Министерство образования Новосибирской области ГБПОУ НСО «Новосибирский авиационный технический колледж имени Б.С. Галущака»

Разработка веб-приложения для игры в шахматы  
с элементами обучения

Пояснительная записка к дипломному проекту

НАТКиГ.101900.010.000 ПЗ

Разработал:

студент группы ПР-21.101

Некрасов Я.А.

**Содержание**

[Введение 4](#_Toc200013186)

[1 Исследование предметной области 5](#_Toc200013187)

[1.1 Описание предметной области 5](#_Toc200013188)

[1.2 Описание групп пользователей и функциональных задач 8](#_Toc200013189)

[1.3 Анализ программных ресурсов необходимых в работе 10](#_Toc200013190)

[1.4 Анализ аналогов и прототипов 12](#_Toc200013191)

[2 Технологический раздел 15](#_Toc200013192)

[2.1 Структура базы данных 15](#_Toc200013193)

[2.2 Разработка структуры сайта 16](#_Toc200013194)

[2.3 Разработка дизайн макета сайта 19](#_Toc200013195)

[2.4 Разработка алгоритмов обработки информации 23](#_Toc200013196)

[2.5 Разработка текстового содержания сайта 31](#_Toc200013197)

[2.6 Разработка мультимедийного контента 34](#_Toc200013198)

[3 Руководство пользователя 39](#_Toc200013199)

[3.1 Введение 39](#_Toc200013200)

[3.2 Назначение и условия применения 40](#_Toc200013201)

[3.3 Подготовка к работе 40](#_Toc200013202)

[3.4 Описание операций 41](#_Toc200013203)

[4 Тестирование программного продукта 46](#_Toc200013204)

[4.1 Выбор стратегии тестирования 46](#_Toc200013205)

[4.2 Разработка сценариев и протоколов тестирования 46](#_Toc200013206)

[Заключение 52](#_Toc200013207)

[Библиография 54](#_Toc200013208)

# Введение

Шахматы являются популярной темой во всем мире. С конца 20 века компьютеры значительно повлияли на шахматное сообщество. На шахматном сервисе «Lichess.org» происходит около 90 миллионов партий между пользователями сервиса ежемесячно. Создание шахматного приложения может повысить доступность этой игры для развития у пользователя логического мышления, навыков планирования.

Дипломный проект по созданию справочного приложения с элементами обучения позволит пользователю осуществлять игру в шахматы с искусственным противником и получать знания о названиях шахматных позиций для дальнейшего самостоятельного изучения.

Целью разработки является создание шахматного приложения с элементами обучения.

Задачами курсового проекта в связи с указанной целью являются:

* изучение предметной области;
* рассмотрение приложения с точки зрения пользователя для выявления функций приложения;
* разработка и реализация дизайна приложения;
* написание кода приложения;
* тестирование полученного продукта.

Практическая значимость проекта состоит в увеличении количества активных игроков в шахматы и увеличение качества игры благодаря учебным материалам, направленным на русскоязычную аудиторию.

# Исследование предметной области

## Описание предметной области

Многим людям, имеющим свободное время приходила идея об отдыхе, и для исполнения данного желания имеется множество видов досуга, в число которых входит времяпровождение за настольными играми. Некоторые из настольных игр не только снимают стресс, но также развивают мышление.

Д. И. Бронштейн и Г. Л. Смолян выделяют четыре фактора, обуславливающие творческий характер шахмат:

* радость, которую получает шахматист, создавая художественные ценности самостоятельно;
* зритель, к которому обращены данные ценности. Без публики шахматисты не испытывают творческое напряжение, и их «остывшая душа оказывается в плену бесчисленных вариантов»;
* красота, для одних шахматистов и зрителей проявляющаяся в строгости, законченности и силе логических построений, для других — в простоте, а для третьих красота рождается в борьбе против очевидного. В целом же, можно выделить «эстетические инварианты», которыми восхищаются и которые ожидают увидеть все неравнодушные к шахматам. К таким инвариантам авторы относят: комбинации, манёвры, ловушки, сложную логику плана и геометрическую гармонию взаимодействия. Гибкая среда, которая позволяет игроку выйти за пределы чистой мысли и проверить силу своих идей на доске;

В своём определении шахмат как искусства Хамбл во многом ссылается на Д. И. Бронштейна и Г. Л. Смоляна. Так, он формулирует следующие условия, которым должна удовлетворять шахматная партия как произведение искусства:

* шахматная партия должна доставлять эстетическое удовольствие;
* шахматная партия должна обладать «эстетическими качествами» (например, целесообразность ходов, точность комбинаций, наличие скрытых угроз и т. д.);
* шахматная партия должна быть уникальной (оригинальность замысла).

Хамбл не рассматривает шахматные партии в качестве великих произведений искусства, так как они «по своей природе не способны раскрыть глубокие проблемы человечества, поднимаемые великим искусством». Поэтому Хамбл определяет шахматы как искусство малых форм, в отличие от живописи, музыки и т. д.

Данное явление не обошло стороной популярную настольную игру «шахматы», соревновательная игра между двумя игроками, действие которой происходят на игровом поле, с разметкой из клеток восемь на восемь, целью которой является с помощью специального набора фигур таких как: пешка, слон, конь, ладья, ферзь, король. Каждая фигура ходит определенным правилами образом. Целью игры является атака фигуры короля оппонента, от которой он не сможет уйти. Ничья может заключаться в случаях:

* договора между двумя игроками;
* отсутствия возможности сделать ход, не противоречащий  
  правилам игры;
* невозможности поставить мат фигурами, оставшимися на доске;
* первые тридцать ходов прошли без взятия фигур;
* семьдесят ходов не происходило ходов пешками и взятия фигур.

Проведение партий шахмат подразделяют по контролю времени, и самыми популярными и общепринятыми, являются классика, рапид и блиц.

При классическом контроле времени на обдумывание каждому игроку дается не мене часа на партию. Наиболее популярный контроль девяносто минут на сорок ходов. Далее возможны варианты: тридцать ходов до конца, двадцать минут с добавлением секунд за каждый сделанный ход и некоторые другие.

Рапид или быстрые шахматы – такой формат игры, в котором каждому из игроков дается время на обдумывания более десяти, но менее шестидесяти минут. Контроль может быть как с добавлением секунд на каждый сделанный ход, так и без добавления сегодня самый популярный контроль времени рапиде – пятнадцать минут и десять секунд за на каждый ход. В блице (разновидность шахмат с жестким лимитом времени) каждый игрок имеет не более 10 минут на всю партию. Такое ограничение ускоряет игру, заставляя игроков принимать решения быстро, что увеличивает динамику и напряженность борьбы.

Шахматная партия традиционно делится на три ключевые стадии:

* дебют (начало партии) – первые 10–15 ходов, где игроки развивают фигуры, контролируют центр и обеспечивают безопасность короля (например, рокировка);
* миттельшпиль (середина игры) – фаза активных тактических и стратегических действий: атаки, защита, жертвы фигур, создание комбинаций;
* эндшпиль (окончание партии) – этап, когда на доске остается мало фигур, и игра сводится к продвижению пешек, активации короля и достижению мата или ничьей;

Каждый ход в шахматах порождает новые возможные позиции, что можно представить в виде дерева вариантов. Первый ход дает 20 возможных варианта (16 ходов пешками и 4 хода конями). С каждым следующим ходом количество возможных позиций растет экспоненциально. К пятому ходу число возможных позиций оценивается в почти 70 триллионов.

Из-за такого огромного количества вариантов невозможно заранее просчитать все лучшие ходы, поэтому шахматисты полагаются на:

* готовые дебютные схемы – заранее изученные последовательности ходов;
* тактическое чутьё – умение видеть комбинации в миттельшпил;
* эндшпильную технику – знание ключевых принципов окончаний.

Наиболее известные дебюты получают имена, связанные с:

турнирами, где они впервые были применены (например, Венская партия).

именами шахматистов, которые их популяризировали (Защита Нимцовича, Гамбит Эванса) или шутливыми ассоциациями (атака жареной печени, кокакольный гамбит).

Метафорами, отражающими особенности расстановки (коготь бобра, дебют Гроба).

Эти названия не просто формальность – они хранят стратегии, исторические партии и типовые планы, помогая игрокам быстрее изучать игру и перенимать опыт мастеров. Таким образом, шахматные дебюты – это важная часть культуры игры, объединяющая теорию, практику и историю шахмат.

## Описание групп пользователей и функциональных задач

В современном мире шахмат онлайн-платформы играют ключевую роль в обучении, анализе и совершенствовании игровых навыков. Однако у существующих сервисов есть ряд особенностей, которые могут ограничивать пользователей. Например, практически все популярные шахматные платформы сохраняют каждую сыгранную партию в открытом доступе, делая её историю частью публичного профиля игрока. Это может создавать психологический дискомфорт для тех, кто только начинает свой путь в шахматах или экспериментирует с новыми стратегиями. Любая ошибка, неверный ход или неудачная партия остаются зафиксированными в аккаунте, что может отпугивать игроков от свободного изучения новых подходов к игре.

Кроме того, в русскоязычном сегменте интернета наблюдается явный дефицит специализированных платформ, где игроки могли бы не только анализировать свои партии, но и создавать, публиковать и изучать пользовательские руководства по шахматам. Существующие ресурсы чаще всего предлагают либо стандартные учебные материалы, либо форумы с разрозненной информацией, что затрудняет систематизированное обучение. Часть терминологии не переведена на русский язык, ярче всего это отражено в наименовании шахматных позиций, эту проблему стремится решить разработанная база данных шахматных дебютов, открытая для использования сторонними разработчиками.

Веб-приложение направлено на решение этих проблем. Проект стремится создать удобный и функциональный сервис, который предоставит пользователям следующие возможности:

* доступ к качественным руководствам – платформа станет местом, где русскоязычные игроки смогут находить подробные, структурированные и проверенные материалы по различным аспектам шахмат: от основ игры до продвинутых стратегий;
* создание собственных учебных материалов – проект предлагает инструменты для тех, кто хочет делиться своим опытом, будь то разбор дебютов, миттельшпиля, эндшпиля или психологических аспектов игры;
* приватность игрового процесса – в отличие от многих популярных сервисов, программный продукт позволит игрокам сохранять конфиденциальность, выбирая, какие партии сделать публичными, а какие оставить только для личного анализа.

Целевая аудитория проекта:

* русскоязычные игроки, ищущие обучающие материалы – начинающие и любители, которым не хватает структурированных руководств на родном языке;
* авторы контента – шахматисты, тренеры и энтузиасты, желающие создавать учебные статьи, видеоразборы и другие форматы обучающих материалов;
* игроки, ценящие приватность – те, кто хочет избежать публичности каждой сыгранной партии и иметь контроль над своей игровой историей.

Сервис объединит в себе удобство обучения, возможность обмена знаниями и гибкость в настройках приватности, что сделает его востребованным среди русскоязычных шахматистов разного уровня подготовки. Функционал пользователей описан на рисунке 1.2.

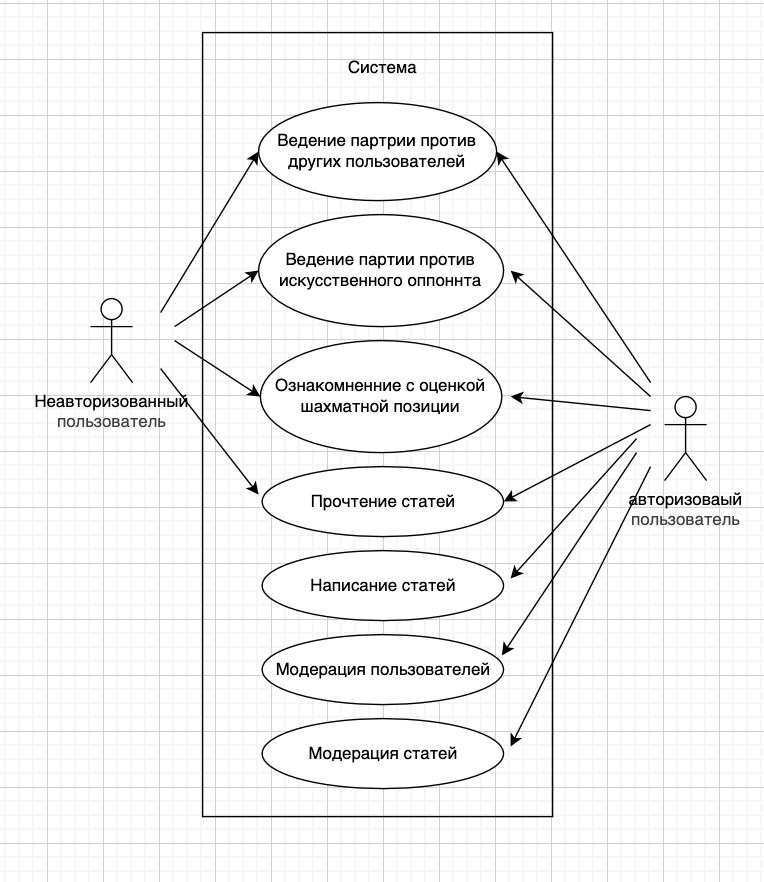


Рисунок 1.2 – Диаграмма прецедентов

В проекте используется система ABAC. Пользователи могут иметь атрибуты, которые отвечают за доступ пользователей к функционалу. Как пример: если пользователь имеет атрибут «admin\_users», то на соответствующей странице он сможет модерировать пользователей. Это позволяет разделить обязанности среди администраторов и выделить отдельных специалистов для работы над конкретными задачами.

## Анализ программных ресурсов необходимых в работе

**1.3.1 Требования для пользователей**

Поскольку проект представляет собой веб-приложение, пользователям не требуется установка дополнительного программного обеспечения. Достаточно иметь:

* любой современный веб-браузер (Google Chrome, Mozilla Firefox, Safari, Microsoft Edge и др.), который предустановлен в большинстве операционных систем (Windows, macOS, Linux, Android, iOS);
* стабильное интернет-подключение для загрузки и работы с приложением.

Возможные проблемы и их решение:

1. если у пользователя устаревший браузер, система может выдать предупреждение о необходимости обновления;
2. в случае несовместимости или низкой производительности рекомендуется:
   1. обновить браузер до последней версии;
   2. использовать альтернативный браузер (В процессе разработки приоритетным браузером являлся Google chrome);
   3. обратиться к документации проекта или в техподдержку для получения инструкций.

