

Practical Steps: Reproducing the System

The complete project source code is available on GitHub: <https://github.com/szelese/ci-cd-gha-aws>

The solution is built on GitHub Actions automation and the AWS Elastic Beanstalk service. Development utilized Python 3.11 and Django 4.2 LTS, with secure authentication provided by AWS IAM OpenID Connect (OIDC) integration. The project goal was to create a working prototype capable of executing every step automatically, from code modification to production deployment in the cloud.

Project Launch: Accounts, GitHub repo, cloning, README, first commit	0-3
Dev Environment Virtual environment, pip, Django installation, requirements file gitignore, runtime.txt	4-9
CI Workflow & Initial Integration .github/workflows, ci.yml, GitHub Actions, First successful run	10-13
Django Project Creation mysite project, hello app, settings (INSTALLED_APPS, ALLOWED_HOSTS), Views and URLs, migration, local test	14-20
Dependencies & Versioning Requirements update and commit	20
AWS Elastic Beanstalk Prep .ebextensions/django.config, manage.py, deploy.zip, EB App and environment creation, basic settings (platform, domain, VPC, monitoring)	21-31
Basic Deployment & Troubleshooting EB launch, health check, environment variables (DJANGO_SETTINGS_MODULE, DEBUG), Root endpoint modification, log analysis, gunicorn/Procfile, successful deploy	32-41
IAM/OIDC Integration IAM provider and role creation, inline policy, ARN, OIDC smoketest workflow	42-51
CI/CD Automation Deploy workflow, permission refinement (admin policy), post-deploy hook (migration + static files), New environment creation, deploy workflow modification for env name, waiting for "Ready & Green" state	52-64
CI Pipeline Expansion flake8, bandit, pytest installation, DJANGO_SECRET_KEY setup, Pytest	65-77
IAM Policy Minimization (Adhering to the Principle of Least Privilege) Admin policy fix, EBDeployMinimal iterative expansion	78-83

Step 0: Preparations

Before starting, you must prepare the accounts and tools used for development and cloud deployment.

Accounts: Create a GitHub account for source code version control and CI/CD pipelines, and an AWS account for cloud infrastructure hosting.

Tools: I recommend installing GitBash and the ZIP command (via MSYS2).

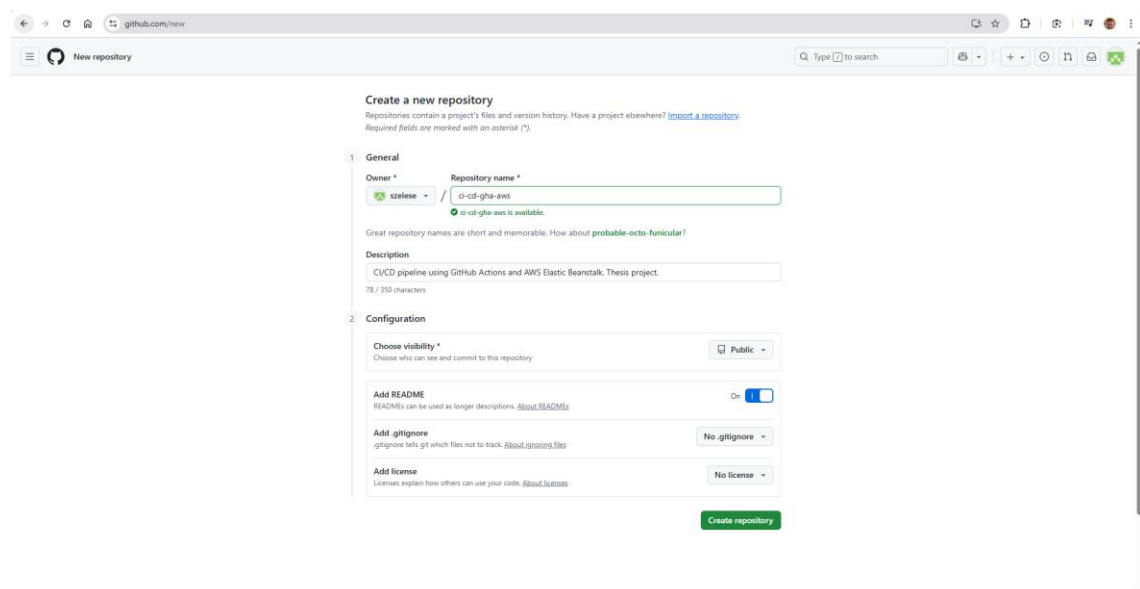
Environment (Optional):

This project was developed using Python 3.11 and managed locally with Visual Studio Code, but it is compatible with other IDEs like PyCharm or IDLE.

While simple text editors work, be cautious of line ending (LF/CRLF) or EOF (End Of File) errors.

Development and testing were performed on Windows 10 Enterprise, but the system can run on any operating system.

Step 1: Create New Repository



The screenshot shows the GitHub 'Create a new repository' page. The form is divided into two sections: 'General' and 'Configuration'.

General

- Owner ***: A dropdown menu showing 'szelise'.
- Repository name ***: A text input field containing 'ci-cd-github-aws'. A green checkmark indicates it is available.
- Description**: A text input field containing 'CI/CD pipeline using GitHub Actions and AWS Elastic Beanstalk. Thesis project.' The character count is 78 / 350.

Configuration

- Choose visibility ***: A dropdown menu set to 'Public'.
- Add README**: A toggle switch set to 'On'.
- Add .gitignore**: A dropdown menu set to 'No .gitignore'.
- Add license**: A dropdown menu set to 'No license'.

A green 'Create repository' button is located at the bottom right of the form.


```
MINGW64/c/szakdolgozat/ci-cd-gha-aws
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ python -c "import django; print(django.get_version())"
4.2.24
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git add requirements.txt
git commit -m "Add Django 4.2 requirements"
git push
[main 4b55783] Add Django 4.2 requirements
 1 file changed, 4 insertions(+)
 create mode 100644 requirements.txt
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 360 bytes | 360.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/szelese/ci-cd-gha-aws.git
 5e858cb..4b55783  main -> main
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ |
```

```
git add requirements.txt
git commit -m „Add Django 4.2 requirements”
```

Step 4: Virtual Environment Creation, Activation, and Pip Upgrade

```
MINGW64/c/szakdolgozat/ci-cd-gha-aws
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ py -m venv .venv
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ source .venv/Scripts/activate
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ python -m pip install --upgrade pip
Requirement already satisfied: pip in c:\szakdolgozat\ci-cd-gha-aws\.venv\lib\site-packages (
25.0.1)
Collecting pip
  Downloading pip-25.2-py3-none-any.whl.metadata (4.7 kB)
  Downloading pip-25.2-py3-none-any.whl (1.8 MB)
  ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.8/1.8 MB 14.3 MB/s eta 0:00:00
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 25.0.1
    Uninstalling pip-25.0.1:
      Successfully uninstalled pip-25.0.1
Successfully installed pip-25.2
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$
```

```
py -m venv .venv
source .venv/Scripts/activate
# (.venv) ha a prompt elején megjeleik, akkor azt jelenti, hogy aktív
python -m pip install --upgrade pip
```

Step 5: Django Installation (LTS 4.2)

```
MINGW64/c/szakdolgozat/ci-cd-gha-aws
Successfully installed pip-25.2
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ pip install "Django==4.2.*"
Collecting Django==4.2.*
  Downloading django-4.2.24-py3-none-any.whl.metadata (4.2 kB)
Collecting asgiref<4,>=3.6.0 (from Django==4.2.*)
  Downloading asgiref-3.9.2-py3-none-any.whl.metadata (9.3 kB)
Collecting sqlparse>=0.3.1 (from Django==4.2.*)
  Downloading sqlparse-0.5.3-py3-none-any.whl.metadata (3.9 kB)
Collecting tzdata (from Django==4.2.*)
  Downloading tzdata-2025.2-py2.py3-none-any.whl.metadata (1.4 kB)
Downloading django-4.2.24-py3-none-any.whl (8.0 MB)
 8.0/8.0 MB 19.6 MB/s 0:00:00
Downloading asgiref-3.9.2-py3-none-any.whl (23 kB)
Downloading sqlparse-0.5.3-py3-none-any.whl (44 kB)
Downloading tzdata-2025.2-py2.py3-none-any.whl (347 kB)
Installing collected packages: tzdata, sqlparse, asgiref, Django
Successfully installed Django-4.2.24 asgiref-3.9.2 sqlparse-0.5.3 tzdata-2025.2
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ |
```

```
pip install "Django==4.2.*"
```

Step 6: Outputting the Requirements File and Quick Verification

```
MINGW64/c/szakdolgozat/ci-cd-gha-aws
  Downloading asgiref-3.9.2-py3-none-any.whl.metadata (9.3 kB)
Collecting sqlparse>=0.3.1 (from Django==4.2.*)
  Downloading sqlparse-0.5.3-py3-none-any.whl.metadata (3.9 kB)
Collecting tzdata (from Django==4.2.*)
  Downloading tzdata-2025.2-py2.py3-none-any.whl.metadata (1.4 kB)
Downloading django-4.2.24-py3-none-any.whl (8.0 MB)
 8.0/8.0 MB 19.6 MB/s 0:00:00
Downloading asgiref-3.9.2-py3-none-any.whl (23 kB)
Downloading sqlparse-0.5.3-py3-none-any.whl (44 kB)
Downloading tzdata-2025.2-py2.py3-none-any.whl (347 kB)
Installing collected packages: tzdata, sqlparse, asgiref, Django
Successfully installed Django-4.2.24 asgiref-3.9.2 sqlparse-0.5.3 tzdata-2025.2
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ pip freeze > requirements.txt
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ python -c "import django; print(django.get_version())"
4.2.24
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$
```

```
pip freeze > requirements.txt
python -c "import django; print(django.get_version())"
# Expected and correct verification value:: 4.2.*
```

Step 7: commit&push

```
MINGW64/c/szakdolgozat/ci-cd-gha-aws
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ python -c "import django; print(django.get_version())"
4.2.24
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git add requirements.txt
git commit -m "Add Django 4.2 requirements"
git push
[main 4b55783] Add Django 4.2 requirements
 1 file changed, 4 insertions(+)
 create mode 100644 requirements.txt
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 360 bytes | 360.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/szelese/ci-cd-gha-aws.git
 5e858cb..4b55783  main -> main
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ |
```

```
git add requirements.txt
git commit -m "Add Django 4.2 requirements"
git push
```

Step 8: .gitignore Creation and Verification

```
MINGW64/c/szakdolgozat/ci-cd-gha-aws
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ printf ".venv/\n__pycache__/\n*.pyc\n.env\n*.sqlite3\n" > .gitignore
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ cat .gitignore
.venv/
__pycache__/
*.pyc
.env
*.sqlite3
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git add .gitignore
git commit -m "Add .gitignore (ignore venv, caches, env, sqlite)"
git push
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the next time Git touches it
[main d8cb82f] Add .gitignore (ignore venv, caches, env, sqlite)
 1 file changed, 5 insertions(+)
 create mode 100644 .gitignore
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 16 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 382 bytes | 382.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/szelese/ci-cd-gha-aws.git
 4b55783..d8cb82f  main -> main
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$
```

Technical justification: The .venv directory is machine and OS-dependent, can be extremely large, and must be rebuilt in CI or on new machines based on

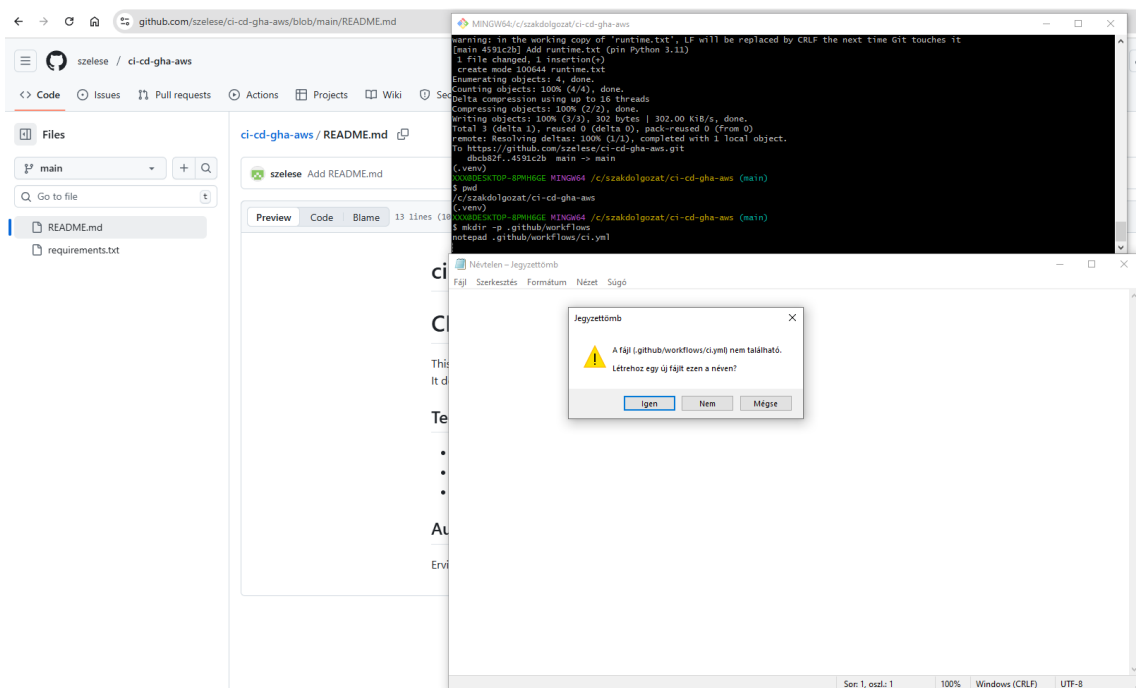
```
printf ".venv/\n__pycache__/\n*.pyc\n.env\n*.sqlite3\n" > .gitignore
cat .gitignore
git status
```

Step 9: Creating runtime.txt with Precise Content, Verification, and Commit/Push

```
MINGW64/c/szakdolgozat/ci-cd-gha-aws
$ echo "python-3.11" > runtime.txt
cat runtime.txt
git add runtime.txt
git commit -m "Add runtime.txt (pin Python 3.11)"
git push
python-3.11
warning: in the working copy of 'runtime.txt', LF will be replaced by CRLF the next time Git touches it
[main 4591c2b] Add runtime.txt (pin Python 3.11)
1 file changed, 1 insertion(+)
create mode 100644 runtime.txt
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 16 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 302 bytes | 302.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/szelese/ci-cd-gha-aws.git
 dcb82f..4591c2b main -> main
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$
```

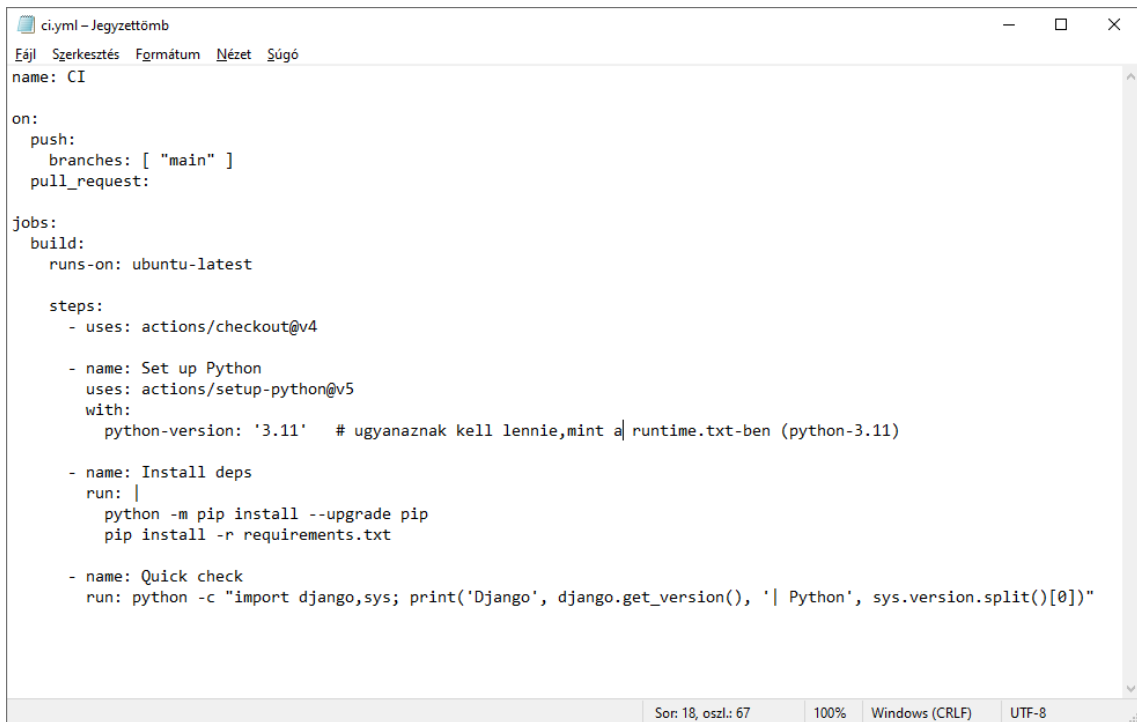
```
$ echo "python-3.11" > runtime.txt
cat runtime.txt
git add runtime.txt
git commit -m "Add runtime.txt (pin Python 3.11)"
git push
python-3.11
```

Step 10: Workflow Folder and ci.yml Creation



```
$ mkdir -p .github/workflows
notepad .github/workflows/ci.yml
```

Step 11: Content of the ci.yml File



```
ci.yml - Jegyzetfőmb
Ejrl Szerkesztés Formátum Nézet Sűgó
name: CI

on:
  push:
    branches: [ "main" ]
    pull_request:

jobs:
  build:
    runs-on: ubuntu-latest

    steps:
      - uses: actions/checkout@v4

      - name: Set up Python
        uses: actions/setup-python@v5
        with:
          python-version: '3.11' # ugyanaznak kell lennie, mint a runtime.txt-ben (python-3.11)

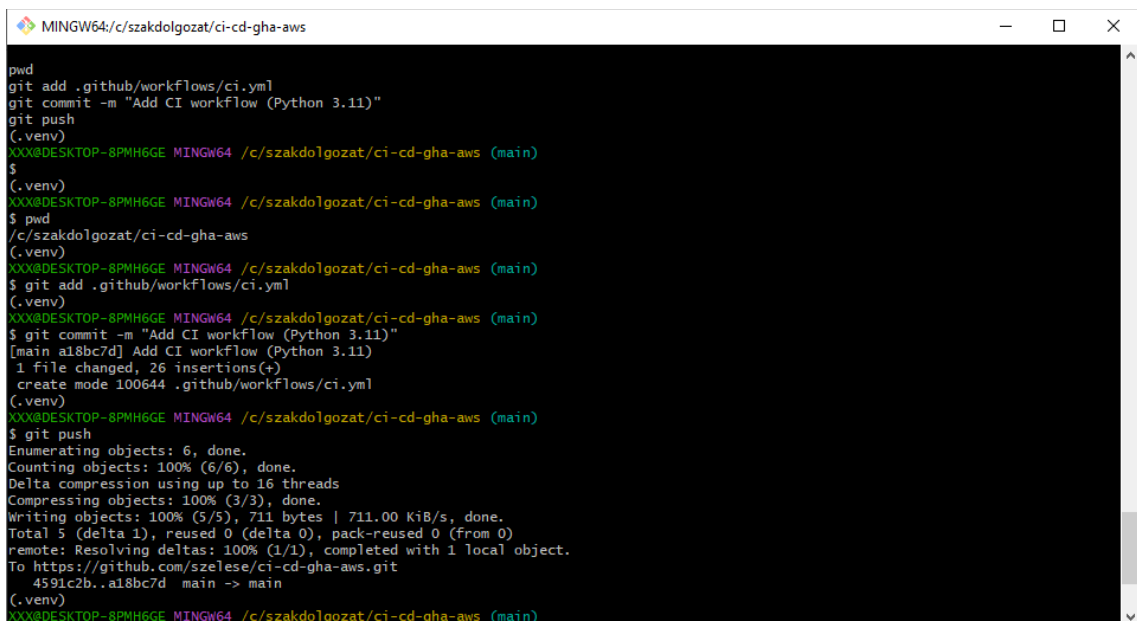
      - name: Install deps
        run: |
          python -m pip install --upgrade pip
          pip install -r requirements.txt

      - name: Quick check
        run: python -c "import django,sys; print('Django', django.get_version(), '| Python', sys.version.split()[0])"

Sor: 18, osz: 67 100% Windows (CRLF) UTF-8
```

The runtime.txt file ensures that Elastic Beanstalk (EB) executes the project using the exact same Python version used during local development.

