TEST TASK

Goals:

We need to restrict the usage of our API to prevent users from abusing our system. These are the conditions/requirements:

1. Implement a basic auth middleware. It could be just an uuid token passed in headers, or it could be a jwt. No need to implement login/register routes. You can just store the token somewhere (env, app, db).
2. Implement 2 types of routes: public and private. Private routes should use the auth middleware.
3. Implement a rate limiter. It should check a token limit for private routes and a ip limit for public routes.
4. **Set a rate limit by token to 200 req/hour**
5. **Set a rate limit by ip to 100 req/hour**
6. Those numbers (token limit and ip limit) should be configurable from the environment
7. When a user reaches the limit, in the response show an error message about current limit for that user account, and display when (time) the user can make the next request
8. **Keep concurrency in mind.**
   Your solution should handle multiple requests at the same time,
   for example, let's say you have 4000 requests per second to public route from the same user, so your solution should respond with 429 status code when the rate limit is reached.
9. Bonus: keep performance in mind.
10. Optional task: Create a different weight of request rate for every URL: 1/2/5 points per request (you can assume we have 5 different end points) depending on end point.

Allowed stack includes:

- Node.js using Express or Nest.js.
- MongoDB
- Redis

Feel free to use additional services, **except ready limiter libraries**.

Output:

Source code of the implementation