Hosted Web Application

Hosted Web Application Link

The live version of the Shorty URL Shortener application can be accessed using the following link:

https://getshortylinks.com/

This link directs users to the publicly accessible frontend, which is built using React Native and Expo Web and deployed via AWS S3 + CloudFront. The backend API is hosted separately with AWS S3 + EC2.

GitHub Repository

The full source code for the project, including both the frontend and backend components, is available in the GitHub repository below:

https://github.com/tgreenleewgu/shorty_capstone

This repository includes the latest version of the codebase submitted for evaluation. All commits are documented with clear messages, and the full branch history can be viewed below. All deployment was done at two sperate repositories and then merged into a single post development. Github doesn't offer a graphical representation.

Backend:

https://github.com/tgreenleewgu/shorty_project/commits/main/

Frontend:

https://github.com/tgreenleewgu/ShortyUrlFrontend/commits/main/

Combined Project:

https://github.com/tgreenleewgu/shorty_capstone/commits/main/

User Guide for Initial Setup and Running the Application

Introduction

This guide provides instructions for setting up and running the Shorty URL Shortener application. It includes steps for both local development and production deployment using AWS services such as EC2, S3, CloudFront, and Route 53. The application uses a Django backend with MongoDB Atlas and a React Native frontend using Expo Web.

Pre-Requisites

Before beginning setup, ensure you have the following:

1. A registered domain name (e.g., getshortylinks.com).
2. A MongoDB Atlas database with URI and credentials.
3. GitHub and Google OAuth credentials (Client ID and Secret).
4. AWS account access to EC2, S3, CloudFront, and Route 53.
5. Python 3.12, Node.js, and npm installed locally.

Local Development Setup

1. Clone the GitHub repository using Git.
2. Navigate to the backend folder of the project.
3. Create a virtual environment using Python and activate it.
4. Install the required packages using the provided requirements.txt file.
5. Create a .env file and fill in your MongoDB URI, secret key, and OAuth credentials.
6. Start the Django server using the Python manage.py runserver command.
7. Navigate to the frontend folder and install dependencies using npm.

8. Create a .env file and specify your local BACKEND_URL (e.g., http://localhost:8000).

9. Start the Expo Web frontend using the npx expo start --web command.

Production Deployment (AWS)

Backend Setup on EC2

1. Launch an Ubuntu EC2 instance and ensure ports 22, 80, and 443 are open.

2. SSH into your instance and update the package list and installed packages.

3. Install Python, pip, Git, and NGINX on the instance.

4. Clone the GitHub repository and navigate to the backend folder.

5. Create and activate a virtual environment, then install the dependencies.

6. Create the .env file and enter your production environment variables.

7. Start the Django application using Gunicorn, bound to localhost on port 8000.

Configure NGINX

1. Edit the NGINX site configuration to proxy requests to Gunicorn.

2. Replace the server block with your domain and proxy settings for localhost:8000.

3. Test the NGINX configuration and restart the service.

Enable SSL with Let's Encrypt

1. Install Certbot and the NGINX plugin.

2. Run Certbot to generate and apply an SSL certificate to your domain.

3. Restart NGINX to apply the secure configuration.

Frontend Deployment via S3 and CloudFront

1. On your local machine, navigate to the frontend folder.

2. Install dependencies and export the project as a static web build.

3. Upload the exported build folder to your S3 bucket for public web hosting.

4. Create a CloudFront distribution pointing to your S3 bucket.

5. Set the default root object to index.html.

Configure DNS with Route 53

1. Create a hosted zone in Route 53 for your domain name.

2. Add an A record pointing api.getshortylinks.com to your EC2 IP address.

3. Add a CNAME record pointing getshortylinks.com to your CloudFront distribution.

4. Update your domain registrar to use the Route 53 nameservers.

Final Verification

Once all services are configured:

- Visit: https://getshortylinks.com to access the frontend.

- Visit: https://api.getshortylinks.com to verify the backend API is active.

- Source code is available at: https://github.com/tgreenleewgu/shorty_capstone