1.3.2 Требования для разработки и отладки

Для разработки, тестирования и развертывания проекта потребуются следующие инструменты:

1. основная среда разработки:
   1. WebStorm платный, мощный инструмент от JetBrains с глубокой интеграцией React и Node.js;
   2. Visual Studio Code (VS Code) бесплатный, популярный редактор с поддержкой расширений для React, ESLint, Prettier;
   3. Sublime Text легковесный редактор с плагинами для веб-разработки;
   4. VIM / Neovim для опытных разработчиков, требует настройки;
   5. Notepad++ минималистичный вариант для быстрого редактирования кода;
2. Среда выполнения, для:
   1. запуска серверной части;
   2. управления зависимостями через npm;
3. дополнительные инструменты:
   1. Git (система контроля версий, интеграция с GitHub/GitLab/Bitbucket);
   2. Postman (тестирование API);
   3. ESLint + Prettier (проверка стиля кода и автоматическое форматирование);
   4. React Developer Tools (расширение для отладки React-приложений в браузере);
4. База данных:
   1. SQLite для базы шахматных дебютов. Позволяет легко хранить и передавать данные в едином файле;
   2. Prisma для базы данных пользователей. Позволяет разрабатывать и управлять базой данных без необходимости установки стороннего программного обеспечения.

## Анализ аналогов и прототипов

В таблице 1.1 приведено сравнение наиболее востребованных шахматных платформ и справочных сайтов, таких как Chess.com, Lichess, Wikipedia. Эти ресурсы выбраны для анализа не случайно: они представляют собой основные площадки, которыми пользуются шахматисты — от новичков до профессионалов.

Пользователи, на которых ориентирован разрабатываемый сервис «Shahmatki», скорее всего, уже знакомы с этими платформами и используют их для:

* игры онлайн (Chess.com, Lichess);
* изучения теории (Wikipedia);
* анализа партий (Lichess, Chess.com);
* общения с другими шахматистами (форумы, чаты).

Таблица 1.1 – Сравнение аналогов

|  |  |  |  |
| --- | --- | --- | --- |
|  | Chess.com | Lichess.org | Wikipedia |
| Доступность | Сервис заблокирован на территории Российской Федерации | Сервис поддерживается на благотворительной основе и не предоставляет круглосуточную работу сайта | Имеет нарушения законодательства Российской Федерации |
| Предоставление ресурсов для разработчиков | Нет | Ежемесячно выставляет в открытый доступ архив партий игроков за месяц | Нет |
| Функционал | Предоставляет множество форматов с альтернативными правилами игры | Классические шахматы | Написание статей |
| Контент сообщества | Форум сообщества | Написание пользовательских руководств (не популярно на русском языке) | Все статьи написаны сообществом |
| Искусственные оппоненты | Большая часть доступна по платной подписке | Несколько уровней сложности | Функционал шахмат отсутствует |
| Доступность на русском языке | Переведен не полностью | Переведён не полностью | Возможность перевести партию на русский язык |

Хотя список не является исчерпывающим, он позволяет выявить ключевые потребности целевой аудитории: удобный интерфейс, обучающие материалы, инструменты для анализа и комфортное сообщество. Сравнение с этими проектами поможет определить, какие функции необходимо внедрить в «Shahmatki», чтобы сервис был конкурентоспособным и привлекательным для пользователей.

Разрабатываемый сервис «Shahmatki» ставит перед собой амбициозную задачу: сочетать лучшие черты ведущих шахматных платформ, но с акцентом на русскоязычную аудиторию и образовательную составляющую.

* пользовательский интерфейс – планируется сделать его интуитивно понятным, на уровне Chess.com, который считается одним из самых удобных шахматных сайтов;
* функционал для статей – как в Wikipedia, но с упором на шахматную тематику: база дебютов, разборы партий, учебные материалы;
* обучение и анализ – в отличие от многих платформ, где обучение платное, «Shahmatk» может предложить бесплатные курсы, интерактивные задачи и детальный разбор ошибок;
* сообщество – важно создать не просто платформу для игры, а дружелюбное сообщество, где игроки смогут общаться, обсуждать партии и делиться знаниями.

«Shahmatki» займет нишу русскоязычного шахматного сервиса с упором на обучение, анализ и комфортное взаимодействие между пользователями. В отличие от зарубежных аналогов, он будет учитывать специфику русскоязычных шахматистов.

Такой подход не только привлечет аудиторию, но и поспособствует популяризации шахмат, особенно среди начинающих игроков, которые часто сталкиваются с нехваткой качественных русскоязычных ресурсов. В перспективе это может сделать «Shahmatki» одним из ключевых шахматных проектов в Российском сегменте интернета.

# Технологический раздел

## Структура базы данных

В проекте присутствует две базы данных для изоляции логики и предоставления одной из них в открытом доступе. ERD баз данных представлены на рисунках 2.1 и 2.2.

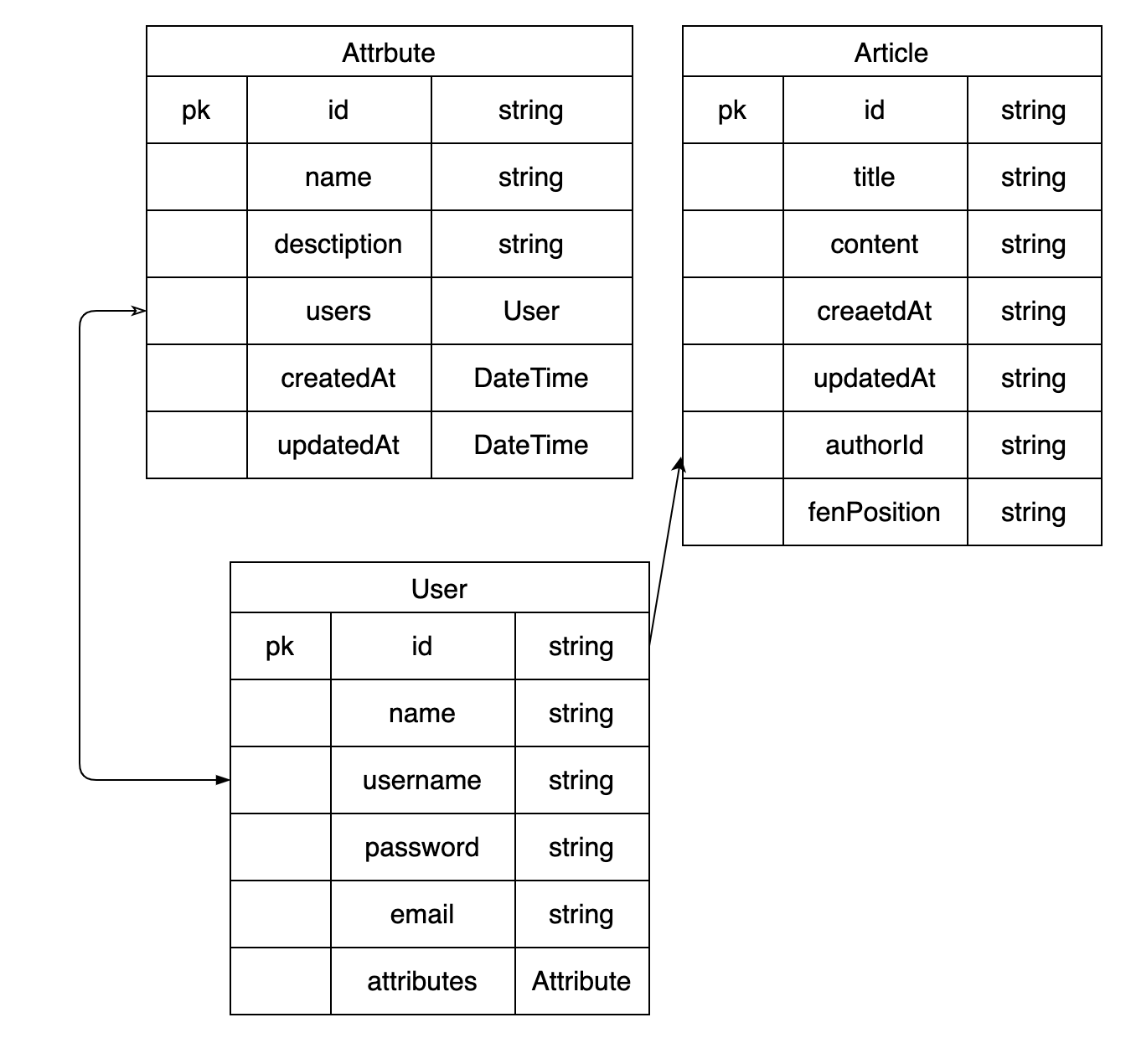


Рисунок 2.1 – База данных пользователей

База данных хранит в себе записи о пользователях, их атрибутах и написанных ими статьях. Для нее используется Prisma так как это позволяет:

* вести разработку без жесткой привязки к СУБД;
* использовать упрощённую систему миграции;
* глубоко интегрировать базу данных в код веб-приложения;
* повысить безопасность чувствительных данных, так как поддерживаются только серверные запросы.

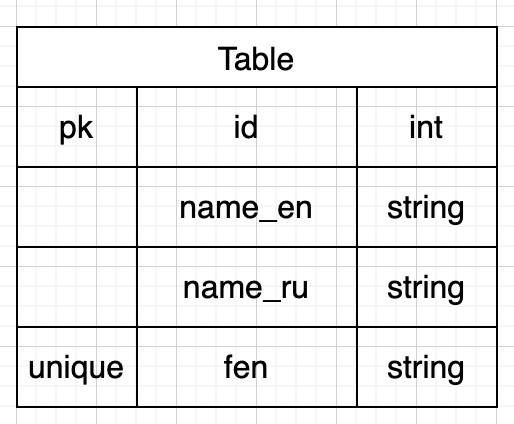


Рисунок 2.2 – База данных шахматных дебютов

База данных шахматных дебютов использует SQLite, так как эта база данных будет предоставлена пользователям в открытый доступ. Подобный выбор СУБД имеет следующие преимущества:

* облегчит распространение базы данных. Так как она хранится в едином файле;
* ускорит интеграцию в пользовательские проекты, благодаря низким требованиям к аппаратному обеспечению.

## Разработка структуры сайта

2.2.1 структура страниц

Старицы организованы с целью изоляции их логики для упрощения отладки и повышения удобства пользователя, так как страницы не перегружены функционалом и выполняют только свои первостепенные задачи. Страницы реализованы с помощью роутинга Next js основанного на файлах и их расположении.

Схема сайта изображена на рисунке 2.3.



Рисунок 2.3 – Схема сайта

2.2.2 структура кода

При разработке кода веб-сайта была выбрана архитектура CBA (Component-Based Architecture), которая предполагает разделение кода на переиспользуемые компоненты. В отличие от классического подхода, где HTML, CSS и JavaScript хранятся в отдельных файлах, в данном проекте применялся метод объединения разметки, функциональности и стилей в едином файле компонента. Это стало возможным благодаря использованию Tailwind CSS и React, где стили задаются непосредственно в JSX-разметке через утилитарные классы. Такой подход имеет свои преимущества и недостатки: с одной стороны, в одном файле сосредоточены разные аспекты компонента (логика, представление, стили), что может усложнить восприятие кода, а с другой – все связанные с компонентом функции, включая управление состоянием, находятся в одном месте, что упрощает разработку динамических интерфейсов и поддержку кода.

Для организации страниц использовались паттерны разработки, предусмотренные Next.js. В этой структуре каждый файл с именем page.js, page.ts, page.jsx или page.tsx, расположенный в папке app, автоматически становится страницей веб-сайта, а его URL-адрес определяется путем к файлу в файловой системе проекта. Например, файл «app/dashboard/page.ts» будет доступен по адресу «/dashboard». Это позволяет интуитивно организовывать маршрутизацию и упрощает навигацию по кодовой базе. Визуальное представление файловой структуры страниц можно увидеть на рисунке 2.4.

API-маршруты в Next.js также следуют строгой конвенции. Каждый файл, находящийся в директории «app/api» и названный «route.js» или «route.ts», автоматически становится API-эндпоинтом, доступ к которому осуществляется через соответствующий HTTP-запрос (GET, POST и т. д.). Это позволяет легко создавать серверную логику без необходимости настройки дополнительных инструментов маршрутизации. Структура API проекта детально отображена на рисунке 2.5.

Использование встроенных возможностей Next.js, таких как файловая маршрутизация и API-роутинг, значительно повышает производительность разработки, особенно в коммерческих проектах. Фреймворк предоставляет готовые решения для fullstack-разработки, включая серверный рендеринг, статическую генерацию и удобную работу с API, что сокращает время на настройку инфраструктуры и позволяет сосредоточиться на бизнес-логике приложения.

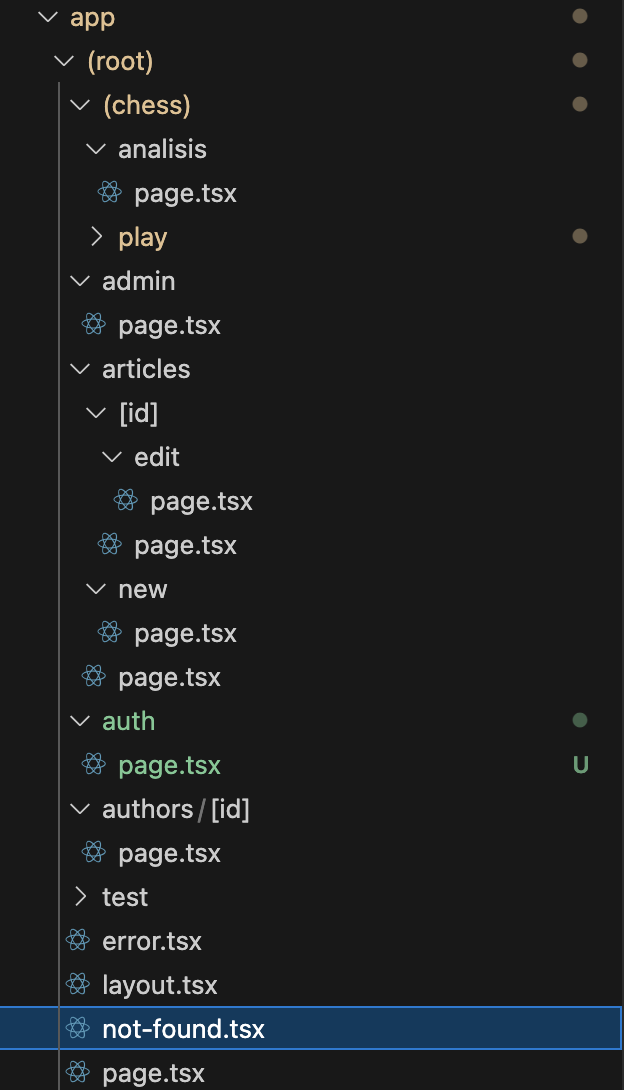


Рисунок 2.4 – Файловая структура страниц



Рисунок 2.5 – Структура api

## Разработка дизайн макета сайта

Макет приложения подготовлен в веб-ресурсе Figma. С соблюдением дизайн-документа Versel. Страницы макета изображены на рисунках 2.6 по 2.12.

