## Шаблон для описания метода Фронт-бэк

### Общая информация CreatePlayer

| **Запрос** | http://{{ domain\_name }}/api/player/create/ |
| --- | --- |
| **Метод операции** | POST |
| **Назначение** | Создает игрока |
| **Необходима авторизация** | Нет |
| **Response status code** | 201 Created |

### Бизнес-логика

Создает игрока, по данным отправленным в теле запроса

### Параметры запроса и ответа

#### Запрос

Описание параметров

| Параметр | REQ | Тип | Описание и дополнительная информация |
| --- | --- | --- | --- |
| game | Нет, может принимать null | integer | Идентификатор игры. Можно не указывать если пользователь хочет создать игру |
| avatar | Да | varchar(255) | Аватар игрока текстом |
| status | Нет, по умолчанию не готов | varchar(2) | Статус игрока  NR - Not Ready (не готов)  RD - Ready (готов) |
| points | Нет, по умолчанию 0 | integer | Количество очков игрока |

#### Ответ

Описание параметров ответа

| Параметр | REQ | Тип | Описание и дополнительная информация |
| --- | --- | --- | --- |
| game | Нет, может принимать null | integer | Идентификатор игры. Можно не указывать если пользователь хочет создать игру |
| avatar | Да | varchar(255) | Аватар игрока текстом |
| status | Нет, по умолчанию NR | varchar(2) | Статус игрока  NR - Not Ready (не готов)  RD - Ready (готов) |
| points | Нет, по умолчанию 0 | integer | Количество очков игрока |

### Примеры запроса и ответа

#### Пример запроса

POST http://localhost:8000/api/player/create/

Body {

“game”: 1,

“avatar”: “lama”,

“status”: “RD”,

“points”: “0”

}

### Пример ответа

201 Сreated

BODY:

{

“id”: 1,

“game”: 1,

“avatar”: “lama”,

“status”: “RD”,

“points”: 0

}

### Общая информация CreateGame

| **Запрос** | http://{{ domain\_name }}/api/game/create/ |
| --- | --- |
| **Метод операции** | POST |
| **Назначение** | Создает игру |
| **Необходима авторизация** | Нет |
| **Response status code** | 201 Created |

### Бизнес-логика

Создает игру, по данным отправленным в теле запроса

### Параметры запроса и ответа

#### Запрос

Описание параметров

| Параметр | REQ | Тип | Описание и дополнительная информация |
| --- | --- | --- | --- |
| creator | Да | integer | Идентификатор игрока, который создал игру. После создания game у этого игрока сменяется на созданный |
| deck | Да | integer | Идентификатор выбранной колоды |
| members\_num | Да | integer | Количество игроков, можно выбрать только 3,4,5,6 |
| points\_to\_win | Да | integer | Количество очков для победы |
| status | Нет, по умолчанию WT | varchar(2) | Статус игры  WT - Waiting  PL - Playing  FN - Finished |

#### Ответ

Описание параметров ответа

| Параметр | REQ | Тип | Описание и дополнительная информация |
| --- | --- | --- | --- |
| creator | Да | integer | Идентификатор игрока, который создал игру. После создания game у этого игрока сменяется на созданный |
| deck | Да | integer | Идентификатор выбранной колоды |
| members\_num | Да | integer | Количество игроков, можно выбрать только 3,4,5,6 |
| points\_to\_win | Да | integer | Количество очков для победы |
| status | Нет, по умолчанию WT | varchar(2) | Статус игры  WT - Waiting  PL - Playing  FN - Finished |

### Примеры запроса и ответа

#### Пример запроса

POST http://localhost:8000/api/game/create/

Body {

“creator”: 1,

“deck”: “starwars”,

“members\_num”: 4,

“points\_to\_win”: 40,

“status”: “PL”,

}

### Пример ответа

201 Сreated

BODY:

{

“id”: 1,

“creator”: 1,

“deck”: “starwars”,

“members\_num”: 4,

“points\_to\_win”: 40,

“status”: “PL”,

}

### Общая информация CreateReview

| **Запрос** | http://{{ domain\_name }}/api/review/create/ |
| --- | --- |
| **Метод операции** | POST |
| **Назначение** | Сохраняет в бд отзыв о игре |
| **Необходима авторизация** | Нет |
| **Response status code** | 201 Created |

### Бизнес-логика

Сохраняет в бд отзыв о игре, по данным отправленным в теле запроса

### Параметры запроса и ответа

#### Запрос

Описание параметров

| Параметр | REQ | Тип | Описание и дополнительная информация |
| --- | --- | --- | --- |
| game | Да | integer | Идентификатор игры |
| player | Да | integer | Идентификатор игрока |
| visual | Да | integer | Оценка визуала |
| gameplay | Да | integer | Оценка геймплэя |
| recomendation | Да | integer | Оценка рекомендации |

#### Ответ

Описание параметров ответа

| Параметр | REQ | Тип | Описание и дополнительная информация |
| --- | --- | --- | --- |
| game | Да | integer | Идентификатор игры |
| player | Да | integer | Идентификатор игрока |
| visual | Да | integer | Оценка визуала |
| gameplay | Да | integer | Оценка геймплэя |
| recomendation | Да | integer | Оценка рекомендации |

### Примеры запроса и ответа

#### Пример запроса

POST http://localhost:8000/api/review/create/

Body {

“game”: 1,

“player”: 1,

“visual”: 4,

“gameplay”: 4,

“recomendation”: 5,

}

### Пример ответа

201 Сreated

BODY:

