

# User Manual for Docks

Team: TripleParity

Client: Compiax



## Contents

<b>1</b>	<b>System Overview</b>	<b>2</b>
<b>2</b>	<b>System Configuration</b>	<b>2</b>
<b>3</b>	<b>Installation</b>	<b>3</b>
3.1	Configuration . . . . .	3
3.1.1	Docks Web Interface . . . . .	3
3.1.2	Docks API . . . . .	3
<b>4</b>	<b>Getting Started</b>	<b>3</b>
<b>5</b>	<b>Using the System</b>	<b>5</b>
5.1	Tasks . . . . .	5
5.1.1	Viewing Tasks . . . . .	5
5.1.2	View Task Logs . . . . .	5
5.2	Services . . . . .	6
5.3	Stacks . . . . .	8
5.3.1	Deploying a Stack . . . . .	9
5.3.2	Updating a Stack . . . . .	10
5.4	User Management . . . . .	11
5.5	Updating Docks . . . . .	13
<b>6</b>	<b>Troubleshooting</b>	<b>13</b>
6.1	Error: bind: address already in use . . . . .	13

# 1 System Overview

This document assumes knowledge of how Docker works. For more information read:

- [About Docker Engine](#)
- [Docker Overview](#)
- [Swarm Mode Key Concepts](#)

Docks provides a web interface for managing a Docker Swarm. Along with an easy to use interface Docks provides security by only allowing registered users to manage Docker. Docks exposes the same functionality as the Docker Command Line Interface.

Docks allows developers and system administrators to manage the deployment of applications without requiring SSH access to a server.

## 2 System Configuration

Docks consists of two subsystems:

- Docks Web Interface
- Docks API Server

The web interface can be served from any static file server such as GitHub pages and will communicate with the Docks API server through the web browser.

The Docks API server is deployed on a Manager Node in the Docker Swarm

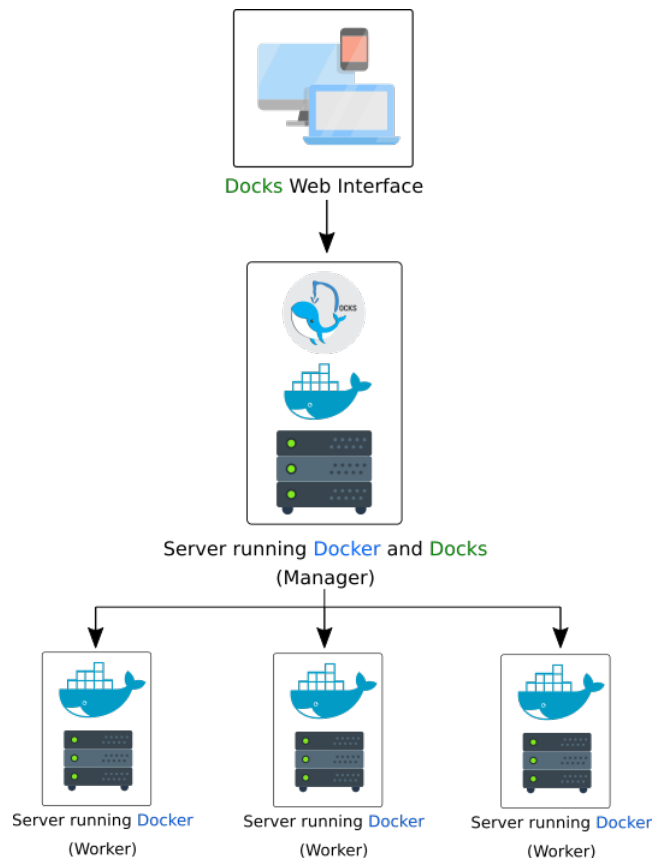


Figure 1: Docks Deployment Diagram

## 3 Installation

1. Install [Docker](#) 17.06.2-ce or higher
2. Install [Docker Compose](#)
3. Create a Swarm using `sudo docker swarm init`
4. Clone `https://github.com/TripleParity/docks.git`
5. Run `sudo docker-compose pull` to download the required images
6. Run `sudo docker stack deploy -c docker-compose.yml docks` to deploy Docks
7. Run `sudo docker stack deploy -c docker-compose-nginx.yml demo` to deploy a sample application
8. Browse to <http://127.0.0.1:4200> to view the Docks web interface
9. To remove Docks from the system run the following commands:
  - `sudo docker stack rm docks`
  - `sudo docker stack rm demo`

