**Соединение:**

• Используйте ws://localhost:8080 для подключения к серверу.

**Отправка сообщений:**

• Отправьте сообщение типа message с JSON-объектом:

{

"type": "message",

"message": "Текст сообщения"

}

\* Сервер отправит сообщение типа message всем авторизованным клиентам с полями:

\* sender: имя пользователя, отправившего сообщение

\* message: текст сообщения

**Авторизация:**

• Отправьте сообщение типа auth с JSON-объектом:

{

"type": "auth",

"username": "имя\_пользователя",

"password": "пароль"

}

\* Сервер ответит сообщением типа auth с параметром success:

\* true - авторизация успешна

\* false - авторизация неудачна

**Отправка сообщений:**

• Отправьте сообщение типа message с JSON-объектом:

{

"type": "message",

"message": "Текст сообщения"

}

\* Сервер отправит сообщение типа message всем авторизованным клиентам с полями:

\* **sender**: имя пользователя, отправившего сообщение, **message**: текст сообщения

**Регистрация:**

\* Отправьте сообщение типа register с JSON-объектом:

{

"type": "register",

"username": "имя\_пользователя",

"password": "пароль"

}

\* Сервер ответит сообщением типа register с параметрами:

\* success:

\* true - регистрация успешна

\* false - регистрация неудачна

\* error: (только если success - false) - описание ошибки (например, "Username already exists")

**Описание сервера:**

• Сервер хранит пользователей в файле users.json в формате:

{

"username": {

"password": "пароль"

},

}

\* При запуске сервер загружает пользователей из файла users.json.

• При отправке сообщения, сервер отправляет его всем авторизованным клиентам.

• Сервер не хранит историю сообщений.

**Рекомендации:**

• Используйте библиотеку ws для работы с WebSocket.

• Используйте формат JSON для обмена данными между клиентом и сервером.

• Не храните конфиденциальные данные (пароли) в открытом виде.