Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)

Утверждаю:\_\_\_\_\_\_\_\_\_\_\_\_\_.

“ ” 2023 г.

**Курсовая работа**

Вариант №39

Техническое задание

5 листов

Исполнители:

Гордеев Н.А. гр. ИУ5-64Б

Федотова А.Д. гр. ИУ5-64Б

Стельмах Я.С. гр. ИУ5-64Б

Москва 2023 г.

Оглавление

[1. Наименование: 3](#_Toc130911881)

[2. Основание для разработки: 3](#_Toc130911882)

[3. Исполнители: 3](#_Toc130911883)

[4. Цель разработки: 3](#_Toc130911884)

[5. Технические требования. 3](#_Toc130911885)

[6. Функциональные требования 4](#_Toc130911886)

[7. Этапы разработки 5](#_Toc130911887)

[8. Техническая документация, предъявляемая по окончанию работы: 5](#_Toc130911888)

[9. Порядок приемки работы: 5](#_Toc130911889)

[10. Дополнительные условия: 5](#_Toc130911890)

# Наименование:

Online игра «Шашки»

# Основание для разработки:

Основанием для разработки является учебный план МГТУ им. Баумана кафедры ИУ5 на 6 семестр.

# Исполнители:

Исполнителями являются студенты МГТУ им. Н.Э. Баумана группы ИУ5-64Б:

* Федотова А. Д. (Frontend)
* Стельмах Я. С. (WebSocket + БД + интеграция)
* Гордеев Н.А. (Backend)

# Цель разработки:

Разработать Online игру «Шашки» в виде веб-приложения для игры в шашки по сети с другим человеком, либо с ИИ. Должен вестись учет статистики для каждого пользователя. Пользователь может выбрать игру как с другим пользователем, так и с ИИ. При регистрации нового пользователя должна производиться проверка указанной почты во избежании создания фиктивных аккаунтов, а также для предотвращения создания аккаунта с использованием чужой почты.

# Технические требования.

Каждый член команды реализует свою часть проекта со своим набором технологий:

* Frontend – TypeScript + React
* Backend – Django(DRF) + MySQL + Redis
* WebSocket – пакет soket.io

Для демонстрации работы программы необходимо:

* Любой браузер, поддерживающий современные функции JavaScript
* Стабильное интернет-соединение (512 Кб/сек).
* Двухъядерный процессор (Core 2 Duo E4600/Athlon 64 X2 5600+)
* Периферийные устройства: монитор, клавиатура и компьютерная мышь

# Функциональные требования

Для авторизации используются JWT-токены из библиотеки djangorestframework-simplejwt