### 3.1 Configuration

#### 3.1.1 Docks Web Interface

The following parameters can be configured in the `docker-compose.yml` file

- `DOCKS_API_ADDRESS` - The address of the Docks API server
- `ports` - The ports to listen on

#### 3.1.2 Docks API

The following parameters can be configured in the `docker-compose.yml` file


- `JWT_SECRET` - The secret key used during authentication requests
- `DOCKS_DB_ADDRESS` - The address of the database
- `POSTGRES_PASSWORD` - The password for the `postgres` database user
- `ports` - The ports to listen on

## 4 Getting Started

The web interface will be available at <http://127.0.0.1:4200> after following the Installation instructions. The database will automatically be initialized and the default user will be created:

- Username: `admin`
- Password: `admin`

It is strongly recommended to change the password as soon as possible. This will be explained further in [User Management](#)



# Login

Username

Password

LOGIN

Figure 2: Login Page

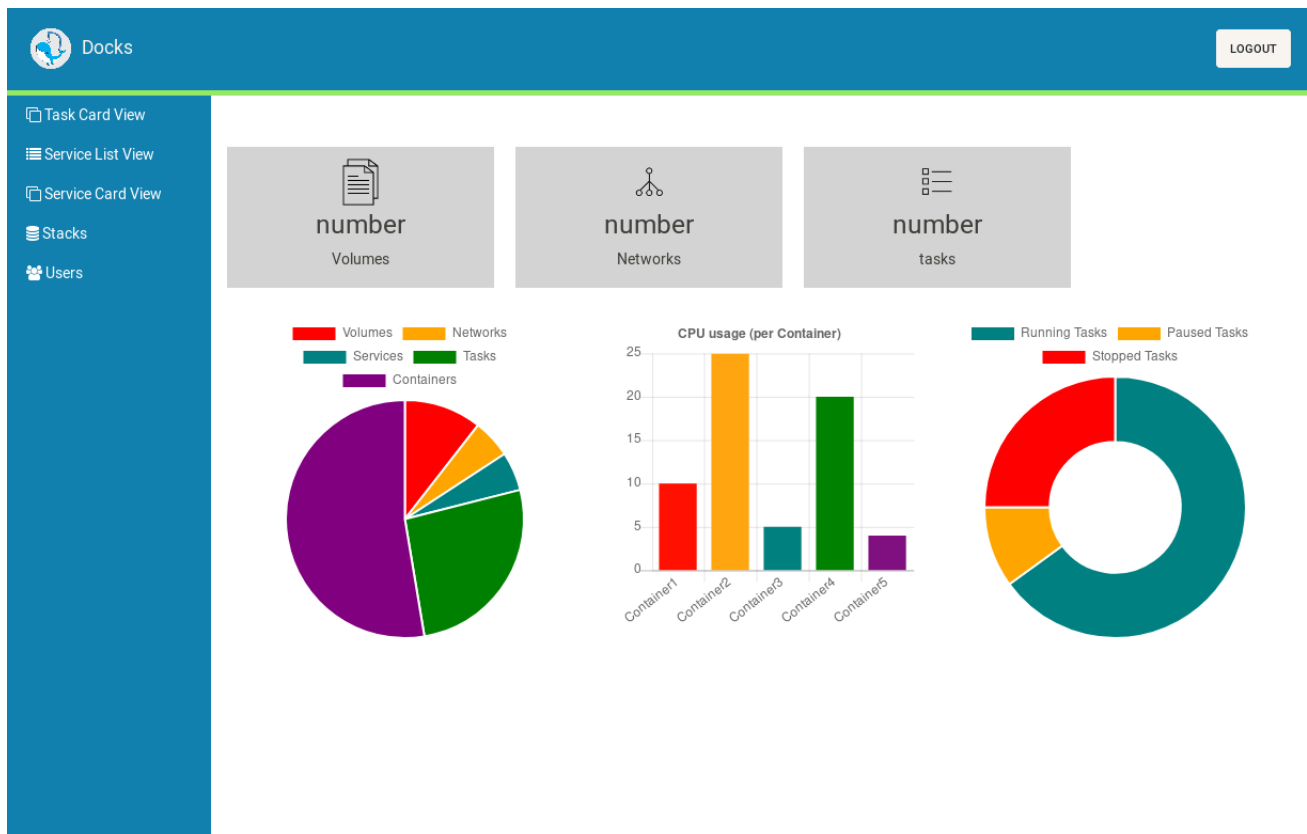


Figure 3: Home Screen

## 5 Using the System

### 5.1 Tasks

A Service consists of Tasks. Tasks can only be viewed - they are managed by Docker as part of services

#### 5.1.1 Viewing Tasks

The card view displays the following information:

- ID of the task
- State of the task. Indicated by the circle and card outline colour. Green means the task is running, red indicating it has stopped and blue indicating it is preparing
- Node ID the task is running on
- Last Updated

ID: 5rsfga ●	ID: acio8d ●	ID: dfoci8 ●
