользователь может быть автором множества бронирований и отзывов; площадка принадлежит одному владельцу и связана с городом и видом спорта, имеет множество временных слотов и отзывов; рейтинговый матч связан с участниками отношением «многие ко многим». Целостность данных обеспечивается ограничениями внешних ключей на уровне базы, а каскадное удаление связанных записей реализовано вручную в сервисном слое для тонкого контроля порядка операций. Первичные ключи генерируются стратегией `IDENTITY` средствами самой СУБД.

Объекты передачи данных. Для обмена информацией между слоями без раскрытия внутренней модели и оптимизации сериализации применяются классы-объекты передачи данных (DTO). Среди них унифицированный формат ответа REST-эндпоинтов с признаком успеха, текстом сообщения и полезной нагрузкой; объект регистрационного запроса с декларативной валидацией через аннотации Jakarta Bean Validation; компактные представления бронирования, участника слота и отзыва. Применение DTO позволяет сократить объём передаваемых данных и исключить утечку чувствительной информации, например хэшей паролей и внутренних идентификаторов.

Реализация слоя репозиториев и бизнес-логики. Слой доступа к дан-

ным состоит из восьми интерфейсов, наследующих `JpaRepository`; большинство методов задаются именованием в стиле Spring Data, а наиболее сложные сценарии реализованы пользовательскими JPQL-запросами. Spring Data автоматически генерирует SQL-запросы и выполняет их через подготовленные выражения, что дополнительно защищает от SQL-инъекций. Центральное место занимает логика бронирования: при записи на слот выполняется двойная проверка — на повторную запись пользователя на тот же слот и на пересечение с другими его бронированиями по времени. Проверка пересечения интервалов реализована эффективным JPQL-запросом на уровне СУБД, что существенно быстрее загрузки всех записей в память. Отдельного внимания заслуживает механизм совместного создания группового чата, основанный на временной блокировке на уровне базы данных сроком на пять минут: он исключает гонку при одновременных действиях нескольких участников, а принудительный сброс изменений гарантирует их запись до возврата ответа клиенту. Для поддержания актуального горизонта планирования временных слотов применяется фоновая задача планировщика, запускаемая ежедневно в полночь и удаляющая устаревшие слоты с одновременным созданием новых. Начальное наполнение справочных таблиц и демонстрационных площадок выполняется идемпотентным сервисом-инициализатором при первом запуске приложения.

Безопасность приложения и REST API. Конфигурация безопасности построена на Spring Security с разграничением доступа на уровне URL-шаблонов и отдельных методов через аннотацию `@PreAuthorize`. Реализованы три сценария входа: классический по паролю, программный через REST и вход по одноразовому коду, отправляемому на электронную почту. Все пароли хранятся в виде хэшей, вычисляемых адаптивным алгоритмом BCrypt с автоматической генерацией соли, что защищает от атак методом полного перебора и применения предвычисленных таблиц. Идентификация пользователя между запросами обеспечивается сессионным механизмом с использованием cookie-файла с флагом, запрещающим доступ к нему из клиентского сценария. Обмен данными с клиентом организован по архитектурному стилю REST поверх протокола HTTP с применением методов `GET`, `POST`, `PUT` и `DELETE` согласно их семантике (безопасность и идемпотентность) и осмысленным набором кодов состояния HTTP. Формат сериализации данных — JSON, передаваемый через DTO-классы, что исключает утечку чувствительной информации.

Интеграция большой языковой модели GigaChat. Интеграция реализована через специализированный сервис и включает два уровня: получение OAuth-токена по REST и отправку запросов к модели по gRPC. Решены задачи автоматического обновления токена при приближении срока его истечения и поддержки корневого сертификата Минцифры РФ, не входящего в стандартный truststore виртуальной машины Java: сертификат загружается из ресурсов и регистрируется в собственном хранилище ключей как для gRPC-канала, так и для HTTPS-клиента. Для каждого пользователя формируется персонализированный системный промпт на основе его игрового рейтинга и статистики сыгранных матчей, что повышает релевантность тренировочных рекомендаций. На случай недоступности внешнего сервиса предусмотрен резервный механизм ответов по ключевым словам, обеспечивающий базовую отказоустойчивость.

Тестирование и развёртывание. Корректность работы серверной части проверялась на ключевых сценариях бизнес-логики, прежде всего на сценариях бронирования (успешная запись, отклонение дубликата, отклонение при пересечении по времени) и разграничения доступа (обращение без авторизации, попытка доступа рядового пользователя к административным эндпоинтам). Локальное развёртывание выполняется командой сборки Maven с последующим запуском исполняемого файла приложения, при первом запуске справочные данные и демонстрационные площадки создаются автоматически. Экспериментальная эксплуатация подтвердила корректность работы реализованных компонентов, в том числе при конкурентных бронированиях. Среднее время отклика REST-эндпоинтов составило около 38 мс, тогда как обращение к языковой модели GigaChat по gRPC занимало в среднем порядка 1,8 секунды и определялось главным образом длиной генерируемого ответа. Полученные показатели свидетельствуют о том, что выбранная архитектура обеспечивает приемлемое качество пользовательского опыта при нагрузке, характерной для небольшого регионального сервиса.

Практическая значимость. Разработанное программное решение объединяет в одном сервисе функции, которые в существующих аналогах представлены лишь по отдельности: бронирование площадок, формирование команд, систему отзывов, рейтинговые матчи и интеграцию с большой языковой моделью. Найденные в ходе работы инженерные решения — механизм временной блокировки создания группового чата на уровне базы данных, защита от пере-

секающихся бронирований средствами СУБД, проксирование gRPC-вызовов к GigaChat с учётом особенностей российской цепочки сертификатов и формирование персонализированного системного промпта — обладают самостоятельной ценностью и могут быть переиспользованы в других проектах со схожей предметной областью.

ЗАКЛЮЧЕНИЕ

В рамках выпускной квалификационной работы была решена задача проектирования и программной реализации серверной части веб-сервиса SportBuddy. Проведён системный анализ предметной области, позволивший выделить четыре типа пользователей и сформулировать функциональные и нефункциональные требования к системе. Сравнительный анализ существующих решений показал, что ниша, объединяющая бронирование площадок, формирование команд, систему отзывов, рейтинговые матчи и интеграцию с большой языковой моделью, остаётся незанятой, что подтверждает актуальность работы.

Спроектирована и реализована модель данных из девяти сущностей, выстроена слоистая архитектура с разделением на контроллеры, сервисы и репозитории, реализована нетривиальная бизнес-логика бронирования с защитой от пересечения временных интервалов и механизм совместного создания группового чата на основе временной блокировки. Конфигурация безопасности обеспечивает строгое разграничение доступа и три сценария входа, а наиболее технически сложной частью стала интеграция отечественной языковой модели GigaChat по протоколу gRPC с поддержкой OAuth-авторизации и российских корневых сертификатов.

Полученное программное решение обладает практической значимостью и может быть использовано как в учебных целях, так и в качестве основы для запуска полноценного коммерческого сервиса. В качестве направлений дальнейшего развития выделены автоматизация обновления игрового рейтинга, добавление push-уведомлений, переход на стриминговую отдачу ответов языковой модели и реализация системы оплаты бронирований. Поставленные во введении цель и задачи достигнуты в полном объёме.