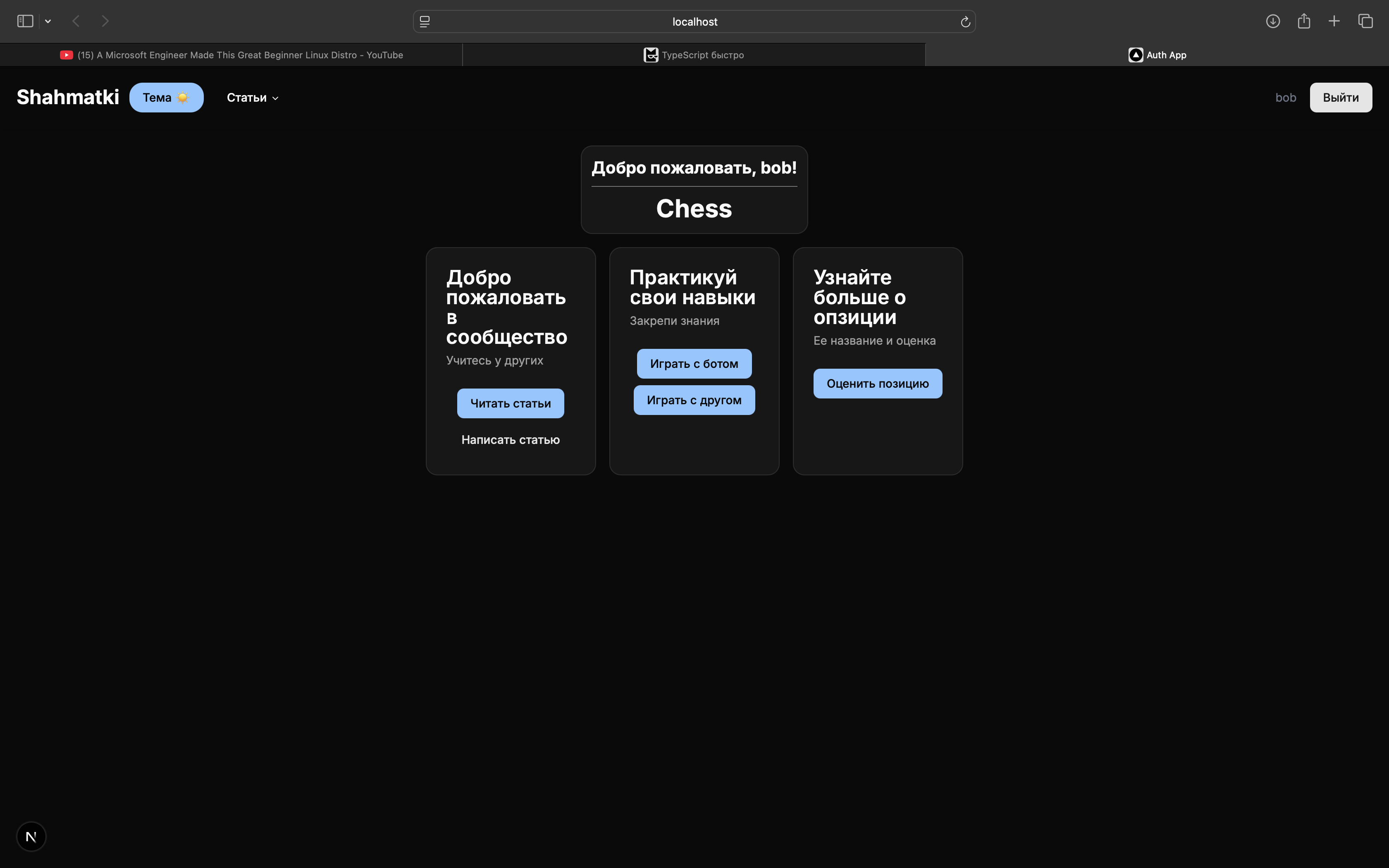


Рисунок 2.6 – Дизайн главной страницы

На главной странице изображено приветствие пользоавтеля и навигационные ссылки на основные разделы сайта.



Рисунок 2.7 – Дизайн страницы анализа позиции

Пользователю предоставляется возможность передвижения фигур и ознакомления с материалами по позиции на доске.

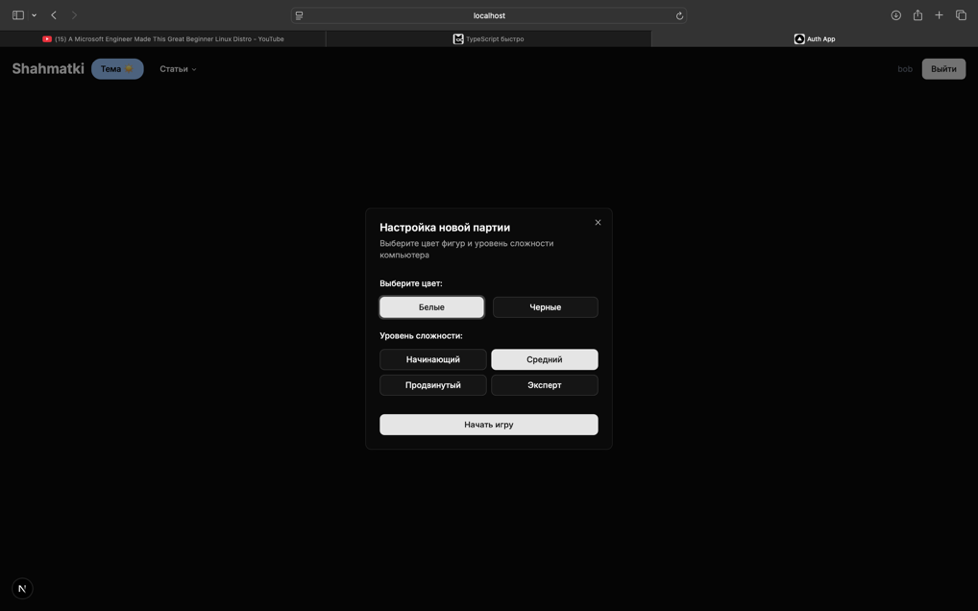


Рисунок 2.8 – Дизайн страницы настнойки искусственного оппонента

Пользоатель проходит настройку бота под свой уровень игры для улучшения игрового процесса.



Рисунок 2.9 – Дизайн страницы шахмат с ботом

Пользователь играет в шахматы и ознакамлиавется с графиком оценки позиции для понимания тогоо, в какой момеент он совершил не лучший ход. Пользователь может вернуть ход и совершить другой, так как этот режим игры направлен на тренеровку, а не реальный процесс игры.

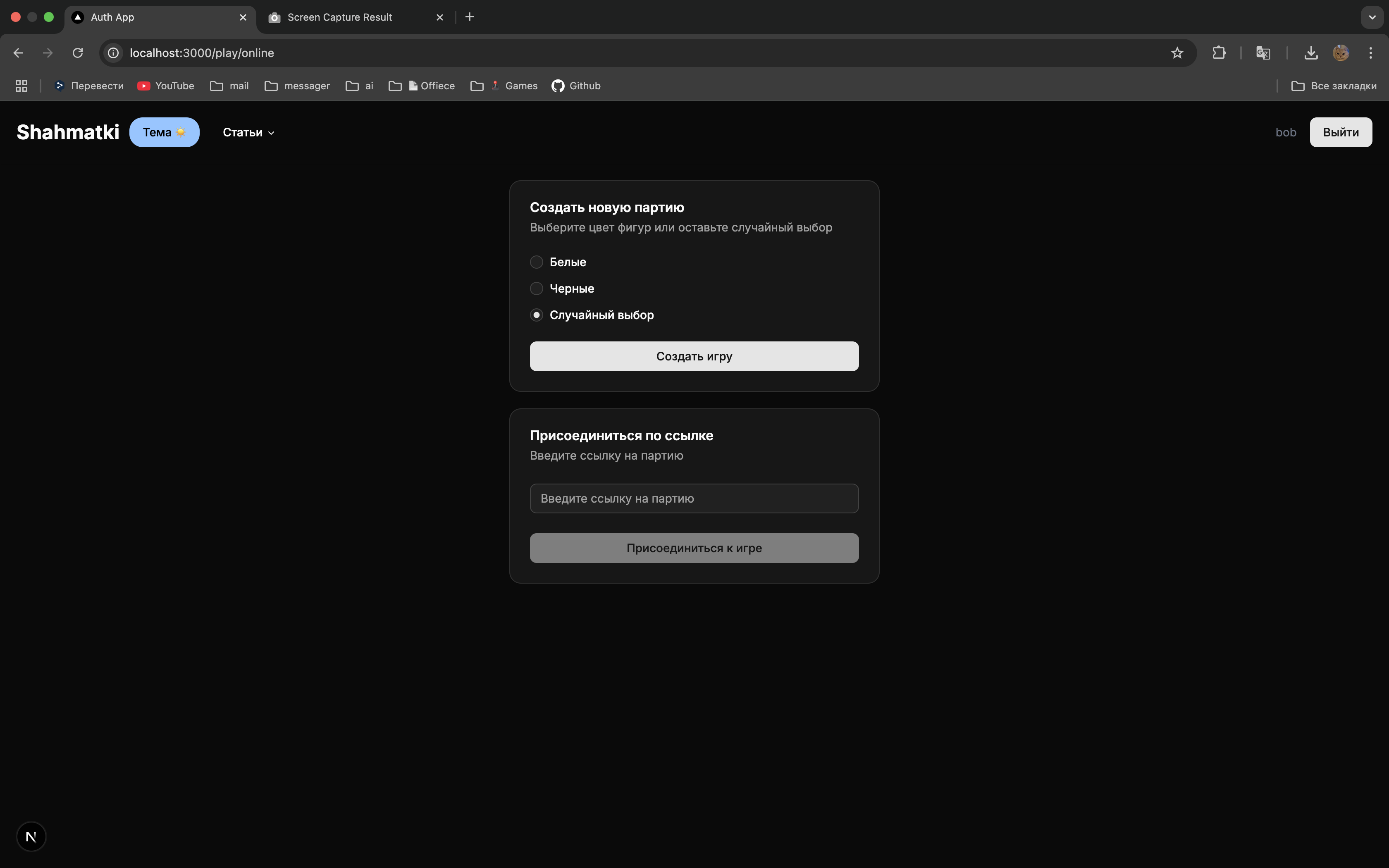


Рисунок 2.10 – Дизайн страницы создания партии онлайн

Пользователь может создать «комнату» чтобы пригласить другого челоека на партию, или присоединиться к уже существвующей.

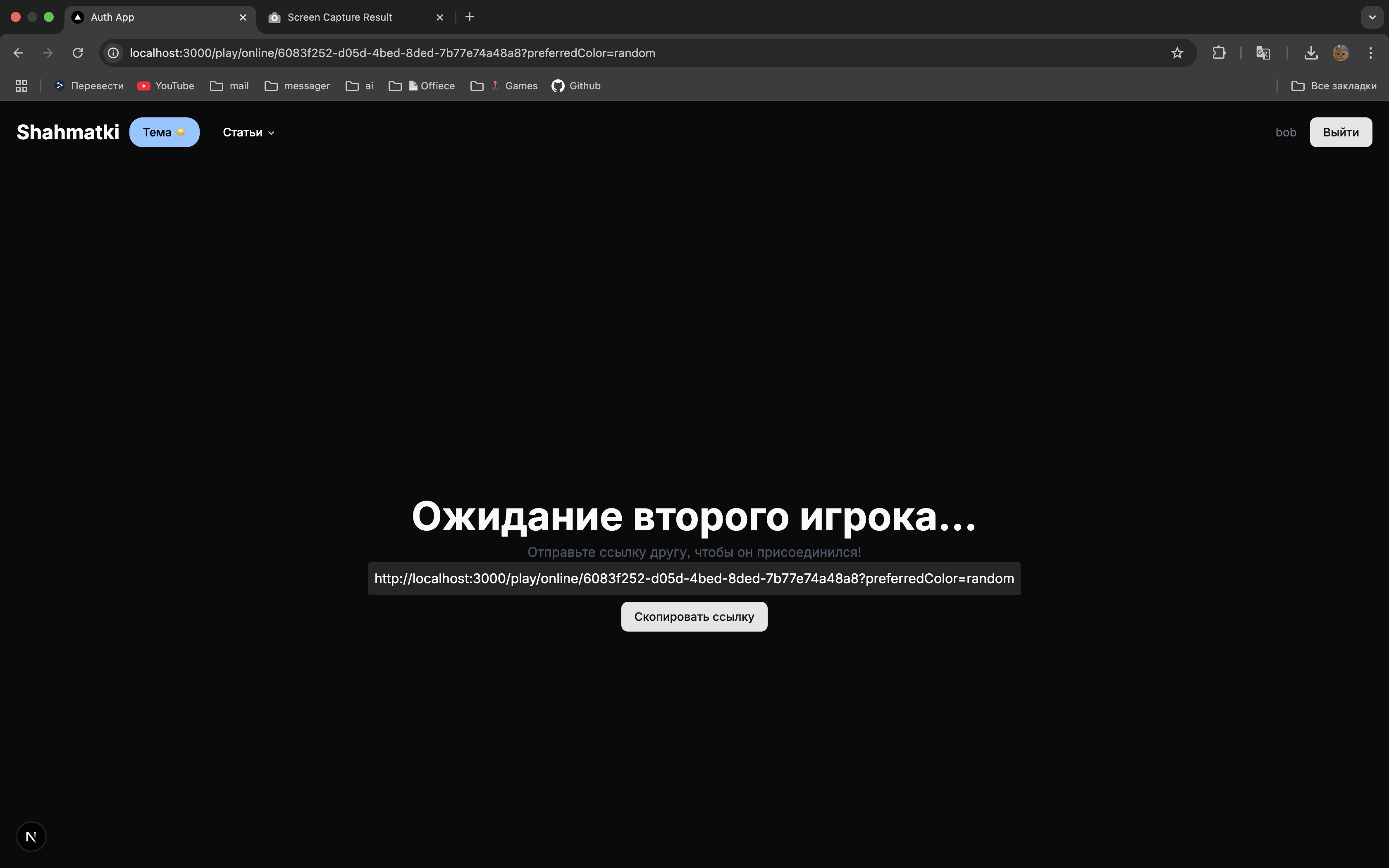


Рисунок 2.11 – Дизайн страницы ожидания второго игрока

Пользователь начавшый партию будет находиться на этом окне пока его соперник не присоедиинится по ссылкке-приглашению. Она изображена на экране, а так же присудствует кнопка для быстрго копирования текста ссылки в буфер обмена устройства пользователя.