ID: 5rsfgadox8cr7wd7jn2kk9s50	ID: acio8d9pft26s0ng94jhr7a0x	ID: dfoci8g0pns181ke981118zza
NodeID: bha3m8lbcjwy87kqds5epv3wb	NodeID: bha3m8lbcjwy87kqds5epv3wb	NodeID: bha3m8lbcjwy87kqds5epv3wb
Image: postgres:latest@sha256:2aa5d4952a8abc	Image: registry:2@sha256:672d519d7fd7bbc7a44	Image: nginxdemos/hello:0.2@sha256:f5a0b2a5f
Last Updated: 2018-07-28 12:08	Last Updated: 2018-07-28 12:08	Last Updated: 2018-07-28 12:08

ID: dn2qew ●	ID: et1bkj ●	ID: fk61x9 ●
ID: dn2qewwlunh5ob3zgden5443d	ID: et1bkjdfa9asiuxtc0jtguhc1	ID: fk61x90uz89sfwq3yfb8anjq8
NodeID: bha3m8lbcjwy87kqds5epv3wb	NodeID: bha3m8lbcjwy87kqds5epv3wb	NodeID: bha3m8lbcjwy87kqds5epv3wb
Image: nginxdemos/hello:0.2@sha256:f5a0b2a5f	Image: nginxdemos/hello:0.2@sha256:f5a0b2a5f	Image: nginxdemos/hello:0.2@sha256:f5a0b2a5f
Last Updated: 2018-07-28 12:09	Last Updated: 2018-07-28 12:09	Last Updated: 2018-07-28 12:08

ID: fn17e9 ●	ID: fydwa8 ●	ID: gk4q2p ●
ID: fn17e98hbywtuudm1th3t7k	ID: fydwa84fzjbhkgkaetpp7wf7j	ID: gk4q2pvm76wtq422y09ugrrp3
NodeID: j8ewl94i7twd2fm0v2sozh82p	NodeID: j8ewl94i7twd2fm0v2sozh82p	NodeID: bha3m8lbcjwy87kqds5epv3wb

Figure 4: Task Card View

#### 5.1.2 View Task Logs


From the task card view, a task can be clicked to bring up a modal for viewing the log associated with the task.



Figure 5: Task Card Modal

## 5.2 Services

The service list view provides a sortable and searchable table for deployed services.


Docks

Task Card View

Service List View

Service Card View

Stacks

Users


CREATE SERVICE

Type to filter the name column...

