Deploying a Python-Based Calculator on AWS App Runner with Firebase Cloud Storage and GitHub Actions

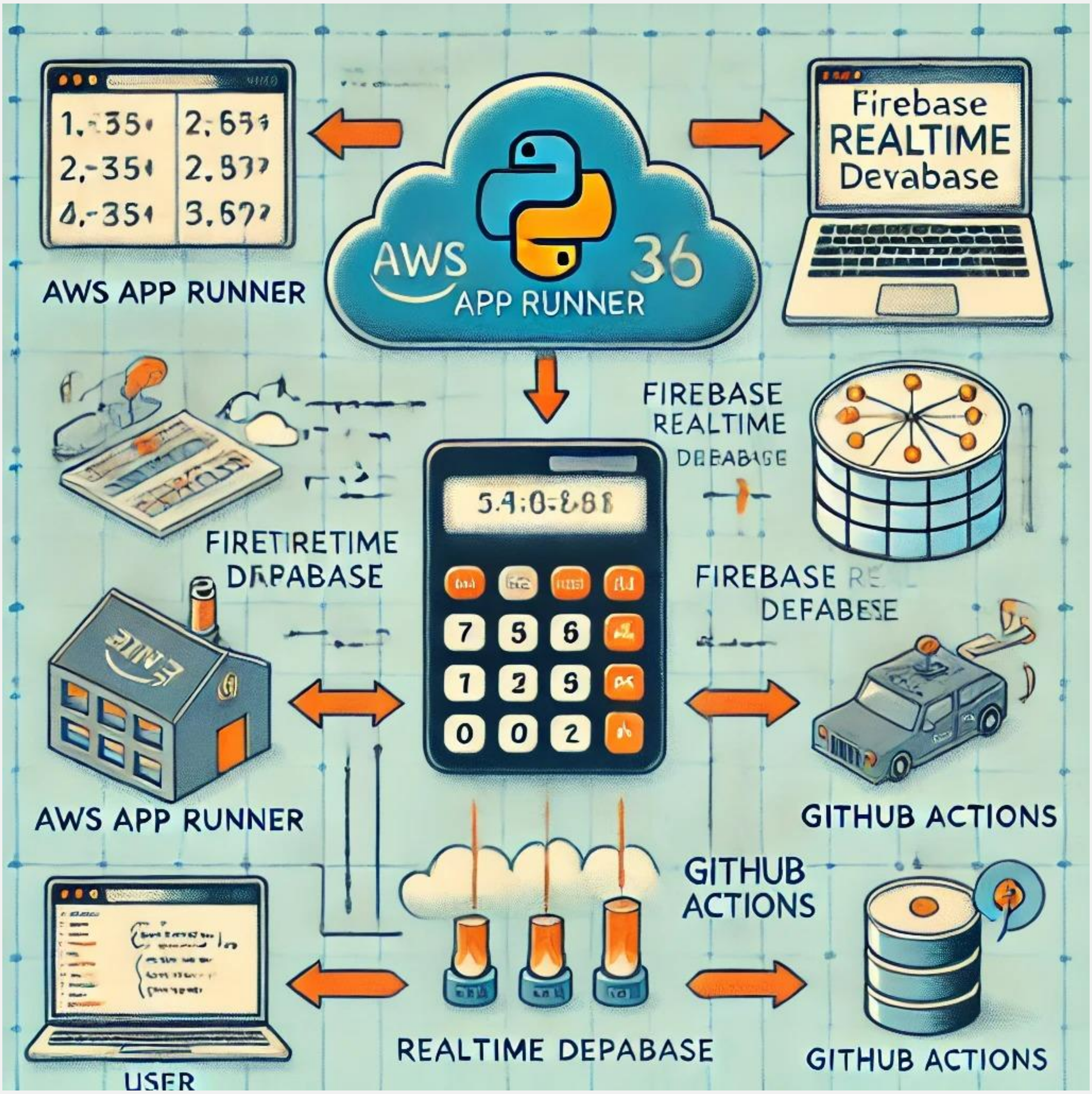
Table of Contents

1. Introduction

This documentation provides a step-by-step guide to deploying a Python-based calculator on AWS App Runner, integrating Firebase Cloud Storage for persistent result storage, and automating deployment using GitHub Actions.

