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-ПЛАТФОРМЫ ДЛЯ ПОИСКА, ВИЗУАЛИЗАЦИИ И АВТОМАТИЧЕСКОЙ ГЕНЕРАЦИИ АККОРДОВЫХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 411 группы
направления 02.03.02 — Фундаментальная информатика и информационные технологии
факультета КНиИТ
Тульцова Александра Алексеевича

Научный руководитель
к. ф.-м. н., доцент _____ А. С. Иванов

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

Саратов 2025

ВВЕДЕНИЕ

Актуальность темы. С каждым годом цифровые технологии все глубже проникают в творческий процесс музыкантов, превращая привычные аккордовые справочники в бесконечные онлайн-библиотеки и таблицы. Поиск нужного перехода во время сочинения нередко превращается в серию рутинных кликов и сверок схем аппликатур, отвлекая от главного — создания музыки.

Веб-платформа объединяет полнотекстовый поиск песен и аккордов, динамическую визуализацию аппликатур, анализ переходов через граф и трансформерную модель для рекомендаций гармонических продолжений. Начав с простого запроса, пользователь получает полный обзор структуры композиции, наглядные схемы и упорядоченные по вероятности варианты следующих аккордов.

В дополнение к описанным возможностям особое внимание уделялось адаптивности платформы под различные уровни подготовки музыкантов. Начинающие пользователи получают пошаговые подсказки и визуальные инструкции, а опытные музыканты могут глубоко анализировать вероятности переходов и экспериментировать с нестандартными прогрессиями. Интеграция модулей выполнена таким образом, чтобы обеспечить низкую задержку отклика и интуитивную навигацию. Технологическое решение позволяет расширять функционал без существенного увеличения нагрузки на сервер. В результате платформа сочетает в себе простоту использования и высокую производительность.

Цель бакалаврской работы. Разработка интеллектуальной веб-платформы для анализа, визуализации и генерации аккордовых последовательностей в песнях.

Для достижения этой цели в работе решаются **следующие задачи**:

1. Выбор и обоснование технологического стека (FastAPI, React, Elasticsearch, PyTorch) для создания приложения.
2. Проектирование пользовательского интерфейса с акцентом на плавные переходы между поиском, анализом гармонии и получением рекомендаций.
3. Реализация полнотекстового поиска по текстам песен и аккордам с использованием Elasticsearch.
4. Построение графа переходов между аккордами и интеграция трансформерной модели для генерации вероятных продолжений.

Структура и объем работы. Бакалаврская работа состоит из введения, 3 разделов, заключения, списка использованных источников и 4 приложений. Общий объем работы – 78 страниц, из них 48 страниц – основное содержание, включая 10 рисунков, цифровой носитель в качестве приложения, список использованных источников информации – 25 наименований.

Первая глава «Анализ предметной области» посвящена постановке задач, обзору существующих аналогов, а также теоретической информации, касающейся инструментов разработки.

Вторая глава имеет название «Инструментарий разработки», данная глава содержит подробное описание технологий и их примеров используемых в работе.

Третья глава имеет название «Трансформер», данная глава посвящена описанию структуры модели, а также ее обучению.

Выпускная квалификационная работа заканчивается заключением, списком использованных источников, а также приложения с кодом А-Г.

1 Основное содержание работы

Постановка задачи. Потребности музыкантов показывают необходимость объединения нескольких функций в едином интерфейсе для сокращения рутинных операций при работе с аккордовой аналитикой. Задача заключается в реализации механизма, позволяющего при вводе запроса «Аккорд A» получить интерактивную схему аппликатур, граф вероятностных переходов на основе марковской модели и список рекомендаций гармонических продолжений, упорядоченных по релевантности. Такой подход устраняет необходимость многократной смены инструментов и повышает фокус на творческом процессе.

Описание продукта. Прежде всего, платформа должна реализовать полнотекстовый поиск по названиям песен, тексту и аккордовым последовательностям, позволяя находить нужный фрагмент за доли секунды. Например, при вводе «Am – F – C» пользователь сразу получает список композиций с таким переходом, а не просматривает десятки нерелевантных страниц.