Name	ID	Stack	Image	Mode	Replicas	Ports	Created At	Updated At	Ownership
docks-ui-ui	4wuz3m723mz		tripleparity/doc ui:branch- develop@sha256:...	vip	1		2018-06-27 21:36	2018-07-28 12:34	
demo_nginx	kh6wsvfo3s0c		nginxdemos/he	vip	5		2018-06-28 07:43	2018-07-28 12:08	
traefik_traefik	n6rpe8h3k9s28		traefik:1.6.4@st	vip	1	80	2018-06-19 21:42	2018-07-28 12:08	
docks-api_db	s8938nox1s9q		postgres:latest	vip	1		2018-06-28 07:40	2018-07-28 12:08	
registry	t4un4bhfkk91j5		registry:2@sha256:...	vip	1	5000	2018-06-30 06:10	2018-07-28 12:08	
docks-api_api	wt8fyobtckhim		tripleparity/doc api:0.2.0@sha256:...	vip	1		2018-06-28 07:40	2018-07-28 12:08	

0 selected / 6 total

Figure 6: Service List View


Docks

LOGOUT

Task Card View

Service List View

Service Card View

Stacks

Users

docks-ui\_ui

ID: 4wuz3m723mzopcrem70m9lzk

Image: tripleparity/docks-ui:branch-develop@sha256:83e545b59d091f259f033

Mode: vip

Replicas: 1

Last Updated: 2018-07-28 12:34

demo\_nginx

ID: kh6wsvfo3s0c18xsty67zh4iy

Image: nginxdemos/hello:0.2@sha256:f5a0b2a5f

Mode: vip

Replicas: 5

Last Updated: 2018-07-28 12:08

traefik\_traefik

ID: n6rpe8h3k9s280wzejp3jfo1c

Image: traefik:1.6.4@sha256:98d82a5fa9a27382d

Mode: vip

Replicas: 1

Last Updated: 2018-07-28 12:08

docks-api\_db

ID: s8938nox1s9qmb644rjk1po3

Image: postgres:latest@sha256:d8011033f12f1ct

Mode: vip

Replicas: 1

Last Updated: 2018-07-28 12:08

registry

ID: t4un4bhfkk91j92grsgrh2y1l

Image: registry:2@sha256:672d519d7fd7bbc7a44

Mode: vip

Replicas: 1

Last Updated: 2018-07-28 12:08

docks-api\_api

ID: wt8fyobtckhinvozkjfcizog

Image: tripleparity/docks-api:0.2.0@sha256:8a54c830892bb8e34c2

Mode: vip

Replicas: 1


Last Updated: 2018-07-28 12:08

Figure 7: Service Card View

### 5.3 Stacks

The concept of a Stack is the core feature of Docker. A Stack describes Services (applications) and how they should be deployed.



 Docks

LOGOUT

Task Card View

Service List View

Service Card View

Stacks

Users

DEPLOY STACK

Search Stacks





Stack Name	Services Count	Operations
demo	1	Edit  <div>DELETE</div>
docks-api	2	Edit  <div>DELETE</div>
docks-ui	1	Edit  <div>DELETE</div>
traefik	1	Edit  <div>DELETE</div>
4 total		

Figure 8: List of Stacks in the Swarm

### 5.3.1 Deploying a Stack

Docks allows the user to deploy their own Stack (Application) by uploading a Stack file. Pre-configured stacks can also be selected for deployment, such as Wordpress, Nginx, and MongoDB. The Stack File should be modified to fit the needs of the administrator and the system.

The Stack should be given a unique name to identify it in the system.

