

Proof of Concept (PoC) – URL Shortener

1. Title

URL Shortener Web Application – Proof of Concept

2. Objective

The objective of this PoC is to demonstrate a simple Python-Flask-based web application that takes a long URL as input and generates a unique shortened version using Base62 encoding. The shortened link redirects to the original URL when accessed.

3. Background

Long URLs can be inconvenient to share and remember.

A **URL shortener** provides a compact alias (short code) that maps to the original link. This is widely used in platforms like **bit.ly** and **tinyurl.com** for:

- Reducing character count for social media posts
- Tracking link usage
- Simplifying link sharing

In this PoC, the application uses:

- **Base62 encoding** to generate unique short slugs
 - **In-memory dictionary storage** for mapping URLs (for demonstration purposes)
 - A minimal **Flask web interface** for user interaction
-

4. Scope

- Accept long URLs via a web form
 - Generate a short URL slug using Base62 encoding
 - Store the mapping temporarily in memory
 - Redirect users from the short URL to the original link
-

5. Methodology

Step 1 – User Input

The user enters a long URL in the web form.

Step 2 – Counter Increment & Encoding

The application maintains a counter that increments for each new link. This counter value is converted into a **Base62 string** to create a short slug.

Step 3 – Mapping Storage

The slug and original URL are stored in an in-memory dictionary (`url_mapping`).

Step 4 – Shortened URL Display

The app returns a clickable shortened link (`/<slug>`).

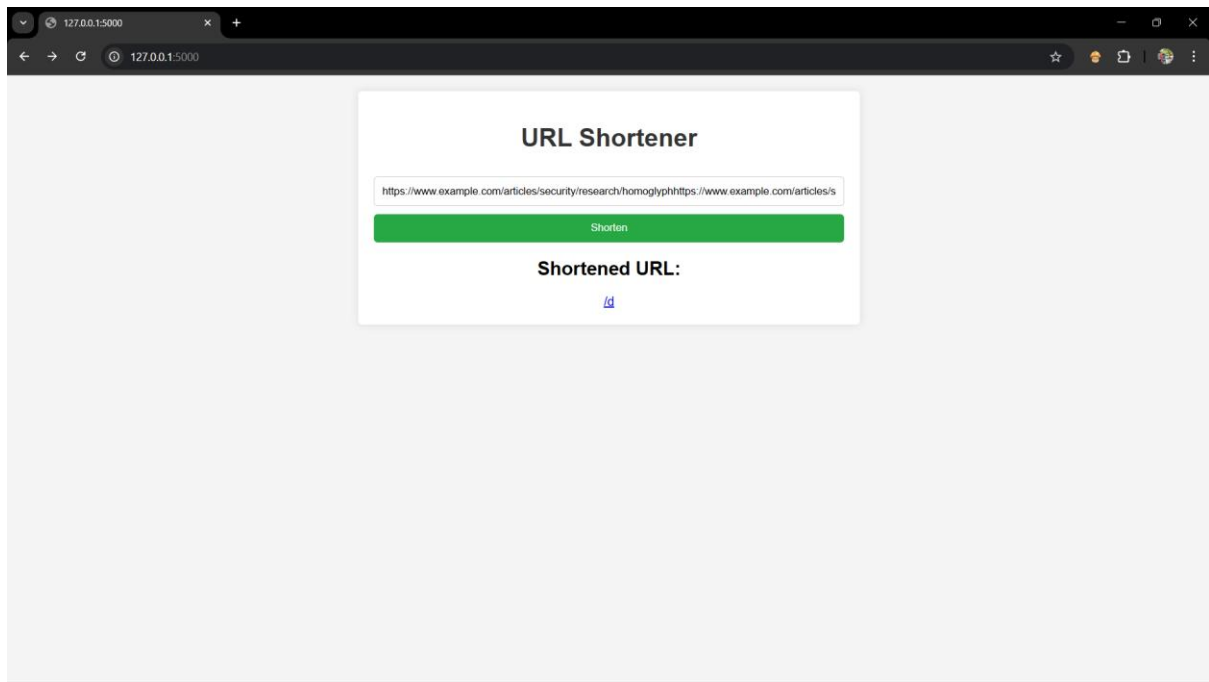
Step 5 – Redirection

When the short link is visited, Flask retrieves the original URL from the mapping and redirects the user.

```

File Edit Selection View Go Run Terminal Help
url_shortener.py X
Homography.py url_shortener.py X
url_shortener.py > home
1 from flask import Flask, request, redirect, render_template_string
2 import string
3
4 app = Flask(__name__)
5
6 # Base62 characters
7 BASE62 = string.ascii_letters + string.digits
8
9 # Dictionary to store URL mappings
10 url_mapping = {}
11 counter = 0
12
13 def encode(num):
14     """Encode a number to a base62 string."""
15     if num == 0:
16         return BASE62[0]
17     base = len(BASE62)
18     encoded = []
19     while num > 0:
20         num, rem = divmod(num, base)
21         encoded.append(BASE62[rem])
22     return ''.join(reversed(encoded))
23
24 @app.route('/', methods=['GET', 'POST'])
25 def home():
26     """Render the home page and handle URL shortening."""
27     if request.method == 'POST':
28         global counter
29         long_url = request.form['url']
30         counter += 1
31         short_slug = encode(counter)
32         url_mapping[short_slug] = long_url
33         return render_template_string('''
34         <style>
35         body { font-family: Arial, sans-serif; background-color: #f4f4f4; margin: 0; padding: 20px; }
36         .container { max-width: 600px; margin: auto; background: white; padding: 20px; border-radius: 5px; box-shadow: 0 0 10px rgba(0, 0, 0, 0.1); }
37         h1 { text-align: center; color: #333; }
38         form { display: flex; flex-direction: column; }
39         input[type="text"] { padding: 10px; margin: 10px 0; border: 1px solid #ccc; border-radius: 5px; }
40         ''')

```



7. Example Execution

User Input:

<https://www.example.com/articles/security/research/homoglyph>

Shortened Output:

/d

Visiting: <http://localhost:5000/b>

Redirects to:

<https://www.example.com/articles/security/research/homoglyph>

8. Limitations

- URLs are stored only in memory; data will be lost when the server restarts.
 - No validation for malicious URLs.
 - No analytics or expiration features.
 - No integration with homoglyph detection (can be added for phishing protection).
-

9. Conclusion

This PoC demonstrates a minimal yet functional URL shortening service using Python's Flask framework. It can be extended with database storage, analytics, user accounts, and phishing prevention techniques such as homoglyph detection.