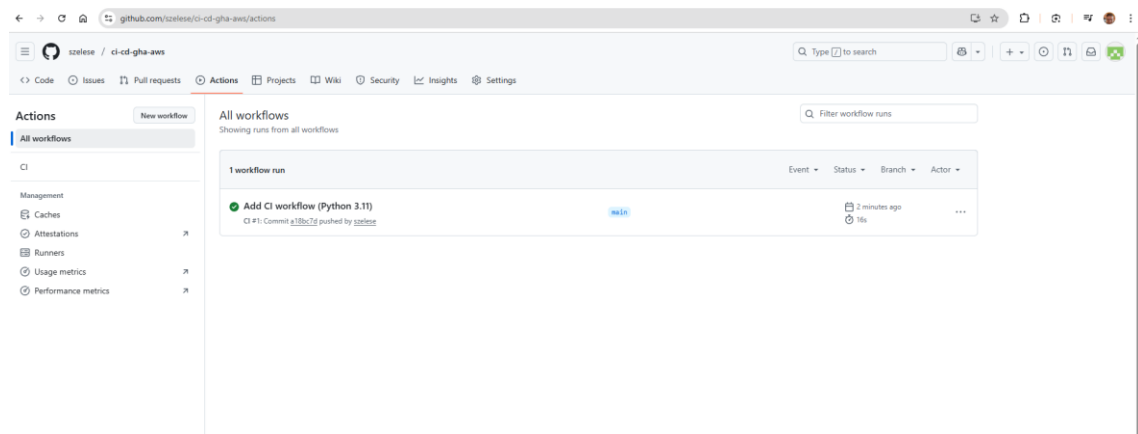
Step 12: commit&push



```
MINGW64:/c/szakdolgozat/ci-cd-gha-aws
pwd
git add .github/workflows/ci.yml
git commit -m "Add CI workflow (Python 3.11)"
git push
(. venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$
(. venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ pwd
/c/szakdolgozat/ci-cd-gha-aws
(. venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git add .github/workflows/ci.yml
(. venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git commit -m "Add CI workflow (Python 3.11)"
[main a18bc7d] Add CI workflow (Python 3.11)
1 file changed, 26 insertions(+)
create mode 100644 .github/workflows/ci.yml
(. venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git push
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (5/5), 711 bytes | 711.00 KiB/s, done.
Total 5 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/szelese/ci-cd-gha-aws.git
4591c2b..a18bc7d main -> main
(. venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
```

```
commit és push ci.yml -t
git add .github/workflows/ci.yml
git commit -m „Add CI workflow (Python 3.11)”
```

Step 13: GitHub Verification



The "CI" run should appear; a green checkmark indicates the process completed successfully.

Step 14: Project and App Creation

```
MINGW64:/c/szakdolgozat/ci-cd-gha-aws
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git commit -m "Add CI workflow (Python 3.11)"
[main a18bc7d] Add CI workflow (Python 3.11)
1 file changed, 26 insertions(+)
 create mode 100644 .github/workflows/ci.yml
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git push
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (5/5), 711 bytes | 711.00 KiB/s, done.
Total 5 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/szelese/ci-cd-gha-aws.git
 4591c2b..a18bc7d main -> main
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git add .github/workflows/ci.yml
git commit -m "Add CI workflow (Python 3.11)"
git push
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
Everything up-to-date
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ django-admin startproject mysite .
python manage.py startapp hello
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ |
```

Initialize the Django project named mysite in the current directory:

```
django-admin startproject mysite .
python manage.py startapp hello
```

Step 15: App Registration and Demo Permissions

```
MINGW64:/c/szakdolgozat/ci-cd-gha-aws
XX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git commit -m "Add CI workflow (Python 3.11)"
[main a18bc7d] Add CI workflow (Python 3.11)
1 file changed, 26 insertions(+)
 create mode 100644 .github/workflows/ci.yml
(.venv)
XX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git push
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (5/5), 711 bytes | 711.00 KiB/s, done.
Total 5 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/szelese/ci-cd-gha-aws.git
 4591c2b..a18bc7d main -> main
(.venv)
XX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git add .github/workflows/ci.yml
$ git commit -m "Add CI workflow (Python 3.11)"
$ git push
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
everything up-to-date
(.venv)
XX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ django-admin startproject mysite .
python manage.py startapp hello
(.venv)
XX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ notepad mysite/settings.py
```

```
ALLOWED_HOSTS = ["*", "localhost", "127.0.0.1"] #ez a sor lett bovitve
# megmondja, hogy milyen hostrol szolgálja ki a Django az oldalt. *-barhonnan masik ketto-helyi futtatas
# elesben nem szabad hasznalni biztonsagi okokbol, kenyelmes a demohoz
```

```
# Application definition
```

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',

    'hello' # ez a sor lett hozza adva
]
```

```
MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
]
```

Sor: 36, oszl.: 27 100% Windows (CRLF) UTF-8

```
notepad mysite/settings.py
INSTALLED_APPS = [
    # ...
    'hello',
]
és
ALLOWED_HOSTS = ["*", "localhost", "127.0.0.1"]
```

Security Warning: The `ALLOWED_HOSTS = ['*']` setting is strictly for initial testing; in a production environment, specific domains must be defined for security reasons !!!

Step 16: Writing a Minimal View

```
MINGW64/c/szakdolgozat/ci-cd-gha-aws
1 file changed, 26 insertions(+)
 create mode 100644 .github/workflows/ci.yml
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git push
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (5/5), 711 bytes | 711.00 KiB/s, done.
Total 5 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/szelese/ci-cd-gha-aws.git
 4591c2b..a18bc7d  main -> main
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git add .github/workflows/ci.yml
git commit -m "Add CI workflow (Python 3.11)"
git push
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
Everything up-to-date
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ django-admin startproject mysite .
python manage.py startapp hello
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ notepad mysite/settings.py
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ notepad hello/views.py
```

```
*views.py - Jegyzetfömb
Ejöl Szerkesztés Formátum Nézet Súgó
from django.http import HttpResponse
```

```
def index(request):
    return HttpResponse("Hello, CI/CD! It works.")
```

```
notepad hello/views.py # views.py megnyitása után:
from django.http import HttpResponse
def index(request):
    return HttpResponse("Hello, CI/CD! It works.")
```

Step 17: Connecting the URL to the Hello Application

```
MINGW64:/c/szakdolgozat/ci-cd-gha-aws
$ git push
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (5/5), 711 bytes | 711.00 KiB/s, done.
Total 5 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/szelese/ci-cd-gha-aws.git
  4591c2b..a18bc7d main -> main
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git add .github/workflows/ci.yml
git commit -m "Add CI workflow (Python 3.11)"
git push
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
Everything up-to-date
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ django-admin startproject mysite .
python manage.py startapp hello
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ notepad mysite/settings.py
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ notepad hello/views.py
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ notepad hello/urls.py

"""urls.py - Jégzettomb

Fájl Szerkesztés Formátum Nézet Súgó
from django.urls import path
from . import views

urlpatterns = [
    path("", views.index, name="index"),
]
```

Create and open the app-level URL file: notepad hello/urls.py.

```
from django.urls import path
from . import views
urlpatterns = [    path("", views.index, name="index"),]
```

Step 18: Connecting App URLs to the Main Project

```
MINGW64/c/szakdolgozat/ci-cd-gha-aws
Counting objects: 100% (6/6), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (5/5), 711 bytes | 711.00 KiB/s, done.
Total 5 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/szelese/ci-cd-gha-aws.git
   4591c2b..a18bc7d  main -> main
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git add .github/workflows/ci.yml
git commit -m "Add CI workflow (Python 3.11)"
git push
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
Everything up-to-date
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ django-admin startproject mysite .
python manage.py startapp hello
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ notepad mysite/settings.py
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ notepad hello/views.py
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ notepad hello/urls.py
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ notepad mysite/urls.py
```

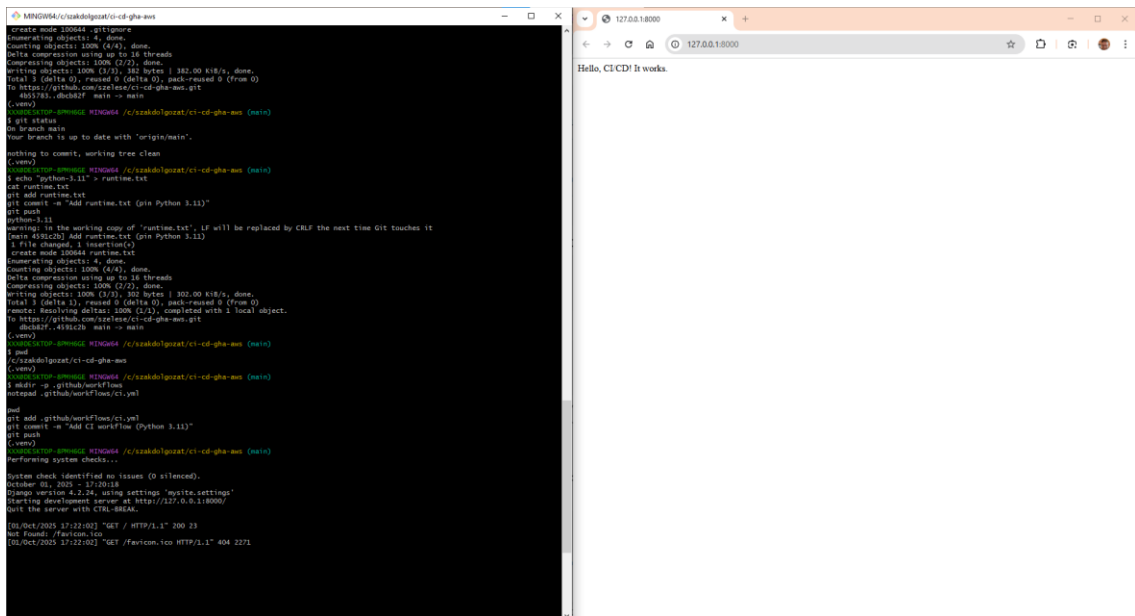
```
urls.py - Jegyzetömb
Fájl Szerkesztés Formátum Nézet Súgó
****
URL configuration for mysite project.

The `urlpatterns` list routes URLs to views. For more information please see:
    https://docs.djangoproject.com/en/4.2/topics/http/urls/
Examples:
Function views
    1. Add an import: from my_app import views
    2. Add a URL to urlpatterns: path('', views.home, name='home')
Class-based views
    1. Add an import: from other_app.views import Home
    2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
****
from django.contrib import admin
from django.urls import path, include # itt az includedal van kiegészítve

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('hello.urls')), # ez a sor lett hozzáadva
]
```

```
notepad mysite/urls.py # megnyitas és modositas:
from django.contrib import admin
from django.urls import path, include
urlpatterns = [ path('admin/', admin.site.urls),
               path('', include('hello.urls')),]
```

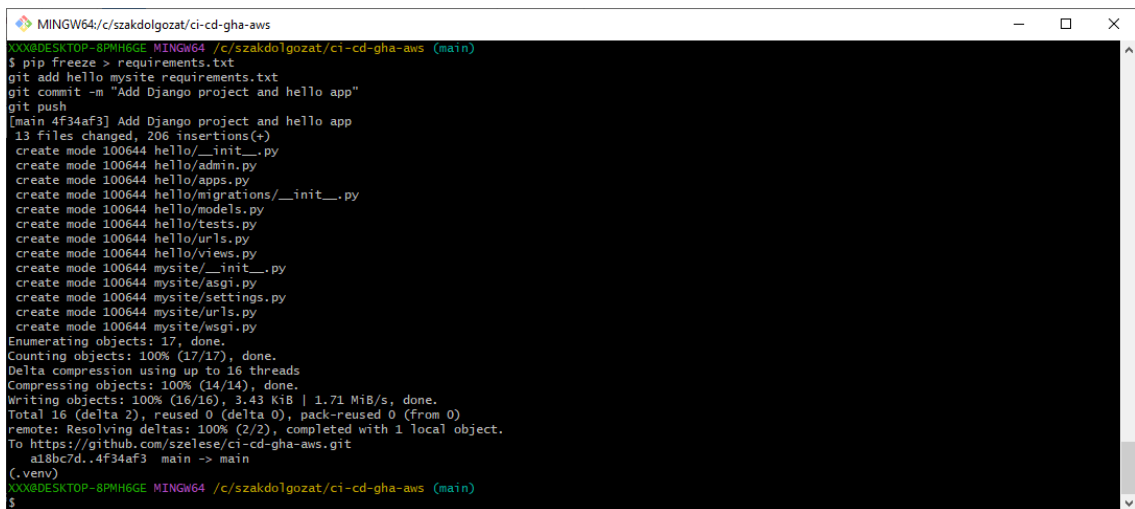
Step 19: Database Initialization and Local Run Test



```
MINGW64/c/szakdolgozat/ci-cd-gha-aws
$ git ignore
$ git add runtime.txt
$ git commit -m "Add runtime.txt (pin Python 3.11)"
$ git push
python:3.11
WARNING: In the working copy of "runtime.txt", LF will be replaced by CRLF the next time Git touches it
1 file changed, 1 insertion(+), 0 deletions(-)
$ git push
$ python manage.py migrate
$ python manage.py runserver
$ curl http://127.0.0.1:8000/
Hello, CI/CD! It works.
```

```
python manage.py migrate # db inicializálása
python manage.py runserver
# utána a böngészőben tesztelés : http://127.0.0.1:8000/
```

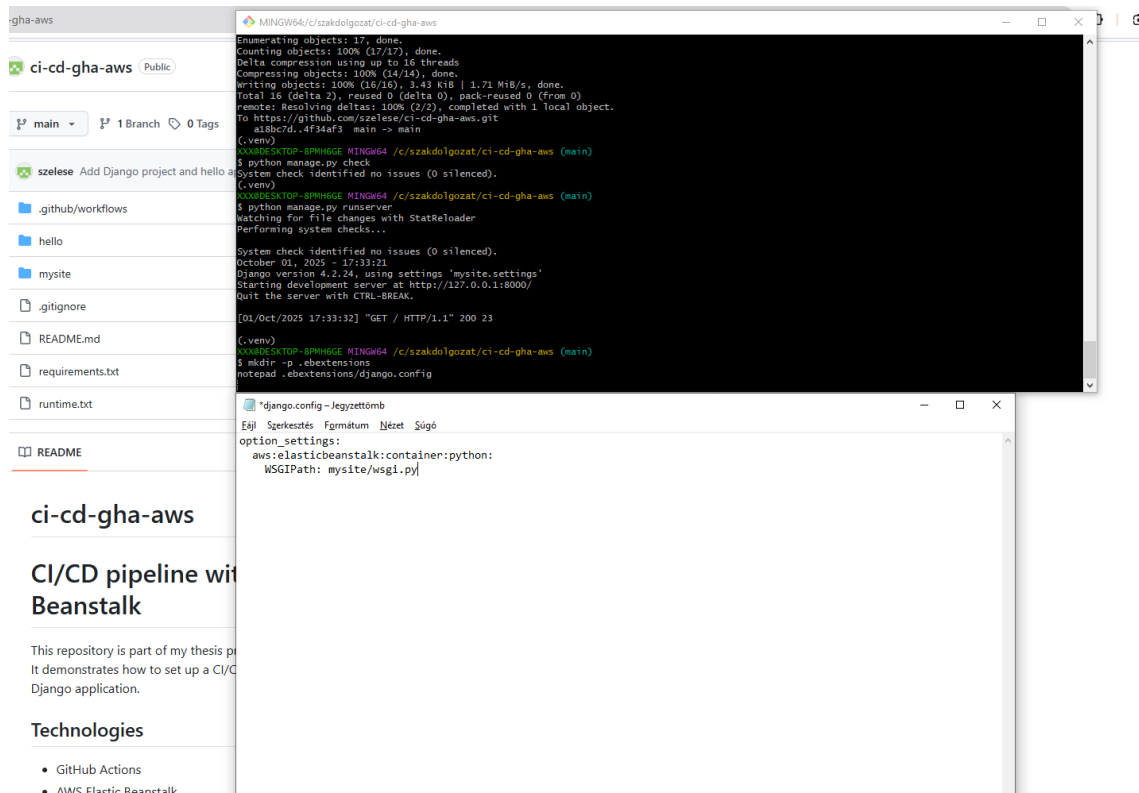
Step 20: Updating Requirements and Pushing to GitHub



```
MINGW64/c/szakdolgozat/ci-cd-gha-aws
$ pip freeze > requirements.txt
$ git add hello mysite requirements.txt
$ git commit -m "Add Django project and hello app"
$ git push
[main 4f34af3] Add Django project and hello app
13 files changed, 206 insertions(+)
create mode 100644 hello/__init__.py
create mode 100644 hello/admin.py
create mode 100644 hello/apps.py
create mode 100644 hello/migrations/__init__.py
create mode 100644 hello/models.py
create mode 100644 hello/tests.py
create mode 100644 hello/urls.py
create mode 100644 hello/views.py
create mode 100644 mysite/__init__.py
create mode 100644 mysite/asgi.py
create mode 100644 mysite/settings.py
create mode 100644 mysite/urls.py
create mode 100644 mysite/wsgi.py
Enumerating objects: 17, done.
Counting objects: 100% (17/17), done.
Delta compression using up to 16 threads
Compressing objects: 100% (14/14), done.
Writing objects: 100% (16/16), 3.43 KiB | 1.71 MiB/s, done.
Total 16 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
To https://github.com/szelese/ci-cd-gha-aws.git
a18bc7d..4f34af3 main -> main
(.venv)
MINGW64/c/szakdolgozat/ci-cd-gha-aws
$
```

```
pip freeze > requirements.txt
git add hello mysite requirements.txt
git commit -m "Add Django project and hello app"
git push
```

Step 21: Configuration for Elastic Beanstalk (EB)



The screenshot shows a GitHub repository for 'ci-cd-gha-aws' with a terminal window open. The terminal displays the following commands and output:

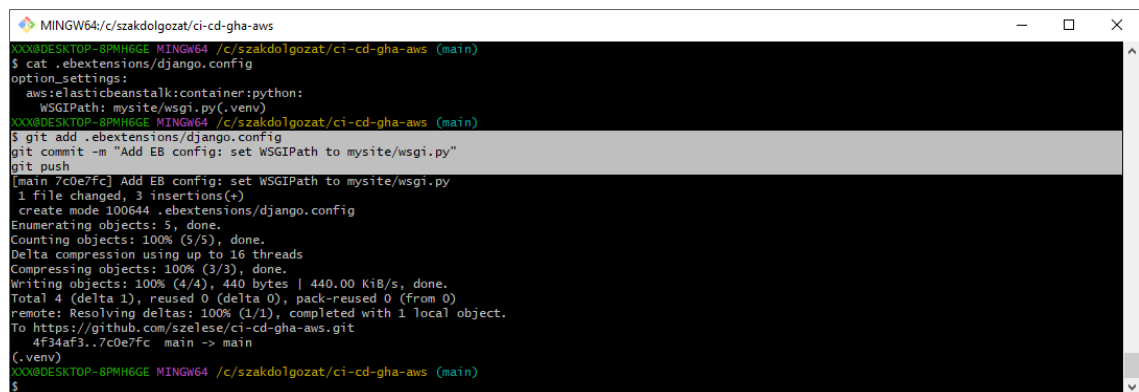
```
Enumerating objects: 17, done.
Counting objects: 100% (17/17), done.
Delta compression using up to 16 threads
Compressing objects: 100% (14/14), done.
Writing objects: 100% (16/16), 3.43 KiB | 1.71 MiB/s, done.
Total 16 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
To https://github.com/szelese/ci-cd-gha-aws.git
a18bc7d..4f34af3 main -> main
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ python manage.py check
System check identified no issues (0 silenced).
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...
System check identified no issues (0 silenced).
October 01, 2025 - 17:33:21
Django version 4.2.24, using settings 'mysite.settings'
Starting development server at https://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
[01/Oct/2025 17:33:32] "GET / HTTP/1.1" 200 23
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ mkdir -p .ebextensions
notepad .ebextensions/django.config
```

The terminal also shows the content of the `django.config` file:

```
option_settings:
  aws:elasticbeanstalk:container:python:
    WSGIPath: mysite/wsgi.py
```

```
mkdir -p .ebextensions
notepad .ebextensions/django.config
django.config file tartalma: # spaces!!!
option_settings:
  aws:elasticbeanstalk:container:python:
    WSGIPath: mysite/wsgi.py
```

Step 22: Verification, Commit & Push



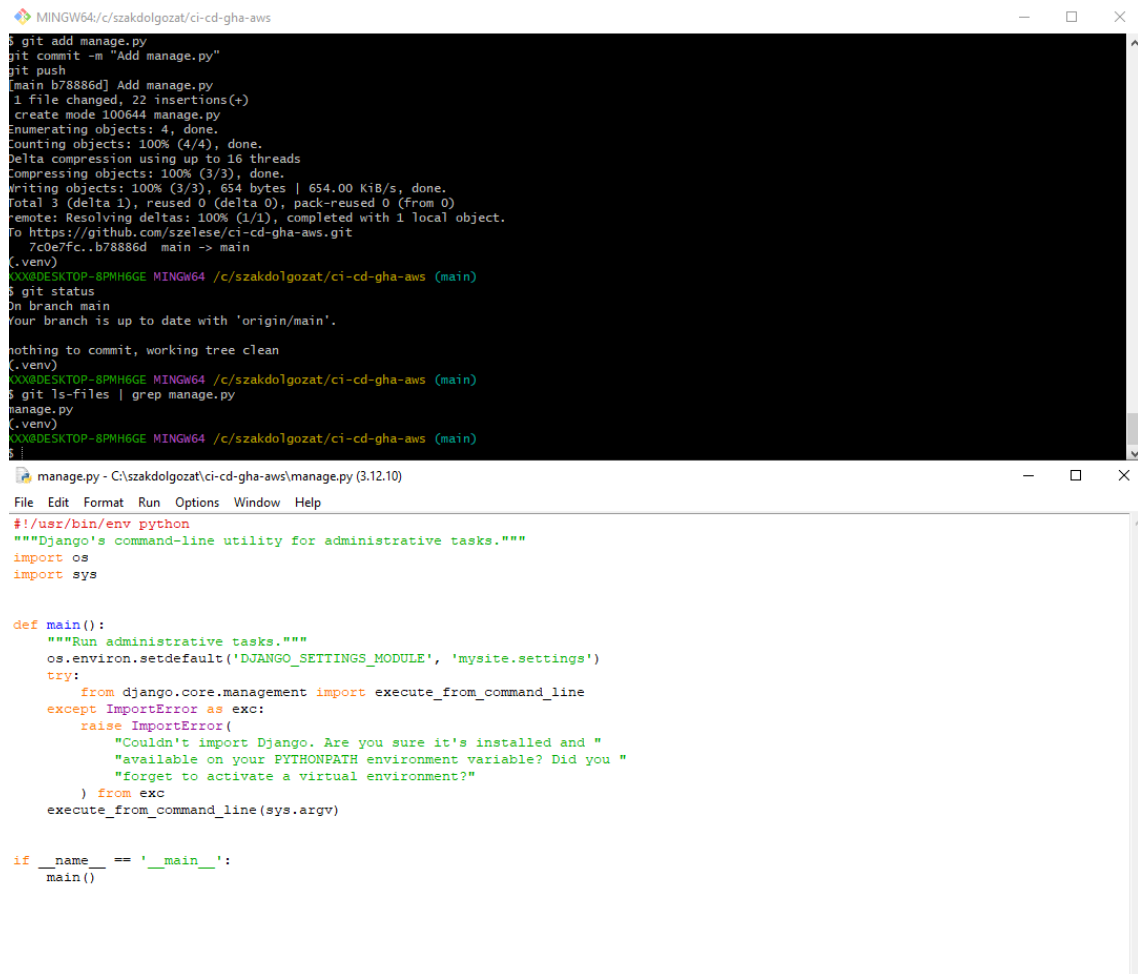
The terminal window shows the following commands and output:

