This command starts the services in detached mode:

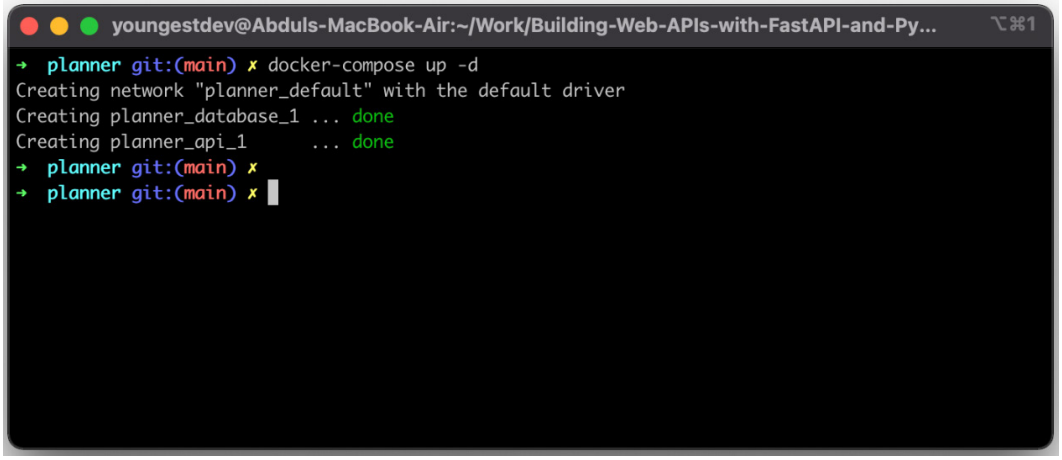

Figure 9.3 – Starting our application using the docker-compose tool

The application services have been created and deployed. Let's verify by checking the list of containers running:

```
(venv)$ docker ps
```

The result is as follows:

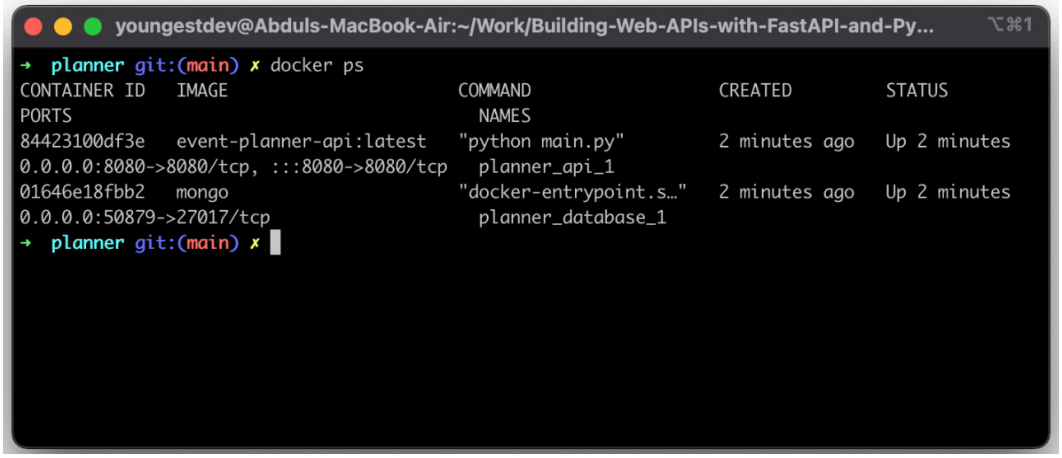

Figure 9.4 – List of running containers

The command returns the list of containers running alongside ports in which they can be accessed. Let's test the working state by sending a `GET` request to the deployed application:

```
(venv)$ curl -X 'GET' \
  'http://localhost:8080/event/' \
  -H 'accept: application/json'
```

We get the following response:

```
[]
```

Great! The deployed application works correctly. Let's verify that the database works as well by creating a user:

```
(venv)$ curl -X 'POST' \
  'http://localhost:8080/user/signup' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
  "email": "fastapi@packt.com",
  "password": "strong!!!"
}'
```