2. Project Overview and Architecture

Architecture Diagram



- **Python Calculator Service**: Handles calculations and interacts with Firebase.
- **AWS App Runner**: Manages the deployment.
- **Firebase Cloud Storage**: Stores persistent results.
- **GitHub Actions**: Automates deployment using a YAML workflow.

3. Prerequisites

- AWS account with permissions for App Runner.
- Firebase account with Cloud Storage enabled.
- GitHub repository for source code and workflow automation.
- Python environment configured locally.

4. Setting Up AWS App Runner

Step 1: Create an AWS App Runner Service

1. Go to **AWS App Runner** in the AWS Console.
2. Click **Create an App Runner service**.
3. Select **Source Code Repository** as the source.
4. Connect to your **GitHub repository**.
5. Choose the branch containing your Python application.
6. Configure the service with the required settings.
7. Click **Deploy**.

App Runner event logs

```
02-16-2025 08:04:42 PM [AppRunner] Deployment with ID : b0b01bcdce0c4f11806c48d3422d7d78
started. Triggering event : SERVICE_CREATE
02-16-2025 08:04:42 PM [AppRunner] Deployment Artifact: [Repo Type: Source], [Repository:
https://github.com/NishantChamate/python-hello], [Branch: main], [SourceDirectory: /]
02-16-2025 08:05:34 PM [AppRunner] Creating pipeline for automatic deployments.
02-16-2025 08:05:34 PM [AppRunner] Successfully created pipeline for automatic
deployments.
02-16-2025 08:05:37 PM [AppRunner] Pulling source code from GITHUB Repository
( https://github.com/NishantChamate/python-hello ).
02-16-2025 08:05:41 PM [AppRunner] Successfully pulled your application source code.
02-16-2025 08:05:41 PM [AppRunner] Successfully validate configuration file.
02-16-2025 08:05:42 PM [AppRunner] Starting source code build.
02-16-2025 08:07:11 PM [AppRunner] Successfully built your application source code.
02-16-2025 08:07:22 PM [AppRunner] Provisioning instances and deploying image for publicly
accessible service.
02-16-2025 08:07:33 PM [AppRunner] Performing health check on protocol `TCP` [Port:
'8080'].
02-16-2025 08:08:24 PM [AppRunner] Health check is successful. Routing traffic to
application.
02-16-2025 08:09:29 PM [AppRunner] Successfully routed incoming traffic to application.
02-16-2025 08:09:30 PM [AppRunner] Deployment with ID : b0b01bcdce0c4f11806c48d3422d7d78
completed successfully.
02-16-2025 08:16:04 PM [AppRunner] Deployment with ID : 8ea8ae27b41749ab888442283e71087b
started. Triggering event : SERVICE_UPDATE
02-16-2025 08:16:04 PM [AppRunner] Deployment Artifact: [Repo Type: Source], [Repository:
https://github.com/NishantChamate/python-hello], [Branch: main], [SourceDirectory: /]
02-16-2025 08:16:20 PM [AppRunner] Provisioning instances and deploying image for publicly
accessible service.
02-16-2025 08:16:30 PM [AppRunner] Performing health check on protocol `TCP` [Port:
'8080'].
02-16-2025 08:17:32 PM [AppRunner] Health check is successful. Routing traffic to
application.
02-16-2025 08:19:07 PM [AppRunner] Successfully routed incoming traffic to application.
02-16-2025 08:19:18 PM [AppRunner] Deployment with ID : 8ea8ae27b41749ab888442283e71087b
completed successfully.
```