```
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ cat .ebextensions/django.config
option_settings:
  aws:elasticbeanstalk:container:python:
    WSGIPath: mysite/wsgi.py(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git add .ebextensions/django.config
git commit -m "Add EB config: set WSGIPath to mysite/wsgi.py"
git push
[main 7c0e7fc] Add EB config: set WSGIPath to mysite/wsgi.py
1 file changed, 3 insertions(+)
create mode 100644 .ebextensions/django.config
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 440 bytes | 440.00 KiB/s, done.
Total 4 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/szelese/ci-cd-gha-aws.git
4f34af3..7c0e7fc main -> main
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$
```

commit&push után még egy ellenőrzés a githubon, hogy létrejött a file ezzel a tartalommal

```
cat .ebextensions/django.config
git add .ebextensions/django.config
git commit -m "Add EB config: set WSGIPath to mysite/wsgi.py"
git push
```

Step 23: Setting up manage.py



```
MINGW64: c:/szakdolgozat/ci-cd-gha-aws
$ git add manage.py
$ git commit -m "Add manage.py"
$ git push
[main b78886d] Add manage.py
1 file changed, 22 insertions(+)
create mode 100644 manage.py
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 654 bytes | 654.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/szelese/ci-cd-gha-aws.git
7c0e7fc..b78886d main -> main
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git ls-files | grep manage.py
manage.py
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
:
manage.py - C:\szakdolgozat\ci-cd-gha-aws\manage.py (3,12,10)
File Edit Format Run Options Window Help
#!/usr/bin/env python
"""Django's command-line utility for administrative tasks."""
import os
import sys


def main():
    """Run administrative tasks."""
    os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'mysite.settings')
    try:
        from django.core.management import execute_from_command_line
    except ImportError as exc:
        raise ImportError(
            "Couldn't import Django. Are you sure it's installed and "
            "available on your PYTHONPATH environment variable? Did you "
            "forget to activate a virtual environment?"
        ) from exc
    execute_from_command_line(sys.argv)

if __name__ == '__main__':
    main()
```

Technical Note: While manage.py is not strictly mandatory for running on EB (as it uses WSGI), it is essential for local development, migrations, collectstatic, and future automation.

```
notepad manage.py
git add manage.py
git commit -m "Add manage.py"
git push
```

Step 24: Creating the deploy.zip Bundle

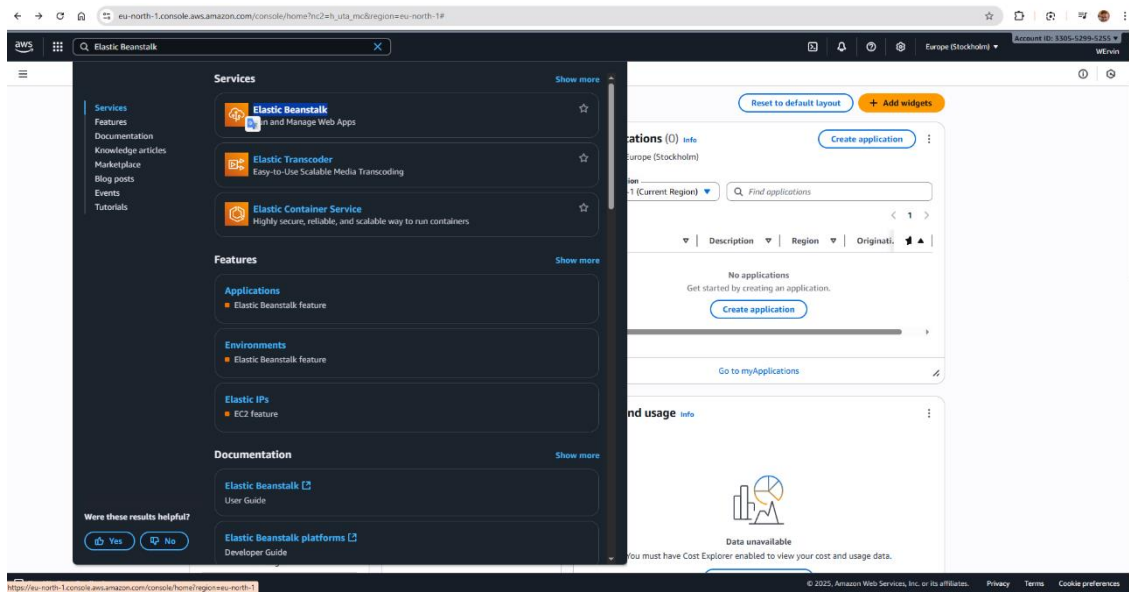


```
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ $ zip -r deploy.zip . -x ".git/*" ".venv/*" "__pycache__/*" "*.pyc"
```

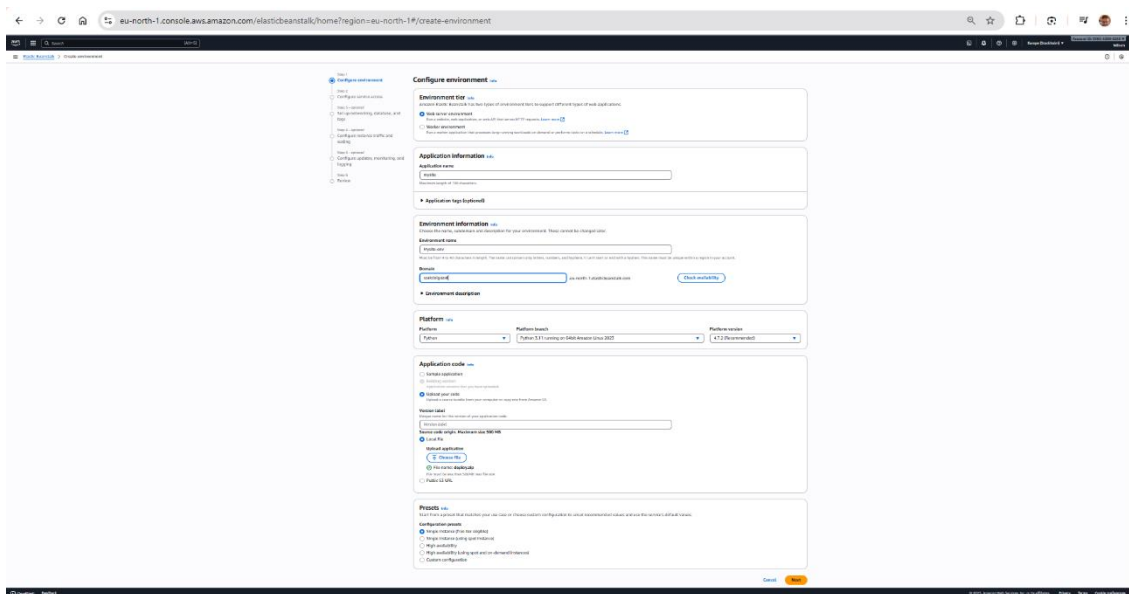
```
zip -r deploy.zip . -x ".git/*" ".venv/*" "__pycache__/*" "*.pyc"
```

Note: The zip tool from MSYS2 was used to handle this via command line, but it can also be done via File Explorer as long as .git and .venv are excluded.

25. lépés: AWS search / Elastic Beanstalk/Create Application



Step 26: AWS Environment Configuration



Environment tier: Web Server Environment # default

Application information: mysite

environment information:

environment name: Mysite.env

Domain: szakdolgozat # or any unique, valid string

Platform: Python

platform branch: Python 3.11 # matching the runtime.txt specification

Version label: mysite-v1

Application code: Upload your code ->local file-> deploy.zip # the bundle generated in Step 24

Presets:

Single instance (free tier eligible) # which is sufficient for demonstration purposes

Step 27: Step 2 - Configure Service Access

The screenshot shows the 'Configure service access' step in the AWS Elastic Beanstalk console. The progress bar on the left indicates that Step 2 is the current step. The main content area is titled 'Configure service access' and contains three sections: 'Service access' (IAM roles), 'Service role' (aws-elasticbeanstalk-service-role), 'EC2 instance profile' (Choose an IAM instance profile), and 'EC2 key pair - optional' (Choose an EC2 key pair). Each section has a 'Create role' or 'Choose' button. At the bottom right, there are 'Cancel', 'Skip to review', 'Previous', and 'Next' buttons.

Service role: For the service role, approve the automatically populated roles and select the specified IAM roles. EC2 key pair: The EC2 key pair (optional) field can be left empty.

Technical Note: SSH settings only need to be configured if you intend to control the server directly from a console.

Step 28: Step 3 - Networking, Database, and Tags (Optional)

The screenshot shows the 'Set up networking, database, and tags - optional' step in the AWS Elastic Beanstalk console. The progress bar on the left indicates that Step 3 is the current step. The main content area is titled 'Set up networking, database, and tags - optional' and contains three sections: 'Instance settings' (Choose a subnet), 'VPC' (Launch your environment in a custom VPC), 'Public IP address' (Assign a public IP address), 'Instance subnets' (Table with columns: Availability Zone, Subnet, CIDR, Name), 'Database' (Integrate an RDS SQL database), and 'Tags' (Apply up to 50 tags). At the bottom right, there are 'Cancel', 'Skip to review', 'Previous', and 'Next' buttons.

VPC: Leave at the default - value; do not create a new VPC or select a subnet.

Public IP address: This can remain disabled; the Single instance preset makes the instance public by default.

Adjusting access: If the instance is not accessible externally later, it can be enabled via Environment → Configuration → Instances → Public IP: Enable.

Database: Do not enable (RDS is not used in this phase).

Tags: Optional; leave at default settings.

Step 29: Step 4 - Configure Instance Traffic and Scaling (Optional)

Configure instance traffic and scaling - optional

Instances

Configure the Amazon EC2 instances that run your application.

Root volume (boot device)

Root volume type	Size	IOPS	Throughput
(Container default)	GB	100 IOPS	125 MB/s

Amazon CloudWatch monitoring

Monitoring interval: 5 minute

Instance metadata service (IMDS)

IMDSv1: Disable

EC2 security groups

Choose security groups

Capacity

Auto scaling group

Environment type: Single Instance

Fleet composition: On-Demand Instance

Architecture: x86_64

Instance types:

- t3.micro
- t3.small

AMI ID: ami-010a32556bd1aaf6

Buttons: Cancel, Skip to review, Previous, Next

Root volume: Container default

Monitoring interval: 5 minutes. IMDSv1: Disable (ensures only IMDSv2 is permitted)

EC2 security groups: leave as the default EB groups.

Capacity: Environment type: Single instance

Fleet composition: On-Demand Architecture: x86_64

Instance types: 1) t3.micro 2) (opcíó) t3.small

AMI ID: Leave at the default value.

Step 30: Step 5 - Updates, Monitoring, and Logging (Optional)

The screenshot displays the 'Configure updates, monitoring, and logging - optional' configuration page in the AWS IAM console. The page is organized into several sections, each with a title and a brief description of the feature. The 'Monitoring' section includes 'Health reporting' (with 'System' set to 'Basic' and 'Enhanced' selected) and 'Health event streaming to CloudWatch logs'. The 'Managed platform updates' section has 'Managed updates' enabled, with a 'Weekly update window' set to 'Monday' through 'Sunday' and 'UTC'. The 'Email notifications' section has a 'Email' address field. The 'Rolling updates and deployments' section includes 'Application deployments' (with 'Deployment policy' set to 'All at once'), 'Configuration updates' (with 'Rolling update type' set to 'Shutdown'), and 'Deployment preferences' (with 'System health check' set to 'None', 'Health threshold' set to '5s', and 'Command timeout' set to '60s'). The 'Platform software' section includes 'Container options' (with 'Proxy server' set to 'Ignore'), 'Amazon E-Kit' (with 'Amazon E-Kit' set to 'None'), 'X-Ray daemon' (with 'X-Ray daemon' set to 'None'), 'S3 log storage' (with 'S3 log storage' set to 'None'), and 'Instance log streaming to CloudWatch logs' (with 'Instance log streaming to CloudWatch logs' set to 'None'). At the bottom, there is a table for 'Environment properties' with columns for 'Name', 'Value', and 'Action'. The table contains one row with 'Name' set to 'PYTHON_VERSION', 'Value' set to '3.7.10', and 'Action' set to 'Remove'. The page ends with 'Cancel', 'Previous', and 'Next' buttons.

Standard settings: The default values are suitable for this project

Step 31: Step 6 - Review and Create

Step 1: Configure environment

Step 2: Configure service access

Step 3: Set up networking, database, and tags

Step 4: Configure instance traffic and scaling

Step 5: Configure updates, monitoring, and logging

Review

Step 1: Configure environment edit

Environment information

<p>Environment tier Web server environment</p> <p>Environment name MyApp-env</p> <p>Platform aws-ec2-elasticbeanstalk-us-east-1-platform/Python3.11 running on 64-bit Amazon Linux 2023/4.2.2</p>	<p>Application name myapp</p> <p>Application code deploy.zip</p>
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------

Step 2: Configure service access edit

Service access

Configure the service role and EC2 instance profile that Elastic Beanstalk uses to manage your environment. Choose an EC2 key pair to securely log in to your EC2 instances.

<p>Service role arn:aws:iam::330652966255:role/aw-elasticbeanstalk-service-role</p>	<p>EC2 instance profile aws-elasticbeanstalk-ec2-role</p>
------------------------------------------------------------------------------------------------	----------------------------------------------------------------------

Step 3: Set up networking, database, and tags edit

Networking, database, and tags

Configure VPC settings, and submit for your environment's EC2 instances and load balancer. Set up an Amazon RDS database that's integrated with your environment.

No options configured

Tags

Key	Value
No tags There are no tags defined.	

Step 4: Configure instance traffic and scaling edit

Instance traffic and scaling

Customize the capacity and scaling for your environment's instances. Select security groups to control instance traffic. Configure the software that runs on your environment's instances by setting platform-specific options.

<p>Instances</p> <p>Instance Disabled</p>	<p>Capacity</p> <table border="0" style="width: 100%;"> <tr> <td style="width: 33%;"> <p>Environment type Single-Instance</p> </td> <td style="width: 33%;"> <p>Fleet composition On-Demand Instance</p> </td> <td style="width: 33%;"> <p>On-demand above base 20</p> </td> </tr> <tr> <td> <p>Processor type x86_64</p> </td> <td> <p>Capacity rebalancing Enabled</p> </td> <td> <p>Instance type t3.micro, t3.small</p> </td> </tr> </table>	<p>Environment type Single-Instance</p>	<p>Fleet composition On-Demand Instance</p>	<p>On-demand above base 20</p>	<p>Processor type x86_64</p>	<p>Capacity rebalancing Enabled</p>	<p>Instance type t3.micro, t3.small</p>	<p>On-demand base 0</p> <p>Scaling cooldown 300</p> <p>AMI ID ami-0156335542af1aef6</p>
<p>Environment type Single-Instance</p>	<p>Fleet composition On-Demand Instance</p>	<p>On-demand above base 20</p>						
<p>Processor type x86_64</p>	<p>Capacity rebalancing Enabled</p>	<p>Instance type t3.micro, t3.small</p>						

Step 5: Configure updates, monitoring, and logging edit

Updates, monitoring, and logging

Define when and how Elastic Beanstalk deploys changes to your environment. Manage your application's monitoring and logging settings, instances, and other environment resources.

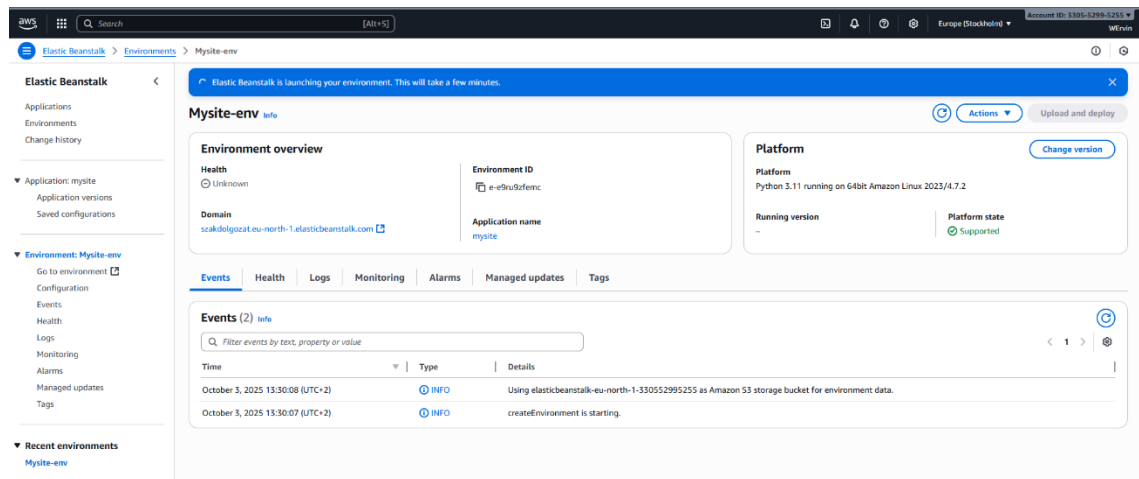
<p>Monitoring</p> <p>System enhanced</p> <p>Log streaming Disabled</p> <p>Updates</p> <p>Managed updates Enabled</p> <p>Command timeout 600</p> <p>Ignore health check false</p> <p>Platform software</p> <p>Lifecycle false</p> <p>NumThreads 16</p> <p>Log retention 7</p> <p>Key enabled Disabled</p>	<p>Cloudwatch custom metrics - Instance —</p> <p>Version T</p> <p>Deployment batch size 100</p> <p>Deployment policy AllAtOnce</p> <p>Instance replacement false</p> <p>Log streaming Disabled</p> <p>WSGIPath application</p> <p>Instance logs Disabled</p>	<p>Cloudwatch custom metrics - environment —</p> <p>Lifecycle false</p> <p>Deployment batch size type Percentage</p> <p>Health threshold OK</p> <p>NumProcesses 1</p> <p>Proxy server nginx</p> <p>Update level minor</p>
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Environment properties

Source	Key	Value
Plain text	PATH_SEPARATOR	/var/app/elasticbeanstalk-us-east-1-1234567890/bin

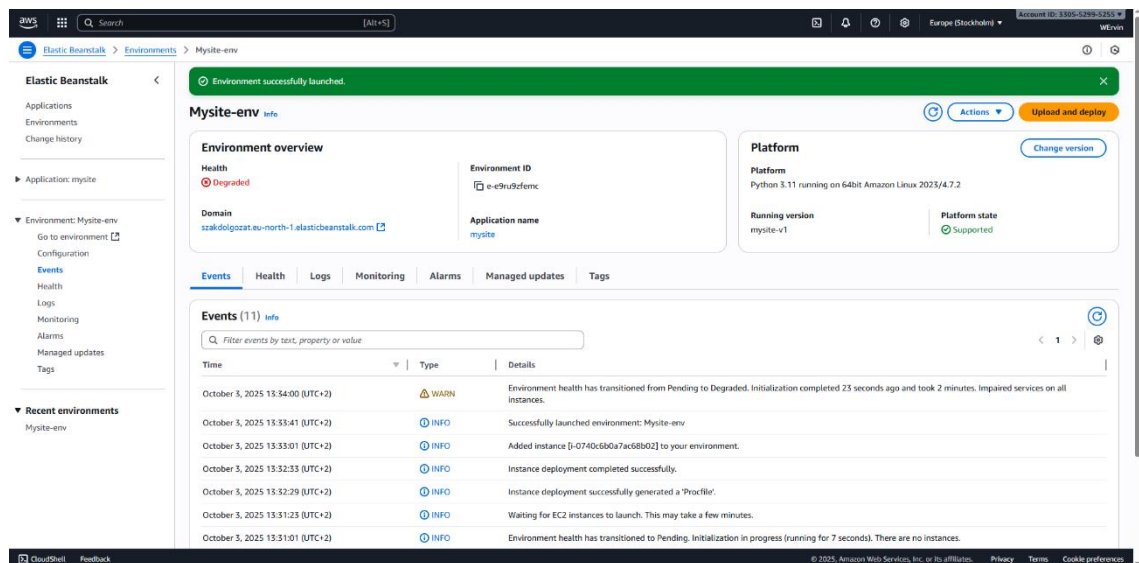
[Cancel](#)
[Previous](#)
[Create](#)

Step 32: EB Environment Launch and Verification



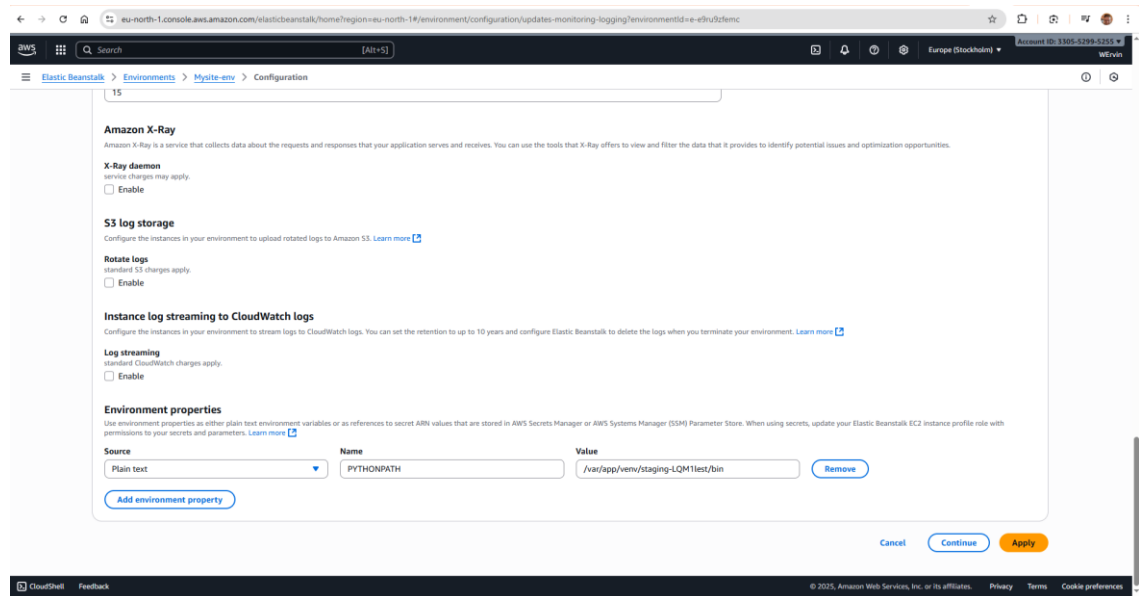
When the Health tab reaches a green "OK" status, the Domain link becomes immediately accessible.

Step 33: Health Check Troubleshooting



This status almost always indicates that the Elastic Beanstalk (EB) health check is receiving a 400, 404, or 5xx error instead of a 200 OK response for the root (/) URL.

Step 34: Adding Environment Properties



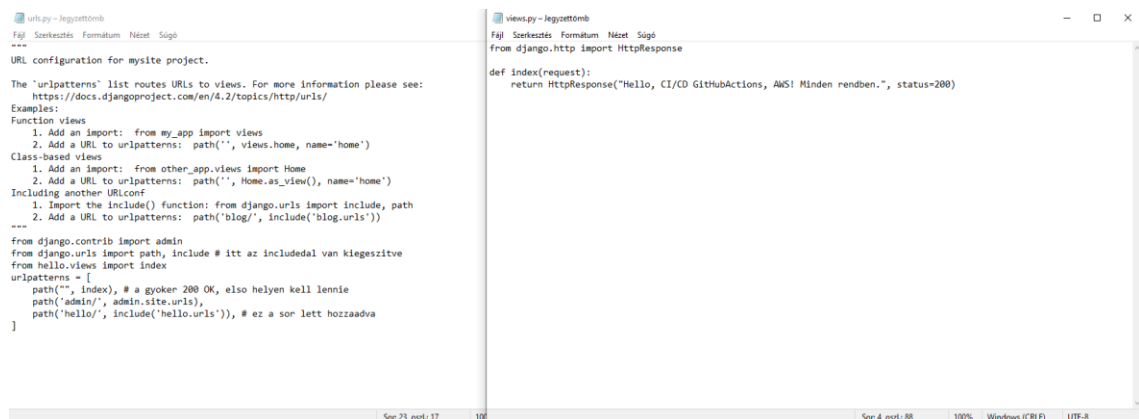
Elastic Beanstalk > Environments > Mysite_env > Configuration > (bottom)
Environment properties > Add

Key: DJANGO_SETTINGS_MODULE → Value: mysite.settings

(optional) Key: DEBUG → Value: False

(optional) Key: SECRET_KEY → Value: A12345678b

Step 35: Local File Modification for Health Check

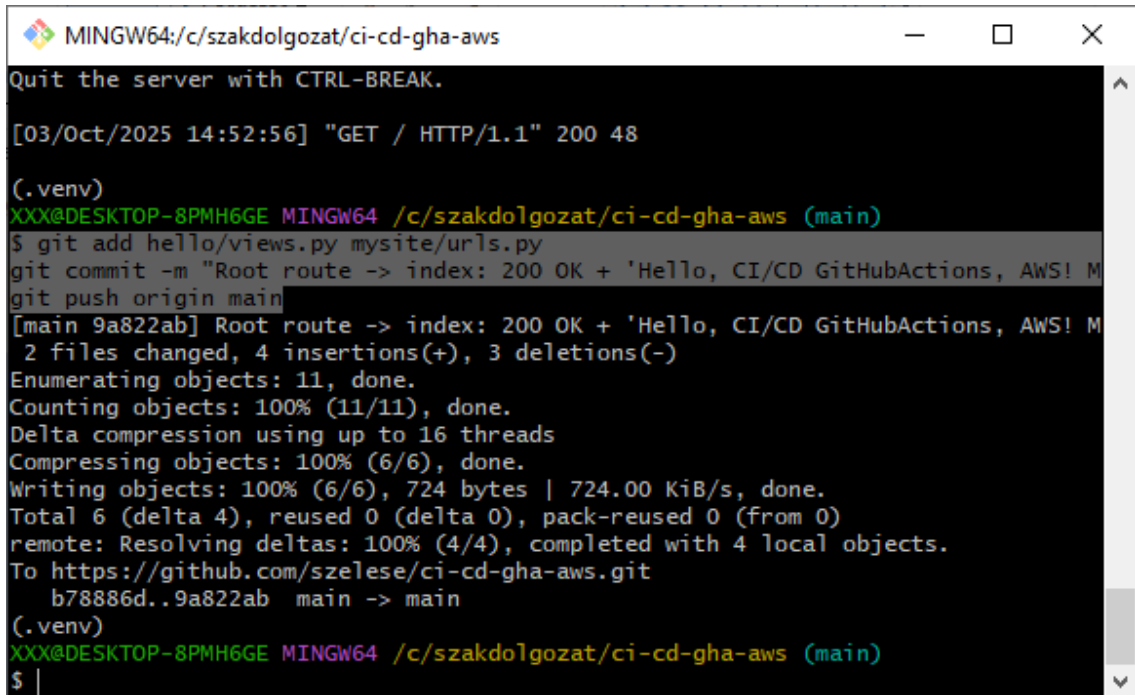


hello/views.py :

