Task: Build a Simple URL Shortener API

Tech Required: Python + **FastAPI** (preferred) OR **Django Rest Framework**
DB: SQLite or Postgres (your choice)
Focus: Clean architecture, clarity of reasoning, correctness, and handling edge cases

---

Time Constraints:

1. Problem Understanding and Doubt Clearing 15mins
2. Coding Time 1hour

Core Requirements

- Users should be able to submit a long URL and receive a shortened version.
    - Optionally, users should be able to specify a custom alias for their shortened URL.
    - Optionally, users should be able to specify an expiration date for their shortened URL.
- Users should be able to access the original URL by using the shortened URL.

1. Create REST Endpoints:

Method	Endpoint	Description
`POST`	`/shorten`	Takes a long URL and returns a shortened code
`GET`	`/{code}`	Redirects the user to the original long URL

---

2. URL Shortening Rules

- The shortened code should be **6–8 characters**.
- Must handle **collisions gracefully**.

---

4. Validation Requirements

- Reject invalid URLs thoughtfully (don’t over-validate).

- Avoid creating duplicate database entries for the same URL (optional but good).

Bonus (Optional, Nice-to-Have)

These are not required, but help identify *strong engineers*:

- Add **expiry time** for links (e.g., `expires_in_seconds`)

- Add `/stats/{code}` endpoint returning:

  - Number of redirects

  - Created timestamp

  - Last accessed timestamp

- Add **pytest**-based unit tests

- Dockerize the application

- Add OpenAPI docs (FastAPI auto-docs already great)

Submission

- Upload the project in Github and provide the **public link in the mail.**
- **Zip the project files** along with other files needed and **attach it with the mail.**

Mail id ayon.das@natureglobal.com , rupanjan.hari@natureglobal.com, hr@natureglobal.com