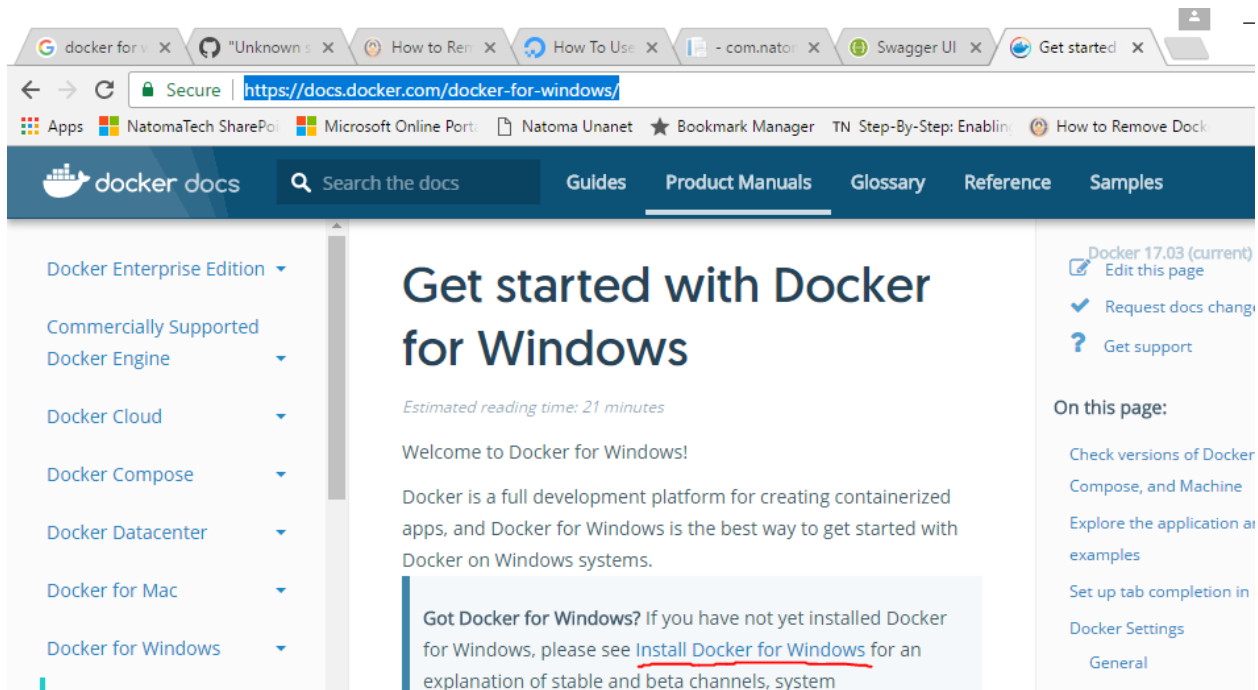


How to install on another Machine

Install “Docker for Windows”. Link can be found [here](#). Click on the “[Install Docker for Windows](#)” link (see screenshot below).



The screenshot shows a web browser window displaying the Docker documentation page for Windows. The browser's address bar shows the URL <https://docs.docker.com/docker-for-windows/>. The page features a dark blue header with the Docker logo and navigation links: Search the docs, Guides, Product Manuals, Glossary, Reference, and Samples. A left sidebar lists various Docker products, with 'Docker for Windows' selected. The main content area is titled 'Get started with Docker for Windows' and includes a sub-header 'Welcome to Docker for Windows!'. The text explains that Docker is a full development platform for creating containerized apps and that Docker for Windows is the best way to get started on Windows systems. A callout box contains the text: 'Got Docker for Windows? If you have not yet installed Docker for Windows, please see [Install Docker for Windows](#) for an explanation of stable and beta channels, system'. On the right side, there are links for 'Docker 17.03 (current)', 'Edit this page', 'Request docs change', and 'Get support'. Below these are 'On this page:' links for 'Check versions of Docker Compose, and Machine', 'Explore the application at examples', 'Set up tab completion in Docker Settings', and 'General'.

After clicking on the link above, choose the “Stable channel” (not the “Edge channel”). Then, execute the downloaded InstallDocker.msi file.

After Docker is installed, open PowerShell or linux and execute the following commands. These commands pull the latest Docker images, stop any running images, remove any existing images, and then run the latest images. Finally, logs are pulled for the latest running images. If any of these commands time out, the command should be executed again.