```
from django.http import HttpResponseRedirect
def index(request):
    return HttpResponseRedirect("Hello, CI/CD GitHubActions, AWS! ALL OK.", status=200)
mysite/urls.py :
```

```
path("", index), # a gyoker 200 OK, also helyen kell lennie
python manage.py runserver # local test if everything is OK:
zip -r deploy.zip . \
  -x '.git/*' '.venv/*' '__pycache__/*' '*.pyc' 'deploy.zip'
```

Step 36: GitBash Commit and Push

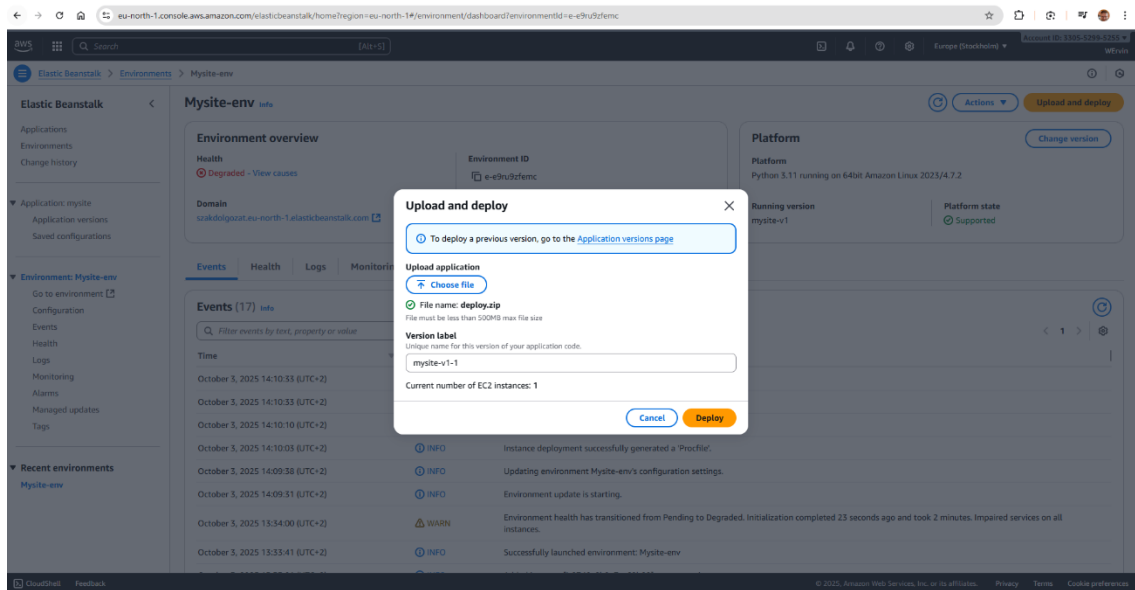


```
MINGW64:/c/szakdolgozat/ci-cd-gha-aws
Quit the server with CTRL-BREAK.
[03/Oct/2025 14:52:56] "GET / HTTP/1.1" 200 48
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git add hello/views.py mysite/urls.py
git commit -m "Root route -> index: 200 OK + 'Hello, CI/CD GitHubActions, AWS! Minden rendben.'"
git push origin main
[main 9a822ab] Root route -> index: 200 OK + 'Hello, CI/CD GitHubActions, AWS! Minden rendben.'"
 2 files changed, 4 insertions(+), 3 deletions(-)
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 16 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 724 bytes | 724.00 KiB/s, done.
Total 6 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 4 local objects.
To https://github.com/szelese/ci-cd-gha-aws.git
   b78886d..9a822ab  main -> main
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ |
```

```
git add hello/views.py mysite/urls.py
git commit -m "Root route -> index: 200 OK + 'Hello, CI/CD GitHubActions, AWS! Minden rendben.'"
git push origin main
```

Technical Note: Frequent small commits and pushes are central to the CI/CD process. Every git push command automatically triggers the GitHub Actions CI process (ci.yml), ensuring the system tests itself immediately after every modification.

Step 37: Manual Upload to Elastic Beanstalk



AWS Console → Elastic Beanstalk → Environments → Mysite-env

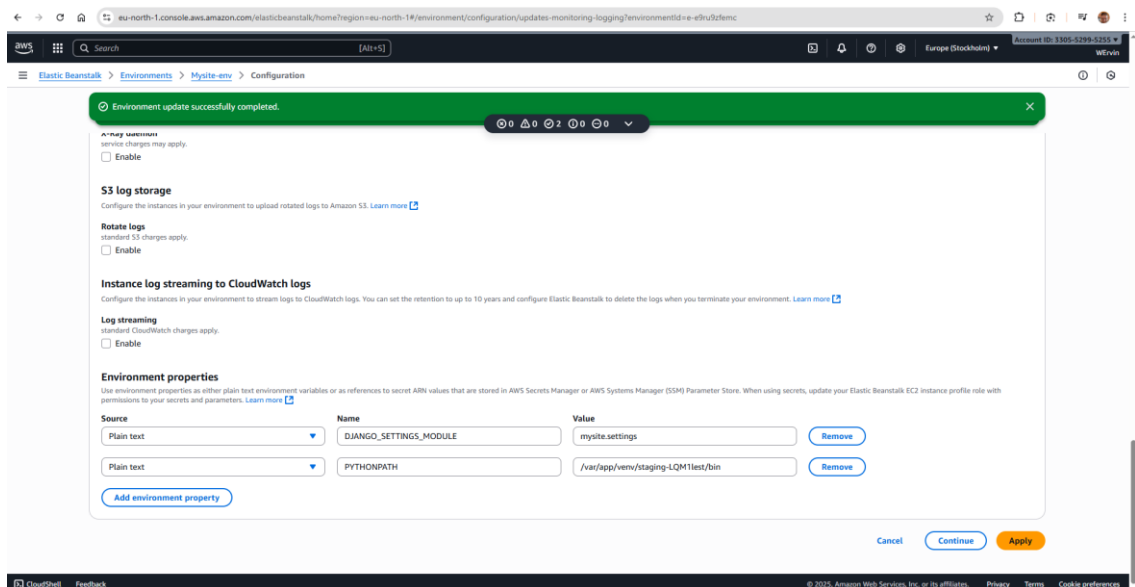
Top right corner: Upload and deploy.

Choose file → select the updated deploy.zip.

Version label: pl. mysite-v1-1 → Deploy.

Step 38: Gunicorn Configuration and Troubleshooting

Failed Attempt: Modifying `/var/app/current` due to `PYTHONPATH` issues did not resolve the error



Log file analysis revealed an ImportError: No module named 'mysite/wsgi'

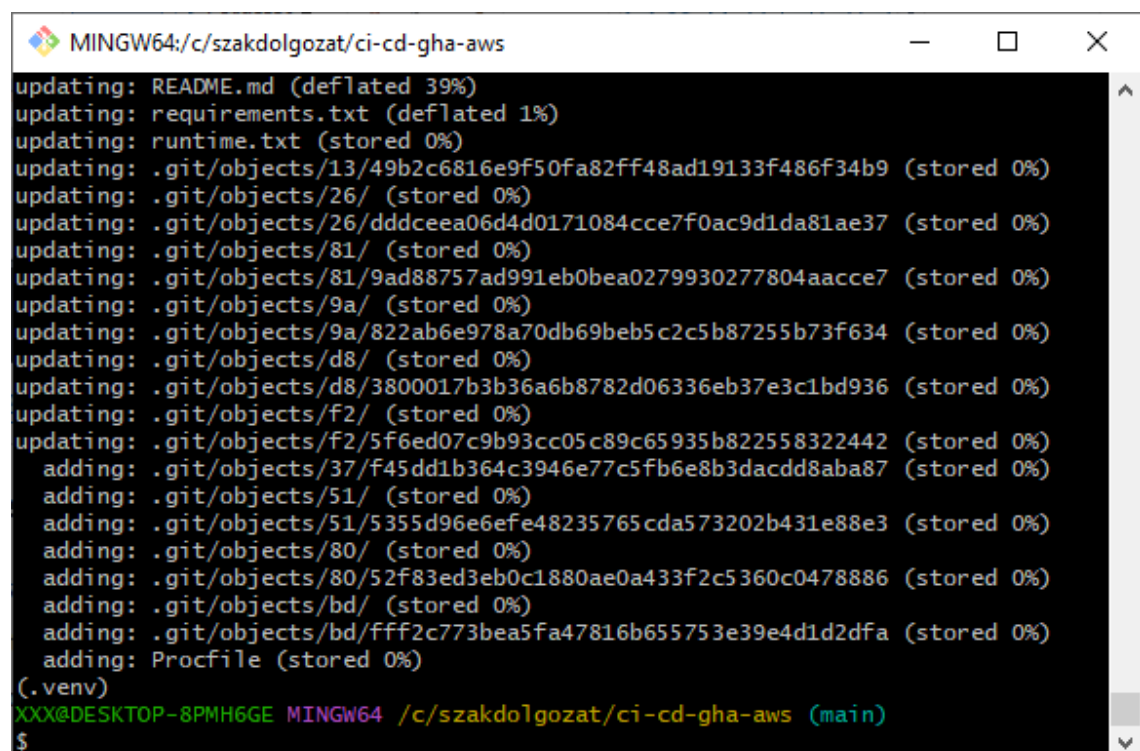
Diagnosis: Gunicorn requires module names to use dots rather than slashes. The correct entry point is `mysite.wsgi:application`.

Currently, EB attempts to launch it as `mysite/wsgi`, which causes a 502 Bad Gateway error. These troubleshooting steps demonstrate practical error handling and testing.

Step 39: Updating Requirements

```
requirements.txt - Jeg
Fájl Szerkesztés Formá
asgiref==3.9.2
Django==4.2.24
sqlparse==0.5.3
tzdata==2025.2
gunicorn>=20
```

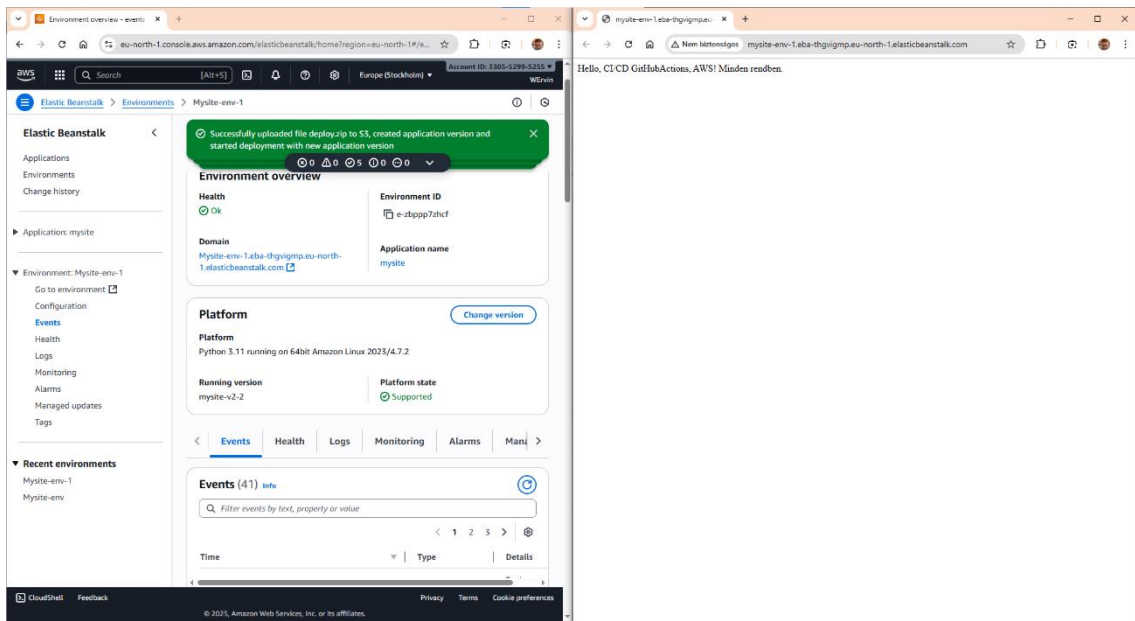
Step 40: Creating the Procfile and Finalizing Push

A terminal window titled 'MINGW64:/c/szakdolgozat/ci-cd-gha-aws' showing the output of a git push command. The output lists various files being updated or added to the repository, including README.md, requirements.txt, runtime.txt, and several .git/objects files. The terminal ends with a prompt '\$' and the current directory path. The window has standard Windows window controls (minimize, maximize, close) in the top right corner.