Наряду с поиском важна динамическая визуализация аппликатур: каждое текстовое обозначение аккорда преобразуется в интерактивную схему грифа, где можно кликнуть на лад и сразу увидеть позицию пальца. Это сокращает время обучения: новичок легко проверит правильность постановки пальцев без необходимости обращения к внешним иллюстрациям.

Далее, система должна предлагать гармонические рекомендации, основанные на анализе графа Маркова и трансформерной модели. При работе с любой песней пользователь получает упорядоченный по вероятности список следующих аккордов — с указанием вероятности каждого перехода.

Не менее важна адаптивность интерфейса: для начинающих необходимо выводить подсказки шаг за шагом, поясняя термины и предлагая готовые шаблоны, а опытным музыкантам — давать доступ к настройкам модели и расширенным фильтрам по стилю, темпу и тональности.

Кроме того, платформа должна обеспечивать низкую задержку отклика (менее 200 мс на основные запросы) благодаря кэшированию в Redis и оптимизированным GraphQL-запросам.

Нефункциональные требования охватывают производительность, кроссплатформенность и безопасность. Время отклика при запросе не должно превышать 200 мс даже при большом объеме данных и сложном вычислении правил трансформера. Поддержка браузерных и мобильных версий обеспечивает

доступ с десктопов и планшетов без потери качества рендеринга аппликатур. Передача пользовательских данных и настроек происходит исключительно по HTTPS с использованием современных алгоритмов шифрования, а хранение персональной информации — в зашифрованном виде.

В итоге, сформулированный набор требований задает четкие критерии приемки платформы и определяет направление для проектирования архитектуры, разработки интерфейса и последующего тестирования.

Анализ существующих решений. Для выявления ограничений современных онлайн-сервисов работы с аккордами рассмотрены три популярных веб-ресурса: am-dm, аккордс.pro и Ultimate Guitar.

Платформа am-dm предоставляет обширный каталог табулатур, однако ее интерфейс выглядит устаревшим: отсутствие адаптивной верстки затрудняет работу на разных устройствах, а статичные текстовые описания аккордов не позволяют визуализировать аппликатуру в реальном времени. При этом поиск по фрагментам последовательности реализован формально — без учета контекста гармоний — из-за чего музыканту приходится перебирать десятки нерелевантных вариантов.

Сайт akkords.pro привлекает пользователя минималистичным дизайном, но при более глубокой проверке становится заметно несколько ограничений. Навигация по списку песен и схематические изображения аппликатур вынуждают прокручивать длинные страницы, что замедляет работу и из-за отсутствия инструментов анализа переходов нет возможности оценить вероятность того или иного гармонического развития, а рекламные блоки добавляют лишние фоновые задержки при загрузке.

Chordify автоматически распознаёт аккорды из аудио-файлов и отображает их в виде синхронизированной схемы, однако точность распознавания часто снижается при сложных аранжировках, а инструменты для ручной корректировки ограничены.

E-Chords предлагает инструмент для редактирования и транспозиции табулатур, но базируется преимущественно на загруженных пользователями результатах, поэтому качество и полнота партий аккордов варьируются.

Наконец, крупнейший в своей категории Ultimate Guitar отличается богатством контента и наличием мобильных приложений, однако многие расширенные функции: автоматическая транспозиция, анимация аппликатур, расширен-

ный поиск доступны только по платной подписке. Кроме того, большое количество пользовательских рецензий и рейтингов вводит элемент субъективности: музыкант тратит время на фильтрацию нерелевантных вариантов и проверку качества табов.

Подводя итоги, несмотря на разнообразие ресурсов, ни одно из существующих решений не объединяет полнотекстовый поиск, динамическую визуализацию аппликатур и интеллектуальные рекомендации в едином интерфейсе. Это обуславливает потребность в разработке новой платформы, способной устранить перечисленные ограничения.