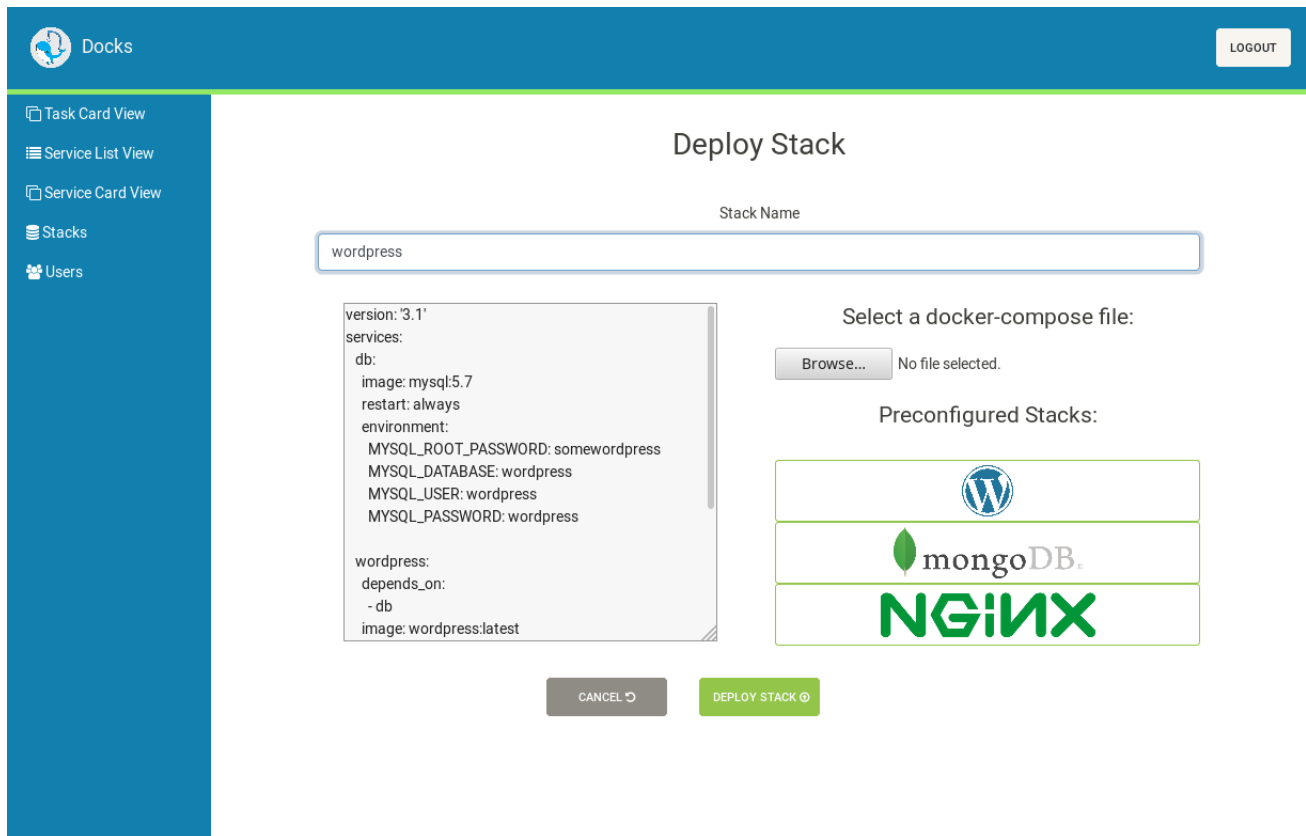


Figure 9: Deploying a Wordpress Stack

### 5.3.2 Updating a Stack

By uploading a modified Stack file Docker will automatically update the deployed Stack to match the Stack file.

Reasons for updating the stack may include:

- Updating to a new version
- Adding a new application to the Stack
- Changing the port for a running application
- Attaching new volumes to applications for extra storage
- Changing environmental variables used for configuration