```
MINGW64:/c/szakdolgozat/ci-cd-gha-aws
updating: README.md (deflated 39%)
updating: requirements.txt (deflated 1%)
updating: runtime.txt (stored 0%)
updating: .git/objects/13/49b2c6816e9f50fa82ff48ad19133f486f34b9 (stored 0%)
updating: .git/objects/26/ (stored 0%)
updating: .git/objects/26/dddceea06d4d0171084cce7f0ac9d1da81ae37 (stored 0%)
updating: .git/objects/81/ (stored 0%)
updating: .git/objects/81/9ad88757ad991eb0bea0279930277804aacce7 (stored 0%)
updating: .git/objects/9a/ (stored 0%)
updating: .git/objects/9a/822ab6e978a70db69beb5c2c5b87255b73f634 (stored 0%)
updating: .git/objects/d8/ (stored 0%)
updating: .git/objects/d8/3800017b3b36a6b8782d06336eb37e3c1bd936 (stored 0%)
updating: .git/objects/f2/ (stored 0%)
updating: .git/objects/f2/5f6ed07c9b93cc05c89c65935b822558322442 (stored 0%)
adding: .git/objects/37/f45dd1b364c3946e77c5fb6e8b3dacdd8aba87 (stored 0%)
adding: .git/objects/51/ (stored 0%)
adding: .git/objects/51/5355d96e6efe48235765cda573202b431e88e3 (stored 0%)
adding: .git/objects/80/ (stored 0%)
adding: .git/objects/80/52f83ed3eb0c1880ae0a433f2c5360c0478886 (stored 0%)
adding: .git/objects/bd/ (stored 0%)
adding: .git/objects/bd/fff2c773bea5fa47816b655753e39e4d1d2dfa (stored 0%)
adding: Procfile (stored 0%)
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$
```

```
printf "web: gunicorn mysite.wsgi:application --bind 127.0.0.1:8000\n" > Procfile
git add Procfile hello/views.py mysite/urls.py requirements.txt
git commit -m "Add Procfile (Gunicorn) + root health endpoint"
git push origin main
```

Step 41: Deployment Verification

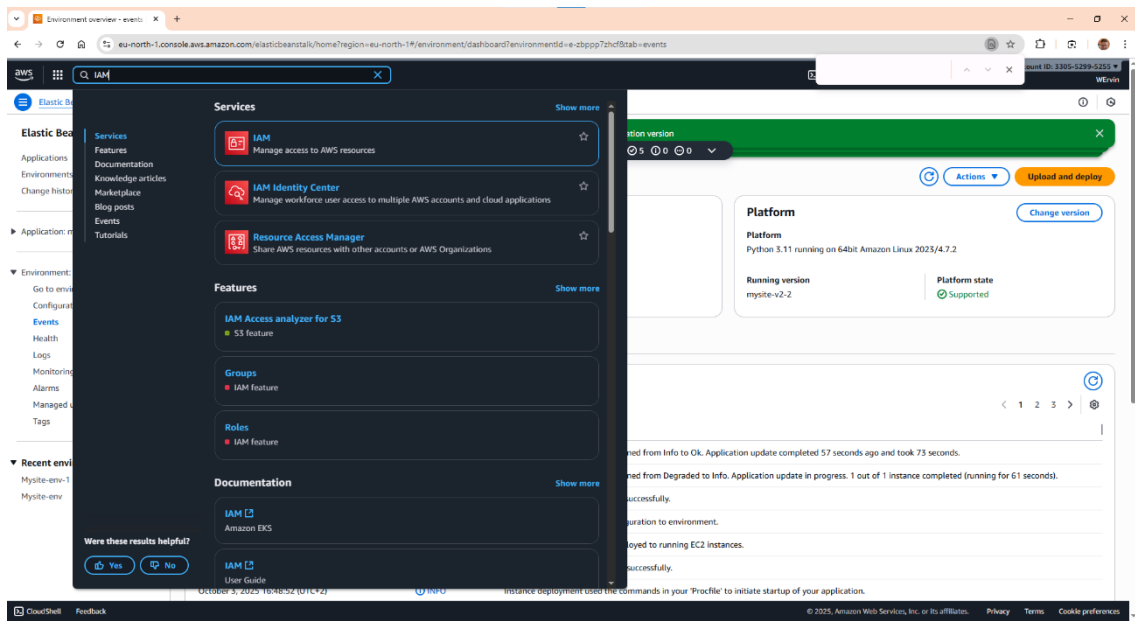


The deployment on AWS Elastic Beanstalk is successful; the `mysite-env-1` environment status is OK.

Running version: `mysite-v2-2` (Python 3.11, AL2023).

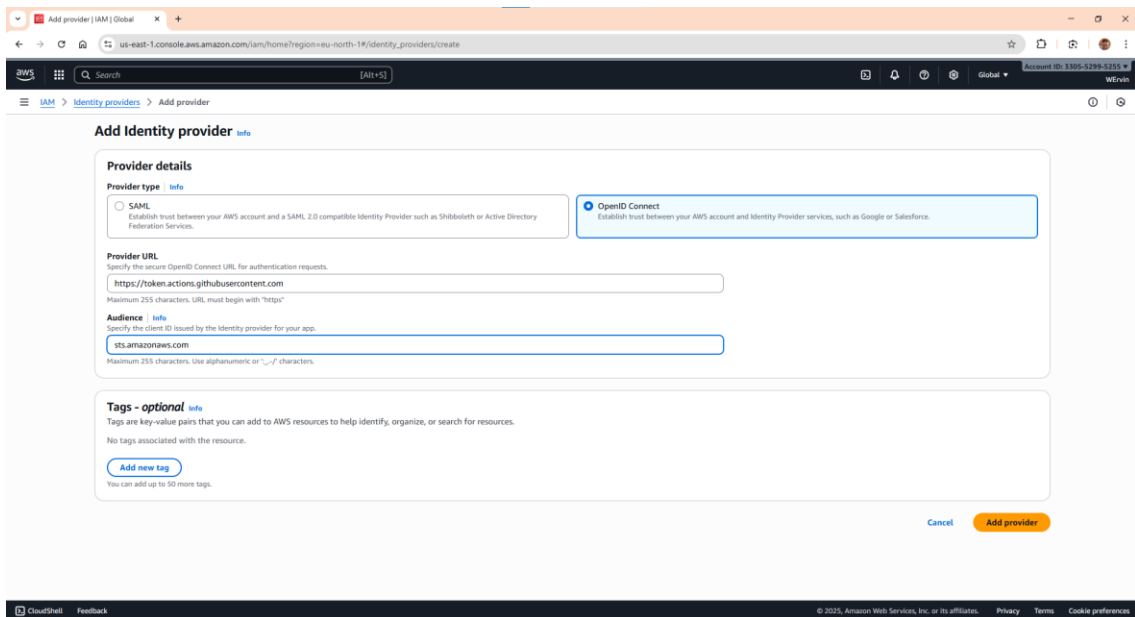
The domain endpoint now returns a 200 OK response: 'Hello, CI/CD GitHubActions, AWS! All OK!'

Step 42: Opening IAM



Type IAM (Identity and Access Management) into the search bar and sign in

Step 43: Adding an Identity Provider



Identify provider -> Add provider

OPENID connect

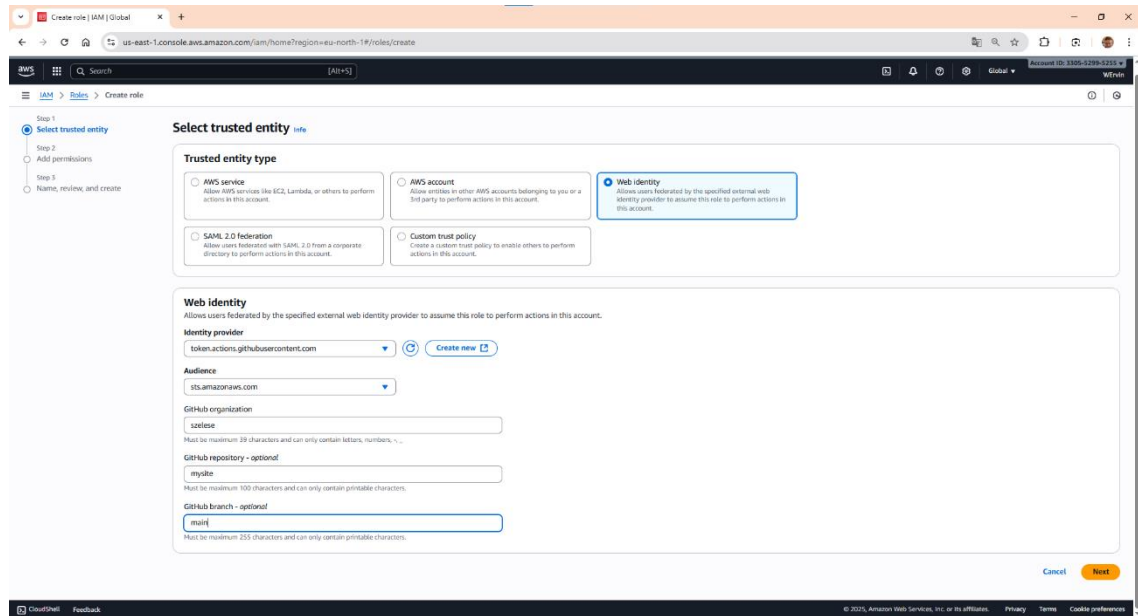
Provider URL:

https://token.actions.githubusercontent.com

Audience:

sts.amazonaws.com

Step 44: Configuring the Trusted Entity (Step 1)



IAM/roles/create role:

WEB Identity

Identity provider: token.actions.githubusercontent.com

Audience: sts.amazonaws.com

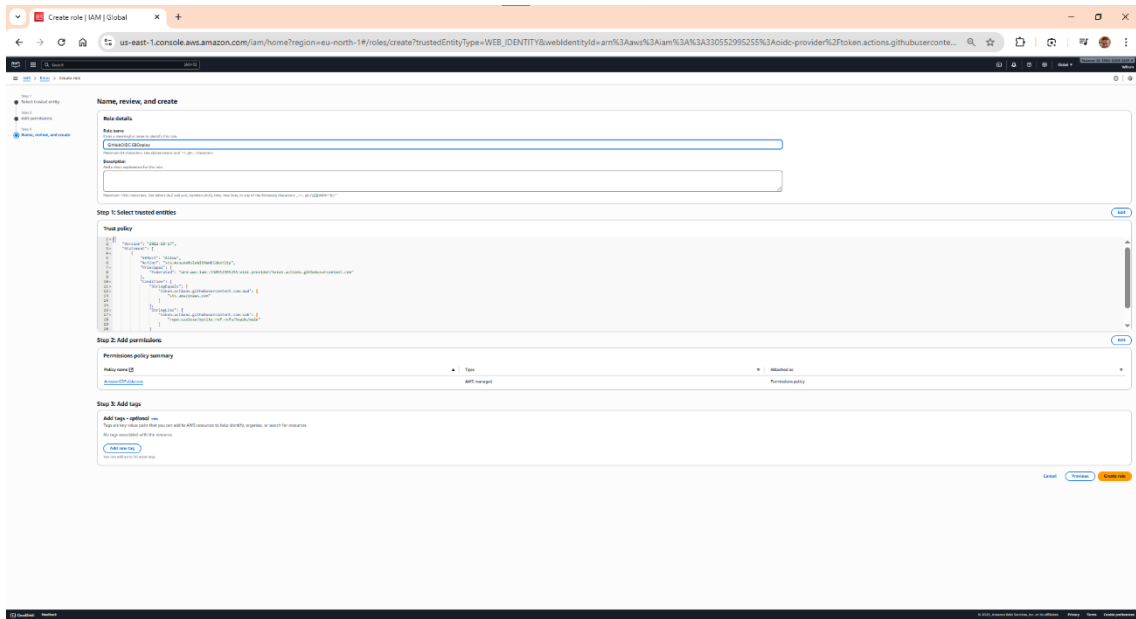
GitHub organization: YourGitHubName # your own github name

GitHub repository: mysite # the repository name for deployment

GitHub branch: main # usually

Security Note: While the repository and branch fields are optional, specifying them is recommended for better security.

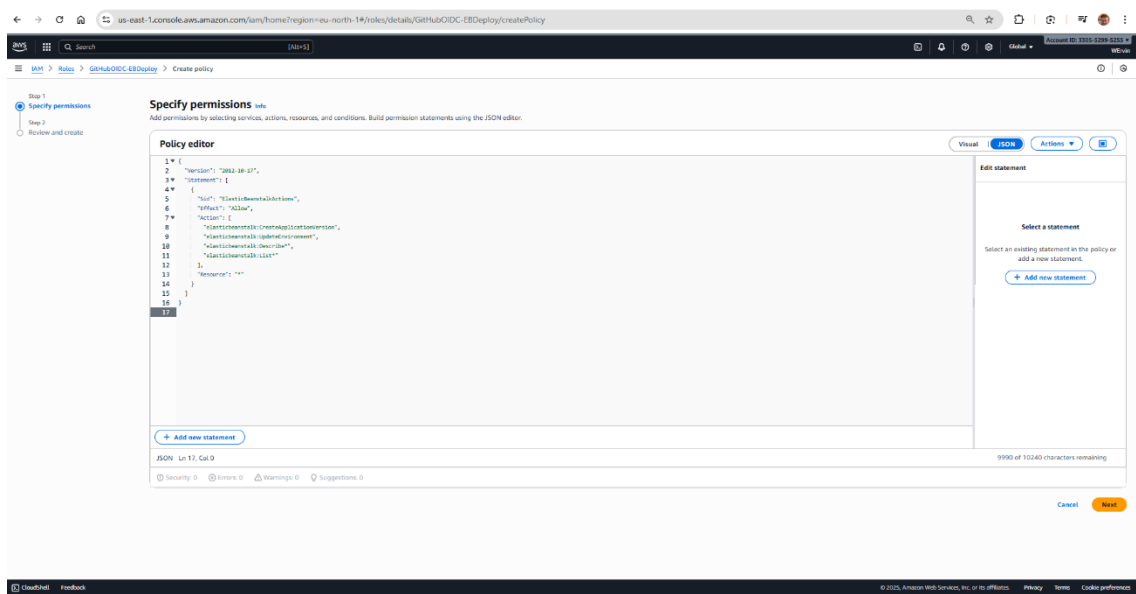
Step 45: Naming the Role (Step 3)



In the Role name field, enter: GitHubOIDC-EBDeploy.

Click Create role.

Step 46: Configuring Permissions



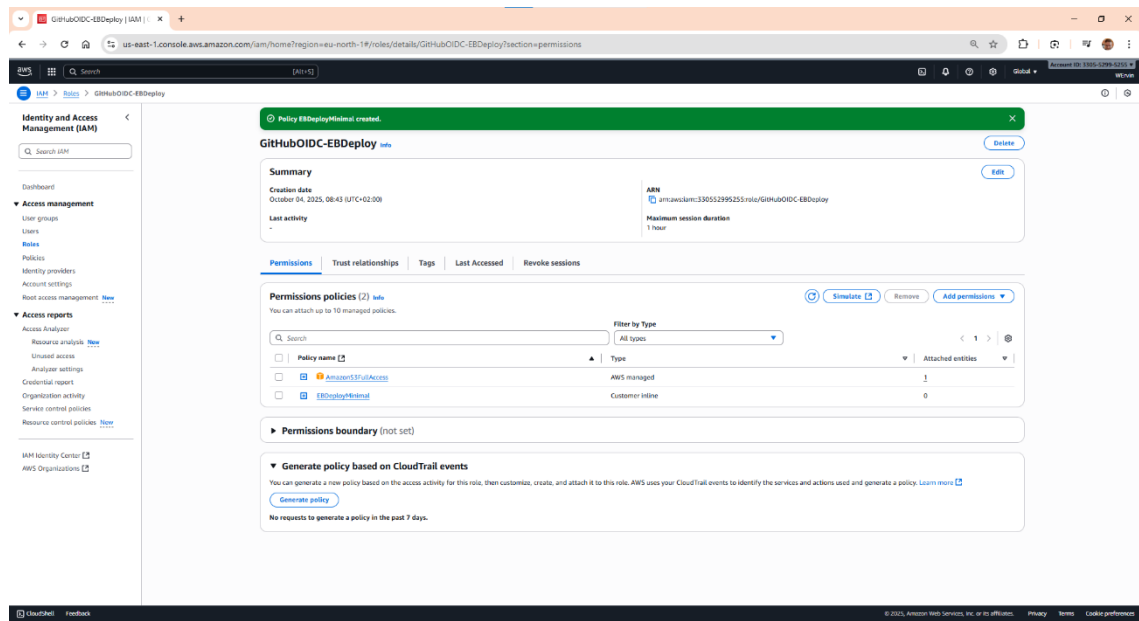
IAM->Roles->GitHubOIDC-EBDeploy->open

Permissions->Add permissions->Create inline policy->JSON

Enter the permission policy as shown in the project screenshots.->NEXT

Név: EBDeployMinimal -> Create policy

Step 47: Policy Verification and ARN Copying



The screenshot shows the AWS IAM console for the role 'GitHubOIDC-EBDeploy'. A green notification banner at the top states 'Policy EBDeployMinimal created.' The 'Summary' section shows the role was created on October 04, 2025, at 08:43 UTC+02:00. The ARN is 'arn:aws:iam::330552995255:role/GitHubOIDC-EBDeploy' with a maximum session duration of 1 hour. The 'Permissions policies' section is expanded, showing two policies: 'AmazonS3FullAccess' (AWS managed, 1 attached entity) and 'EBDeployMinimal' (Customer inline, 0 attached entities). There is also a section for 'Generate policy based on CloudTrail events' with a 'Generate policy' button.

Verify that the Permission policies section displays both expected policies.

Copy the Role ARN for later use: `arn:aws:iam::330552995255:role/GitHubOIDC-EBDeploy`.

Step 48: GitBash and commit & push

```
MINGW64:/c/szakdolgozat/ci-cd-gha-aws
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ mkdir -p .github/workflows
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ cat > .github/workflows/oidc-smoke.yml <<'YAML'
name: OIDC smoke test
on: workflow_dispatch

permissions:
  id-token: write
  contents: read

jobs:
  test:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4

      - name: Configure AWS credentials
        uses: aws-actions/configure-aws-credentials@v4
        with:
          role-to-assume: arn:aws:iam::330552995255:role/GitHubOIDC-EBDeploy
          aws-region: eu-north-1

      - name: Verify identity
        run: aws sts get-caller-identity

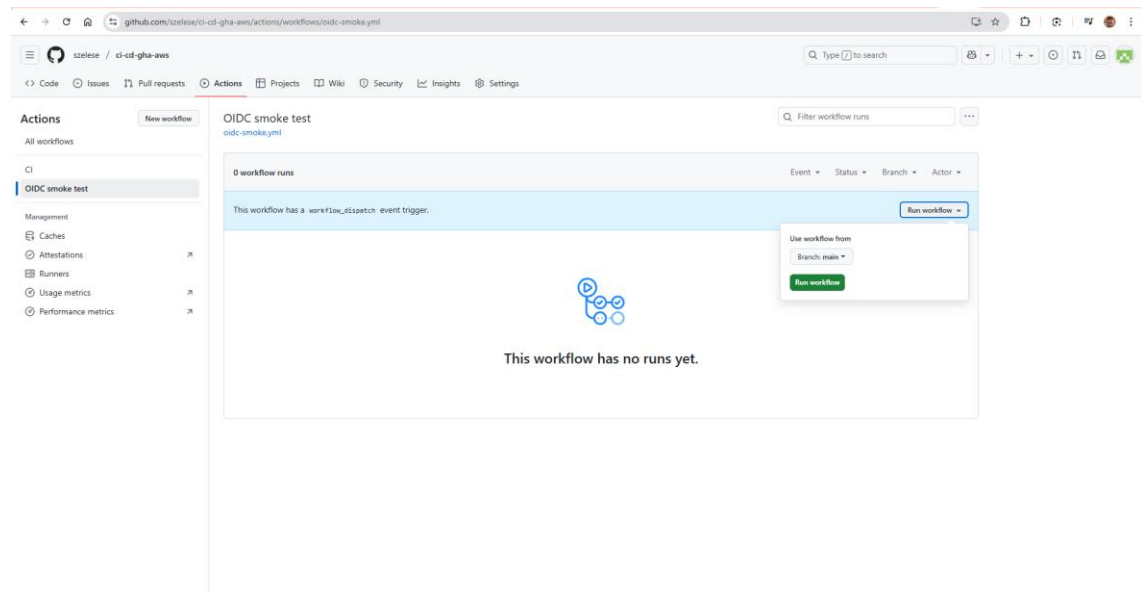
YAML
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git add .github/workflows/oidc-smoke.yml
git commit -m "Add OIDC smoke test workflow"
git push origin main
```