We get a positive response as well:

```
{
  "message": "User created successfully"
}
```

Now that we have tested both routes, you can go ahead to test the other routes. To stop the deployment server after exploring, the following command is run from the root directory:

```
(venv)$ docker-compose down
```

The result is as follows:

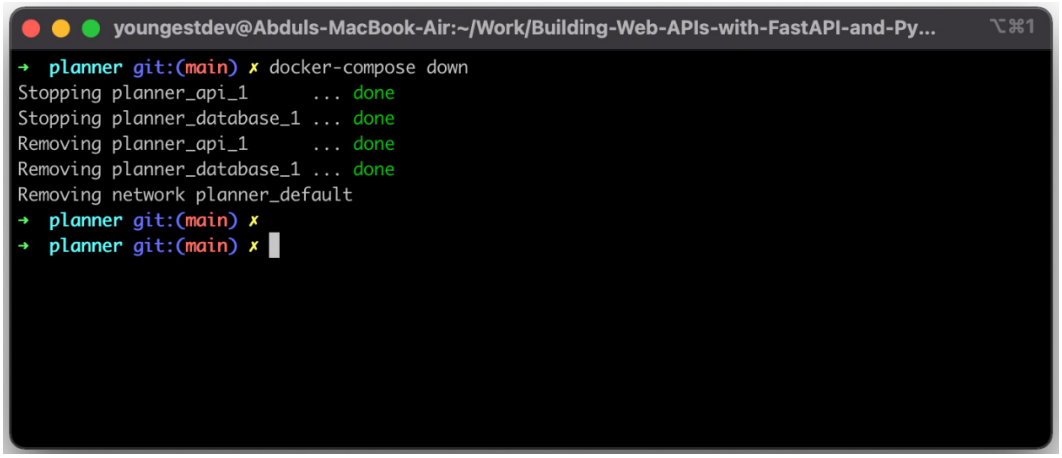

Figure 9.5 – Stopping application instances

Deploying Docker images

In the last section, we learned how to build and deploy Docker images locally. These images can be deployed on any virtual machine and on serverless platforms such as Google Cloud and AWS.

The regular mode of operation involves pushing your Docker images to a private registry on the serverless platform. The process involved in deploying Docker images on serverless platforms varies from provider to provider and, as a result, the links to select serverless service providers have been provided here:

- Google Cloud Run: `https://cloud.google.com/run/docs/quickstarts/build-and-deploy/python`
- Amazon EC2: `https://docs.aws.amazon.com/AmazonECS/latest/developerguide/getting-started-ecs-ec2.html`
- Deploying to Microsoft Azure: `https://docs.microsoft.com/en-us/azure/container-instances/container-instances-tutorial-deploy-app`

The steps covered in the previous section can be followed when the Docker images are to be installed on a machine or traditional server.

Deploying databases

Platforms such as Google Cloud, AWS provide the option to host your database containers. However, this might be expensive in terms of running cost and overall manageability.

For platforms that do not support the deployment of docker-compose manifests, the MongoDB database can be hosted on MongoDB Atlas (`https://www.mongodb.com/atlas/database`) and the `DATABASE_URL` environment variable overwritten with the connection string. A detailed guide on setting up a database on MongoDB atlas can be found at `https://www.mongodb.com/docs/atlas/getting-started/`.

Summary

In this chapter, you learned how to prepare your application for deployment. You started by updating the dependencies used for the application and saving them in the `requirements.txt` file before moving on to managing the environment variables used for the API.

We also covered the steps involved in deploying an application to production: building the Docker image from the Dockerfile, configuring the compose manifest for the API and database services, and then deploying the application. You also learned new commands to check the list of running containers as well as start and stop Docker containers. Finally, you tested the application to ensure the deployment was successful.

This marks the end of this book, and you should now be ready to build, test, and deploy a FastAPI application on the web. We covered various concepts and ensured that each concept is properly discussed with adequate examples: routing, templating, authentication, connecting to the database, and deploying the application. Be sure to check out the external resources mentioned occasionally in the book to gain more knowledge!