API Документация для "Шахматы: Дополнение"

1. Введение

Данная документация описывает API для взаимодействия клиента и сервера в игре "Шахматы: Дополнение". Игра представляет собой дополнение к классическим шахматам с элементами карточной игры и фэнтезийными механиками. Взаимодействие осуществляется по протоколу HTTP/HTTPS с использованием JSON для передачи данных.

2. Общие принципы

Формат запросов/ответов: JSON.

Авторизация: Для некоторых запросов может потребоваться токен авторизации (например, JWT), который будет передаваться в заголовке Authorization: Bearer <token>.

Коды состояния HTTP:

200 OK: Запрос успешно выполнен.

201 Created: Ресурс успешно создан.

204 No Content: Запрос успешно выполнен, но нет содержимого для возврата.

400 Bad Request: Некорректный запрос (например, неверные параметры).

401 Unauthorized: Требуется авторизация или токен недействителен.

403 Forbidden: Доступ запрещен (например, у пользователя нет прав).

404 Not Found: Ресурс не найден.

409 Conflict: Конфликт при выполнении запроса (например, попытка создать существующий ресурс).

500 Internal Server Error: Внутренняя ошибка сервера.

3. Структура данных (Объекты)

3.1. Пользователь (User)

```
{
    "id": "string",
    "username": "string"
}
```

3.2. Комната (Room)

Представляет собой игровую комнату, где игроки ожидают матча или уже играют.

```
{
    "id": "string",
    "name": "string",
    "players": [
        {
            "id": "string",
            "username": "string"
        }
    ],
    "status": "string" // "waiting", "in_game", "finished"
}
```

3.3. Игровая доска (Board State)

Полное состояние игровой доски. Для карт указываются только те значения, которые пользователь может видеть, то есть для вражеских карт указывается исключительно id и mode, и только если mode прозрачная, то все остальные свойства.

```
{
    "board": [
        [
            {
                "figure_id": "string",
                "name": "string",
                "color": "string", // "white", "black"
                "description": "string",
                "copied_figure": "string", // id скопированной морфом фигуры, в начальном состоянии свое собственное id
                "unavailable_copy": [
                    "string",
                ], // id копированных ранее морфом фигур
                "mode": "integer" // обычная – 1, старая – 2 и т.д., 0 при отсутствии
                "hero": "string" // название второй фигуры для Легендарного война
                "death": "integer", // 1 если фигура съедена, 0 если на поле
                "aura": "integer", // вдохновения – 1, гипноза – 2 и т.д. 0 при отсутствии
                "condition": "integer", // уставший – 1, парализованный – 2 и т.д.
                "row": "integer",
                "col": "integer",
                "move_creation": "integer", // номер хода, на котором фигура была создана
                "walk_count": "integer", // число совершенных фигурой перемещений
            },
            ...
        ],
```

```
],
"current_player": "string", // "white" or "black",
"white_hand": [
    {
        "card_id": "string",
        "name": "string",
        "description": "string"
        "type": "string" // "spell", "creature", "aura"
        "mode": "integer" // обычная – 1, старая – 2 и т.д.
        "move_add": "integer", // номер хода, на котором карта была получена
    },
    …
],
"black_hand": [
    {
        "card_id": "string",
        "name": "string",
        "description": "string"
        "type": "string" // "spell", "creature", "aura"
        "mode": "integer" // обычная – 1, старая – 2 и т.д.
        "move_add": "integer", // номер хода, на котором карта была получена

    },
    …
],
"white_double_add": "integer", // число возможных для использования Двойных доборов
"black_double_add": "integer",
"game_log": [
    "string", // Список текстовых логов событий в виде, определенном ТЗ.
],
```