```
mkdir -p .github/workflows
```

Create the file with the specified content.

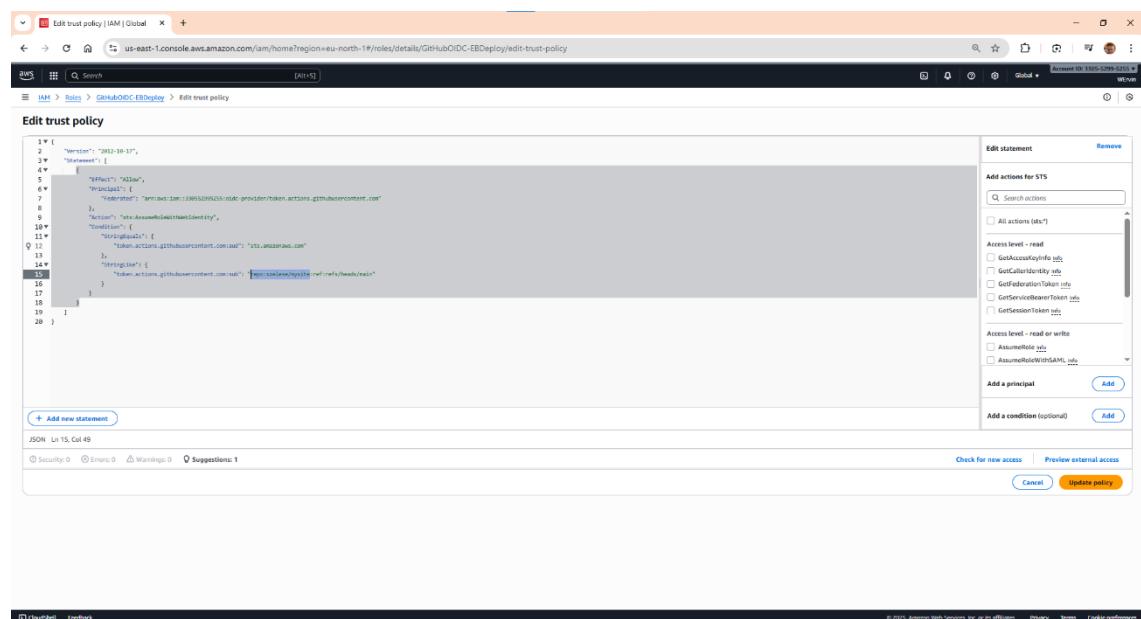
```
# commit&push
git add .github/workflows/oidc-smoke.yml
git commit -m "Add OIDC smoke test workflow"
git push origin main
```

Step 49: Running the Smoke Test



GitHub -> Actions ->OIDC smoke test -> Branch -> Run workflow

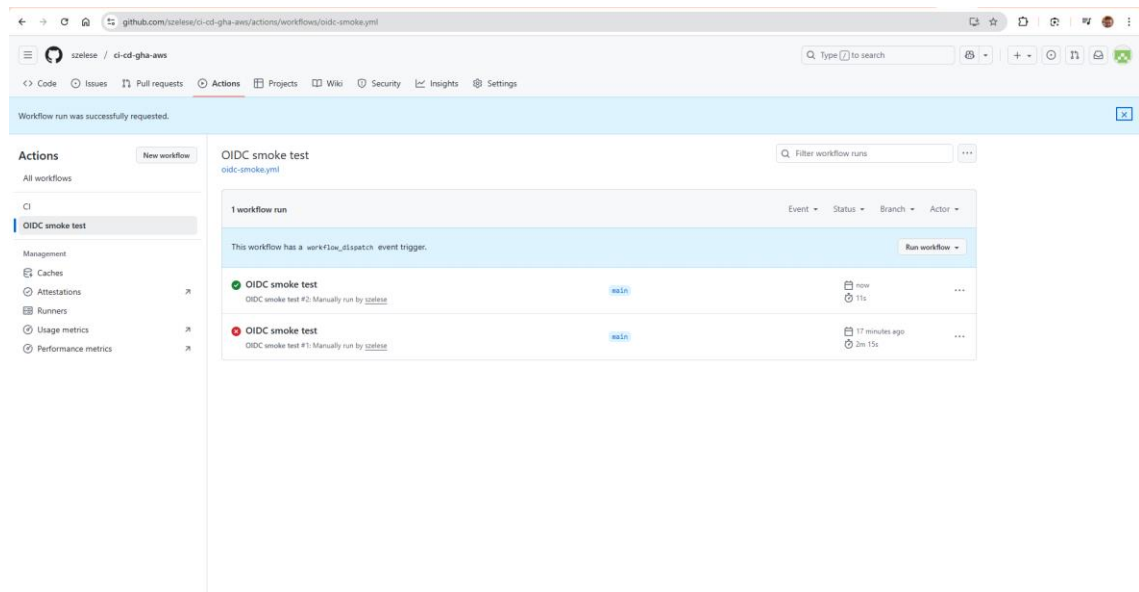
Step 50: Verification and Debugging



The default repository configuration points to the wrong location.

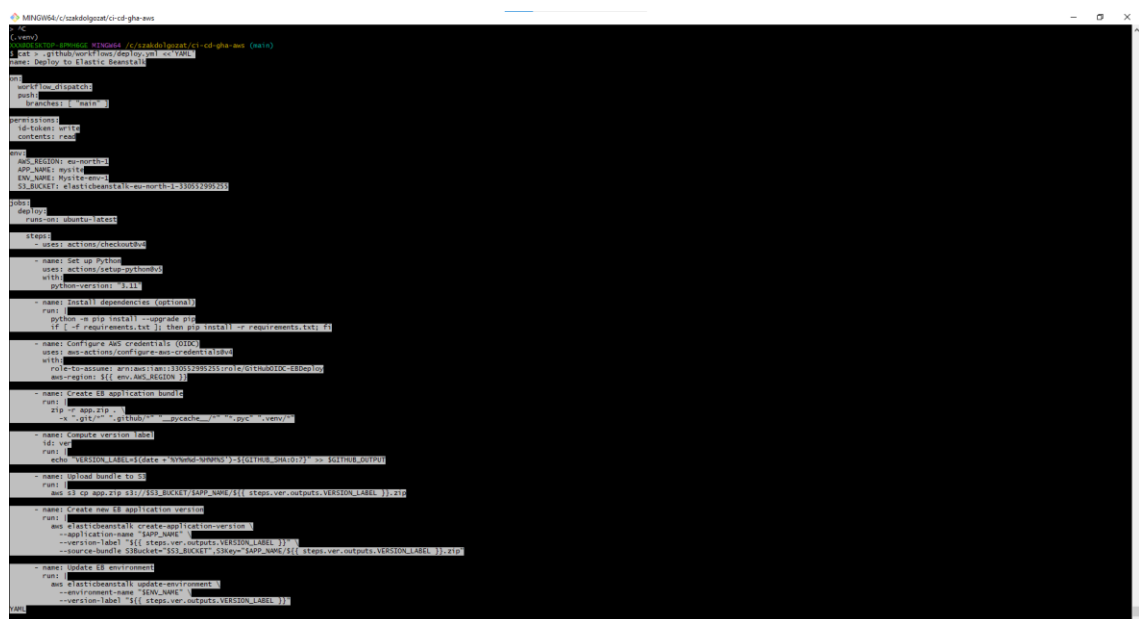
The actual repository is szelese/ci-cd-gha-aws; correct this so the smoke test can run properly. Note: Change to your repoName/projectName

Step 51: Re-running the Smoke Test

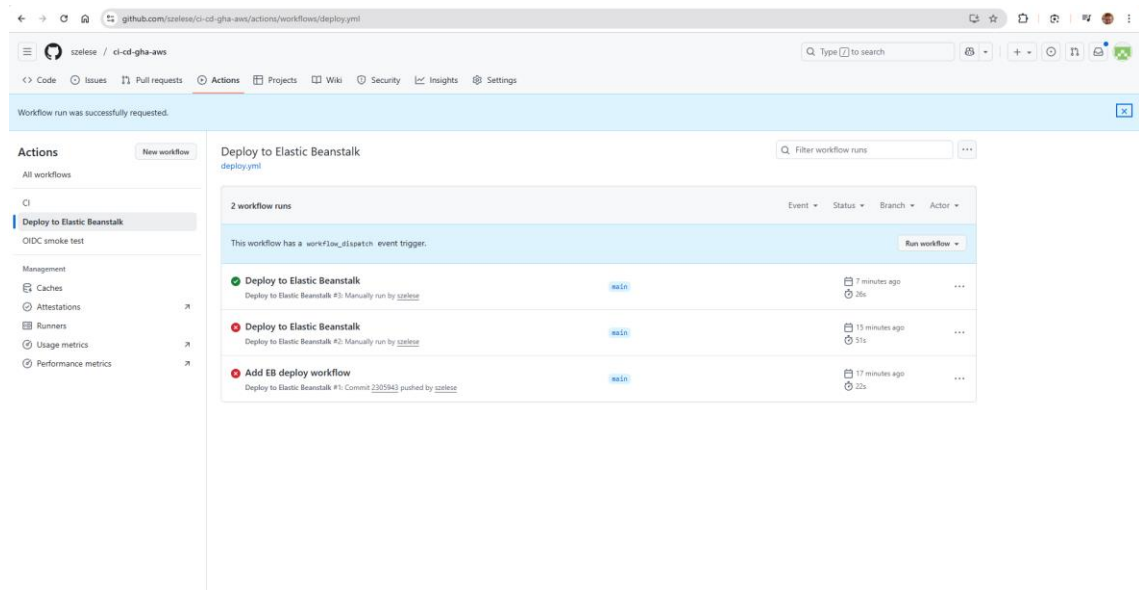


Successful OIDC smoke test between GitHub Actions and the AWS IAM role—the connection is verified and working (indicated by a green checkmark).

Step 52: Creating the Deploy File



Step 55: GitHub Actions -> Re-run jobs



As shown in the logs, the most recent "Deploy to Elastic Beanstalk" run is now green—the CI→CD chain is fully functional.

Step 56: Post-deploy Hook (migrate + collectstatic)

```
MINGW64:/c/szakdolgozat/ci-cd-gha-aws
To https://github.com/szelese/ci-cd-gha-aws.git
7dc0ba7..2305943 main -> main
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git checkout main
Already on 'main'
Your branch is up to date with 'origin/main'.
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ mkdir -p .platform/hooks/postdeploy
cat > .platform/hooks/postdeploy/00_django_tasks.sh <<'BASH'
set -e
>
# EB app mappa (AL2/AL2023 eset)
cd /var/app/staging 2>/dev/null || cd /var/app/current

# venv aktiválás (wildcard, mert a mappa neve verziófüggő)
if [ -d "/var/app/venv" ]; then
    source /var/app/venv/*/bin/activate
fi

# Django parancsok
python manage.py migrate --noinput
python manage.py collectstatic --noinput
BASH
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git add .platform/hooks/postdeploy/00_django_tasks.sh
git update-index --chmod=+x .platform/hooks/postdeploy/00_django_tasks.sh
git commit -m "Add postdeploy hook: migrate + collectstatic"
git push origin main
warning: in the working copy of '.platform/hooks/postdeploy/00_django_tasks.sh', LF will be replaced by CRLF the next
time Git touches it
[main d78be06] Add postdeploy hook: migrate + collectstatic
1 file changed, 13 insertions(+)
 create mode 100755 .platform/hooks/postdeploy/00_django_tasks.sh
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (6/6), 672 bytes | 672.00 KiB/s, done.
Total 6 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/szelese/ci-cd-gha-aws.git
2305943..d78be06 main -> main
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$
```

Technical Note: While not strictly required for operation, it is highly recommended to automate migrations, static file collection, and server-side secret management.

```
mkdir -p .platform/hooks/postdeploy
cat > .platform/hooks/postdeploy/00_django_tasks.sh <<'BASH'
git add .platform/hooks/postdeploy/00_django_tasks.sh
git update-index --chmod=+x .platform/hooks/postdeploy/00_django_tasks.sh
git commit -m "Add postdeploy hook: migrate + collectstatic"
git push origin main
```

Step 57: GitHub Actions Status

The screenshot shows the GitHub Actions interface for the workflow 'Deploy to Elastic Beanstalk' (deploy.yml) in the repository 'szelise / ci-cd-gha-aws'. The workflow is currently in a 'pending' state, indicated by a yellow lightning bolt icon. The interface shows a list of workflow runs:

Run Name	Status	Event	Branch	Actor	Time
Add postdeploy hook: migrate + collectstatic	In progress	now	main	szelise	10:38 AM
Deploy to Elastic Beanstalk #3	Failed	Manually run by szelise	main	szelise	Today at 10:30 AM
Deploy to Elastic Beanstalk #2	Failed	Manually run by szelise	main	szelise	Today at 10:27 AM
Add EB deploy workflow	Failed	Commit 2305943 pushed by szelise	main	szelise	Today at 10:27 AM

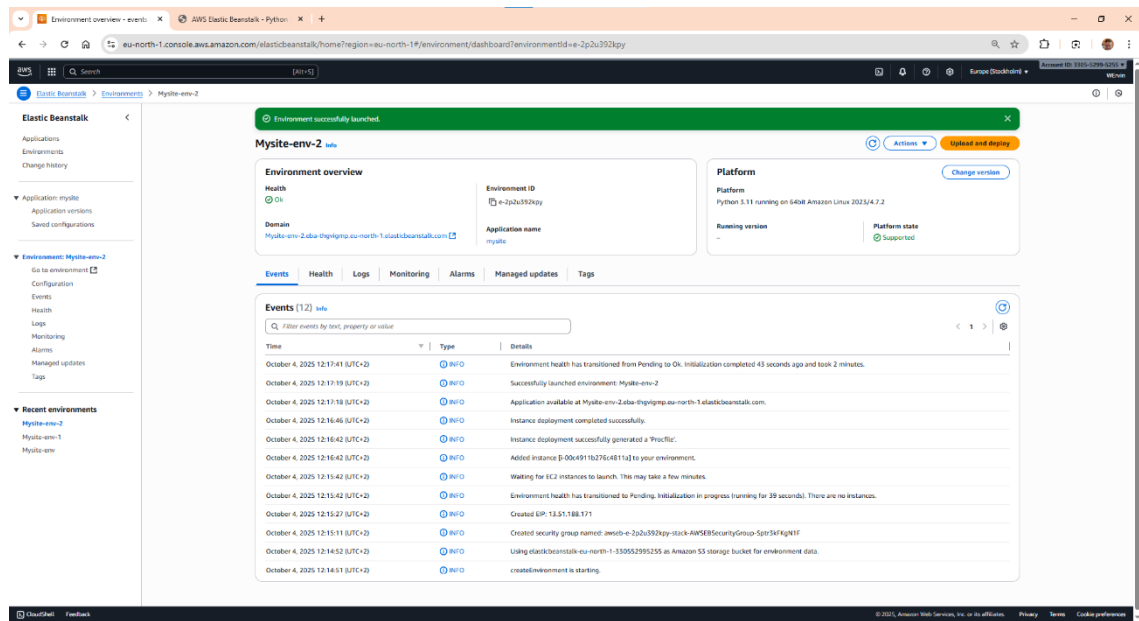
After the push and commit, GitHub Actions automatically begins execution.

Step 58: Successful Run

The screenshot shows the GitHub Actions interface for the workflow 'Deploy to Elastic Beanstalk' (deploy.yml) in the repository 'szelise / ci-cd-gha-aws'. The workflow is now in a 'completed' state, indicated by a green checkmark icon. The interface shows a list of workflow runs:

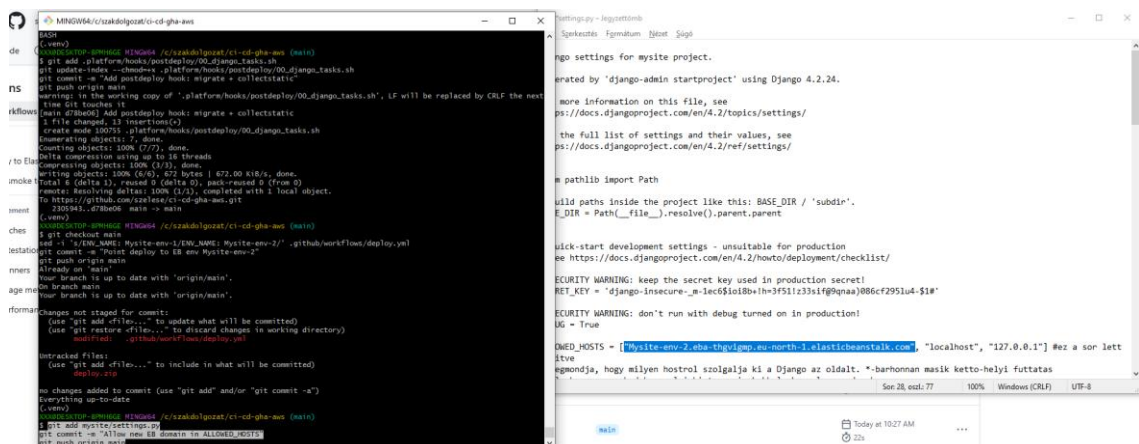
Run Name	Status	Event	Branch	Actor	Time
Add postdeploy hook: migrate + collectstatic	Completed	3 minutes ago	main	szelise	27s
Deploy to Elastic Beanstalk #3	Completed	Today at 10:38 AM	main	szelise	26s
Deploy to Elastic Beanstalk #2	Failed	Today at 10:30 AM	main	szelise	31s
Add EB deploy workflow	Failed	Today at 10:27 AM	main	szelise	22s

Step 59: Creating a New Server: mysite-env-2



Technical Note: For security reasons, mysite-env-1 was later placed into a Suspended state.

Step 60: Adjusting the Deploy Workflow for the New Environment



Modify settings.py to restrict ALLOWED_HOSTS to the specific domain:

Mysite-env-2.eba-thgvgmp.eu-north-1.elasticbeanstalk.com.

Note: use your own repo details.

Commit and push the security update.

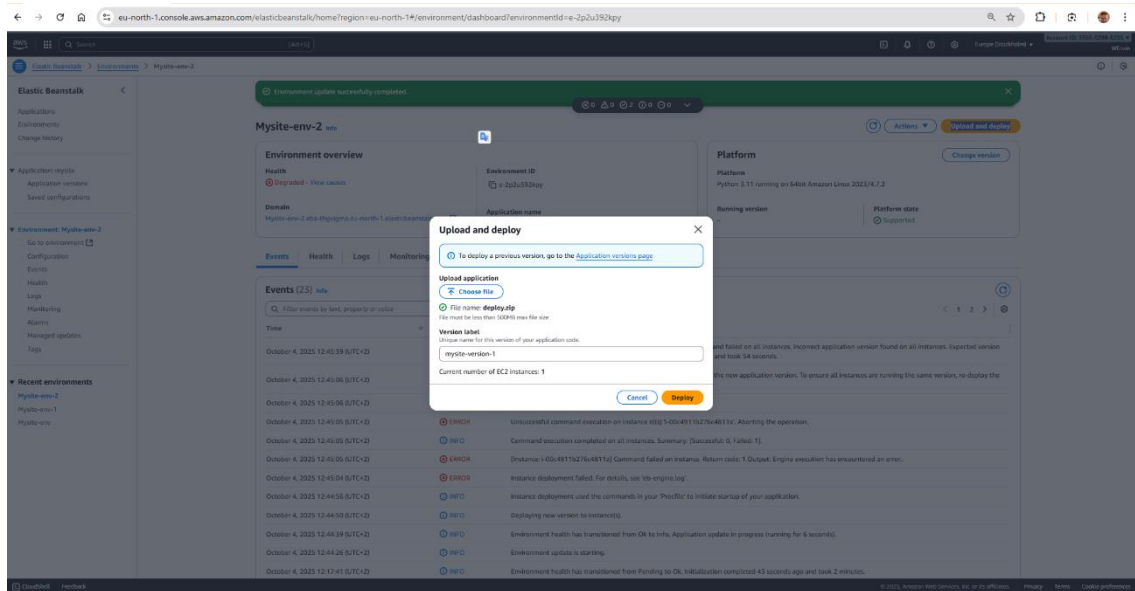
Step 61: Local vs Repo Environment Synchronization

Locally the target is Mysite-env-2, but the repository still references environment 1; correct this via commit and push:

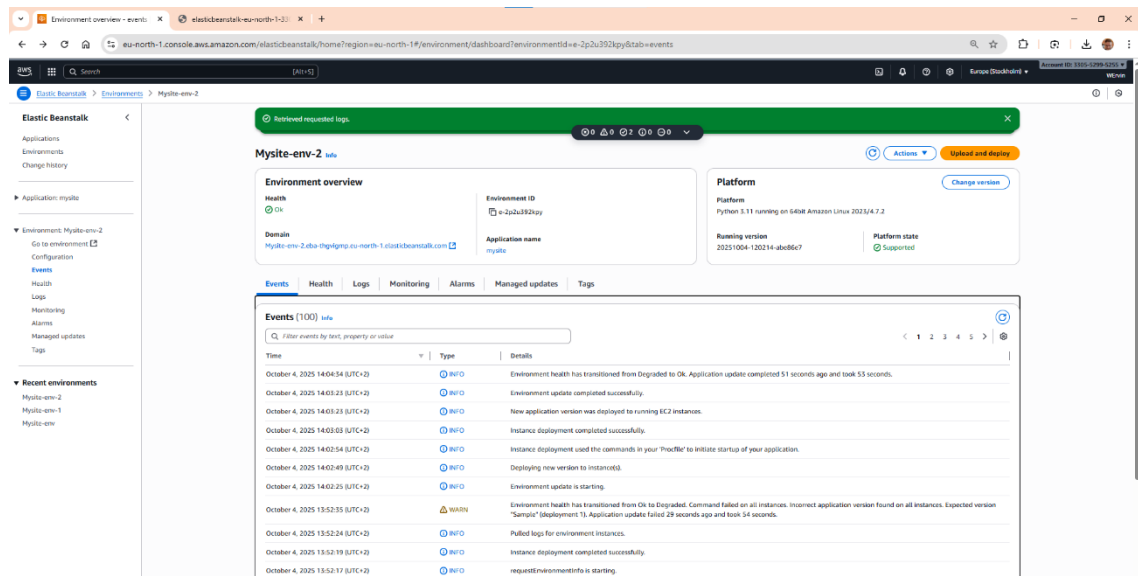
```
MINGW64/c/szakdolgozat/ci-cd-gha-aws
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git add .github/workflows/deploy.yml
git commit -m "Switch ENV_NAME to Mysite-env-2"
git push origin main
warning: in the working copy of '.github/workflows/deploy.yml', LF will be replaced by CRLF the next time Git touches it
[main d4c0db3] Switch ENV_NAME to Mysite-env-2
1 file changed, 1 insertion(+), 1 deletion(-)
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 16 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 467 bytes | 467.00 KiB/s, done.
Total 5 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/szelese/ci-cd-gha-aws.git
d78be06..d4c0db3 main -> main
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$
```