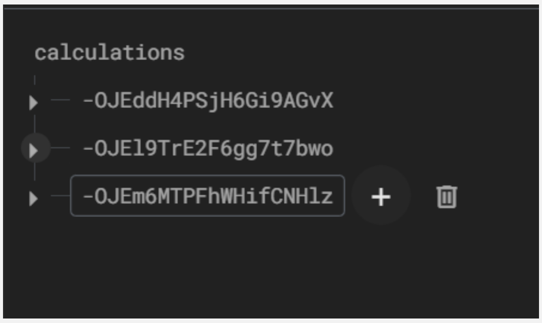
5. Integrating Firebase Cloud Storage

Step 1: Create a Firebase Project

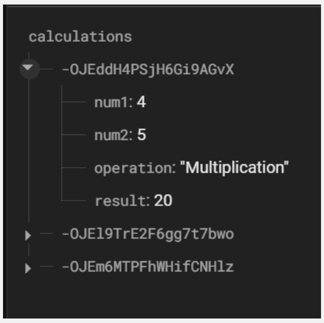
1. Go to the Firebase Console.
2. Click **Create a project** and follow the setup wizard.
3. Navigate to **Storage** and set up Cloud Storage.

Step 2: Get Firebase Credentials

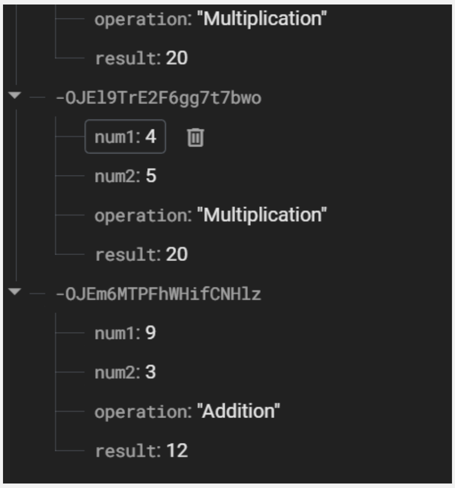
1. In Firebase Console, go to **Project Settings**.
2. Select **Service accounts**.
3. Click **Generate new private key**.
4. Store the JSON key securely.



```
calculations
  -OJEddH4PSjH6Gi9AGvX
  -OJEl9TrE2F6gg7t7bwo
  -OJEm6MTPFhWHifCNHlz
```



```
calculations
  -OJEddH4PSjH6Gi9AGvX
      num1: 4
      num2: 5
      operation: "Multiplication"
      result: 20
  -OJEl9TrE2F6gg7t7bwo
  -OJEm6MTPFhWHifCNHlz
```



```
      operation: "Multiplication"
      result: 20
  -OJEl9TrE2F6gg7t7bwo
      num1: 4
      num2: 5
      operation: "Multiplication"
      result: 20
  -OJEm6MTPFhWHifCNHlz
      num1: 9
      num2: 3
      operation: "Addition"
      result: 12
```