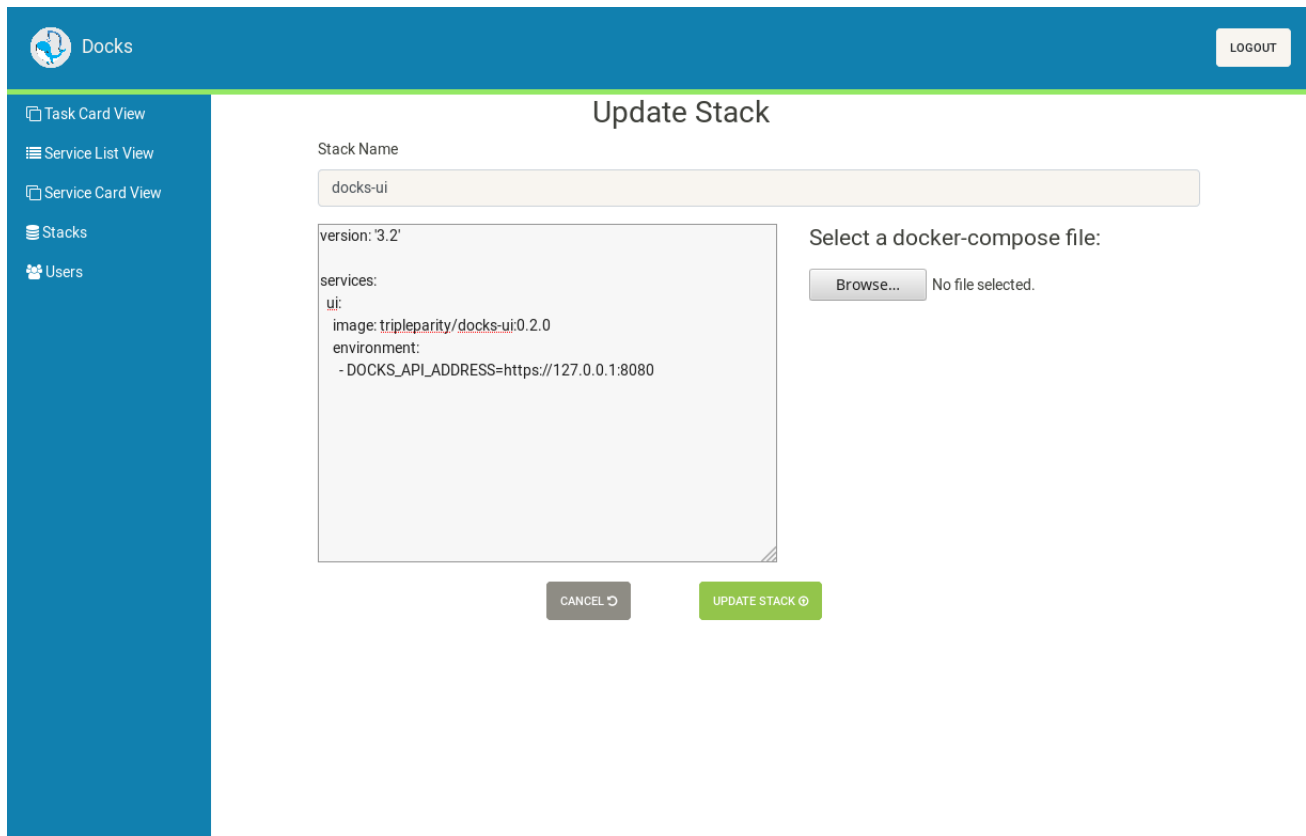



Figure 10: Updating the Docks Web Interface Stack

## 5.4 User Management

Only registered users can use Docks. An admin user can create more admin users. Currently there exists only admin users, so be sure to trust the person that will be using the account.

Users can be created using the Create User Page (Users -> Create):

Passwords can be changed using the Change Password page (Users -> Edit)

Docks

Task Card View

Service List View

Service Card View

Stacks

Users

LOGOUT

Users

CREATE USER

List of users

Filter username...

Username	Operations
admin	<div>EDIT</div> <div>DELETE</div>
demo	<div>EDIT</div> <div>DELETE</div>
2 total	

Figure 11: Users

The screenshot shows the Docks application interface. At the top, there is a blue header bar with the 'Docks' logo on the left and a 'LOGOUT' button on the right. On the left side, there is a vertical blue sidebar containing navigation links: 'Task Card View', 'Service List View', 'Service Card View', 'Stacks', and 'Users'. The 'Users' link is highlighted. The main content area is white and titled 'Create User'. It contains three input fields: 'Username' with the value 'john', and two 'Password' fields, both masked with dots. Below the input fields are two buttons: 'CANCEL' (grey) and 'SUBMIT' (green).

Figure 12: Users

## 5.5 Updating Docks

Once Docks is deployed, it can be updated as described in [Updating a Stack](#)

Alternatively the respective Stack file can be modified locally as deployed using `sudo docker stack deploy -c docker-c`

## 6 Troubleshooting

### 6.1 Error: bind: address already in use

Another service is most likely running on port `4200` , `8080` or `8081` . The ports for Docks and nginx-demo can be specified in the *docker-compose.yml* and *docker-compose.yml* files.

For example to run on port 9000 instead of 4200 make the following changes:

```
ports:
  - 4200:80
```

to

```
ports:
  - 9000:80
```