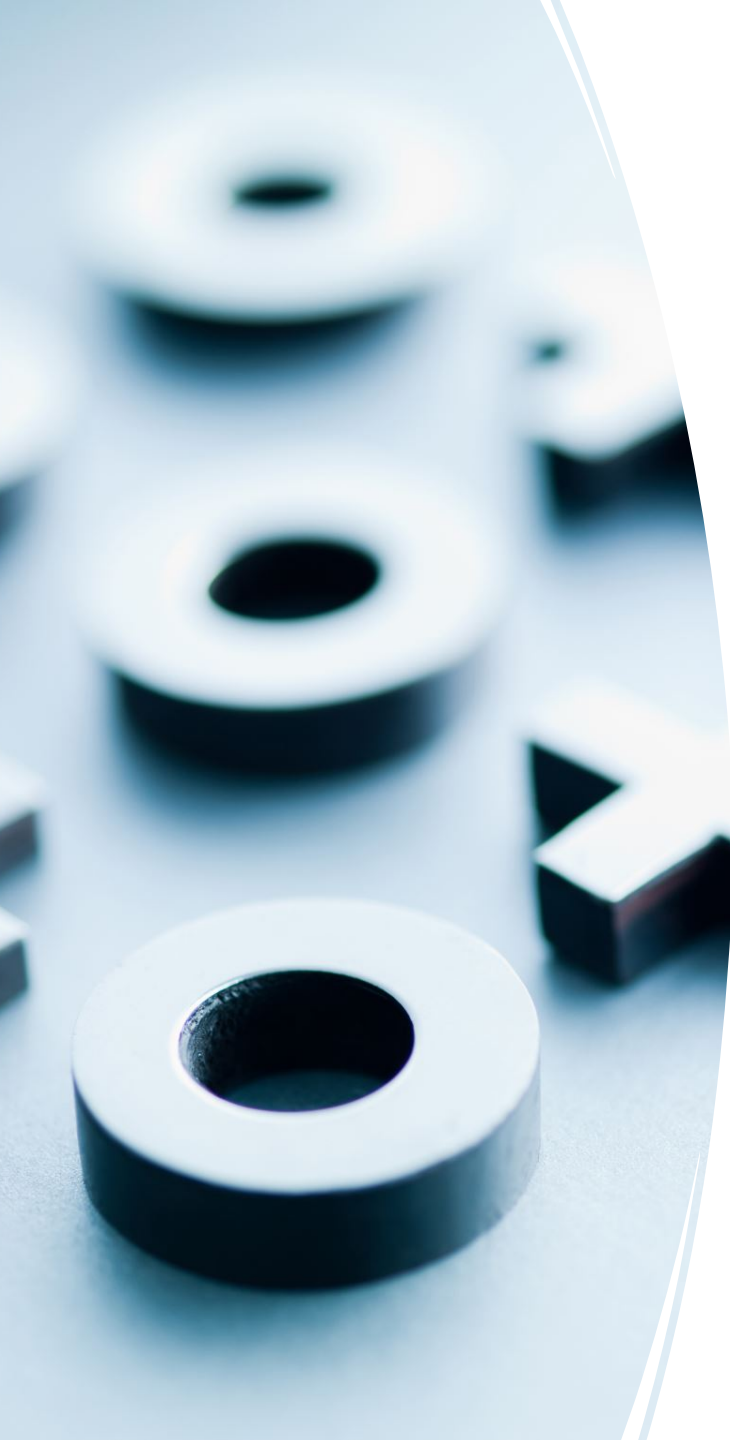




Challenge Task: Tic Tac Toe

Distributed Systems

Project Requirements

Load balancing of two Instances

Failover

Persistent database

JWT based authentication

Single command startup

Load testing

Technology Stack

Frontend: React (Vite)

Backend: Node.js + Express

Database: PostgreSQL 18

Containers: Docker

Authentication: JWT

Load Testing: wrk

JWT Authentication

User registration and login

Passwords hashed before storage

JWT tokens expire after 1d/10m

Protected API endpoints

```
router.post('/', verifyJWT, async  
(req, res) => {
```

Load Balancing & Failover

- Nginx upstream round-robin balancing (default algorithm)
- Traffic distributed across backend1/backend2
- If one instance is not healthy, nginx temporarily removes it from rotation
- Creates group of backend instances:

```
upstream backend_cluster {  
    server backend1:3000 max_fails=3 fail_timeout=10s;  
    server backend2:3000 max_fails=3 fail_timeout=10s;  
}
```

- Forwards api requests to one of those instances:

```
proxy_pass http://backend_cluster/api/;
```

Database & Persistence

PostgreSQL persistent storage

Docker volume persistence

Games and users stored permanently

Indexes for efficient DB reads

Shared database between instances

Load Testing

```
docker run --rm williamyeh/wrk `
-t4 -c50 -d30s `
-H "Authorization: Bearer $token" `
$TEST_URL
```

Load Testing Results

Baseline
Requests/sec:
~1895

Failover
Requests/sec: ~
1082

Latency baseline:
~9.43ms

Latency during
failover: ~11.7ms

System continued
serving requests
after failure

Single Command Deployment

`docker compose up --build`

Automatically starts all services

Creates DB schema automatically

No manual setup required

Questions?