"moves_log": [

"string", // Список совершенных ходов в формате ‘1.tFIGaccx1y1x2y2 x3y3acc… tFIGaccx1y1 … 2.…‘, где t – тип фигуры, FIG – буквенный идентификатор фигуры, a – буквенное обозначение ауры перемещаемой фигуры после совершения хода, cc – буквенные обозначения наложенных состояний после совершения хода, x1y1 – место, где находится фигура или где она появляется в результате действия карты, x2y2 – координаты, куда перемещается фигура, x3y3, x4y4… – координаты изменяемой фигуры, после которых идет описание приобретенных состояний

],

"cards_count": "integer", // число оставшихся в колоде карт

"game_status": "string", // "playing", "check", "checkmate", "stalemate", "draw", "resigned"

}

3.4. Карта (Card)

{

"card_id": "string",

"name": "string",

"description": "string",

"type": "string" // "spell", "creature", "aura"

"mode": "integer" // обычная – 1, старая – 2 и т.д.

"move_add": "integer", // номер хода, на котором карта была получена

}

3.5 Фигуры (Figure)

{

"figure_id": "string",

"name": "string",

"color": "string", // "white", "black"

"description": "string",

"copied_figure": "string", // id скопированной морфом фигуры, в начальном состоянии свое собственное id

"unavailable_copy": [

"string",

], // id копированных ранее морфом фигур

"mode": "integer" // обычная – 1, старая – 2 и т.д., 0 при отсутствии

"hero": "string" // название второй фигуры для Легендарного война

"death": "integer", // 1 если фигура съедена, 0 если на поле

"aura": "integer", // вдохновения – 1, гипноза – 2 и т.д. 0 при отсутствии

"condition": "integer", // уставший – 1, парализованный – 2 и т.д.

"row": "integer",

"col": "integer",

"move_creation": "integer", // номер хода, на котором фигура была создана

"walk_count": "integer", // число совершенных фигурой перемещений

}

4. API Эндпоинты

4.1. Авторизация и Управление Пользователями

POST /auth/register - Регистрация нового пользователя

Описание: Создает нового пользователя в системе.

Запрос:

{

"username": "string",

"password": "string"

}

Ответ:

201 Created

{

```
    "user_id": "string",
    "username": "string",
    "token": "string" // Токен авторизации
}
```

400 Bad Request (если имя пользователя занято или пароль не соответствует требованиям)

POST /auth/login - Вход пользователя

Описание: Аутентифицирует пользователя и возвращает токен авторизации.

Запрос:

```
{
    "username": "string",
    "password": "string"
}
```

Ответ:

200 OK

```
{
    "user_id": "string",
    "username": "string",
    "token": "string" // Токен авторизации
}
```

401 Unauthorized (неверные учетные данные)

4.2. Управление Игровыми Комнатами

GET /rooms - Получить список доступных комнат

Описание: Возвращает список всех открытых игровых комнат, ожидающих игроков.

Запрос: Отсутствует

Ответ:

200 OK

```
[
    {
        "id": "string",
        "name": "string",
        "players": [
            {
                "id": "string",
                "username": "string"
            }
        ],
        "status": "string"
    },
]
```

POST /rooms - Создать новую игровую комнату

Описание: Создает новую игровую комнату.

Запрос: (с токеном авторизации)

```
{
    "name": "string" // Необязательное имя комнаты
}
```

Ответ:

201 Created

```
{
    "room_id": "string",
    "name": "string",
    "players": [
        {
            "id": "string",
            "username": "string"
        }
    ],
    "status": "waiting"
}
```


401 Unauthorized

POST /rooms/{room_id}/join - Присоединиться к комнате

Описание: Позволяет игроку присоединиться к существующей комнате.

Запрос: (с токеном авторизации):


```
{
    "id": "string"
}
```


Ответ:

200 OK (возвращает обновленное состояние комнаты)


```
{
    "room_id": "string",
    "name": "string",
    "players": [
        {
```

```
            "id": "string",
            "username": "string"
        },
        {
            "id": "string",
            "username": "string"
        }
    ],
    "status": "in_game" // Если комната заполнилась и игра началась
}
```


404 Not Found (комната не найдена)

409 Conflict (комната уже полна)

POST /rooms/{room_id}/leave - Покинуть комнату

Описание: Позволяет игроку покинуть комнату.