Архитектурный стиль. Одностраничные приложения (SPA) получают широкое распространение в технологической сфере. С ростом спроса на сложные веб-приложения и насыщенные пользовательские интерфейсы архитектурный шаблон SPA выбирается все чаще. Характерными достоинствами такого подхода являются высокая скорость и плавность откликов при взаимодействии пользователя, относительная простота разработки и отладки, а также удобство последующего перехода к мобильным платформам за счет разделения серверной и клиентской частей. Примерами ежедневно эксплуатируемых SPA являются Gmail, Google Docs, Facebook, Twitter и GitHub.

2 Инструментарий разработки

React — это библиотека JavaScript с компонентной архитектурой, изначально созданная Facebook. Она упрощает разработчикам создание интерактивных пользовательских интерфейсов, одновременно эффективно управляя состоянием компонентов. Благодаря возможности комбинировать множество компонентов для построения сложных приложений без потери их состояния в DOM (Document Object Model) браузера React стал серьезным преимуществом для многих разработчиков [1].

TypeScript — это язык с открытым исходным кодом от Microsoft, компилирующийся в JavaScript. Он оказался настолько востребованным, что в 2019 году занял 7-е место по распространенности и 5-е место по темпам роста на GitHub. При этом любой корректный код на JavaScript автоматически остается валидным TypeScript-кодом. Благодаря введению опциональной системы типов появляется возможность использовать мощные инструменты для масштабных JavaScript-приложений на любом браузере, сервере или операционной системе [2].

Axios — это библиотека HTTP-клиента, которая позволяет отправлять запросы к любому API-эндпоинту: внешнему сервису или вашему собственному серверу на Node.js. ее методы буквально повторяют HTTP-глаголы (.get(), .post(), .put(), .delete()), а сама библиотека заботится о преобразовании JSON, заголовках и выбросе ошибок – все это экономит массу строчек кода.

Scss — это препроцессор CSS, который расширяет возможности этого языка, делая его более удобным и мощным. Он позволяет писать CSS-код с помощью переменных, вложенных правил, миксинов и других полезных функций. Sass генерирует стандартный CSS-код, который затем может быть использован браузерами.

D3.js (Data-Driven Documents) представляет собой JavaScript-библиотеку для создания динамических, интерактивных визуализаций на основе стандартов веба (SVG, HTML, CSS). При этом сама аббревиатура D3 подчеркивает ключевой подход библиотеки: «data-driven documents», то есть «документы, управляемые данными» — где данные напрямую влияют на формирование и обновление элементов DOM.

SVGuitar это небольшой JavaScript-пакет, умеющий на лету рисовать диаграммы гитарных аккордов в формате SVG, опираясь на заранее подготовленные

данные о расположении пальцев.

Tone.js это библиотека для создания интерактивных инструментов и генерации музыки непосредственно в браузере. Она построена поверх Web Audio API и включает обширный набор абстракций: синтезаторы, эффекты, генераторы тонов и утилиты, предназначенные для работы с аудиоконтекстом. Web Audio API обеспечивает выполнение всех аудиоопераций внутри единого аудиоконтекста и поддерживает модульную маршрутизацию сигналов. Базовые операции выполняются через аудио-узлы (Audio Nodes), которые соединяются в граф аудиомаршрутизации.

FastAPI — это фреймворк на Python для создания веб-API и веб-приложений. Он поддерживает современные возможности языка, такие как асинхронная обработка и аннотации типов, благодаря чему получается быстро и эффективно. Кроме того, FastAPI использует стандарт ASGI (Asynchronous Server Gateway Interface) для асинхронного и конкурентного обслуживания клиентов, но при необходимости умеет работать и по WSGI.

SQLAlchemy это ORM-библиотека, для работы с базами данных в Python. SQLAlchemy состоит из двух взаимосвязанных API — Core и ORM. Они выстроены один над другим, где Core задает фундамент, а ORM расширяет его возможностями объектно-реляционного отображения [3].