{

“id”: 1,

“game”: 1,

“player”: 1,

“visual”: 4,

“gameplay”: 4,

“recomendation”: 5,

}

### Общая информация GameWaiting WebSocket

| **Запрос** | ws://{{ domain\_name }}/ws/game/waiting/{{ game\_id }}/ |
| --- | --- |
| **Метод операции** |  |
| **Назначение** | Сокет, который используется до начала игры, когда игроки только присоединяются к игре |
| **Необходима авторизация** | Нет |
| **Response status code** |  |

### Бизнес-логика

При подключении к сокету отправляет данные

{

“isReadyToPlay”: isReadyToPlay,

“members”: members

}

где isReadyToPlay булево значение, которая означает можно ли начать игру, становится true, если подключилось максимальное кол-во игроков и у всех статус Ready. В members хранится информация о игроках в виде

{

player\_id: {

“avatar”: player\_avatar,

“status”: player\_status

}

}

Пример ответа:

{

“isReadyToPlay”: false,

“members”: {

1: { “avatar”: “lama”, “status”: “RD” },

2: { “avatar”: “lama”, “status”: “NR” }

}

}

Можно отправить данные о готовности или неготовности игрока в виде

{

“player\_id”: 1,

“status”: “RD”

}

При отправке данных в бд изменяется статус игрока и player\_id сохраняется в сессии. Затем в вебсокет отправляются данные о готовности игроков, такой же который отправляется при подключении. Для создателя игры также нужно будет отправить информацию о готовности чтобы его player\_id сохранилась в его сессии. После того как isReadyToPlay становится true, нужно отключится от этого вебсокета и подключится к двум другим, которые я опишу ниже.

### Общая информация GamePublic WebSocket

| **Запрос** | ws://{{ domain\_name }}/ws/game/playing/{{ game\_id }}/ |
| --- | --- |
| **Метод операции** |  |
| **Назначение** | Сокет для отправки общих информаций, информация отправленная сюда будет видна всем игрокам |
| **Необходима авторизация** | Нет |
| **Response status code** |  |

### Бизнес-логика

При подключении к вебсокету статус игры сменяется на Playing и создается 1-раунд игры. После этого рандомно выбираются карты для данного раунда, выбираются карты для каждого игрока, сохраняются в бд и отправляются через приватный сокет игрока GamePrivate WebSocket.

Ведущий может отправить информацию о тексте ассоциации и карте ассоциации в виде

{

“association\_text”: “мал да удал”,

“association\_card”: 5 // id карты

}

Статус ведущего становится Ready, а у остальных Not ready. В ответ на это сообщение, вебсокет отправляет всем игрокам текст ассоциации в виде

{

“association\_text”: “мал да удал”

}

После того, как они выбирают свою карту, информацию о готовности они должны отправить сюда в виде

{

“status”: “RD”

}

После этого ведущему через приватный сокет отправляются данные о количестве очков у каждого игрока и готовность игрока. Если все игроки готовы, то каждому игроку через его приватный сокет отправляются выбранные другими игроками карты и карта которую выбрал он.

После того как все выберут карту ведущего, начинаются вычисления результатов

{

“who\_chose\_youc\_cards”: [1, 3, 4], // id игроков которые выбрали вашу карту

“points\_for\_round”: 3,

“all\_points”: 3,

“guess\_right”: false // не отображается для ведущего

}

Если есть победитель, он сохраняется в бд, статус игры переводится в Finished и отправляются данные о победителе в виде

{

“winner\_id”: 2

}

Иначе, создается новый раунд и игра продолжается.

### Общая информация GamePrivate WebSocket

| **Запрос** | ws://{{ domain\_name }}/ws/game/playing/{{ game\_id }}/  {{ player\_id }}/ |
| --- | --- |
| **Метод операции** |  |
| **Назначение** | Сокет для отправки личных информаций игрока, какую карту выбрал и т.д. |
| **Необходима авторизация** | Нет |
| **Response status code** |  |

### Бизнес-логика

При подключении к вебсокету проверяет совпадают ли player\_id из url с player\_id из данных сессии и принимает подключение если они совпадают. По команде из публичного сокета, может отправить карты игрока в виде

{

“game\_status”: “PL”,

“leader\_id”: 2,

“player\_cards”: {

1: <http://localhost:8080/media/1.jpg>,

2: [http://localhost:8080/media/2.jpg](http://localhost:8080/media/1.jpg) // id: url

},

“players”: {

1: {

“avatar”: “lama”,

“status”: “WT”,

“points”: 0

}

…

}

}

При изменении статуса у одного из игроков, отправляет ведущему через приватный сокет данные в виде

{

1: {

“status”: “RD”,

“points”: 0

},

…

}

После того как все выберут карту каждому игроку отправляются данные о выбранных картах и его карте в виде

{

“your\_card”: 2,

“placeCards”: {

1: <http://localhost:8080/media/1.jpg>,

2: [http://localhost:8080/media/2.jpg](http://localhost:8080/media/1.jpg) // id: url

}

}

Может получать данные о карте связанный с ассоциацией, данные должны быть отправлены в виде

{

“association\_card”: 2 // id карты

}

Может получать данные о том, какой по его мнению карта у ведущего в виде

{

“choice”: 3 // id карты

}

при отправке его статус меняется на Ready

Если все игроки выбрали карты, то в публичном сокете вычисляются результаты.