```
git add .github/workflows/deploy.yml
git commit -m "Switch ENV_NAME to Mysite-env-2"
git push origin main
```

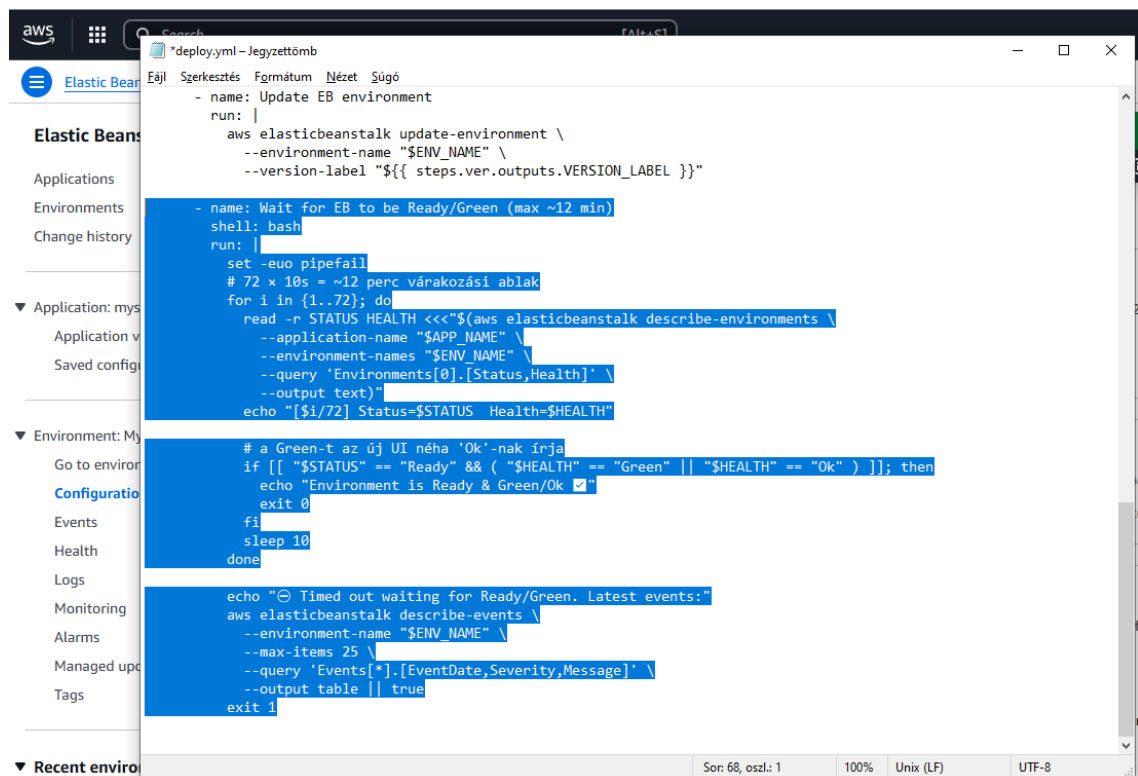
Step 62: End of Manual Deployment



Step 63: New Server Stability



Step 64: Troubleshooting and Prerequisites



The following configuration ensures the deployment workflow only completes successfully if the Elastic Beanstalk (EB) environment reaches a Ready + Green/Ok status. If this state is not reached, the workflow fails (marked in red) and outputs the most

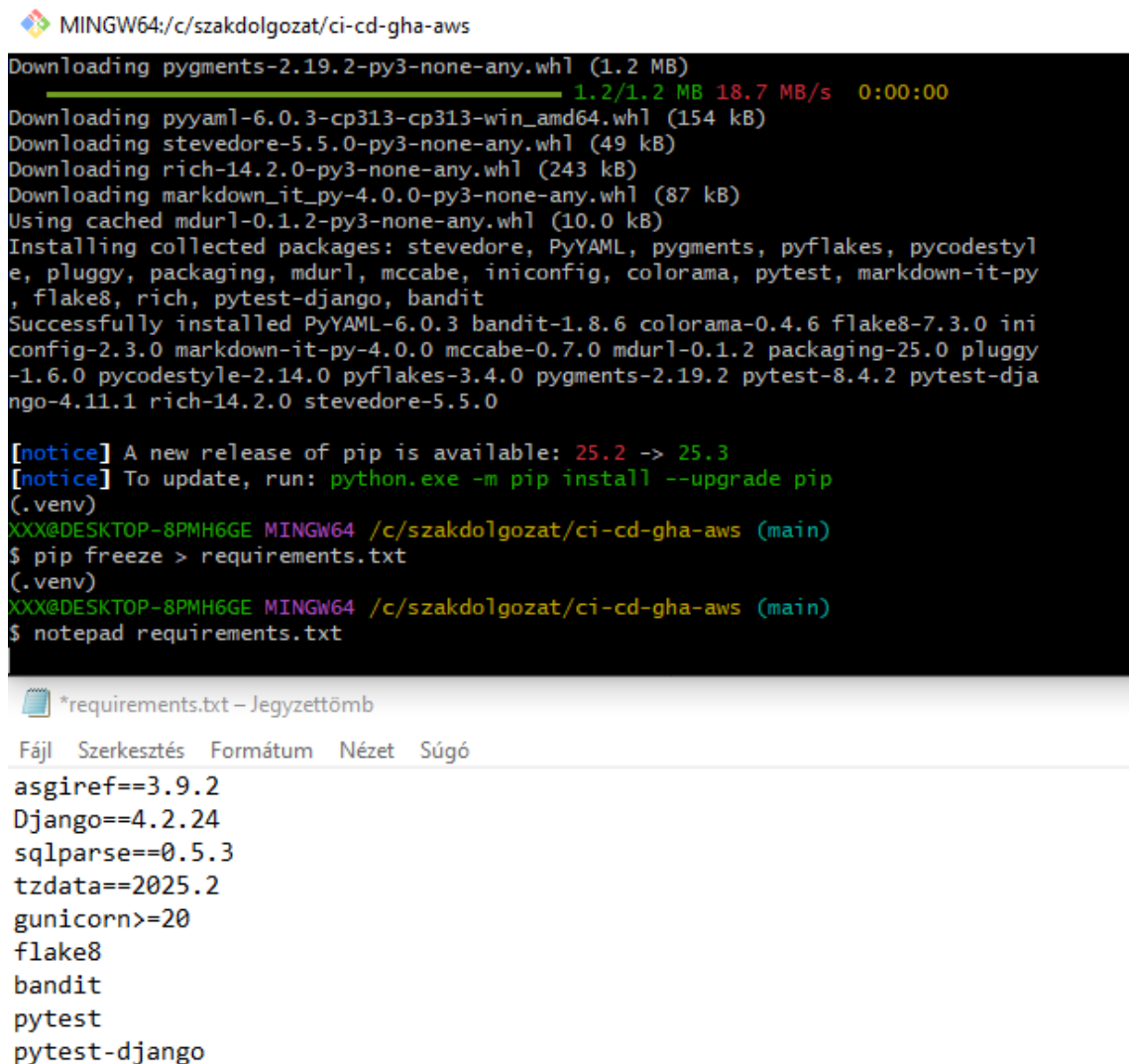
recent EB events. This serves as a simple diagnostic check and a prerequisite ensuring the server only goes live when the system is stable. Commit&push the changes GitBash:

```
git checkout main
git add .github/workflows/deploy.yml
git commit -m "Deploy: wait until EB is Ready & Green"
git push origin main
```

Step 65: CI Expansion: flake8, bandit, pytest

To strengthen the CI pipeline, install essential quality assurance tools: flake8 (style guide enforcement), bandit (security vulnerability scanning), and pytest-django (unit testing). Add these dependencies to requirements.txt to ensure the GitHub Actions runner installs them.

Technical Note: Instead of using `pip freeze > requirements.txt` (which can include unnecessary packages), manually append the new dependencies to the end of the file to maintain a clean environment.



```
MINGW64:/c/szakdolgozat/ci-cd-gha-aws
Downloading pygments-2.19.2-py3-none-any.whl (1.2 MB)
 1.2/1.2 MB 18.7 MB/s 0:00:00
Downloading pyyaml-6.0.3-cp313-cp313-win_amd64.whl (154 kB)
Downloading stevedore-5.5.0-py3-none-any.whl (49 kB)
Downloading rich-14.2.0-py3-none-any.whl (243 kB)
Downloading markdown_it_py-4.0.0-py3-none-any.whl (87 kB)
Using cached mdurl-0.1.2-py3-none-any.whl (10.0 kB)
Installing collected packages: stevedore, PyYAML, pygments, pyflakes, pycodestyle, pluggy, packaging, mdurl, mccabe, iniconfig, colorama, pytest, markdown-it-py, flake8, rich, pytest-django, bandit
Successfully installed PyYAML-6.0.3 bandit-1.8.6 colorama-0.4.6 flake8-7.3.0 iniconfig-2.3.0 markdown-it-py-4.0.0 mccabe-0.7.0 mdurl-0.1.2 packaging-25.0 pluggy-1.6.0 pycodestyle-2.14.0 pyflakes-3.4.0 pygments-2.19.2 pytest-8.4.2 pytest-django-4.11.1 rich-14.2.0 stevedore-5.5.0

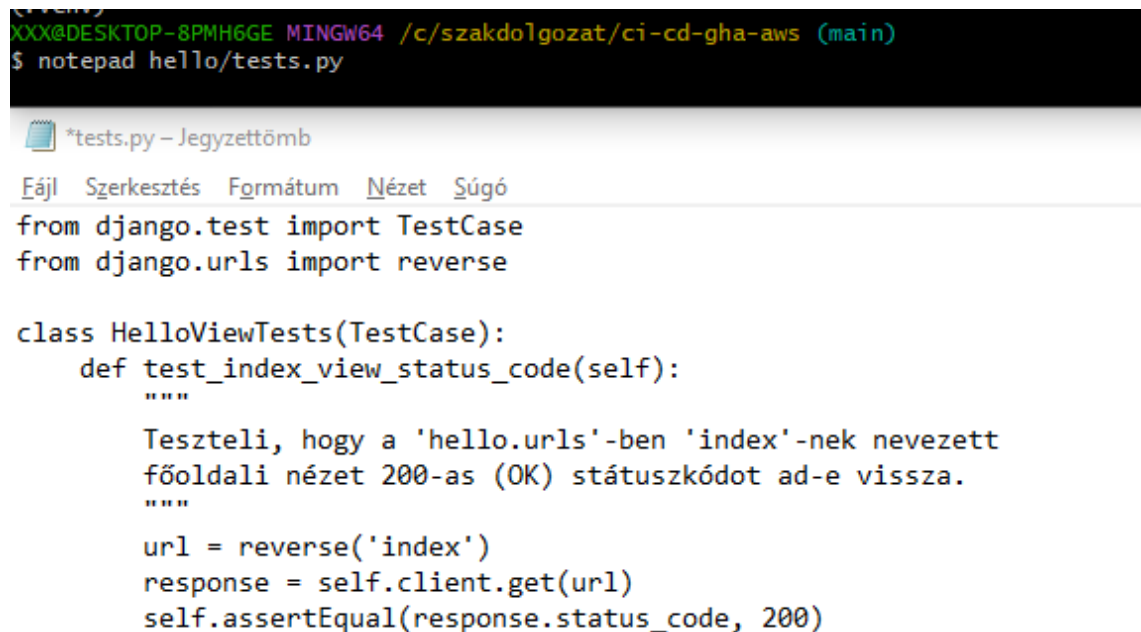
[notice] A new release of pip is available: 25.2 -> 25.3
[notice] To update, run: python.exe -m pip install --upgrade pip
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ pip freeze > requirements.txt
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ notepad requirements.txt

*requirements.txt - Jegyzetömb
Fájl Szerkesztés Formátum Nézet Súgó
asgiref==3.9.2
Django==4.2.24
sqlparse==0.5.3
tzdata==2025.2
gunicorn>=20
flake8
bandit
pytest
pytest-django
```

```
pip install flake8 bandit pytest pytest-django
notepad requirements.txt #manuálisan beírni
# pip freeze > requirements.txt
```

Step 66: Basic Unit Test Creation

This test verifies that the application's main page (index view) loads successfully by checking for a 200 OK status code.



```
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ notepad hello/tests.py

*tests.py - Jegyzetömb

Fájl Szerkesztés Formátum Nézet Súgó

from django.test import TestCase
from django.urls import reverse

class HelloViewTests(TestCase):
    def test_index_view_status_code(self):
        """
        Teszteli, hogy a 'hello.urls'-ben 'index'-nek nevezett
        főoldali nézet 200-as (OK) státuszkodeket ad-e vissza.
        """
        url = reverse('index')
        response = self.client.get(url)
        self.assertEqual(response.status_code, 200)
```

notepad hello/tests.py # insert the TestCase code

67. lépés: a ci.yml workflow kibővítése

Modify `.github/workflows/ci.yml` by inserting three new steps before the 'Quick check': 'Linting check (flake8)', 'Security check (bandit)', and 'Run Unit Tests (pytest)'. These steps function as quality gates: if any check finds an error, the pipeline terminates, preventing faulty code from being integrated.

This completes the CI portion of the pipeline.

```
(. venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ notepad .github/workflows/ci.yml

*ci.yml - Jegyzetfömb
Fájl Szerkesztés Formátum Nézet Súgó
pull_request:

jobs:
  build:
    runs-on: ubuntu-latest

    steps:
      - uses: actions/checkout@v4

      - name: Set up Python
        uses: actions/setup-python@v5
        with:
          python-version: '3.11' # ugyanaznak kell lennie, mint a runtime.txt-ben (python-3.11)

      - name: Install deps
        run: |
          python -m pip install --upgrade pip
          pip install -r requirements.txt
#új lépések kezdete
      - name: Linting check (flake8)
        run: |
          flake8 . --count --select=E9,F63,F7,F82 --show-source --statistics
          flake8 . --count --exit-zero --max-complexity=10 --max-line-length=127 --statistics

      - name: Security check (bandit)
        run: |
          bandit -r .


      - name: Run Unit Tests (pytest)
        run: |
          pytest
#új lépések vége

      - name: Quick check
        run: python -c "import django,sys; print('Django', django.get_version(), '| Python', sys.version.split()[0])"

Sor: 25, oszl.: 37    100%    Windows (CRLF)    UTF-8
```

notepad .github/workflows/ci.yml

Step 68: Commit & Push

 MINGW64:/c/szakdolgozat/ci-cd-gha-aws

```
bash: XXX@DESKTOP-8PMH6GE: command not found
(. venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git add requirements.txt hello/tests.py .github/workflows/ci.yml
git commit -m "CI quality gates"
git push
[main f1e54a8] CI quality gates
 3 files changed, 24 insertions(+), 6 deletions(-)
Enumerating objects: 13, done.
Counting objects: 100% (13/13), done.
Delta compression using up to 16 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (7/7), 1.04 KiB | 1.04 MiB/s, done.
Total 7 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To https://github.com/szelese/ci-cd-gha-aws.git
 8a7410d..f1e54a8  main -> main
(. venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$
```

```
git add requirements.txt hello/tests.py .github/workflows/ci.yml
git commit -m "CI quality gates"
git push
```

Step 69: Verification of Expanded CI Pipeline

After pushing, GitHub Actions automatically triggers the expanded pipeline.

Observation: The run failed at the 'Security check (bandit)' step.

```

1 ▶ Run bandit -r .
11 [main] INFO  profile include tests: None
12 [main] INFO  profile exclude tests: None
13 [main] INFO  cii include tests: None
14 [main] INFO  cii exclude tests: None
15 [main] INFO  running on Python 3.11.13
16 Run started:2025-10-25 07:33:11.276455
17
18 Test results:
19 >> Issue: [B105:hardcoded_password_string] Possible hardcoded password: 'django-insecure_m-1ec65i0b~!n~3f51i233i@Qnna)886c2951u4-51a'
20 Severity: Low  Confidence: Medium
21 CWE: CWE-259 (https://cwe.mitre.org/data/definitions/259.html)
22 More info: https://bandit.readthedocs.io/en/3.8.0/plugins/B105_hardcoded_password_string.html
23 Location: ./mysite/settings.py:23:13
24 # SECURITY WARNING: keep the secret key used in production secret!
25 SECRET_KEY = 'django-insecure_m-1ec65i0b~!n~3f51i233i@Qnna)886c2951u4-51a'
26
27
28 -----
29
30 Code scanned:
31 Total lines of code: 159
32 Total lines skipped (#nosec): 0
33 Total potential issues skipped due to specifically being disabled (e.g., #nosec BXXX): 0
34
35 Run metrics:
36 Total issues (by severity):
37   Undefined: 0
38   Low: 1
39   Medium: 0
40   High: 0
41 Total issues (by confidence):
42   Undefined: 0
43   Low: 0
44   Medium: 1
45   High: 0
46 Files skipped (0):
47 Error: Process completed with exit code 1.

```

Step 70: Modifying mysite/settings.py

```

XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ notepad mysite/settings.py

```

Generated by 'django-admin startproject' using Django 4.2.24.

For more information on this file, see <https://docs.djangoproject.com/en/4.2/topics/settings/>

For the full list of settings and their values, see <https://docs.djangoproject.com/en/4.2/ref/settings/>

```

from pathlib import Path
import os
from django.core.management.utils import get_random_secret_key

# Build paths inside the project like this: BASE_DIR / 'subdir'.
BASE_DIR = Path(__file__).resolve().parent.parent

# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/4.2/howto/deployment/checklist/

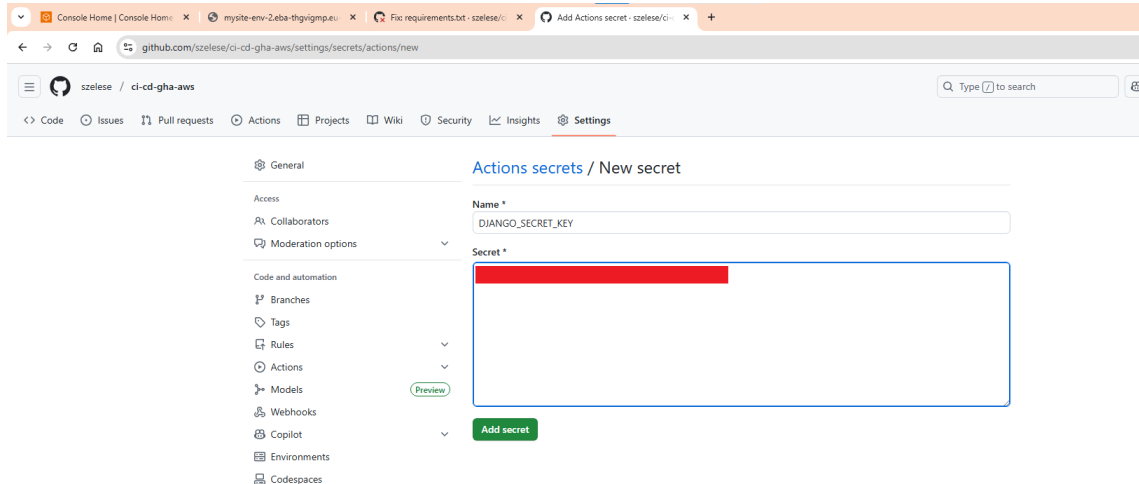
# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = os.environ.get("DJANGO_SECRET_KEY", get_random_secret_key())

```

Diagnosis: Bandit flagged the hardcoded SECRET_KEY.

Replace the static key with logic that prioritizes the DJANGO_SECRET_KEY environment variable, falling back to a randomly generated key if the variable is missing.

Step 71: Setting GitHub Repository Secret



Settings/Secrets and Variables/Actions/New repository secret

Name: DJANGO_SECRET_KEY

Secret: *****

Step 72: Updating the Workflow File

```
MINGW64:/c/szakdolgozat/ci-cd-gha-aws
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ notepad mysite/settings.py
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ python - <<'PY'
from django.core.management.utils import get_random_secret_key
print(get_random_secret_key())
PY
=4!-p-psx^rde=sj!@o_zkbr_mfwc57cuvv(ydmi!cp1r!hw!5
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ notepad .github/workflows/ci.yml
```

```
*ci.yml - Jegyzetfömb
Fájl Szerkesztés Formátum Nézet Súgó
name: CI

on:
  push:
    branches: [ "main" ]
  pull_request:

jobs:
  build:
    runs-on: ubuntu-latest
    env:
      DJANGO_SECRET_KEY: ${ secrets.DJANGO_SECRET_KEY }
    steps:
      - uses: actions/checkout@v4

      - name: Set up Python
        uses: actions/setup-python@v5
        with:

env:
  DJANGO_SECRET_KEY: ${ secrets.DJANGO_SECRET_KEY }
```

Step 73: Commit & Push

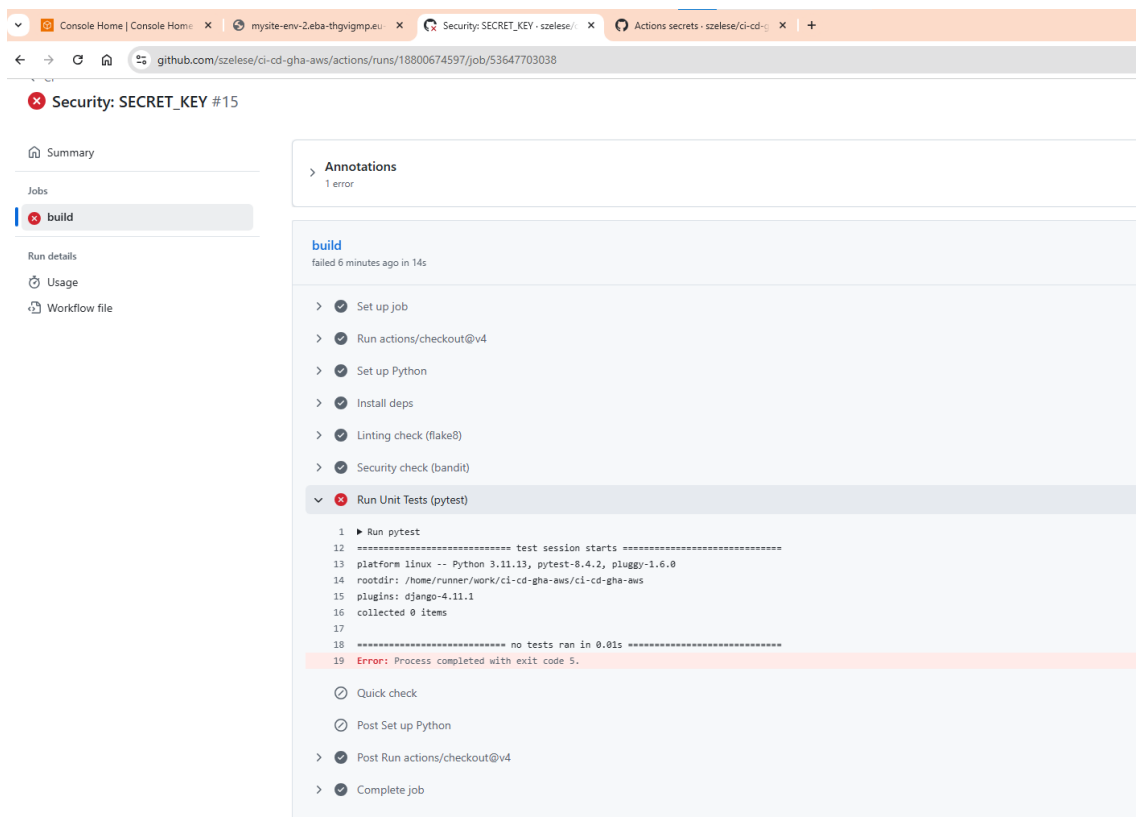
```
MINGW64:/c/szakdolgozat/ci-cd-gha-aws
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ notepad .github/workflows/ci.yml
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>.." to update what will be committed)
  (use "git restore <file>.." to discard changes in working directory)
    modified:   .github/workflows/ci.yml
    modified:   mysite/settings.py

no changes added to commit (use "git add" and/or "git commit -a")
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git add mysite/settings.py .github/workflows/ci.yml
$ git commit -m "Security: SECRET_KEY"
$ git push
```

