CI/CD Pipeline

Continuous Integration / Continuous Deployment (CI/CD)

This document explains the automated workflows that build, test, and deploy the **API Threat Assessment Tool (AT-AT)**. The CI/CD pipeline is powered by **GitHub Actions** and triggered on every push and pull request to the `dev` and `main` branches.

CI/CD Goals

- Automatically test code changes on every push
- Ensure main branch stability
- Provide automated deployment hooks for future releases
- Minimize manual errors and speed up delivery

GitHub Workflow Files

All workflows are located in:

```
.github/workflows/
```

File	Purpose
`test.yml`	Run backend tests on push and PR
`lint.yml`	Lint frontend + backend
`deploy.yml` (future)	Reserved for deployment automation

Triggers

Workflows are triggered by:

```
on:
  push:
    branches: [ "dev", "main" ]
  pull_request:
    branches: [ "dev", "main" ]
```

This ensures all changes are validated before merging into production branches.

Example Workflow: Lint + Test

`.github/workflows/test.yml`

```yaml
name: Run Backend Tests

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

jobs:
  test-backend:
    runs-on: ubuntu-latest
    defaults:
      run:
        working-directory: backend
    steps:
      - uses: actions/checkout@v4
      - name: Set up Python
        uses: actions/setup-python@v5
        with:
          python-version: "3.10"
      - name: Install dependencies
        run: |
          python -m venv venv
          source venv/bin/activate
          pip install -r requirements.txt
      - name: Run tests
        run: pytest
```

Other workflows follow a similar pattern for linting and scanning.

Manual Deployment

Until deployment is fully automated, developers deploy manually using:

- Local `.env` setup
- Python backend via `main.py`
- Frontend via `react-scripts start`

Once stable, this will be replaced by a production deploy action.

Future Improvements

- □ Add automated Docker build and push
- □ Enable deployment to Vercel/Render/Fly.io
- □ Tag-based release workflow

Conclusion

The AT-AT GitHub Actions pipeline enforces code quality through linting and testing. CI/CD enables faster iteration, safer merges, and confidence in deploys.