Рисунок 2.12 – Дизайн страницы шахмат против другого игрока

При отображении шахматной доски используются контрастные цвета, что помогает пользователям находить доску на экране.

Страницы разработаны с использованием «белого пространства», что позволяет:

* повысить читамость;
* облегчить восприятие визуальной информации на странице;
* повысить «Call to action» (время за которое пользователь поимает что он должен сделать, чтобы выполнить действие на странице).

## Разработка алгоритмов обработки информации

2.4.1 Разработка с учетом выбора технологий

Методы обработки пользовательского ввода зависят от конкретных элементов, с которыми производятся взаимодействия, их можно разделить на 4 категории:

* ссылки. Используются для переходов между страницами сайта и на внешние ссылки. В проекте реализованы через компонент «Link» из пакета «next». Является аналогом HTML тэга «a», используемого для переходов по ссылкам, однако при навигации по страницам веб-приложения не обновляет глобальные ресурсы сайта, такие как: глобальные css атрибуты, метаданные;
* кнопки. Используются для вызова действий. Реализован через «button». Чтобы реализовать вызов функции по нажатию необходимо использовать атрибут onClick в формате «onClick={()=> [название функции]}»;
* формы. Используются для ввода текста и выполнения запросов к API.  При нажатии «submit button» данные из input попадают в соответствующую переменную и передаются в функцию автоматически.
* шахматная доска. Реализуется через компонент «Chessboard». При вызове принимает в себя функции через «props» и при срабатывании события вызывает соответствующие функции.

На рисунке 2.13 приведен код обработки пользовательского ввода.

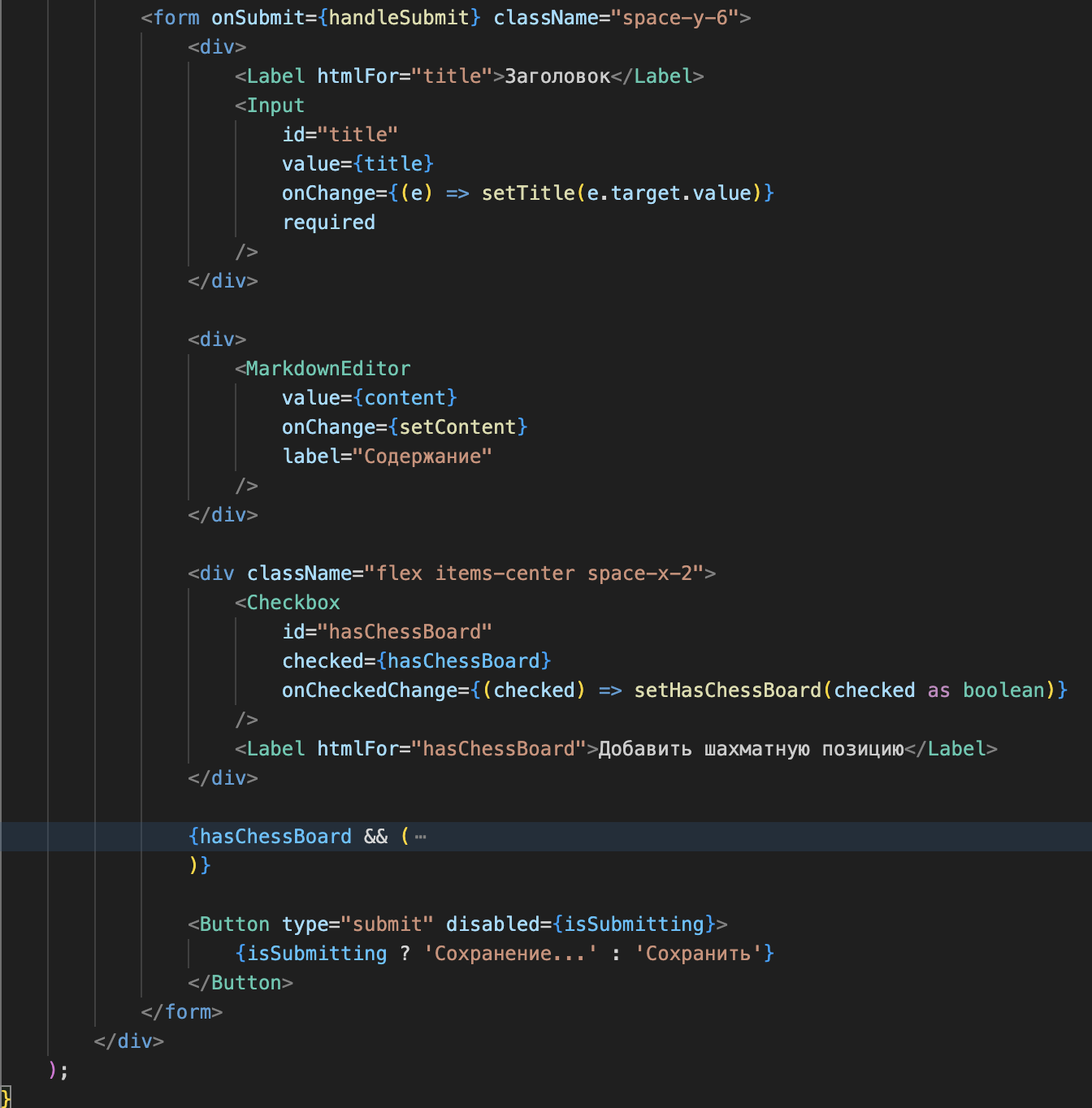


Рисунок 2.13 – Пример обработки пользовательского ввода

В форму передается «onSubmit={handleSubmit}», что означает вызов функции handleSubmit при подтверждении пользовательского ввода. Для подтверждения реализована кнопка с «type="submit"», при нажатии вызывается функция, передаваемая в форму.

2.4.2 Разметка пользовательских статей

Для отображения пользовательских статей используется разметка markdown, это позволяет пользователям оформлять текст руководства. В базе данных хранится запись с контентом статьи. При переходе пользователем на динамический роут отправляется запрос к базе данных. Полученный текст передается в компонент «ReactMarkdown» пакета «react-markdown», который преобразует текст с учетом разметки и оформления. Также пользователь может прикрепить шахматную позицию к руководству, для хранения используется нотация Форсайта – Эдвардса:

* положение фигур со стороны белых. Позиция описывается цифрами и буквами по горизонталям сверху вниз начиная с восьмой горизонтали и заканчивая первой. Расположение фигур на горизонтали записывается слева направо, данные каждой горизонтали разделяются косой чертой /. Белые фигуры обозначаются заглавными буквами. K, Q, R, B, N, P — соответственно белые король, ферзь, ладья, слон, конь, пешка. k, q, r, b, n, p — соответственно чёрные король, ферзь, ладья, слон, конь, пешка. Обозначения фигур взяты из англоязычного варианта алгебраической нотации. Цифра задаёт количество пустых полей на горизонтали, счёт ведётся либо от левого края доски, либо после фигуры (восемь означает пустую горизонталь);
* активная сторона: w — следующий ход принадлежит белым, b — следующий ход чёрных;
* возможность рокировки. k — в сторону королевского фланга (короткая), q — в сторону ферзевого фланга (длинная). Заглавными указываются белые. Невозможность рокировки обозначается «-»;
* возможность взятия пешки на проходе. Указывается проходимое поле, иначе «-»;
* счётчик полуходов. Число полуходов, прошедших с последнего хода пешки или взятия фигуры. Используется для определения применения правила 50 ходов;
* номер хода. Любой позиции может быть присвоено любое неотрицательное значение (по умолчанию 1), счётчик увеличивается на 1 после каждого хода чёрных.

Например, позиция, записанная в базе данных строкой «rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR w KQkq-01» изображена на рисунке 2.14.

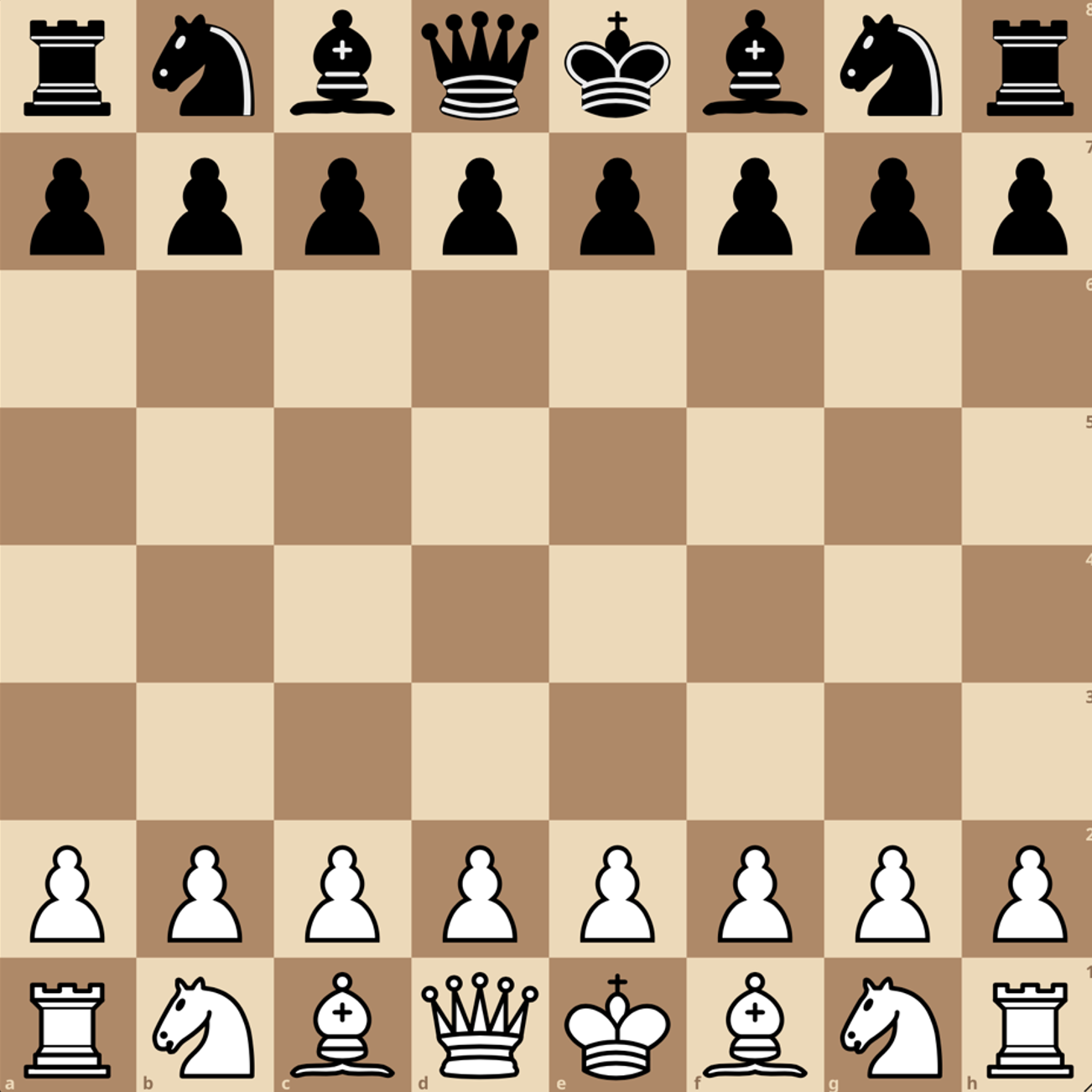


Рисунок 2.14 – Начальная позиция

2.4.3 Взаимодействие между пользователями в реальном времени

В проекте необходимо реаизовать взаимодействие между пользователямси в реальном времени для ведения шахматных партий онлайн. Для этого необходимо реализовать:

* соединение двух пользователей с сервером;
* минимальные задержки между пользователем и сервером;
* минимизировать нагрузки на сеервер.

Сравнение техноголий приведео в таблице 2.1.

Таблица 2.1 – Сравнение технологий клиент-серверного взаимодействия

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Критерий | WebSocket | HTTP polling | HTTP long polling | Server-sent events |
| 1 | 2 | 3 | 4 | 5 |
| Тип связи | Двухсторонняя (Full-duplex) | Односторонняя | Односторонняя | Односторонняя |

Продолжение таблицы 2.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| Эффективность | Высокая благдаря единому соединению | Низкая из-за большого количества заголовков | Низкая из-за времени ожидания и количества запросов | Высокая благодаря единому соедиеннию |
| Серверная нагррузка | Низкая (единое соединение) | Высокая (множество запросов) | Высокая (время ожидания и колиечство запросов) | Низкая (единое одностороннее соединение) |
| Поддержка браузеров | Все | Все | Все | Все кроме  Internet Explorer |

Из сравнения видно преимущество webSocket при необходимости реализации двухстороннего соединения. Изображение примера наботы WebSocket находится на рисунке 2.15.

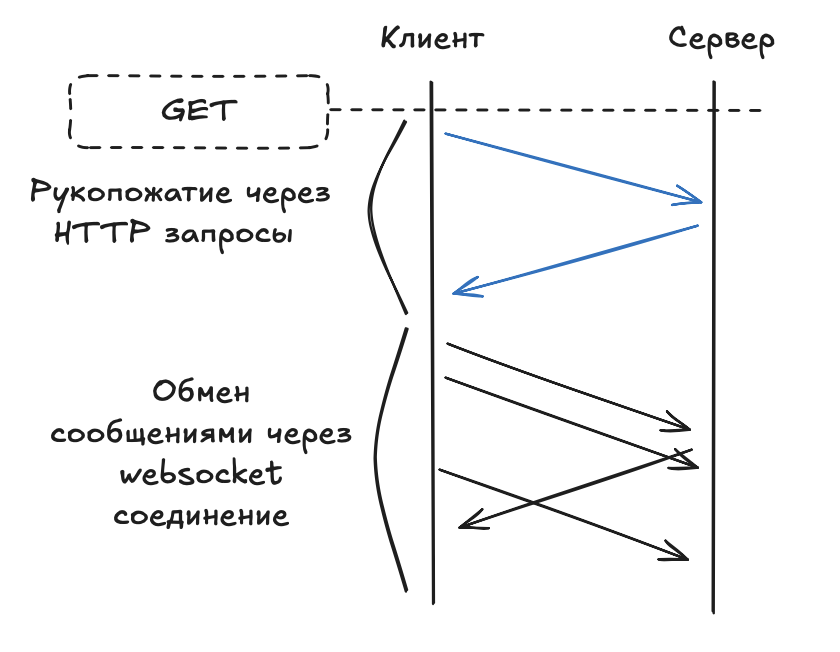


Рисунок 2.15 – Пример работы WebSocket cоединения

WebSocket — это протокол двусторонней связи основанный на TCP, который позволяет устанавливать постоянное соединение между клиентом и сервером. Это делает его эффективным решением для приложений, требующих обмена данными в реальном времени (чаты, онлайн-игры, трейдинг, коллаборативные инструменты).

