### \*\*Docker Setup Instructions\*\*

#### \*\*Prerequisites\*\*

1. Install [Docker](https://docs.docker.com/get-docker/).

2. Install [Docker Compose](https://docs.docker.com/compose/install/) (optional, but recommended).

---

### \*\*Step 1: Project Structure\*\*

Create a folder with the following files:

```

your-project-folder/

|\_\_ app.py

|\_\_ requirements.txt

|\_\_Dockerfile

|\_\_ docker-compose.yml (optional)

### \*\*Step 2: Build the Docker Image\*\*

Open a terminal in the project folder and run:

* docker build -t receipt-processor .

Purpose: Builds a Docker image named `receipt-processor` using the `Dockerfile`.

Notes:

* The `-t` flag tags the image for easier reference.
* The `.` at the end specifies the build context (current directory).

---

### \*\*Step 3: Run the Container\*\*

#### \*\*Option 1: Basic Docker Run\*\*

* docker run -p 5000:5000 --name receipt-service receipt-processor

```

Flags:

* -p 5000:5000`: Maps port 5000 on your machine to port 5000 in the container.
* --name`: Assigns a name to the container for easier management.

#### \*\*Option 2: Docker Compose\*\*

* docker-compose up --build

```

Purpose: Uses `docker-compose.yml` to build and start the service.

Notes:

* The `--build` flag ensures the image is rebuilt if changes are detected.
* To run in detached mode (background), add `-d`:

docker-compose up -d --build

---

### \*\*Step 4: Environment Variables (Optional Configuration)\*\*

The service supports these environment variables:

|  |  |  |
| --- | --- | --- |
| Variable | Default | Purpose |
| FLASK\_HOST | 0.0.0.0 | Binds the server to all network interfaces (required for Docker). |
| FLASK\_PORT | 5000 | Sets the port the server listens on. |
| FLASK\_DEBUG | 0 | Disables debug mode (1 enables it, but avoid in production). |

#### \*\*Override Variables at Runtime\*\*

# Example: Change port to 8080 and enable debug mode

docker run -p 8080:5000 -e FLASK\_PORT=5000 -e FLASK\_DEBUG=1 receipt-processor

---

### \*\*Step 5: Verify the Service\*\*

#### \*\*Test the API Endpoints\*\*

1. \*\*Submit a Receipt\*\*:

curl -X POST http://localhost:5000/receipts/process \

-H "Content-Type: application/json" \

-d '{

"retailer": "Target",

"purchaseDate": "2022-01-01",

"purchaseTime": "13:01",

"items": [

{"shortDescription": "Mountain Dew 12PK", "price": "6.49"},

{"shortDescription": "Emils Cheese Pizza", "price": "12.25"}

],

"total": "35.35"

}'

Expected Response:

{"id": "a1b2c3d4-e5f6-7g8h-9i0j-k1l2m3n4o5p6"}

```

2. \*\*Retrieve Points\*\*:

* curl http://localhost:5000/receipts/a1b2c3d4-e5f6-7g8h-9i0j-k1l2m3n4o5p6/points

Expected Response:

{"points": 28}

---

### \*\*Troubleshooting\*\*

1. \*\*Port Conflicts\*\*:

- Ensure port `5000` (or your custom port) is not in use by another service.

2. \*\*Debug Mode\*\*:

- If the service crashes, temporarily enable `FLASK\_DEBUG=1` to see detailed logs:

* docker run -p 5000:5000 -e FLASK\_DEBUG=1 receipt-processor

```

3. \*\*View Logs\*\*:

- For Docker Compose:

* docker-compose logs

```

- For standalone Docker:

* docker logs receipt-service

```

---

\*\*Cleanup\*\*

1. Stop the Container:

* docker stop receipt-service

```

2. Remove the Container:

* docker rm receipt-service

```

3. Remove the Image (optional):

* docker rmi receipt-processor

```

---

This setup ensures the application runs securely in a containerized environment with minimal configuration. The provided `docker-compose.yml` simplifies orchestration, while the Dockerfile guarantees consistency across deployments.