```
git add mysite/settings.py .github/workflows/ci.yml
git commit -m "Security: SECRET_KEY"
git push
```

Step 74: Unit Test Failure (Pytest Configuration)



The screenshot shows a GitHub Actions workflow run for a repository named 'szelese/ci-cd-gha-aws'. The workflow is titled 'Security: SECRET_KEY #15'. The left sidebar shows the 'build' job is failed. The main content area shows the 'Annotations' section with 1 error. The 'build' job failed 6 minutes ago in 14s. The workflow steps are: Set up job, Run actions/checkout@v4, Set up Python, Install deps, Linting check (flake8), Security check (bandit), Run Unit Tests (pytest), Post Run actions/checkout@v4, and Complete job. The 'Run Unit Tests (pytest)' step is expanded, showing the following output:

```
1 ▶ Run pytest
12 ===== test session starts =====
13 platform linux -- Python 3.11.13, pytest-8.4.2, pluggy-1.6.0
14 rootdir: /home/runner/work/ci-cd-gha-aws/ci-cd-gha-aws
15 plugins: django-4.11.1
16 collected 0 items
17
18 ===== no tests ran in 0.01s =====
19 Error: Process completed with exit code 5.
```

Observation: The run failed again because Pytest could not find the tests or identify the Django settings automatically.

Step 75: Setting DJANGO_SETTINGS_MODULE in CI



```
Enumerating objects: 13, done.
Counting objects: 100% (13/13), done.
Delta compression using up to 16 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (7/7), 749 bytes | 749.00 KiB/s, done.
Total 7 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 4 local objects.
To https://github.com/szelese/ci-cd-gha-aws.git
   77a0897..122fa0c main -> main
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ notepad .github/workflows/ci.yml
```

```
*ci.yml - Jegyzetfömb
Fájl Szerkesztés Formátum Nézet Súgó
name: CI

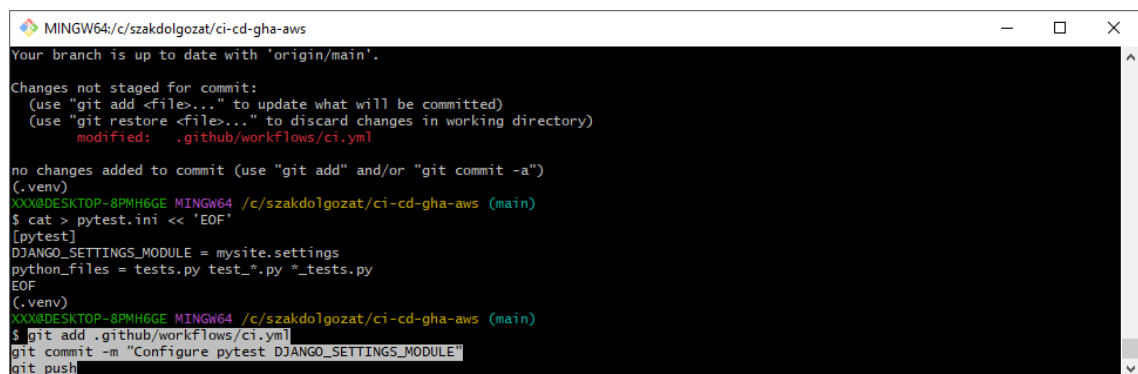
on:
  push:
    branches: [ "main" ]
  pull_request:

jobs:
  build:
    runs-on: ubuntu-latest
    env:
      DJANGO_SECRET_KEY: ${ secrets.DJANGO_SECRET_KEY }
      DJANGO_SETTINGS_MODULE: mysite.settings
    steps:
      - uses: actions/checkout@v4

      - name: Set up Python
        uses: actions/setup-python@v5
        with:
```

```
DJANGO_SETTINGS_MODULE: mysite.settings
...
- name: Run Unit Tests (pytest)
  run: |
    pytest --ds=mysite.settings
```

Step 76: Creating pytest.ini and Commit & Push

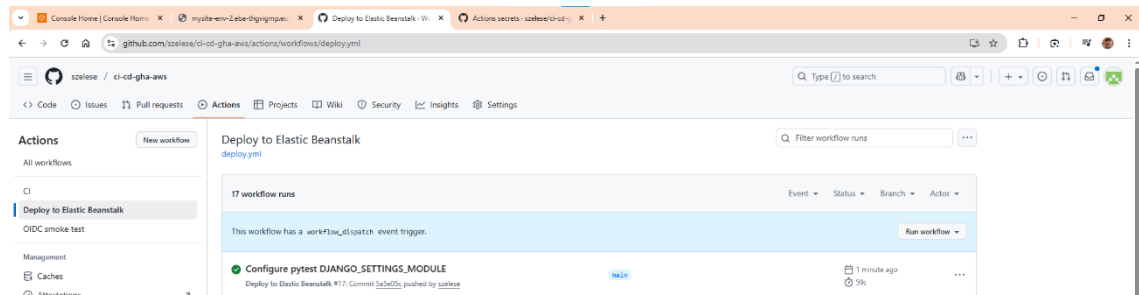


```
Your branch is up to date with 'origin/main'.
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   .github/workflows/ci.yml

no changes added to commit (use "git add" and/or "git commit -a")
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ cat > pytest.ini << 'EOF'
[pytest]
DJANGO_SETTINGS_MODULE = mysite.settings
python_files = tests.py test_*.py *_tests.py
EOF
(.venv)
XXX@DESKTOP-8PMH6GE MINGW64 /c/szakdolgozat/ci-cd-gha-aws (main)
$ git add .github/workflows/ci.yml
git commit -m "Configure pytest DJANGO_SETTINGS_MODULE"
git push
```

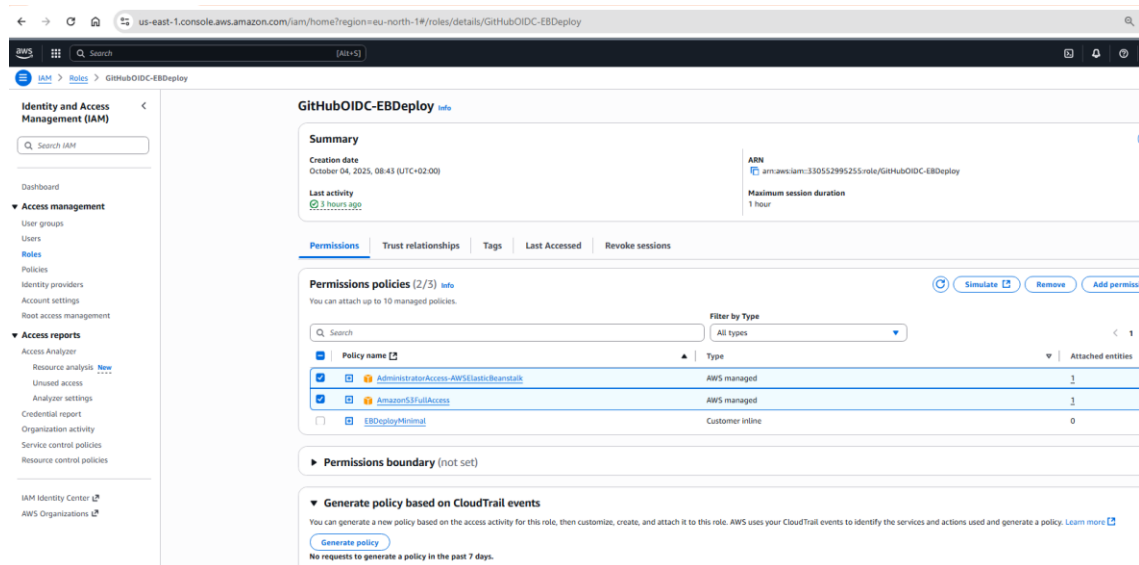
```
cat > pytest.ini << 'EOF'
[pytest]
DJANGO_SETTINGS_MODULE = mysite.settings
python_files = tests.py test_*.py *_tests.py
EOF
git add pytest.ini .github/workflows/ci.yml
git commit -m "Configure pytest DJANGO_SETTINGS_MODULE"
git push
```

Step 77: Verification of Successful CI Pipeline



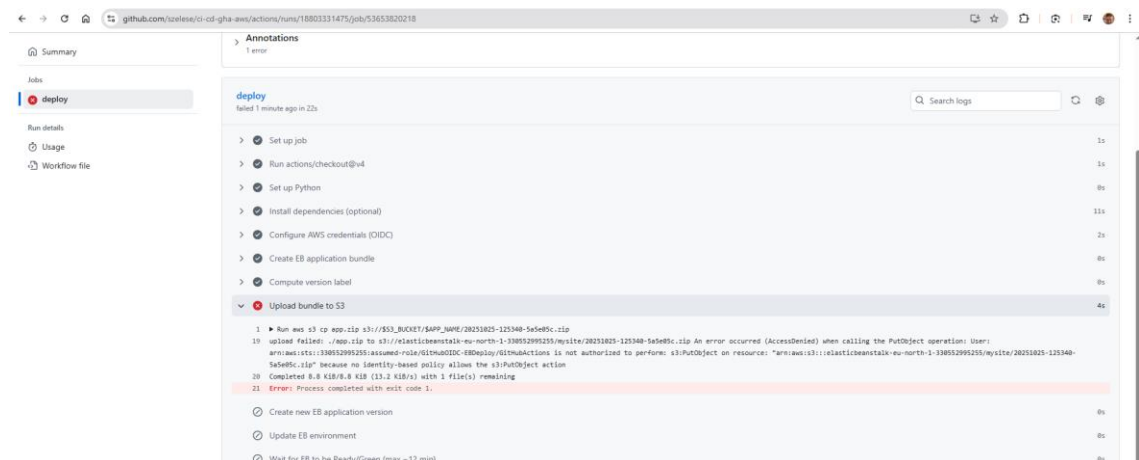
The GitHub Actions pipeline now runs successfully, passing all quality gates.

Step 78: Admin Policy Correction



To adhere to the Principle of Least Privilege, remove the "lazy" administrative policies: AmazonS3FullAccess and AdministratorAccess-AWSElasticBeanstalk.

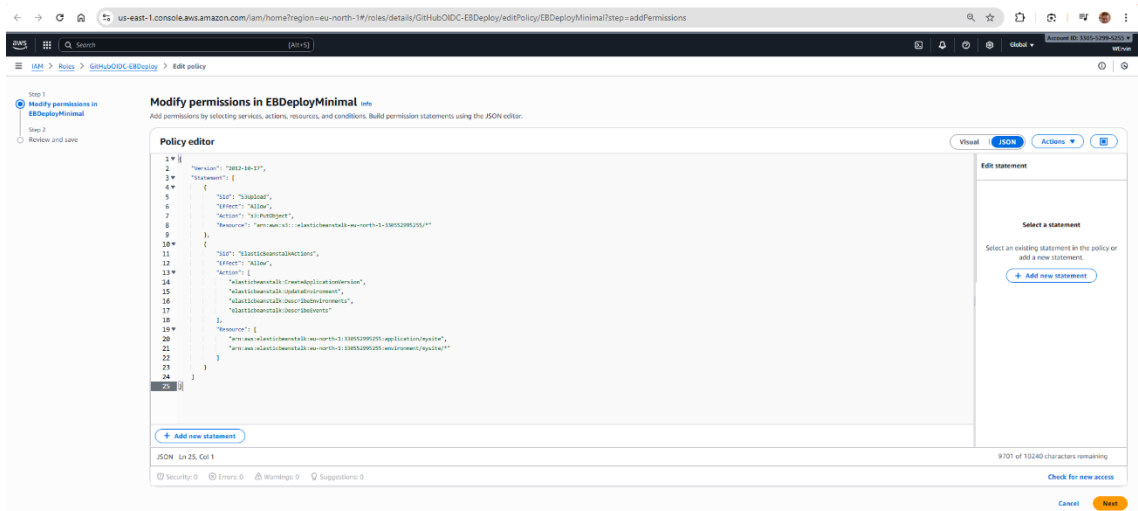
Step 79: Iterative Debugging Cycle



Repeatedly trigger the deploy.yml workflow to identify precisely which permissions are missing.

Observation: The first failure indicates s3:PutObject (file upload) is unauthorized.

Step 80: Policy Modification

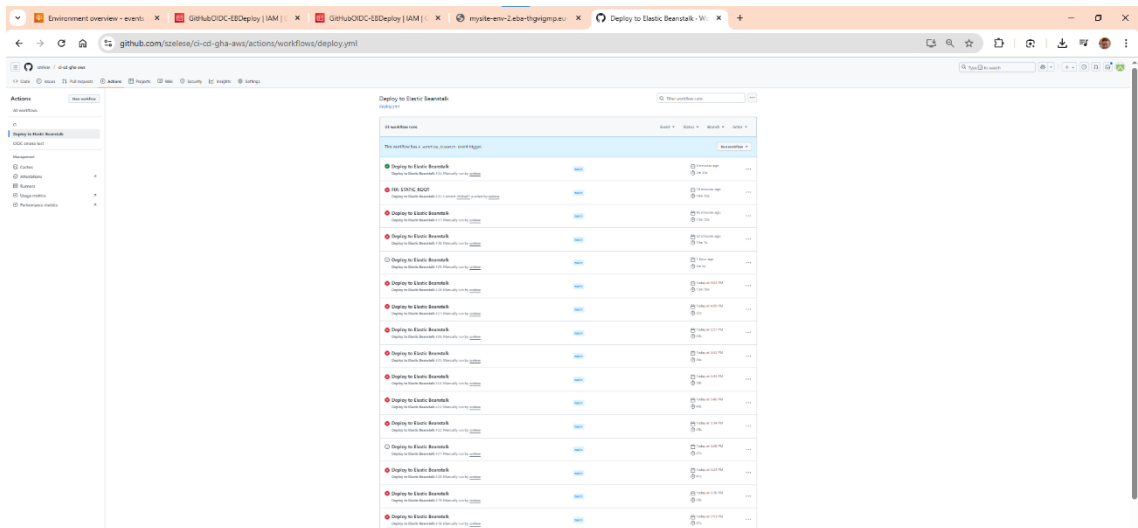


The screenshot shows the AWS IAM console's 'Policy editor' for the 'EBDeployMinimal' inline policy. The policy document is displayed in a code editor, showing a 'Statement' block with a 'Deny' action. The 'Deny' action is for the 'iam:PassRole' action on the resource 'arn:aws:iam::123456789012:role/*'. The 'Deny' action is currently selected in the 'Edit statement' panel on the right.

Manually update the EBDeployMinimal inline policy with the missing permission identified in the previous step.

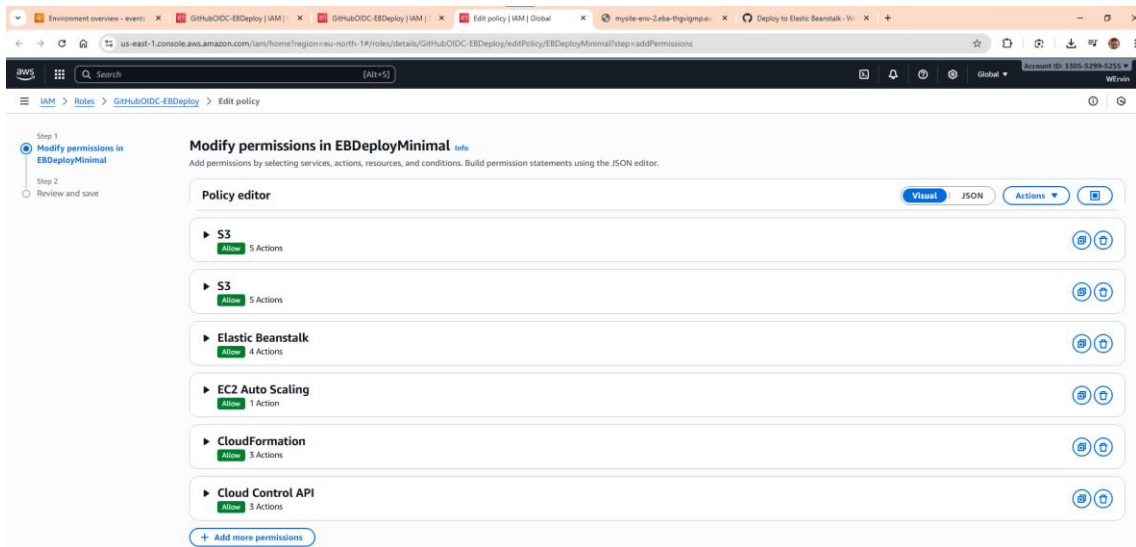
Save the policy and trigger the deployment workflow again.

Step 81: Successful Deployment After 15 Iterations



The screenshot shows the GitHub Actions workflow 'Deploy to Elastic Beanstalk'. The workflow is currently in a 'Completed' state. The runs are listed in a table with columns for 'Run ID', 'Status', and 'Created At'. All 15 runs are marked as 'Completed'.

Step 82: Final Minimal Policy Configuration



JSON:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "S3objectAccess",
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "s3:GetObject",
        "s3:GetObjectAcl",
        "s3:PutObjectAcl",
        "s3:DeleteObject"
      ],
      "Resource": "arn:aws:s3:::elasticbeanstalk-eu-north-1-330552995255/*"
    },
    {
      "Sid": "S3BucketAccess",
      "Effect": "Allow",
      "Action": [
        "s3:CreateBucket",
        "s3:PutBucketPolicy",
        "s3:PutBucketOwnershipControls",
        "s3:ListBucket",
        "s3:GetBucketPolicy"
      ],
      "Resource": "arn:aws:s3:::elasticbeanstalk-eu-north-1-330552995255"
    },
    {
      "Sid": "ElasticBeanstalkCloudformationAutoscalingAccess",
      "Effect": "Allow",
      "Action": [
        "elasticbeanstalk:CreateApplicationVersion",
```

```

        "elasticbeanstalk:UpdateEnvironment",
        "elasticbeanstalk:DescribeEnvironments",
        "elasticbeanstalk:DescribeEvents",
        "cloudformation:GetTemplate",
        "cloudformation:DescribeStackResource",
        "cloudformation:DescribeStackResources",
        "autoscaling:DescribeAutoScalingGroups"
    ],
    "Resource": [
        "arn:aws:elasticbeanstalk:eu-north-1:330552995255:application/mysite",
        "arn:aws:elasticbeanstalk:eu-north-1:330552995255:environment/mysite/*",
        "arn:aws:elasticbeanstalk:eu-north-1:330552995255:applicationversion/mysite/*",
        "arn:aws:cloudformation:eu-north-1:330552995255:stack/awseb-*",
        "arn:aws:autoscaling:eu-north-1:330552995255:autoScalingGroup:*:autoScalingGroupName/*"
    ]
}
]
}
}

```

Step 83: Enforcing Sequential CI→CD Flow

Analysis of runtimes showed that CI and Deployment workflows were previously triggering in parallel upon a push.

Modify `deploy.yml` to use the `workflow_run` event.

This configuration forces the Deployment to start only if the CI workflow has completed with a success conclusion.

```

ci-cd-gha-aws > .github > workflows > ⌘ deploy.yml
1  name: Deploy to Elastic Beanstalk
2
3  on:
4  - workflow_run:
5  - workflows: ["CI"]
6  - types:
7  - completed
8
9  permissions:
10 - id-token: write
11 - contents: read
12
13 env:
14 - AWS_REGION: eu-north-1
15 - APP_NAME: mysite
16 - ENV_NAME: Mysite-env-2
17 - S3_BUCKET: elasticbeanstalk-eu-north-1-330552995255
18
19 jobs:
20 - deploy:
21 - if: ${{ github.event.workflow_run.conclusion == 'success' }}
22 - runs-on: ubuntu-latest
23
24 - steps:
25 - uses: actions/checkout@v4
26 - with:
27 - ref: ${{ github.event.workflow_run.head_sha }}
28 - fetch-depth: 0
29
30 - name: Set up Python
31 - uses: actions/setup-python@v5

```

commit & push
git add .github/workflows/deploy.yml

```
git commit -m " CI quality gates enforced before deployment (sequential CI->CD flow)"  
git push
```

Summary

This project successfully established a fully functional CI/CD pipeline that automates code integration, testing, and deployment to AWS Elastic Beanstalk.

Secure authentication via AWS IAM OIDC replaces static credentials.

The application remains in a stable, "Healthy" state in production, demonstrating the effectiveness of automated DevOps practices.