Problem:

On a cloud platform of your choice, provision a service, using Infrastructure as Code, that serves a HTML page.

The content of the page must be ..

<h1>The saved string is dynamic string</h1>

.. Where "dynamic string" can be set to whatever is requested without having to re-deploy.

Solutions:

I chose to use **Python**, **Terraform**, and **AWS**. Based on that, here are 3 possible solutions:

1- Python server on EC2:
Create a Python server (e.g using Flask) on an EC2 instance (e.g as a systemd service).
The server exposes 2 routes:

- “GET /” to serve the HTML page
- “POST /update” to update the page with the new dynamic string

This solution is straightforward and low in cloud service coupling (vendor-lock) but it’s also legacy. It’s not easily scalable and charge constant costs since the EC2 instance is always running, which makes it less ideal for low-traffic scenarios.

2- Lambda function with DynamoDB (serverless):
Create a lambda function to handle both get and update actions:

- GET requests return the HTML content with the stored dynamic string in DynamoDB
- POST requests update the string in DynamoDB

This is a modern, serverless and cost-effective solution for low to moderate traffic. Creating a semi-static HTML page dynamically could be its most important drawback.
If request rate is high, we can use CloudFront with the help of API Gateway to cache the output and improve performance and reducing costs. However, caching adds some more complexity because we should invalidate the CloudFront cache in case of update in dynamic string value. A simpler alternative is to use a short TTL.
If dynamic string is changed rarely by an admin person, we can use SSM Parameter Store instead of DynamoDB. It removes the dynamic string update part in the Lambda function and make it very simple. However, since the value should be fetched on every read request, caching is essential (in the Lambda or via CloudFront).

3- S3 for reads and Lambda for updates (serverless):
This solution uses:

- An S3 bucket to host the HTML page
- A Lambda function to update the HTML file with a new dynamic string

This is a very simple and cost-effective solution for normal situations when the read rate is high and much more than writes (updates).
We can improve proformance by putting S3 bucket directly behind the CloudFront but we still need to choose between using a short TTL value or invalidate the cache on updates. Rate of update requests and level of consistency could help us to select between them. This is obvious

that adding cache invalidation increase the complexity of the Lambda function. The trade-off here is between simplicity and content consistency.

Final Choice:

I chose the third solution (S3+Labda) because it's:

- Simple (serving static HTML content)
- Modern (serverless and scalable)
- Cost-efficient for typical usecases (many reads, few updates)

Although, it's very easy to improve performance by putting it behind CloudFront.
I didn't use the S3 website hosting which lacks HTTPS support, but it can be added via CloudFront if needed.

What would I do if I had more time?

- Add authentication to the Lambda function to restrict unauthorized acess
- Sanitize user input to prevent HTML injection (e.g using html.escape)
- Add a unit test for the HTML generation logic of the Lambda function
- Support separate environments in Terrafom file (e.g staging and production)
- Add an automated end to end test in the staging environment
- Integrate CloudFront for better performance