Elasticsearch, позволяет строить специализированный индекс с набором фильтров и анализаторов для русского и английского языков, а также edge-ngram для автодополнения. В скрипте создания индекса задаются фильтры russian_stop, english_stemmer и edge_ngram, что обеспечивает удаление стоп-слов, приведение слов к корню и поиск по начальным символам [4].

PyTorch — это фреймворк для языка программирования Python, предназначенный для машинного обучения. Он включает в себя набор инструментов для работы с моделями, используется в обработке естественного языка, компьютерном зрении и других похожих направлениях.

3 Трансформер

Архитектура трансформера построена по классической схеме «энкодер—декодер энкодер сначала преобразует входную последовательность символов $(x_1, ..., x_n)$ в набор непрерывных векторных представлений $(z_1, ..., z_n)$, а затем декодер авторегрессивно генерирует выходную последовательность $(y_1, ..., y_n)$.

Трансформер придерживается общей архитектуры, используя каскад слоев внимания (attention) и покомпонентных полносвязных слоев как в энкодере, так и в декодере, что показано в левой и правой частях (1) соответственно.

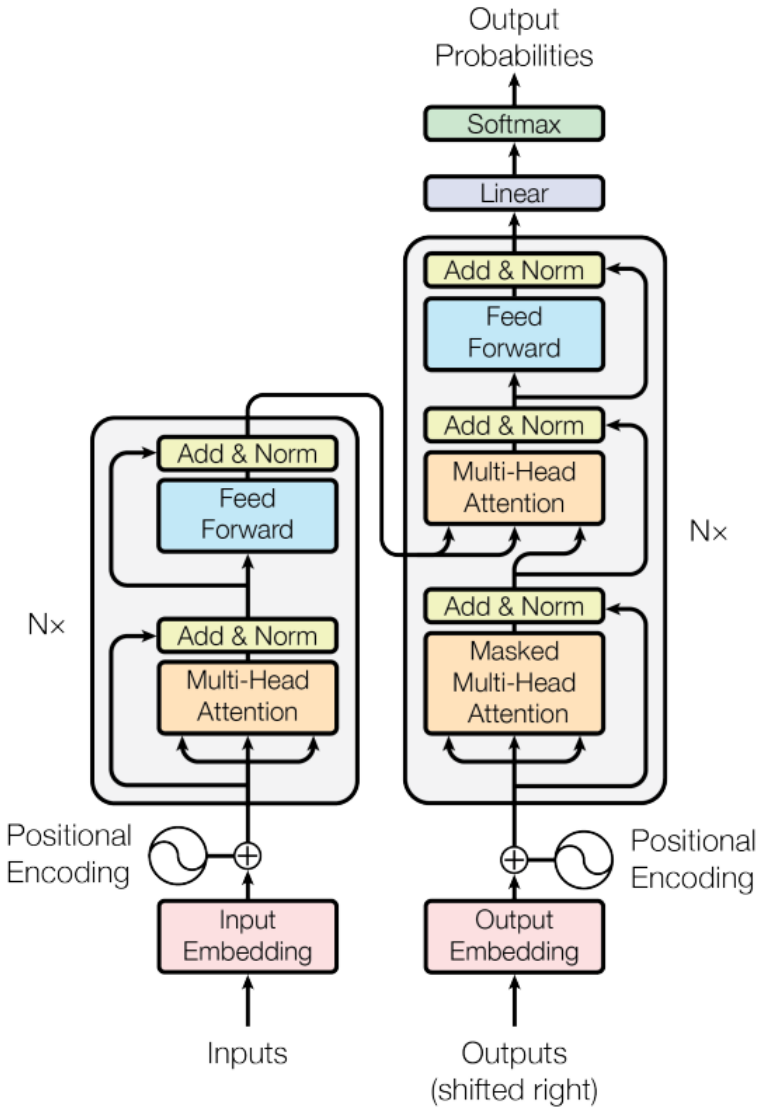

Рисунок 1 – Общая схема архитектуры Transformer

Энкодер состоит из стека $N = 6$ идентичных слоев. Каждый слой включает два подслоя: механизм многоголового внимания и покомпонентную полносвязную сеть прямого распространения. Вокруг каждого подслоя используется