При создании соедиинения передается:

* url. Соединение создается в отдельном от опсновного приложения потоке для большей устойчивости сервиса;
* roomId. Уникальный идентификатор партии;
* playerId. Уникальный идентификатор пользователя;
* preferredColor. Цвет фигур, выбранный пользователем (белые, черные, случайные).

На рисунке 2.16 приведен код подключения пользователей к WebSocket.

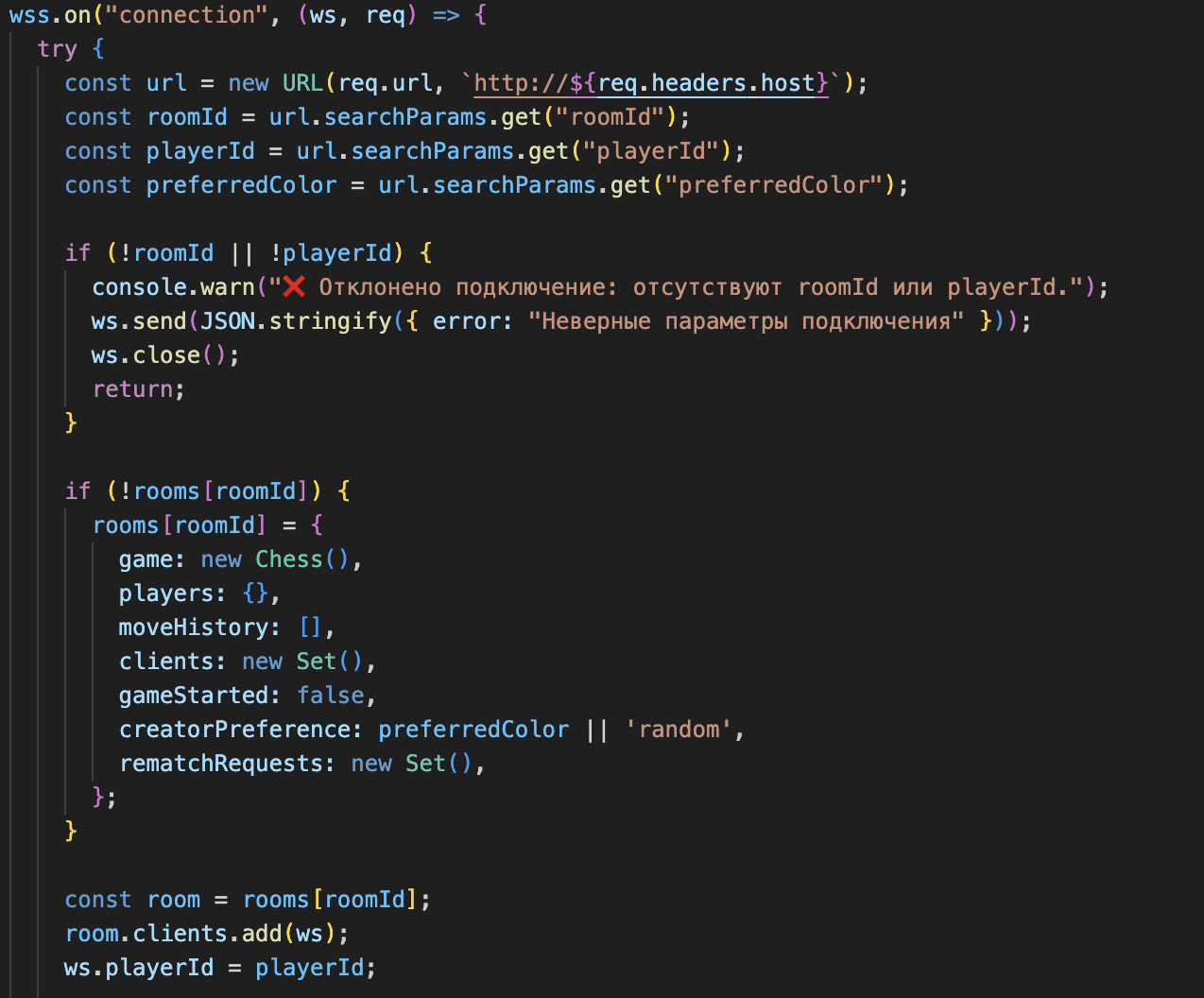


Рисунок 2.16 – Код подключения пользвователей к WebSocket

Пользователь-инициатор комнаты перенаправляется на страницу ожидания, на которой ему предлагается скопировать ссылку-приглошение. Как только второй пользователь перейдет по этой ссылке – создастся запрос на подключение к webSocket и игра между двумя пользователями начнется. Примеры сообщиний между польщователем и webSocket   соединением приведены на  
рисунках

2.17 и 2.18.

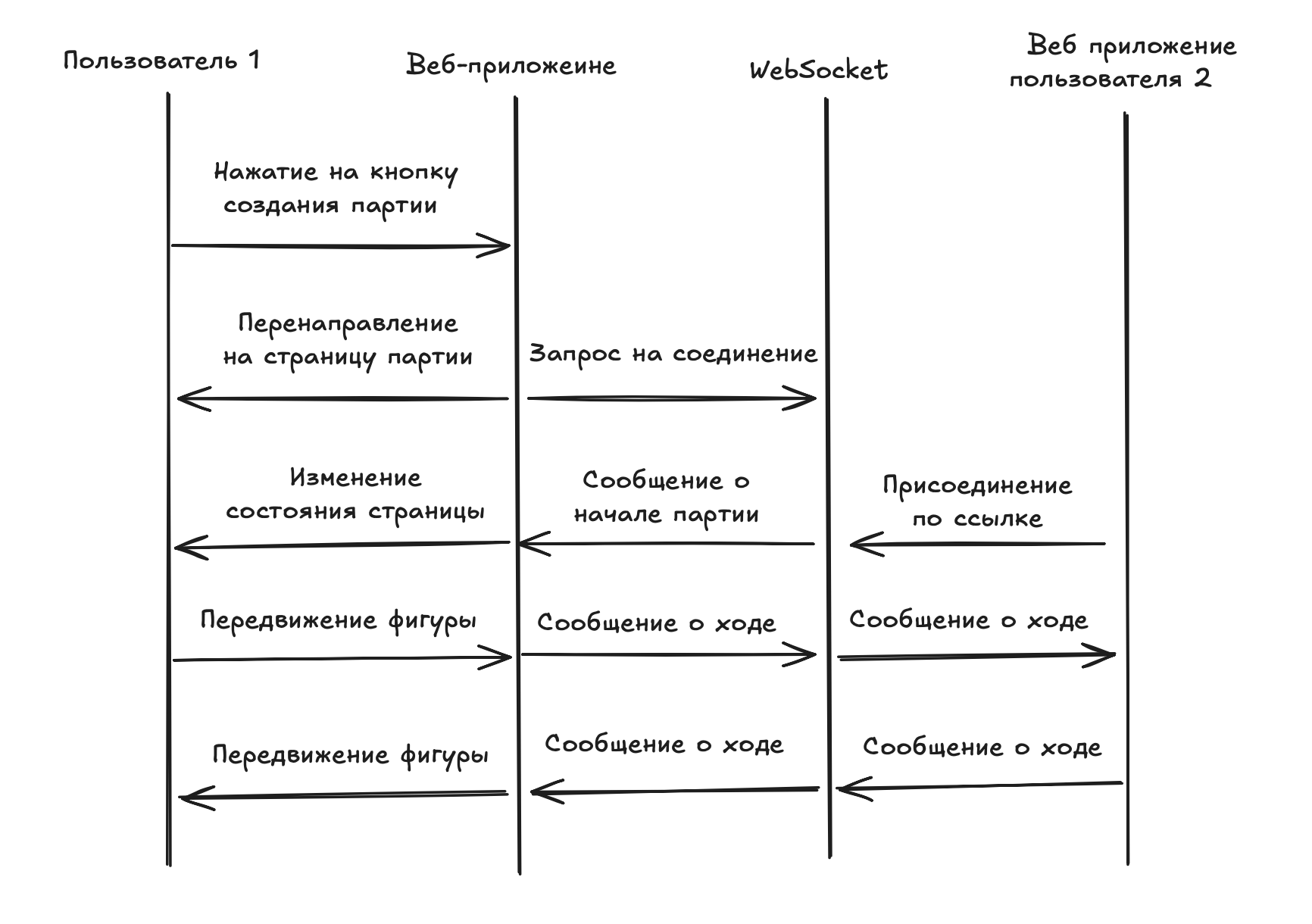


Рисунок 2.17 –  Пример обмена сообщениями при создании комнаты

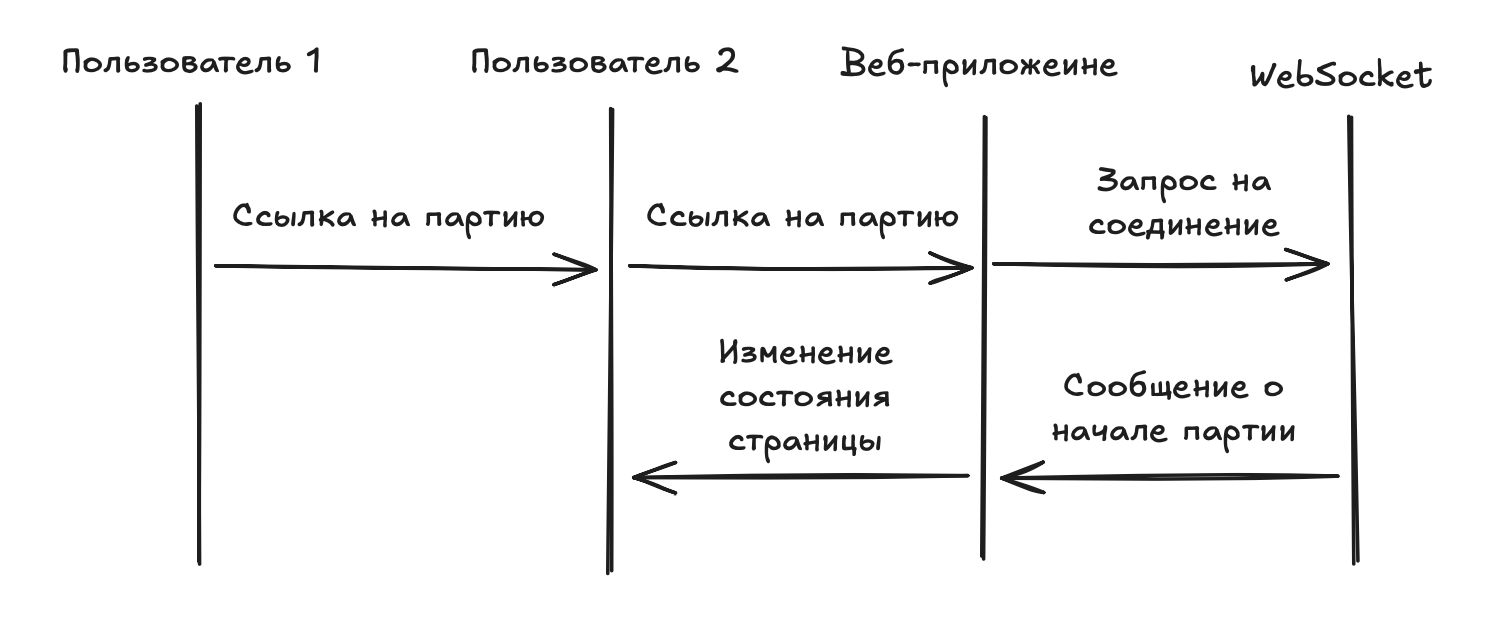


Рисунок 2.18 –  Пример обмена сообщениями присоединении второго пользователя

2.4.4 Верификация пользователя

Для безопасной передачи чувствительных данных пользователю необходимо строго соблюдать процедуры аутентификации и верификации, чтобы предотвратить несанкционированный доступ третьих лиц к учетным записям пользователей.

В Next.js-приложениях для верификации пользовательских запросов используется комбинация технологий:

* JWT (JSON Web Token) - стандартный метод для безопасной передачи данных между сторонами в виде JSON-объекта;
* Auth.js (ранее NextAuth.js) - популярная библиотека для аутентификации в Next.js-приложениях.

Процесс аутентификации пользователя:

1. регистрация и вход пользователя. При успешной регистрации или входе система генерирует два типа токенов:
   1. Refresh Token - долгоживущий токен для продления сессии;
   2. Access Token - краткосрочный токен для доступа к защищенным ресурсам;
2. хранение токенов:
   1. Refresh Token хранится в HTTP-only, Secure, SameSite=Strict cookie с максимальным сроком жизни 30 дней;
   2. Access Token передается в ответе аутентификации и хранится в памяти клиентского приложения (не в localStorage);
3. использование токенов:
   1. при каждом запросе к защищенным API отправляется Access Token в заголовке Authorization;
   2. Если Access Token истек (через 15 минут), клиент использует Refresh Token для получения нового Access Token без необходимости повторного входа;
4. ключевые моменты реализованные в верификации пользователя:
   1. используется PrismaAdapter для связи NextAuth с базой данных;
   2. все операции с пользователями выполняются через Prisma;
   3. пользовательские данные включают связанную сущность attributes для реализации ABAC;
5. поток работы:
   1. пользователь вводит email и пароль на странице /auth;
   2. данные валидируются и ищутся в базе;
   3. при успехе создаётся JWT токен с данными пользователя;
   4. при последующих запросах токен используется для: проверки аутентификации, доступа к данным пользователя, обдонления даных пользователя при запросах;
6. особенности реализации:
   1. постоянная синхронизация с базой: JWT колбэк всегда запрашивает свежие данные пользователя, что гарантирует актуальность информации;
   2. расширенная модель пользователя: В отличие от стандартной конфигурации, здесь пользователь имеет дополнительные атрибуты, которые включаются в токен и сессию;
   3. отладка: Добавлено логирование процесса загрузки атрибутов для отслеживания проблем;
   4. безопасность: Используется секрет из переменных окружения, пароли сравниваются напряму.

## Разработка текстового содержания сайта

Разработка текстового содержания выделятся при разработке такового отдельно от оформления стилей и функционала, как оно велось бы при разработке сайта с использованием HTML напрямую, однако в большинстве современных решений этого не происходит.

При разработке веб-приложений на React функционал и текстовое содержание разрабатываются вместе. В проекте используется Tailwind. Это означает, что стили также интегрированы в текстовое содержание сайт напрямую это изображено на рисунках 2.19 и 2.20.