Запрос: (с токеном авторизации)


```
{
    "id": "string"
}
```


Ответ:

204 No Content

401 Unauthorized

404 Not Found (комната не найдена или игрок не находится в ней)

4.3. Игровой процесс

GET /game/{room_id}/state - Получить текущее состояние игры

Описание: Возвращает полное текущее состояние игровой доски и других игровых параметров.

Запрос: (с токеном авторизации)

```
{
    "id": "string" // id комнаты
}
```

Ответ:

200 OK

См. Board State.

401 Unauthorized

404 Not Found (игра не найдена или не началась)

POST /game/{room_id}/move - Сделать ход фигурой

Описание: Отправляет информацию о ходе фигуры.

Запрос: (с токеном авторизации)

```
{
    "id": "string" // id комнаты
    "from_row": "integer",
    "from_col": "integer",
    "to_row": "integer",
    "to_col": "integer",
    "promotion_to": "string" // Необязательно, если ход - превращение пешки (например, "queen", "rook", "bishop", "knight" и т.п.)
```

"morph_reincarnation": "string" // Необязательно, id фигуры, в которую перевоплощается Морф

}

Ответ:

200 OK (возвращает обновленное состояние игры)

См. Board State.

400 Bad Request (некорректный ход, не очередь игрока)

401 Unauthorized

404 Not Found (игра не найдена)

409 Conflict (ход невозможен в текущем состоянии)

POST /game/{room_id}/play_card - Использовать карту

Описание: Игрок использует карту из своей руки.

Запрос: (с токеном авторизации)

{

"id": "string" // id комнаты

"card_id": "string",

"target_type": "string", // "figure", "cell", "player", "self", etc. (зависит от типа карты)

"target_data": {} // Данные о цели (например, { "row": 1, "col": 2 } для клетки или { "figure_id": "some_id" } для фигуры)

}

Ответ:

200 OK (возвращает обновленное состояние игры)

См. Board State.

400 Bad Request (карта применена неверно, неверная цель)

401 Unauthorized

404 Not Found (игра не найдена)

409 Conflict (использование карты невозможно в текущем состоянии)

POST /game/{room_id}/surrender - Сдаться

Описание: Игрок сдает партию.

Запрос: (с токеном авторизации):

```
{
    "id": "string" // id комнаты
}
```

Ответ:

200 OK (возвращает обновленное состояние игры, где game_status будет "resigned")

См. Board State.

401 Unauthorized

404 Not Found (игра не найдена)

5. Дополнительные соображения

WebSocket: Для получения обновлений игрового состояния в реальном времени (ходы других игроков, срабатывание эффектов карт, таймеры) настоятельно рекомендуется использовать WebSocket-соединение. Клиент будет подписываться на события, связанные с room_id. API, описанное выше, может использоваться для начальной загрузки состояния и отправки команд, а WebSocket - для пуша изменений.

ID Сущностей: Все ID (пользователей, комнат, карт, фигур) должны быть уникальными строками (например, UUID).

Валидация: Сервер должен строго валидировать все входящие запросы для предотвращения читерства и некорректных действий.

Идемпотентность: Некоторые запросы (например, POST для создания ресурсов) могут быть не идемпотентными. Это необходимо учитывать при проектировании клиента и обработке ошибок.

Обработка ошибок: Ответы с ошибками должны содержать достаточно информации для отладки на стороне клиента (например, code и message поля в JSON).

Безопасность: Использовать HTTPS, валидировать токены авторизации, применять механизмы защиты от CSRF/XSS.

Расширяемость: Структура данных и API должны быть достаточно гибкими для добавления новых типов фигур, карт, механик в будущем.