остаточное соединение, за которым следует нормализация слоя [5].

$$\mathrm{LayerNorm}\big(x + \mathrm{Sublayer}(x)\big)$$

А размерность выходного пространства всех подслоев и эмбеддингов задается как

$$d_{\mathrm{model}} = 512.$$

Декодер также состоит из стека $N = 6$ идентичных слоев. Каждый слой включает три подслоя:

1. Маскированное многоголовое внимание (закрывает доступ к «будущим» позициям).
2. Многоголовое внимание к выходам энкодера.
3. Позиционно-зависимая полносвязная сеть.

Вокруг каждого подслоя применяется остаточное соединение с последующей нормализацией.

$$\mathrm{LayerNorm}\big(x + \mathrm{Sublayer}(x)\big)$$

Где для маскированного внимания используется модифицированная формула:

$$\mathrm{MaskedAttention}(Q, K, V) = \mathrm{softmax}\Big(\frac{QK^{\mathsf{T}}}{\sqrt{d_k}} + M\Big)\,V$$

Здесь матрица маски M содержит $-\inf$ для запрещенных (будущих) позиций и 0 — для разрешенных, что вместе со «сдвигом» выходных эмбеддингов гарантирует, что предсказание на позиции i зависит только от ранее сгенерированных токенов.

Для внимания над выходами энкодера используется та же базовая формула без маски:

$$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}\Big(\frac{QK^{\mathsf{T}}}{\sqrt{d_k}}\Big)\,V$$

Основной цикл обучения В процессе обучения модель переводится в режим тренировки, что включает в себя активацию механизмов регуляризации (dropout) и подготовку всех параметров к накоплению градиентов. Затем через загрузчик батчей последовательно поступают тензоры входных последовательностей аккордов и соответствующих жанров, которые подаются на вход модели.

При этом при каждой итерации по батчу обеспечивается последовательное выполнение прямого прохода, вычисления ошибки, обнуления градиентов, обратного распространения и шага оптимизации. После прохода по всем батчам накопленное значение потерь нормируется на число итераций для получения средней метрики эпохи.

Параллельно сохраняется историю значений потерь в списки train_losses и val_losses для последующей визуализации динамики обучения.

Такой непрерывный цикл без явной нумерации шагов позволяет компактно и удобно интегрировать логику обучения в скрипт, обеспечивая при этом наглядный вывод ключевых метрик(2).