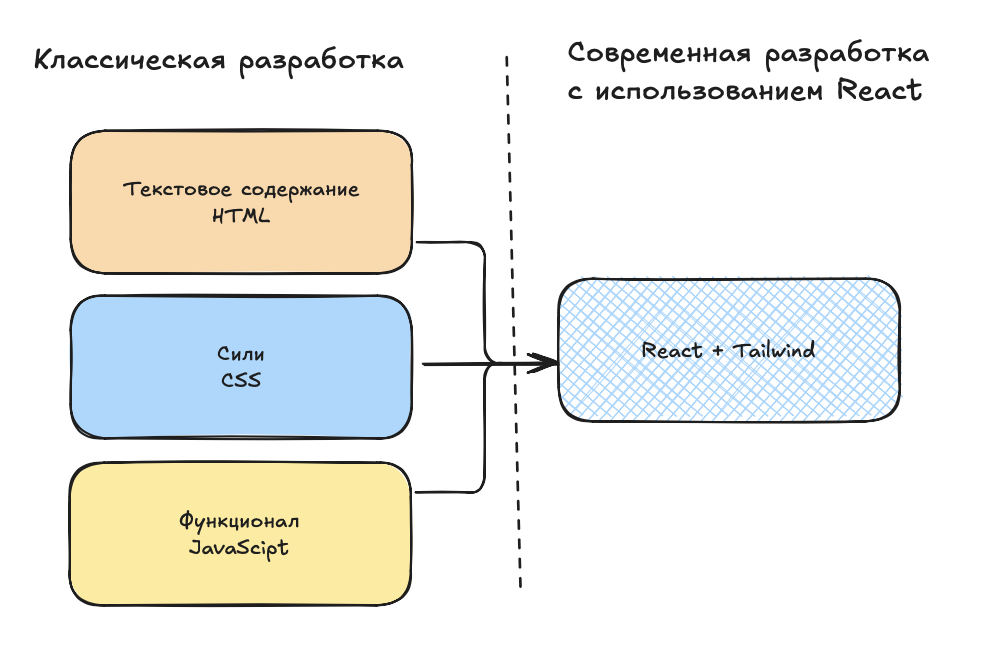


Рисунок 2.19 – Сравнение классической разработки сайтов и React

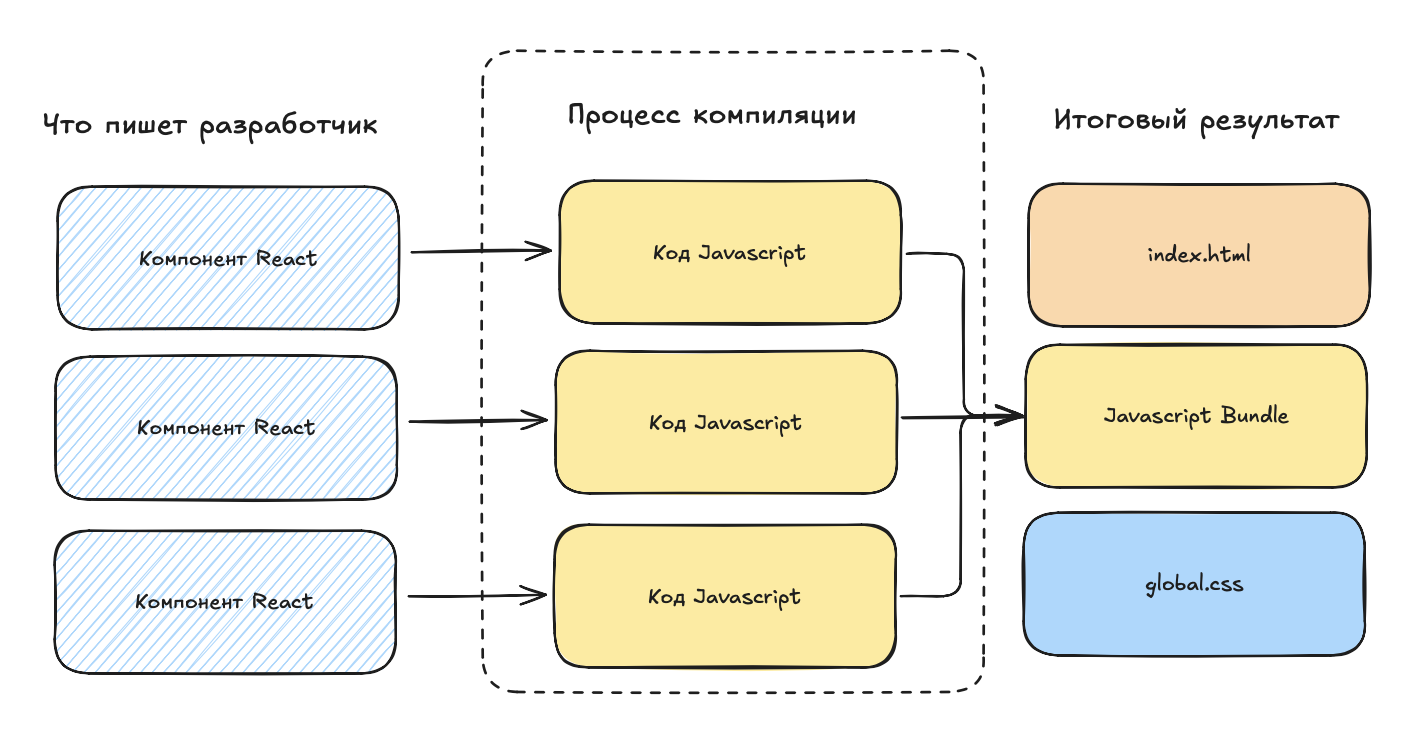


Рисунок 2.20 – Процесс компиляции React

Финальный пользователь получет пакет Javascript и разметку сайта HTML, CSS. Далее браузер клиента исоплняет JavaScript, который генерирует все текстовое содержание и стили сайта. Это самый простой пример, так как современные технологии комплекснее. Процесс компиляции можт включать в себя использоание сторонних сервисов и компиляторов, упрощенная система отображена на рисунке 2.21.

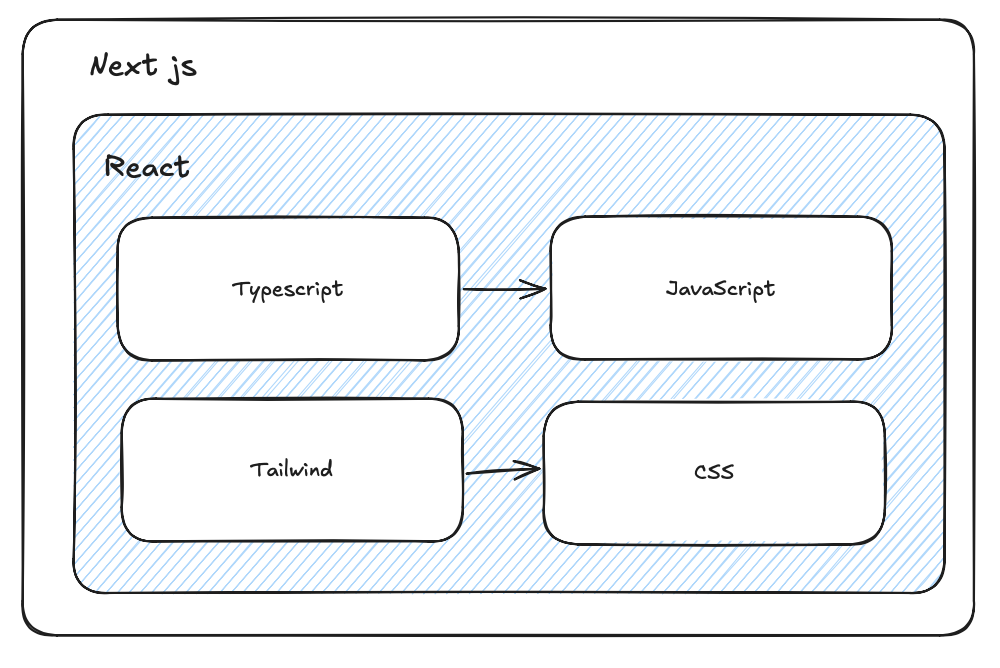


Рисунок 2.21 – Комплескность комписяции современных фреймворков

В проекте используется метафреймвок Next js, основанный на фреймворке React.  Одним из значительных изменений является добавление SSR (рендеринг на стороне сервера), разница между ними отображена на рисунке 2.22.

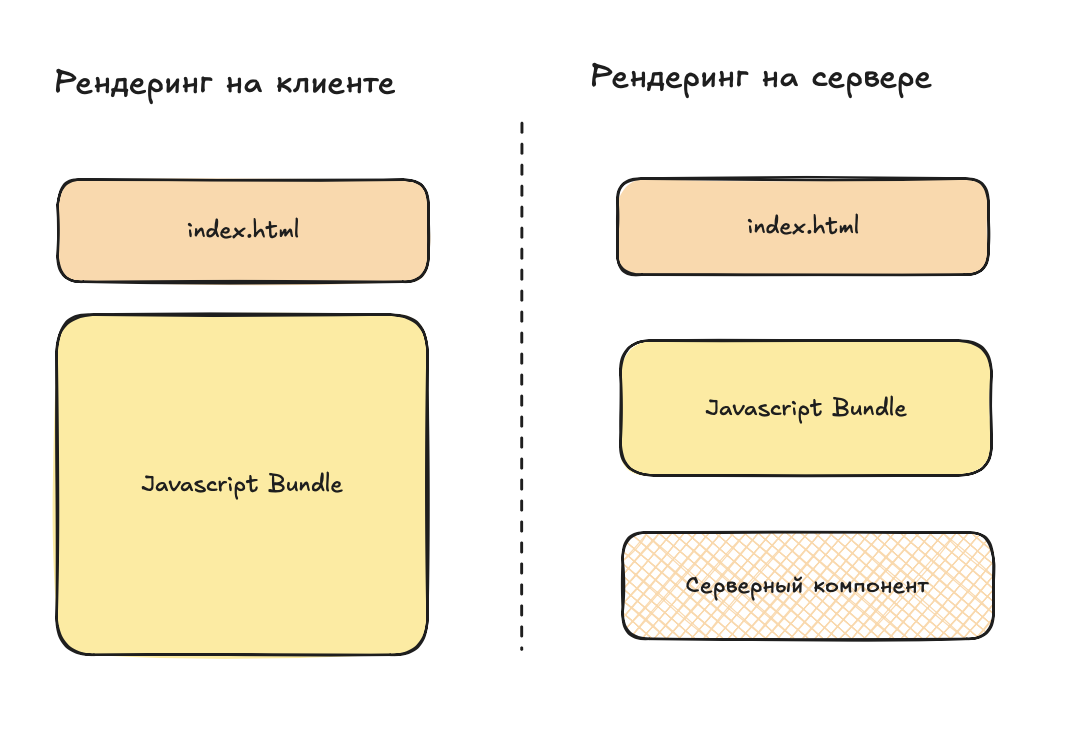


Рисунок 2.22 – Пример разницы между серверным и клиентским рендерингом

Использование серверного рендеринга позволяет уменьшить размер пакета JavaScript так как расчет компонента произошел на сервере. Это позволят снизить нагрузку на устройство и интернет-соединение клиента.

Часть контента зависит от базы данных и ее записей. Как пример – пользовательские статьи. В базе данных хранится запись с контентом статьи. При переходе пользователем на динамический роут отправляется запрос к базе данных. Полученный текст передается в компонент «ReactMarkdown» пакета «react-markdown», который преобразует текст с учетом разметки и оформления. Такая функция позволяет авторам акцентировать внимание на определенных аспектах текстового содержания, упростить чтение пользователям.

## Разработка мультимедийного контента

Для изображения фигур на шахматной доске использована библиотека «react-chessboard», которая в свою очередь использует изображения с Wikipedia. Фигуры различимы между собой, но имеют единую стилистику и не требуют от пользователя дополнительного периода привыкания к их внешнему виду.

При зажатии пользователем основной кнопки мыши на изображении фигуры у него появляется возможность ее перетаскивать в границах окна приложения браузера. При отпускании кнопки если фигура находится над полем, на которое она может совершить ход согласно правилам шахмат – фигура переместится и дальнейшие действия зависят от контекста, в котором происходит взаимодействие с пользователем. Если фигура не может совершать перемещение на клетку – она вернется на позицию до начала перемещения пользователем. Подобная логика взаимодействия с элементами используется для перемещения элементов на рабочем столе компьютера, при перемещении файлов в приложениях компьютера, так что подобное взаимодействие знакомо пользователю и не вызовет дискомфорта в использовании.

Для повышения удобства использования веб-приложения разработано несколько визуальных тем графического оформления сайта. Для отображения того, на что заменится текущая тема пользователям отображается разный текст на кнопке смены визуального оформления. Оба варианта изображены на  
рисунках 2.23 и 2.24.



Рисунок 2.23 – Кнопка смены темы на светлую



Рисунок 2.24 – Кнопка смены светлой темы на темную

Подобный подход использован в нескольких местах сайта для повышения комфорта пользовательского использования программного продукта. Изображения основных интерактивных элементов находятся на риунках 2.25 по 2.27. При наведении пользователем на меню оно открывается и символ переворачивается для отображения сосотояния элемента интерфейса.



