**Node.js Task Queue with Rate Limiting Documentation**

**Project Overview**

This project is a Node.js application that implements a task queue with rate limiting for handling tasks per user. Each user is allowed a maximum of 1 task per second and 20 tasks per minute. The application uses Redis for managing the task queue and rate limiting.

**Tech Used**

* Node.js
* Express.js
* Redis (with Docker)
* Bull (for task queue)
* ioredis (for Redis client)
* Cluster (for Node.js clustering)
* fs(for logging)

**Prerequisites**

* [Node.js](https://nodejs.org/) (version 14 or later)
* [Docker](https://www.docker.com/) (for Redis setup)

**Installation**

1. download the code zip file.
2. Navigate to the project directory.
3. Install the required dependencies:

npm install

**Running the Application**

1. Start Redis using Docker (see [Docker Setup for Redis](#docker-setup-for-redis) below).
2. Run the Node.js application:

node index.js

**Docker Setup for Redis**

To run Redis with Docker, use the following command:

docker run -d --name redis-stack -p 6379:6379 -p 8001:8001 redis/redis-stack:latest

For more details, refer to the [Redis installation guide](https://redis.io/docs/latest/operate/oss_and_stack/install/install-stack/docker/).

**API Endpoints**

* GET /
* Description: A simple endpoint to check if the server is running.
* Response:
  + Status: 200 OK
  + Body: Hello, welcome to this task
* **POST /task**
* **Description**: Endpoint to submit a task for processing.
* **Request Body**:

{

"userid": "string" // Required user ID

}

* **Responses**:
* **200 OK**: Task queued successfully.
* **400 Bad Request**: Missing userid.
* **429 Too Many Requests**: Rate limit exceeded.

**Rate Limiting**

The application implements rate limiting using the express-rate-limit package. Each user can submit:

* 1 task per second
* 20 tasks per minute

If a user exceeds the limits, their requests are queued for processing.

**Logging**

The application logs task processing times to a file named task\_log.txt. Each log entry includes:

* User ID
* Task start time
* Task completion time

**Test:**

For test use postman and this is api for test

**Localhost:3000/task**

**Task Queue**

The application utilizes the Bull library to manage a task queue. Tasks that exceed the rate limits are queued for later processing, ensuring they are handled in the order they are received.