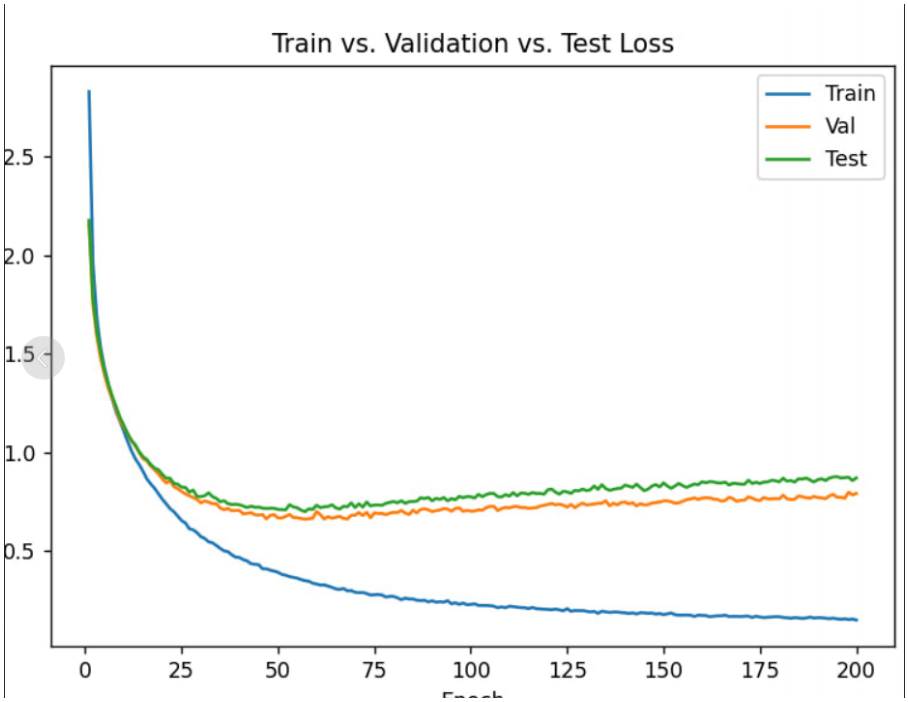

Рисунок 2 – Визуализация динамики обучения

На графике видно, что после примерно 50–60 эпох кривая валидационной (оранжевой) и тестовой (зелёной) потерь достигает своего минимума и затем начинает постепенно расти, это означает, что модель начинает переобучаться после 50 эпохи, поэтому была выбрана данная точка на графике.

ЗАКЛЮЧЕНИЕ

Итогом проведенной работы стало создание полнофункциональной веб-платформы, объединяющей поиск, визуализацию и интеллектуальную генерацию аккордовых последовательностей. В результате внедрения FastAPI и React была обеспечена отзывчивость интерфейса, позволяющая получать результаты поиска по песням и аккордам за доли секунды. Elasticsearch решает задачу гибкого поиска: будь то точный подбор по исполнителю или нечеткий поиск с опечатками — все сценарии обрабатываются автоматически.

Благодаря построению графа переходов между аккордами и интеграции трансформерной модели пользователи могут получать рекомендации музыкальных продолжений, опирающиеся на реальные практики гармонизации. Например, при вводе базовой прогрессии «Am–F–C» система предлагает наиболее вероятное «G» и альтернативы в стиле джазовых аппликатур. Короткий пример такой работы — моментальное появление трех вариантов продолжения, упорядоченных по вероятности, что позволяет сразу же приступить к экспериментам.

Особое внимание уделялось адаптивности: начинающим музыкантам выдаются поясняющие подсказки и визуальные инструкции по аппликатурам, а продвинутые пользователи получают доступ к статистике переходов. Такая гибкость делает платформу полезной как для обучения, так и для профессиональной практики.

Технологическая архитектура построена с расчетом на масштабирование и модульное расширение. Каждый компонент — поиск, граф гармоник, трансформер — оперирует независимо, что снижает нагрузку на сервер при добавлении новых функций. В результате разработанная система сочетает в себе простоту использования, высокую производительность и готовность к будущему развитию.

В результате работы были успешно решены все поставленные задачи: подобран многофункциональный стек (FastAPI, React, Elasticsearch, PyTorch), реализован полнотекстовый поиск и интегрирована модель музыкальных рекомендаций на основе трансформера и графа переходов.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Раушмайер, Д,. Front-End разработка: Современный подход / Раушмайер, Д. — Эксмо, 2021.

2 Черни, Б,. TypeScript и JavaScript. Полный справочник / Черни, Б.— Аль пина Паблишер, 2020.

3 SQLAlchemy часть 4: продвинутый SQLAlchemy.— [Электронный ресурс], URL: https://medium.com/@umair.qau586/ sqlalchemy-seriespart-4-advanced-sqlalchemy-features-for-power-users (Дата обращения 1.04.2025). Яз. англ..

4 Строим продвинутый поиск с Elasticsearch.— URL: https: //dou.ua/lenta/columns/building-advanced-search-with-elasticsearch/ (Дата обращения 10.02.2025). Яз. рус.

5 Attention Is All You Need [Электронный ресурс].— [Электронный ресурс] URL: https://paperswithcode.com/paper/attention-is-all-you-need (Дата обращения 15.02.2025). Яз. англ.