Рисунок 2.25 – Закрытое меню

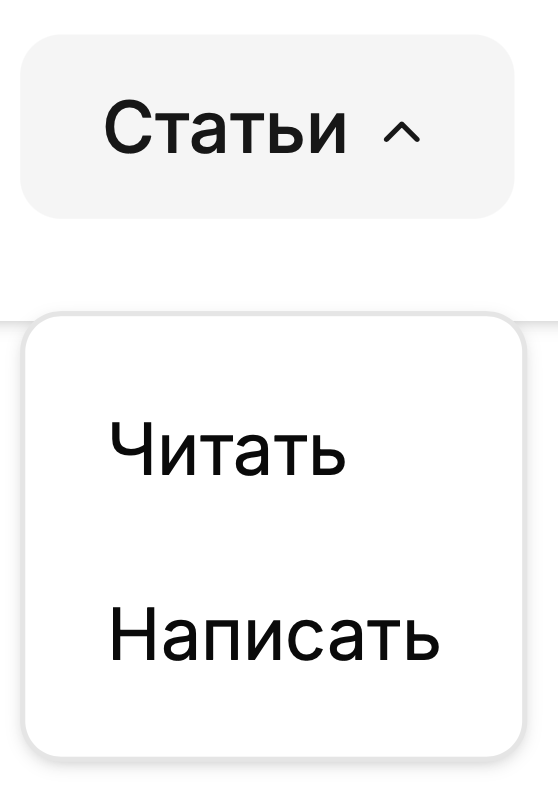


Рисунок 2.26 – Открытоое меню

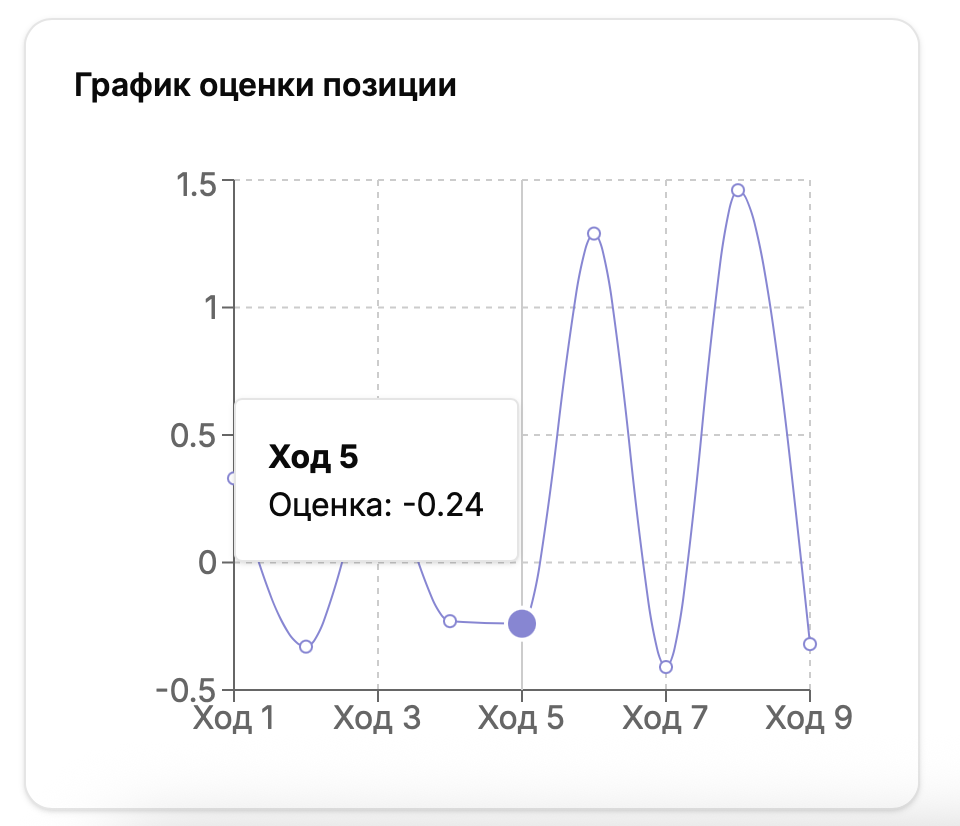


Рисунок 2.27 –  График оценки позиции

График строится с учетом данных пользователя и обновляется в реальном времени. При наведении пользователем на конкретную точку, ему покажется всплывающая подсказка с данными графика.

При наведении пользователем на нажимаемый элемент интерфейса курсор изменится на соответствующий, а также компонент изменится для отображения возможности взаимодействия с пользователем.

Так как веб приложение использует CSS для стилизации компонентов необходимо различать несколько видов свойств:

* дискретные (отображение элемента, шрифт);
* интерполированные (цвет, размер).

В контексте стилизации разметки они различаются возможностями в переходах между состояниями, дискретные таковой не имеют. При наведении на кнопку, на которую пользователь может нажать таковая изменяет свой цвет для индикации о своей интерактивности пользователю, для обеспечения плавности перехода между начальным и конечным цветом используется css transition. Он позволяет плавно изменять значения интерполированных свойств css в заданных пределах времени.

Пример взаимодействия с пользователем через интерактивные элементы приведен ниже на рисунке 2.28.

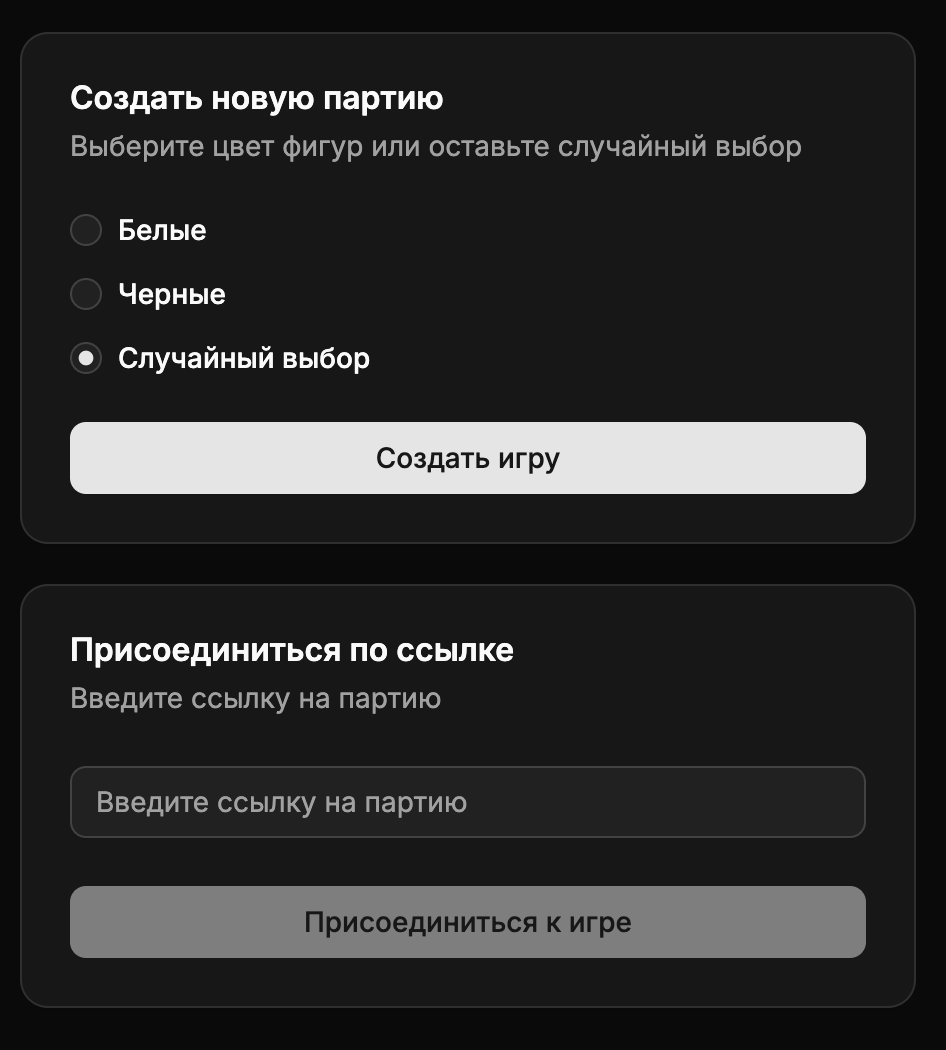


Рисунок 2.28 –  Пример взаимодействия с пользователем через визуальные средства интерфейса

На рисунке 2.28 изображены две интерактивные карточки:

* для создания партии – позволяет пользователю начать новую игровую сессию;
* для присоединения к партии – дает возможность подключиться к уже существующей игре по приглашению.

Обе карточки содержат кнопки, с помощью которых пользователь подтверждает свое действие после заполнения необходимых данных. Однако в случае с присоединением к партии кнопка остается неактивной, пока пользователь не введет корректную ссылку-приглашение. Это сделано для того, чтобы предотвратить преждевременное нажатие и обеспечить корректное выполнение действия.

# Руководство пользователя

## Введение

3.1.1 Область применения

Требования настоящего документа применяются при:

* предварительных комплексных испытаниях;
* опытной эксплуатации;
* приемочных испытаниях;
* промышленной эксплуатации.

3.1.2 Краткое описание возможностей

Сервис «Shahmatki» предназначен для обучения пользователя шахматам на:

* теоретических руководствах, написанных другими пользователями;
* играх против компьютера и реальных оппонентов;
* справочных материалах.

Программный продукт предоставляет следующие возможности:

* ведение партии против искусственного оппонента;
* предоставление справочных материалов для конкретных позиций;
* ведение партии против другого игрока онлайн;
* написание руководств на шахматную тематику;
* предоставление руководств, написанных другими игроками.

3.1.3 Уровень подготовки пользователя

Пользователь веб-приложения для игры в шахматы с элементами обучения должен иметь опыт работы в веб-браузере. Управляться с мышью. Так же для определенного функционала необходимы базовые знания шахмат. Для написания шахматных статей и игры с ботами определенного уровня требуются углубленные познания в шахматах.

## Назначение и условия применения

Справочное веб-приложение для игры в шахматы с элементами обучения представляет собой универсальную платформу, предназначенную для комплексного обучения пользователей шахматной игре. Оно сочетает в себе теоретические материалы, практические упражнения и интерактивные элементы, позволяя игрокам любого уровня – от новичков до продвинутых – осваивать стратегию, тактику и ключевые принципы шахмат в удобном формате. Приложение не только помогает изучать правила и основы игры, но и предлагает инструменты для анализа партий, разбора типовых позиций и совершенствования навыков.

Взаимодействие со справочным приложении становится возможным в любой момент, когда у пользователя возникает желание сыграть в шахматы, повторить теорию или улучшить свои игровые навыки. Единственное необходимое условие — наличие доступа к интернету, что делает приложение доступным. Благодаря веб-формату, для начала использования не требуется установка дополнительного программного обеспечения – достаточно открыть браузер на компьютере, планшете или смартфоне.

Доступ к веб-приложению открыт для всех пользователей, независимо от их возраста, уровня подготовки или географического расположения. Интерфейс разработан с учетом дизайнерских паттернов, обеспечивая простоту навигации и понятное взаимодействие даже для тех, кто только начинает знакомиться с шахматами. При этом система адаптируется под индивидуальные потребности каждого игрока, предлагая персонализированные рекомендации и материалы для эффективного обучения.

## Подготовка к работе

3.3.1 Состав и содержание дистрибутивного носителя данных

Для взаимодействия с веб-приложением, предназначенным для игры в шахматы и обучения, пользователю потребуется веб-браузер с поддержкой стандарта ES2017 (ECMAScript 2017), а также включённой поддержкой JavaScript.

Наиболее стабильная и корректная работа приложения обеспечивается при использовании последней версии Google Chrome или других браузеров на движке Chromium (например, Microsoft Edge, Opera, Brave). Это связано с тем, что данные браузеры полностью поддерживают современные возможности HTML и CSS, а также обеспечивают высокую производительность при выполнении JavaScript-кода.

При использовании альтернативных браузеров (таких как Firefox, Safari или устаревшие версии браузеров) функциональность приложения сохранится, однако возможны незначительные отклонения в визуальном отображении элементов интерфейса. Это может повлиять на удобство взаимодействия, но не нарушит основную логику работы шахматного движка и обучающих модулей.

Рекомендуется регулярно обновлять браузер до актуальной версии для обеспечения максимальной совместимости, производительности и безопасности при использовании веб-приложения.

3.3.2 Порядок загрузки данных и программ

Перед началом взаимодействия с сайтом необходимо открыть веб-браузер и в поисковой строке выполнить соответствующий запрос в поисковой строке браузера.

## Описание операций

3.4.1 Выполняемые функции и задачи

Описание операций, проводимых внутри веб-сервиса изложено   
в таблице 3.1.

Выполнение задач позволит сервису достигнуть поставленных целей и быть востребованным в шахматной среде. Факт выполнения требует значительных навыков в сфере web разработки.

Таблица 3.1 – Описание операций вер-сервиса

|  |  |  |
| --- | --- | --- |
| Функции | Задачи | Описание |
| Запуск игры в шахматы против искусственного оппонента | Практика навыка игры в шахматы против бота | Открытие окна с шахматной доской, предварительно пользователю будет предложено выбрать цвет фигур и сложность оппонента |
| Запуск игры в шахматы против другого пользователя в реальном времени | Практика навыка игры в шахматы | Пользователь получает ссылку-приглашение. Как только она будет открыта другим пользователем откроется окно с шахматной доской и возможностью поочередно передвигать фигуры. По окончании партии пользователи могут предложить реванш, скачать запись партии |
| Предоставление справочных материалов | Получение справочных данных для выбранной позиции | Пользователю открывается шахматная доска, при перемещении фигур шахматный компьютер предлагает лучший ход, а также отображается название шахматной позиции на двух языках с возможностью дальнейшего изучения за пределами сервиса |
| Редактор пользовательских статей | Предоставление другим пользователям теоретических знаний на выбранную тему | Пользователь может писать текстовое содержание статьи и оформлять его согласно стандарту «markdown», при необходимости пользователь может прикрепить к руководству шахматную позицию для отображения ее читателям |
| Предоставление доступа к пользовательским руководствам | Чтение руководств по игре, написанных другими пользователями | Пользователь может выбрать статью из списка предложенных для ознакомления, по нажатии на ссылку откроется окно с содержанием руководства |

3.4.2 Описание операций технологического процесса обработки данных, необходимых для выполнения задач

Операция 1 – настройка партии и искусственного оппонента:

1. условия: сайт открылся исправно, пользователь находится на странице «/play/bot»;
2. подготовительные действия не требуются;
3. основные действия:
   1. выбрать цвет, которым пользователь будет играть в процессе партии;
   2. выбрать сложность шахматного бота из предложенных;
4. заключительные действия: подтвердить выбор, нажатием на соответствующую кнопку.

Операция 2 – ведение парти против искусственного оппонента:

1. условия: сайт открылся исправно, пользователь находится на странице «/play/bot»;
2. подготовительные действия: настройка партии и искусственного оппонента;
3. основные действия: в соответствии с шахматными правилами перемещать фигуры по изображенной доске, предварительно нажав на фигуру основной кнопкой мыши.
4. заключительные действия: нет.

Операция 3: создание «комнаты» для игры против другого пользователя в реальном времени:

1. условия: сайт открылся исправно, пользователь находится на странице «/play/online»;
2. подготовительные действия: настройка партии и искусственного оппонента;
3. основные действия: 1 выбор цвета фигур, за которые будет играть пользователь в процессе партии;
4. заключительные действия: подтверждение выбора.

Операция 4 – присоединение к «комнате» для игры против другого пользователя в реальном времени:

* условия: сайт открылся исправно, пользователь находится на странице «/play/online»;
* подготовительные действия: получение ссылки-приглашения;
* основные действия: ввод ссылки в соответствующее поле на странице;
* заключительные действия: подтверждение ввода.

Операция 5 – игра против другого пользователя в режиме реального времени:

* условия: сайт открылся исправно, оба пользователя находятся на странице «/play/online/[код комнаты]»;
* подготовительные действия: создание комнаты, присоединение второго игрока по ссылке-приглашению;
* основные действия: в соответствии с шахматными правилами перемещать фигуры по изображенной доске, предварительно нажав на фигуру основной кнопкой мыши;
* заключительные действия: по завершении партии пользователям отобразится меню с итогом партии.

