



# Build and Deployment Guide

Edit

New page

[Jump to bottom](#)Aditi edited this page now · [2 revisions](#)

## Build and Deployment Guide

This guide provides instructions on how to build the AI-Driven Testing backend application and deploy it, primarily using Docker.

### 1. Building the Application

#### 1.1. Local "Build" (Development Setup)

For local development and running the application directly without Docker, the "build" process involves setting up the Python environment and installing dependencies. Refer to the [Backend Installation and Setup Guide](#) for detailed steps.

#### 1.2. Building the Docker Image

The project includes a `Dockerfile` in the `backend/` directory to package the backend application into a Docker container image.

##### Prerequisites:

- Docker must be installed and the Docker daemon/service running.

##### Steps:

1. Navigate to the `backend` directory of the project:

```
cd amos2025ss04-ai-driven-testing/backend
```



2. Run the `docker build` command:

```
docker build -t amos-ai-testing-backend .
```



- `-t amos-ai-testing-backend` : Tags the image with the name `amos-ai-testing-backend` (you can choose a different tag).
- `.` : Specifies that the build context (including the `Dockerfile`) is the current directory.

### Understanding the `Dockerfile` ( `backend/Dockerfile` ):

- `FROM python:3.12.3-slim-bookworm` : Specifies the base image (a slim version of Python 3.12.3).
- `WORKDIR /app` : Sets the working directory inside the container to `/app`.
- `COPY requirements.txt .` : Copies the Python dependencies file into the container.
- `RUN pip install --no-cache-dir -r requirements.txt` : Installs the dependencies using pip. `--no-cache-dir` helps keep the image size smaller.
- `COPY . .` : Copies all files from the build context (your `backend` directory) into the `/app` directory in the container.
  - **Important:** It's highly recommended to have a `.dockerignore` file in your `backend` directory to exclude unnecessary files and folders (like `.git`, `venv/`, `__pycache__`, `*.pyc`, `.pytest_cache/`, local `.env` files) from being copied into the image. This keeps the image lean and secure.
- `EXPOSE 8000` : Documents that the application inside the container will listen on port 8000. This does not publish the port; it's informational.
- `CMD ["uvicorn", "main:app", "--host", "0.0.0.0", "--port", "8000"]` : Defines the default command to run when a container is started from this image. It starts the Uvicorn server for the FastAPI application ( `main:app` or `api:app` ).

## 2. Deploying the Application

The primary method for deploying the built application is by running the Docker container.

### 2.1. Running the Docker Container

#### Prerequisites:

- The Docker image must be built (e.g., tagged as `amos-ai-testing-backend`).
- Docker must be running.

#### Command:

```
docker run -d -p 8000:8000 \
  -e OLLAMA_BASE_URL="http://host.docker.internal:11434" \
  --name my-ai-testing-app amos-ai-testing-backend
```



+ Add a custom footer

Pages 40

Find a page...

▶ [Home](#)

▶ [AI-Model Benchmark](#)

▶ [Architecture](#)

▶ [Backend API Design](#)

▶ [Backend Installation and Setup](#)

▶ [Benefits of chaining LLMs](#)

▼ [Build and Deployment Guide](#)

Build and Deployment Guide

1. Building the Application

1.1. Local "Build" (Development Setup)

1.2. Building the Docker Image

2. Deploying the Application

2.1. Running the Docker Container

▶ [CLI Design](#)

▶ [CLI Pipeline](#)

▶ [Code Complexity](#)

▶ [Code Coverage](#)

▶ [Contributing](#)

▶ [DeepCoder](#)

▶ [DeepSeek-Coder V1](#)

▶ [Docker Performance](#)

Show 25 more pages...

+ Add a custom sidebar

### Clone this wiki locally

<https://github.com/amosproj/amos2025ss04-ai-driven-testing/wiki.git>