* 1. **Frontend**
     1. На главной странице приложения должны быть следующие функциональные кнопки:
        + Авторизация
        + Регистрация
        + Личный кабинет
        + Правила
        + Найти игру
        + Общий рейтинг игроков
     2. Страницы авторизации и регистрации.
     3. Страница правил игры «Шашки». На странице должны быть подробно расписаны правила игры в «Шашки» и правила начисления очков рейтинга.
     4. Страница поиска игр. Функциональные возможности:
        + Возможность присоединиться к игре из списка
        + Создать новую игру
     5. Страница игры. Функциональный возможности:
        + Возможность передвигать пешки по игровому полю
        + Просмотр всех предыдущих ходов в специальной таблице
        + Возможность сдаться
        + Уведомление о победе/проигрыше/ничье
     6. Страница общего рейтинга игроков. На странице должна быть таблица со следующими столбцами: имя пользователя, кол-во игр всего, кол-во победных игр, кол-во очков рейтинга. Должна быть реализована возможность проводить сортировку по любому из доступных столбцов по возрастанию или по убыванию.
     7. Страница личного кабинета. На странице должна быть информация о кол-ве побед пользователя, кол-во проведенных и очков рейтинга. Также должна быть возможность посмотреть результаты предыдущих игр.
  2. **Backend**
     1. Все запросы проверяются на наличие JWT токена
     2. Пользователь (GET, POSTPUTCH, DELETE)
        + /user /registration/POST (email\*, username\*, password\*) – создание нового пользователя
        + /user/login/POST (email\*, password\*) – авторизация пользователя
        + /user/logout/POST (refresh \*) – деавторизация пользователя
        + /user/refresh-token/POST (refresh) – получение нового access-token
        + /user/GET – получить полную информацию о текущем пользователе
        + /user/PUTCH (email, username, password) – изменение данных пользователя
        + /user/DELETE – удаление всех данных о пользователе
     3. Профиль пользователя (GET, PUTCH)
        + /profile/GET (ordering, search) – получение статистики пользователей с возможностью сортировки по полям «games», «wins» и «rating», а также поиском по полю «username»
        + /profile/{id}/GET – получение статистики пользователя
        + /profile/{id}/PUTCH (wins, games, rating) – изменение данных профиля определенного пользователя
     4. Игры (GET, POST, PUTCH, DELETE)
        + /game/GET (search, status) – получение информации об игре: «id», «user\_1», «user\_2», «status», «start\_at», «finish\_at» и «moves»
        + /game/POST (username\_1\*, username\_2\*) – создание начальной записи об игре
        + /game/{id}/GET – получение информации об игре с возможностью поиска по «id», «user\_1», «user\_2», «status», «start\_at», «finish\_at» и «moves»
        + /game/{id}/PUTCH (moves, start\_at, finish\_at, status) – добавление новых ходов в поле moves (JSON), запись времени начала и окончания игры и изменение статуса игры
        + /game/{id}/DELETE – удаление записи из БД
     5. Ходы (GET, POST, DELETE)
        + /move/GET (checker\_id, game, user) – получение информации о ходах: «id», «game», «user», «checker\_id», «new\_position», «is\_last\_move» и «is\_white»
        + /move /POST (game\*, user\*, checker\_id\*, new\_position\*, is\_last\_move, s\_white) – создание начальной записи об игре
        + /move /{id}/GET – получение информации о ходе: «id», «game», «user», «checker\_id», «new\_position», «is\_last\_move» и «is\_white»
        + /move /{id}/DELETE – удаление всех записей из БД с совпадающим game\_id
  3. **WebSocket**
     1. Connect/WS-протокол – установление WebSocket соединения
     2. Move/WS-протокол – передача информации о сделанном ходе и статусе игры
     3. newMove/gRPC - протокол – новый ход игрока
     4. updateGameStatus/gRPC-протокол (game\_id) – обновляет статус игры: «CREATED», «FINISHED» или «IN\_PROCESS»

# Этапы разработки

* 1. 1 неделя - согласование темы разработки с преподавателем
  2. 4 неделя - предварительный вариант ТЗ.
  3. 8 неделя - окончательный вариант ТЗ с окончательным вариантом API
  4. 14 неделя – защита КР

# Техническая документация, предъявляемая по окончанию работы:

* 1. Технический проект. Расчетно-пояснительная записка, включающая в приложении комплект технической документации на программный продукт, содержащий
     1. Приложение 1 - Техническое Задание;
     2. Приложение 2 - ER-диаграмма базы данных;
     3. Приложение 3 - Диаграмма развертывания;
     4. Приложение 4 - Диаграмма прецедентов;
     5. Приложение 5 - Руководство пользователя;
     6. Приложение 6 - Программа и методика испытаний;
     7. Приложение 7 - Руководство системного администратора;
     8. Приложение 8 - Спецификация интерфейсов;
     9. Приложение 9 - Sequence диаграмма;
     10. Приложение 10 - Activity диаграмма
  2. Папка с технической и программной документацией в формате: <группа>\_ <Фамилия И.О. студента> \_КР\_СТ\_в\_АСОИУ.zip.

# Порядок приемки работы:

Приемка работы осуществляется в соответствии с "Программой и методикой испытаний."

# Дополнительные условия:

Данное Техническое Задание может дополняться и изменяться в установленном порядке.