Операция 6 – редактирование новой пользовательской статьи:

1. условия: сайт открылся исправно, пользователь находится на странице «/articles/new»;
2. подготовительные действия: получение ссылки-приглашения;
3. основные действия:
   1. ввод текста статьи в поле ввода;
   2. оформление текста в соответствии с форматом «markdown» при помощи соответствующих кнопок меню;
4. заключительные действия: подтверждение ввода.

Операция 7 – редактирование пользовательской статьи:

1. условия: сайт открылся исправно, пользователь находится на странице «/articles/[идентификатор статьи]», пользователь должен быть автором статьи;
2. подготовительные действия: нажатие на кнопку редактирования;
3. основные действия:
   1. ввод текста статьи в поле ввода;
   2. оформление текста в соответствии с форматом «markdown» при помощи соответствующих кнопок меню;
4. заключительные действия: подтверждение ввода.

Операция 8: ознакомление со справочной информацией о позиции:

* условия: сайт открылся исправно, пользователь находится на странице «/analysis»;
* подготовительные действия не требуются;
* основные действия: отображение на шахматной доске расположения фигур;
* заключительные действия: пользователю отобразится оценка позиции, график оценки, название позиции на двух языках, если она есть в базе данных.

# Тестирование программного продукта

## Выбор стратегии тестирования

Для проверки работоспособности системы был выбран метод тестирования по стратегии черного ящика.

Этот подход к тестированию фокусируется на поведении системы независимо от ее внутренней структуры и реализации. В рамках этого метода тестировщик осуществляет ввод данных, а затем анализирует вывод системы, рассматривая его как часть процесса проверки.

Такой подход позволяет выявить, как система реагирует на ожидаемые и неожиданные действия пользователя, оценить время реакции, сложность использования и проблемы с надежностью.

Тестирование по стратегии черного ящика является эффективным подходом, поскольку позволяет проверить работу системы в целом, аналогично тому, как пользователи ожидают получить соответствующий ответ на свои запросы, не вдаваясь в детали ее внутренней работы.

При этом осуществляется исследование всех компонентов системы, включая интерфейс пользователя, веб-сервер, базу данных и другие зависимые и интегрированные системы.

## Разработка сценариев и протоколов тестирования

Сценарий 1 – Получение ссылки-приглашения. Для выполнения данного сценария необходимо проверить:

* переход между страницами;
* получение пользовательского ввода;
* присоединение к WebSocket;
* генерацию ссылки а основе uuid;
* копирование ссылки при помощи api браузера.

Тестирование сценария приведено на таблице 4.1.

Таблица 4.1 – Протокол создания «комнаты»

|  |  |
| --- | --- |
| Наименование | Описание |
| Код | 1 |
| Приоритет тестирования | Высокий |
| Название тестирования | Получение ссылки для ведения партии онлайн |
| Шаги тестирования | 1. Вход на страницу «play/online» 2. Выбор цвета фигур 3. Подтверждение выбора 4. Перемещение на страницу «play/online [идентификатор комнаты]» |
| Данные тестирования | 1. Ссылка на присоединение к онлайн партии |
| Ожидаемый результат | Логирование о создании комнаты и присоединении пользователя |
| Фактический результат | Логирование о создании комнаты и присоединении пользователя |

Сценарий 2 – Использование ссылки-приглашения. Для выполнения данного сценария необходимо проверить:

* переход между страницами;
* получение пользовательского ввода;
* присоединение к WebSocket.

Тестирование сценария приведено на таблице 4.2.

Таблица 4.2 – Протокол присоединения к комнате

|  |  |
| --- | --- |
| Наименование | Описание |
| Код | 2 |
| Приоритет тестирования | Высокий |
| Название тестирования | Получение ссылки для ведения партии онлайн |
| Шаги тестирования | 1. Вход на страницу «play/online» 2. Ввод ссылки 3. Подтверждение выбора 4. Перемещение на страницу «play/online [идентификатор комнаты]» |
| Данные тестирования | 1. Ссылка на присоединение к онлайн партии |
| Ожидаемый результат | Логирование о присоединении пользователя |
| Фактический результат | Логирование о присоединении пользователя |

Сценарий 3 – Ведение партии против бота. Для выполнения данного сценария необходимо проверить:

* переход между страницами;
* получение пользовательского ввода;
* работу шахматного бота.

Тестирование сценария приведено на таблице 4.3.

Таблица 4.3 – Протокол ведения игры в шахматы против бота

|  |  |
| --- | --- |
| Наименование | Описание |
| Код | 2 |
| Приоритет тестирования | Высокий |
| Название тестирования | Ведение партии против бота |
| Шаги тестирования | 1. Вход на страницу «play/bot» 2. Настройка партии и сложности 3. Подтверждение выбора 4. Совершение хода согласно шахматным правилам |
| Данные тестирования | нет |
| Ожидаемый результат | Шахматный бот совершает ходы согласно шахматным правилам |
| Фактический результат | Шахматный бот совершает ходы согласно шахматным правилам |

Сценарий 4 – Создание пользовательской статьи. Для выполнения данного сценария необходимо проверить:

* переход между страницами;
* получение пользовательского ввода;
* сохранение записи в базе данных.

Тестирование сценария приведено на таблице 4.4.

Таблица 4.4 – Протокол создания пользовательской статьи

|  |  |
| --- | --- |
| Наименование | Описание |
| 1 | 2 |
| Код | 4 |
| Приоритет тестирования | Высокий |

Продолжение таблицы 4.4

|  |  |
| --- | --- |
| 1 | 2 |
| Название тестирования | Создание пользовательской статьи |
| Шаги тестирования | 1. Вход на страницу articles/new» 2. Заполнение полей ввода 3. Подтверждение выбора |
| Данные тестирования | 1. Пользовательская статья |
| Ожидаемый результат | Создание записи о партии в базе данных |
| Фактический результат | Создание записи о партии в базе данных |

Сценарий 5 – Редактирование статьи. Для выполнения данного сценария необходимо проверить:

* переход между страницами;
* получение пользовательского ввода;
* изменение записи в базе данных.

Тестирование сценария приведено на таблице 4.5.

Таблица 4.5 – Протокол изменения пользовательской статьи

|  |  |
| --- | --- |
| Наименование | Описание |
| Код | 5 |
| Приоритет тестирования | Высокий |
| Название тестирования | Изменение пользовательской статьи |
| Шаги тестирования | 1. Вход на страницу articles/ [код статьи]» 2. Заполнение полей ввода 3. Подтверждение изменений |
| Данные тестирования | Пользовательская статья |
| Ожидаемый результат | Изменение записи о партии в базе данных |
| Фактический результат | Изменение записи о партии в базе данных |

Сценарий 6 – Удаление статьи. Для выполнения данного сценария необходимо проверить:

* переход между страницами;
* получение пользовательского ввода;
* удаление записи в базе данных.

Тестирование сценария приведено на таблице 4.6.

Таблица 4.6 – Протокол удаления пользовательской статьи

|  |  |
| --- | --- |
| Наименование | Описание |
| Код | 6 |
| Приоритет тестирования | Высокий |
| Название тестирования | Удаление пользовательской статьи |
| Шаги тестирования | 1. Вход на страницу articles/ [код статьи]» 2. Нажатие на кнопку |
| Данные тестирования | Пользовательская статья |
| Ожидаемый результат | Удаление записи о партии в базе данных |
| Фактический результат | Удаление записи о партии в базе данных |

Сценарий 7 – Получение справочной информации о позиции. Для выполнения данного сценария необходимо проверить:

* переход между страницами;
* получение пользовательского ввода;
* обработка пользовательского ввода.

Тестирование сценария приведено на таблице 4.1.

Таблица 4.7 – Протокол анализа позиции

|  |  |
| --- | --- |
| Наименование | Описание |
| Код | 7 |
| Приоритет тестирования | Высокий |
| Название тестирования | Получение справочной информации о позиции |
| Шаги тестирования | 1. Вход на страницу «analysis» 2. Совершение хода 3. Получение информации |
| Данные тестирования | Справочная информация о позиции |
| Ожидаемый результат | Вывод названия позиции и лучшего хода |
| Фактический результат | Вывод названия позиции и лучшего хода |

Сценарий 8 – Создание аккаунта пользователя. Для выполнения данного сценария необходимо проверить:

* переход между страницами;
* получение пользовательского ввода;
* обработка пользовательского ввода.

Тестирование сценария приведено на таблице 4.8.

Таблица 4.8 – Протокол создания аккаунта

|  |  |
| --- | --- |
| Наименование | Описание |
| Код | 7 |
| Приоритет тестирования | Высокий |
| Название тестирования | Создание пользовательского аккаунта |
| Шаги тестирования | 1. Вход на страницу «auth» 2. Выбор опции создания нового аккаунта 3. Заполнение пользовательских данных |
| Данные тестирования | Личные данные пользователей |
| Ожидаемый результат | Создание записи о пользователе |
| Фактический результат | Создание записи о пользователе |

# Заключение

В ходе выполнения дипломного проекта разработано веб-приложение для игры в шахматы с элементами обучения. Основной целью проекта являлась популяризация шахмат и повышение доступности обучающих материалов для широкой аудитории. Разработанная система решает актуальные задачи, связанные с обеспечением удобного доступа к шахматным знаниям и инструментам, а также способствует повышению осведомлённости пользователей о различных аспектах игры.

В рамках проекта создан удобный и современный интерфейс для администраторов, который обеспечивает быстрый и точный контроль доступов, управление контентом и мониторинг активности пользователей. Это позволяет эффективно администрировать платформу и оперативно реагировать на потребности аудитории.

В процессе работы над проектом выполнены следующие ключевые задачи: проведён детальный анализ существующих решений, выявлены их ограничения в контексте современных потребностей пользователей; спроектирована структура системы с выделением ключевых функциональных модулей и их взаимосвязей; подобраны и реализованы оптимальные программные средства, включая интуитивно понятный веб-интерфейс и надежную серверную часть.

Таким образом, внедрение разработанного веб-приложения позволит существенно увеличить число активных игроков, сформировать обширную базу обучающих материалов, а также обеспечить высокую доступность и простоту использования для пользователей с разным уровнем подготовки.

Изучение предметной области проходило при активном участии тренера сборной Новосибирской области по шахматам Владимирова Михаила Яковлевича. Его экспертные советы помогли точно определить задачи, необходимые для реализации проекта, ориентированного на обучение. В процессе исследования и совершенствования навыков игры были выявлены ключевые потребности целевой аудитории, что позволило создать по-настоящему полезный продукт.

При разработке визуального оформления использовалась дизайн-система «Geist» от Vercel, которая придала проекту минималистичный, современный и запоминающийся внешний вид. Это способствует удобству взаимодействия пользователей с платформой и улучшает общее восприятие интерфейса.

Написание кода велось на метафреймворке «Next.js» от Vercel, что значительно упростило интеграцию визуальных решений, ускорило процесс разработки и позволило оптимизировать проект для конечных пользователей. Благодаря этому обеспечивается высокая производительность, быстрая загрузка и отзывчивость интерфейса.

# Библиография

Нормативно-правовые акты:

1. ГОСТ Р 2.105-2019.ЕСКД. Общие требования к текстовым документам. –Москва:Стандартинформ, 2019. – 36 с;
2. ГОСТ Р 7.0.100-2018. Библиографическая запись.

Библиографическое описание. Общие требования и правила составления национальный стандарт Российской Федерации : издание официальное : 35 утвержден и введен в действие Приказом Федерального агентства по техническому регулированию и метрологии от 3 декабря 2018 года №1050-ст : введен впервые : дата введения 2019-07-01/ подготовлен Информационным агентством России, филиалом Российской книжной палаты, Российской государственной библиотекой, Российской национальной библиотекой. – Москва :Стандартинформ, 2018. – 65 c.

Электронные ресурсы:

1. Кайл Симпсон, Вы пока еще не знаете JS} Познакомьтесь, JavaScript // Яндекс книги [сайт] – URL: <https://books.yandex.ru/books/EZFARflo> (дата обращения: 06.07.2024;
2. Шелли Пауэрс, Адам Д. Скотт, Мэтью МакДоналд, JavaScript. Рецепты для разработчиков // Яндекс книги  [сайт] – URL: <https://books.yandex.ru/books/htEqBlMm> (дата обращения: 08.08.2024);
3. Трэйси Осборн, Веб-дизайн для недизайнеров – Текст электронный // Яндекс книги [сайт] – URL: <https://books.yandex.ru/books/nTn6cwvg> (дата обращения: 18.09.2024);
4. Стоян Стефанов, React. Быстрый старт – Текст электронный // Яндекс книги [сайт] –  URL: <https://books.yandex.ru/books/MRULy4Cu> (дата обращения: 24.09.2024);
5. Бэнкс Алекс, Ева Порселло, React: современные шаблоны для разработки приложений // Яндекс книги [сайт] – URL: <https://books.yandex.ru/books/LlcjwvGM> (дата обращения: 13.11.2024);
6. Яков Файн, Антон Моисеев, Typescript быстро  // Яндекс книги [сайт] – URL: <https://books.yandex.ru/books/BjLK3QAY> (дата обращения 9.01.2025);
7. Борис Черный, Профессиональный TypeScript. Разработка масштабируемых JavaScript-приложений // Яндекс книги [сайт] – URL: <https://books.yandex.ru/books/XQje6ux1> (дата обращения 19.01.2025);
8. Арнольд Роббинс, Элберт Ханна, Изучаем vi и Vim. Не просто редакторы // Яндекс книги [сайт] – URL: <https://books.yandex.ru/books/U4fGdTaO> (дата обращения: 15.02.2025);
9. Chessboard [Электронный ресурс]. – Документация по React Chessboard. – URL: <https://react-chessboard.vercel.app/?path=/story/chessboard--default> (дата обращения 19.03.2025);
10. TailwindCSS [Электронный ресурс]. – Документация по Tailwind CSS – URL: <https://tailwindcss.com/> (дата обращения 03.04.2025);
11. React dev [Электронный ресурс]. – Документация по React – URL: <https://react.dev/> (дата обращения 04.01.2025);
12. Next js [Электронный ресурс]. – Документация по Next js – URL: <https://nextjs.org/> (дата обращения 19.01.2025);
13. Shadcn UI [Электронный ресурс]. – Документация по Shadcn UI – URL: <https://ui.shadcn.com/> (дата обращения 29.01.2025);
14. Prisma [Электронный ресурс]. – Документация по Prisma – URL: <https://www.prisma.io/> (дата обращения 30.03.2025).