6. Deploying the Python Calculator on AWS App Runner

Application Requirements

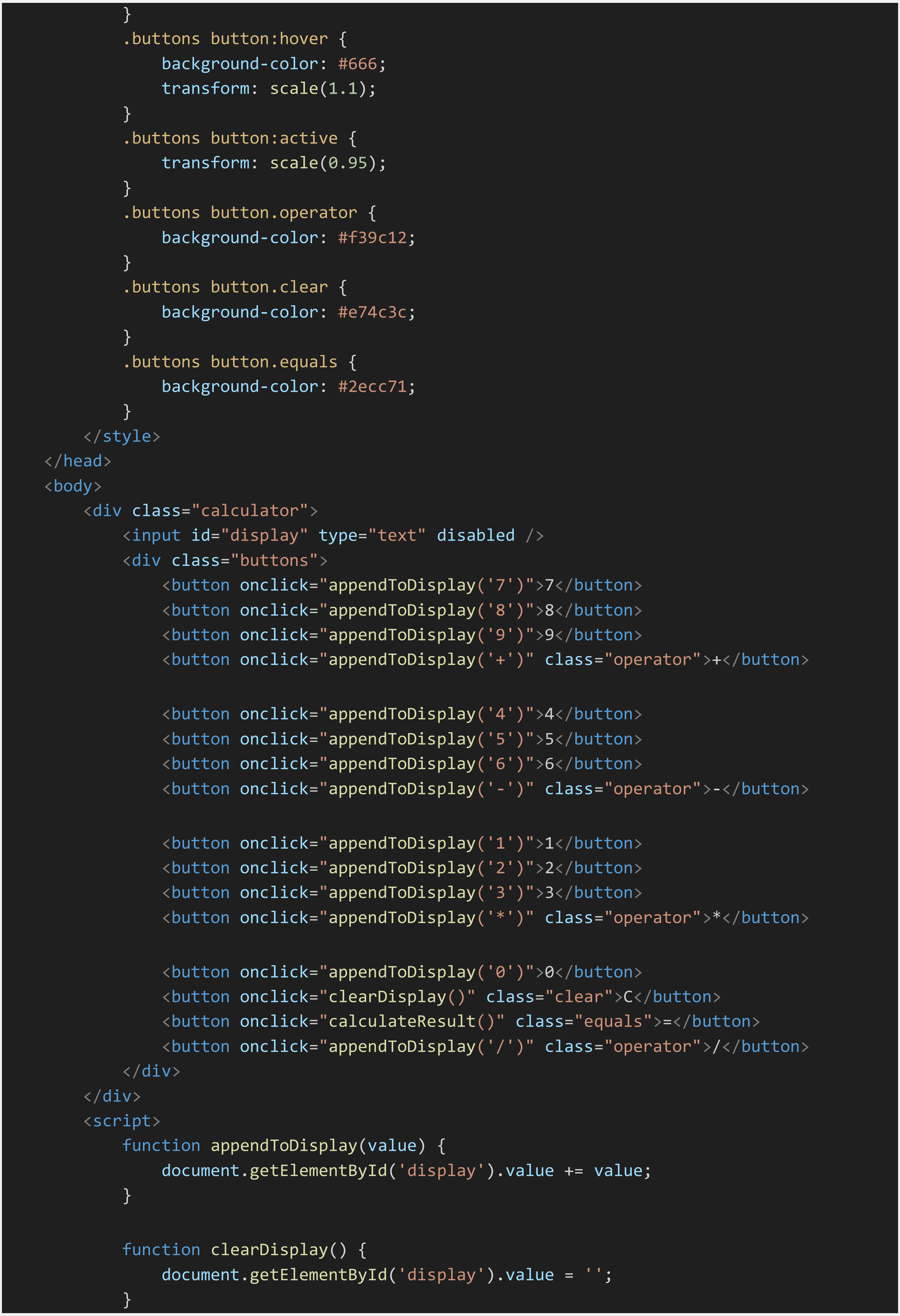
Ensure your application has a `requirements.txt` file specifying dependencies, such as:

```
pyramid==2.0
firebase-admin
waitress
requests
urllib3==1.26.16
```