[Pull the releases from dockerhub](#)

```
docker pull natoma/adpq2:web.release
```

```
docker pull natoma/adpq2:api.release
```

```
docker pull natoma/adpq2:postgresdb.release
```

[stop the docker instances](#)

```
docker stop adpq2prodweb
```

```
docker stop adpq2prodapi
```

```
docker stop adpq2proddb
```

```
docker rm adpq2prodweb
```

```
docker rm adpq2prodapi
```

```
docker rm adpq2proddb
```

[Start the docker instances](#)

```
docker run -d -p 5432:5432 --name adpq2proddb natoma/adpq2:postgresdb.release
```

```
docker run -d -p 5050:5050 --name adpq2prodapi natoma/adpq2:api.release
```

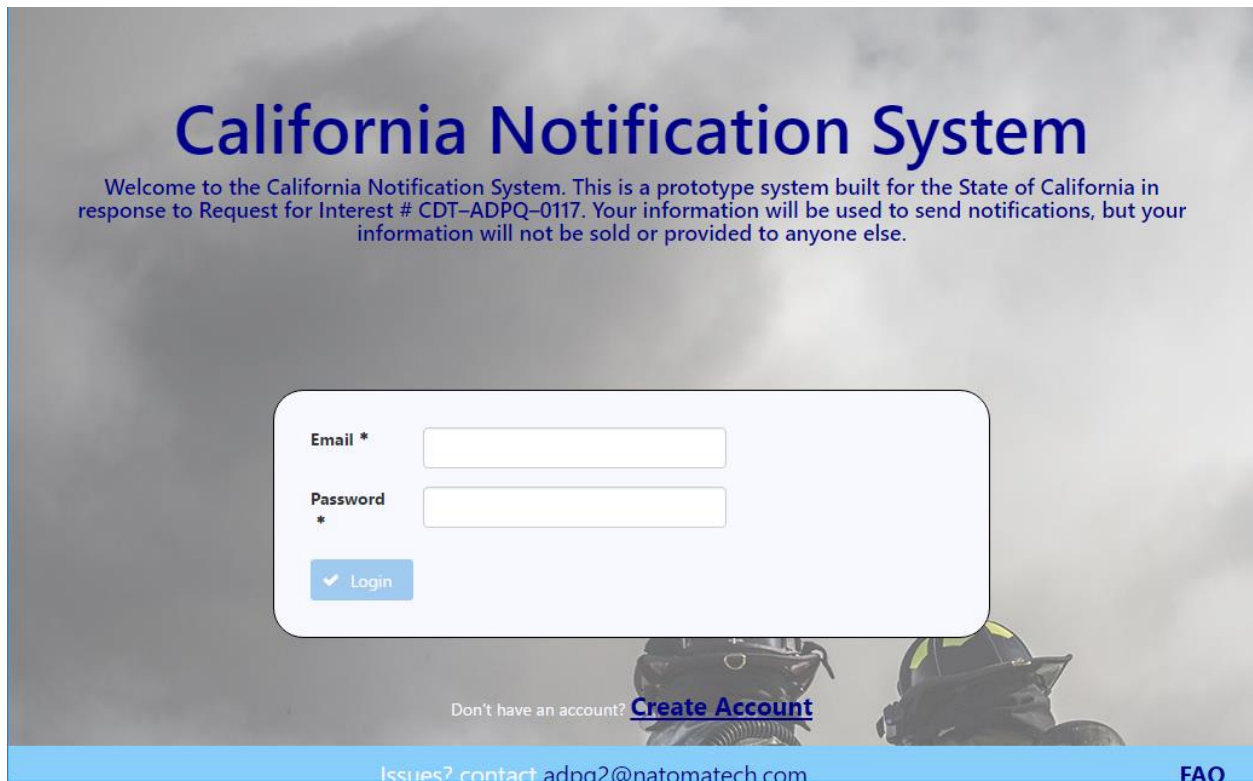
```
docker run -d -p 5000:5000 --name adpq2prodweb natoma/adpq2:web.release
```

```
docker logs adpq2prodweb
```

```
docker logs adpq2prodapi
```

```
docker logs adpq2proddb
```

At this point, the Web server and the Web API server and the Database server should all be ready to test. To validate this, open a browser and key the following URL into the address bar: <http://localhost:5000>. If the following is displayed, the Web server is responding.



You can do the same on the API at <http://localhost:5050/swagger>

Additional information:

The default setup assumes that the Web server Docker container and the Web API server Docker container are both running on the same local machine (i.e. localhost), on the default configured ports (i.e., 5000 and 5050). If this assumption is not true, config files must be changed on one or both Docker containers. The files to change are the `appsettings.production.json` and `appsettings.Production.json` on the `webapi` and `web` projects respectively.

If you encounter any issues connecting to the database you may want to install Postgres locally and run the script named `adpq_db_create_postgres.sql` to create user and database from `Com.Natoma.Adpq.Prototype.Api/DB`.

The Docker images ran well and were used for production on our Amazon Linux environment.