Main Application File (`server.py`)

```
from wsgiref.simple_server import make_server
from pyramid.config import Configurator
from pyramid.response import Response
import os

def hello_world(request):
    html = """
    <html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Calculator</title>
        <style>
            body {
                font-family: Arial, sans-serif;
                background-color: #f7f7f7;
                display: flex;
                justify-content: center;
                align-items: center;
                height: 100vh;
                margin: 0;
            }
            .calculator {
                background-color: #333;
                border-radius: 15px;
                padding: 20px;
                box-shadow: 0px 0px 15px rgba(0, 0, 0, 0.1);
            }
            .calculator input {
                width: 100%;
                height: 50px;
                text-align: right;
                font-size: 24px;
                margin-bottom: 15px;
                padding: 10px;
                border: none;
                border-radius: 10px;
                background-color: #222;
                color: #fff;
            }
            .buttons {
                display: grid;
                grid-template-columns: repeat(4, 1fr);
                gap: 10px;
            }
            .buttons button {
                padding: 20px;
                font-size: 24px;
                background-color: #444;
                color: white;
                border: none;
                border-radius: 10px;
                cursor: pointer;
                transition: background-color 0.3s ease, transform 0.2s ease;
```



```
            }
            .buttons button:hover {
                background-color: #666;
                transform: scale(1.1);
            }
            .buttons button:active {
                transform: scale(0.95);
            }
            .buttons button.operator {
                background-color: #f39c12;
            }
            .buttons button.clear {
                background-color: #e74c3c;
            }
            .buttons button.equals {
                background-color: #2ecc71;
            }
        </style>
    </head>
    <body>
        <div class="calculator">
            <input id="display" type="text" disabled />
            <div class="buttons">
                <button onclick="appendToDisplay('7')">7</button>
                <button onclick="appendToDisplay('8')">8</button>
                <button onclick="appendToDisplay('9')">9</button>
                <button onclick="appendToDisplay('+')" class="operator">+</button>

                <button onclick="appendToDisplay('4')">4</button>
                <button onclick="appendToDisplay('5')">5</button>
                <button onclick="appendToDisplay('6')">6</button>
                <button onclick="appendToDisplay('-')" class="operator">-</button>

                <button onclick="appendToDisplay('1')">1</button>
                <button onclick="appendToDisplay('2')">2</button>
                <button onclick="appendToDisplay('3')">3</button>
                <button onclick="appendToDisplay('*')" class="operator">*</button>

                <button onclick="appendToDisplay('0')">0</button>
                <button onclick="clearDisplay()" class="clear">C</button>
                <button onclick="calculateResult()" class="equals">=</button>
                <button onclick="appendToDisplay('/')" class="operator">/</button>
            </div>
        </div>
        <script>
            function appendToDisplay(value) {
                document.getElementById('display').value += value;
            }

            function clearDisplay() {
                document.getElementById('display').value = '';
            }
            function calculateResult() {
                try {
                    let result = eval(document.getElementById('display').value);
                    document.getElementById('display').value = result;
                } catch (e) {
                    document.getElementById('display').value = 'Error';
                }
            }
        </script>
    </body>
    </html>
    """
    return Response(html)

if __name__ == '__main__':
    port = int(os.environ.get("PORT", 8080))
    with Configurator() as config:
        config.add_route('hello', '/')
        config.add_view(hello_world, route_name='hello')
        app = config.make_wsgi_app()
    server = make_server('0.0.0.0', port, app)
    server.serve_forever()
```

7. Automating Deployment with GitHub Actions

YAML Workflow Example

```
name: Deploy to AWS App Runner

on:
  push:
    branches:
      - main

jobs:
  deploy:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout code
        uses: actions/checkout@v3

      - name: Set up Python
        uses: actions/setup-python@v4
        with:
          python-version: '3.9'

      - name: Install dependencies
```

```
        run: |
          pip install -r requirements.txt

      - name: Set up Firebase Credentials
        env:
          FIREBASE_CREDENTIALS: ${{ secrets.FIREBASE_CREDENTIALS }}
          DATABASE_URL: ${{ secrets.DATABASE_URL }}
        run: |
          echo "$FIREBASE_CREDENTIALS" > firebase_credentials.json
          export GOOGLE_APPLICATION_CREDENTIALS=firebase_credentials.json

      - name: Deploy to AWS App Runner without Docker
        env:
          SERVICE_NAME: CalculatorApp
        run: |
          aws apprunner update-service --service-arn ${{ secrets.APP_RUNNER_SERVICE_ARN }}
\
            --source-configuration
CodeRepository={RepositoryUrl=${{ secrets.REPO_URL }},SourceCodeVersion={Type="BRANCH",Val
ue="main"}}
```

8. Configuring GitHub Secrets for Secure Credentials

1. Go to **GitHub Repository > Settings > Secrets**.
2. Click **New repository secret**.
3. Add the following secrets:
    a. `DATABASE_URL`
    b. `FIREBASE_CREDENTIALS` (Base64 encoded JSON key)

9. Deployment Process

1. Push code to the `main` branch.
2. GitHub Actions triggers the CI/CD pipeline.
3. AWS App Runner automatically builds and deploys the Python application.

10. Scaling & Monitoring

- Enable **auto-scaling** in App Runner.
- Use **AWS CloudWatch** for logs and monitoring.
- Configure **Firebase usage alerts** for cost management.

11. Best Practices & Troubleshooting

- **Security**: Use IAM roles and least privilege access.

- **Performance**: Optimize the Python application for efficiency.
- **Debugging**: Use `apprunner list-deployments` to check status.