

# Application Configuration On Fly.io

## Questions Covered:-

- How to install fly CLI
- How to deploy new project on fly
- How to deploy existing project on fly
- How to setup MySQL on fly
- How to access MySQL using proxy connection
- **How to connect fly terminal using SSH**
- How to add, edit ENV variable
- How to check fly logs using terminal
- Detail explanations on fly.toml file's configuration
- List of useful command with explanation
- List of issues we faced and how we resolve it

Prepared By:-

Pallavi Thakre & Sagar Deshmukh

Date: 4 Mar 2023

# APPLICATION CONFIGURATION

## (How to install fly CLI)

Flyctl is a command-line utility that lets you work with Fly.io, from creating your account to deploying your applications. It runs on your local device so you'll want to install the version that's appropriate for your operating system

**We are operating on Ubuntu 20.04 so we can run below command**

### Linux

Run the install script:

```
$ curl -L https://fly.io/install.sh | sh
```

**If We operating on Windows so we Can below command**

### Windows

Run the Powershell install script:

```
$ powershell -Command "iwr https://fly.io/install.ps1 -useb | iex"
```

- If this is your first time with Fly.io, your next step will be to [Sign up](#).
- If you already have a Fly.io account, then your next step is to [Sign in to Fly.io](#).

## Sign Up Process

### (How to deploy new project on fly)

#### Sign up

If it's your first time using Fly.io, you'll need to sign up for an account. To do so, run:

```
$ flyctl auth signup
```

This will take you to the sign-up page where you can either:

- **Sign up with email:** Enter your name, email and password.
- **Sign up with GitHub:** If you have a GitHub account, you can use that to sign up. Look out for the confirmatory email we'll send you which will give you a link to set a password; you'll need a password set so we can actively verify that it is you for some Fly.io operations.

## Sign In Process

### (How to deploy existing project on fly)

#### Sign In

If you already have a Fly.io account, all you need to do is sign in with flyctl. Simply run:

```
$ fly auth login
```

Your browser will open up with the Fly.io sign-in screen, enter your user name and password to sign in. If you signed up with GitHub, use the [Sign in with GitHub](#) button to sign in.

Whichever route you take you will be returned to your command line, ready to use Fly.io.

## Launch an App on Fly

## **Command:- flyctl launch**

```
$ flyctl launch --image flyio/hellofly:latest
```

```
? App Name (leave blank to use an auto-generated name):
```

## **Select Organization**

We are Having 2 Options over there are as follows

- 1)Organizations
- 2)Personal

**Organizations:** Organizations are a way of sharing applications and resources between Fly.io users. Every Fly.io account has a personal organization, called `personal`, which is only visible to your account. Let's select that for this guide.

If you did not add another organization to your account, the `personal` organization will be selected automatically.

```
? Select organization: Personal (personal)
```

## **Select Region:**

On our custom pricing wholesale app we select MAA region for INDIA

```
? Select region: ord (Chicago, Illinois (US))
```

Next, you'll be prompted to select a region to deploy in. The closest region to you is selected by default. You can use this or change to another region.

We get 2 option here for set up in database,we was select NO !!

```
? Would you like to set up a Postgresql database now? (y/N)
```

Then CLI will ask if you need a Postgresql database. You can reply no for now.

At this point, `flyctl` creates an app for you and writes your configuration to a `fly.toml` file. The `fly.toml` file now contains a default configuration for deploying your app.

Lastly, the CLI will ask you if you would like to deploy now - please say no if your app doesn't use `internal_port = 8080`.

Now, it will build the process or the image with fly.io.

```
? Would you like to deploy now? Yes
==> Building image
```

```
app = "hellofly"

[build]
  image = "flyio/hellofly:latest"

[[services]]
  internal_port = 8080

...
```

Building Image:-It will Fetch On When we HIT the URL . URL is created after final deployment.

After that we run **command:- flyctl deploy**

```
$ flyctl deploy
```

### **After Deployment:- Checking the Hostname**

Hitting the Hostname on search bar -bring in build image  
"Hello from fly"

### Application Information

Personal details and application.

Hostname	custom-pricing-wholesale.fly.dev	VM size	shared-cpu-1x
Memory	256MB	CPU cores	1.0
IP addresses	<div>66.241.125.210 <span>Shared v4</span></div> <div>2a09:8280:1::a:b7b4 <span>v6</span></div>		

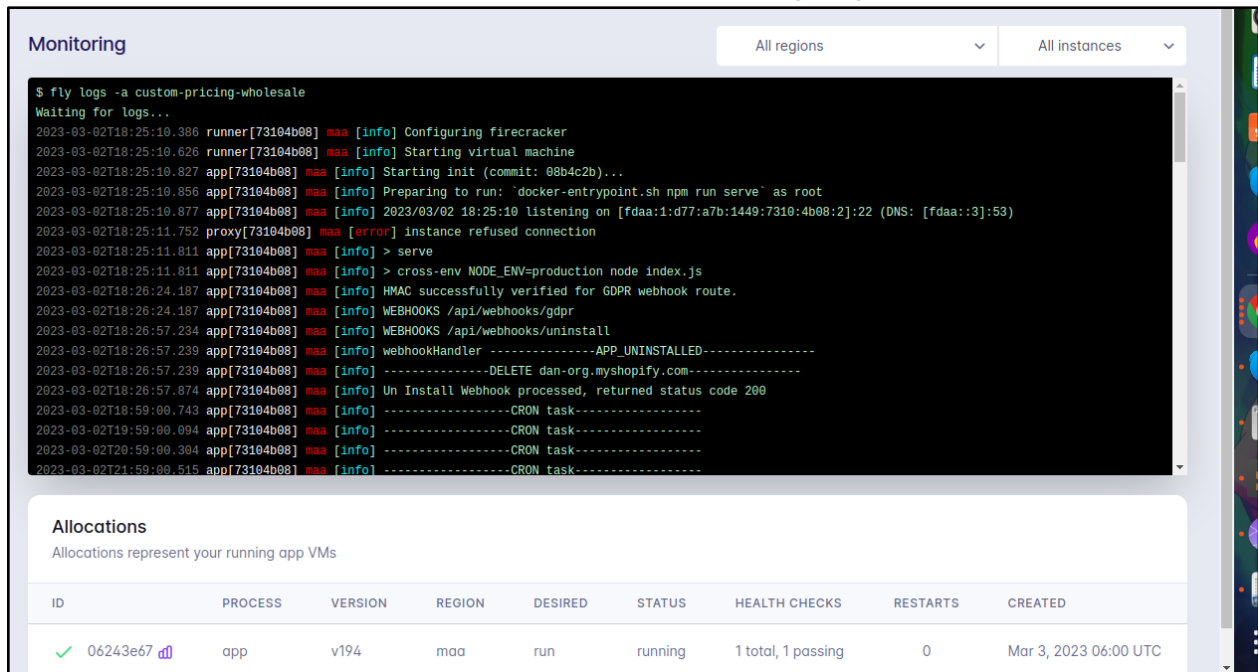
Hostname Allocated (Dashboard)

ip's Allocated



**Monitoring:- Log's**  
**(How to check fly logs using terminal )**

## Command:-flyctl logs —(If You Want monitoring logs on terminal.)

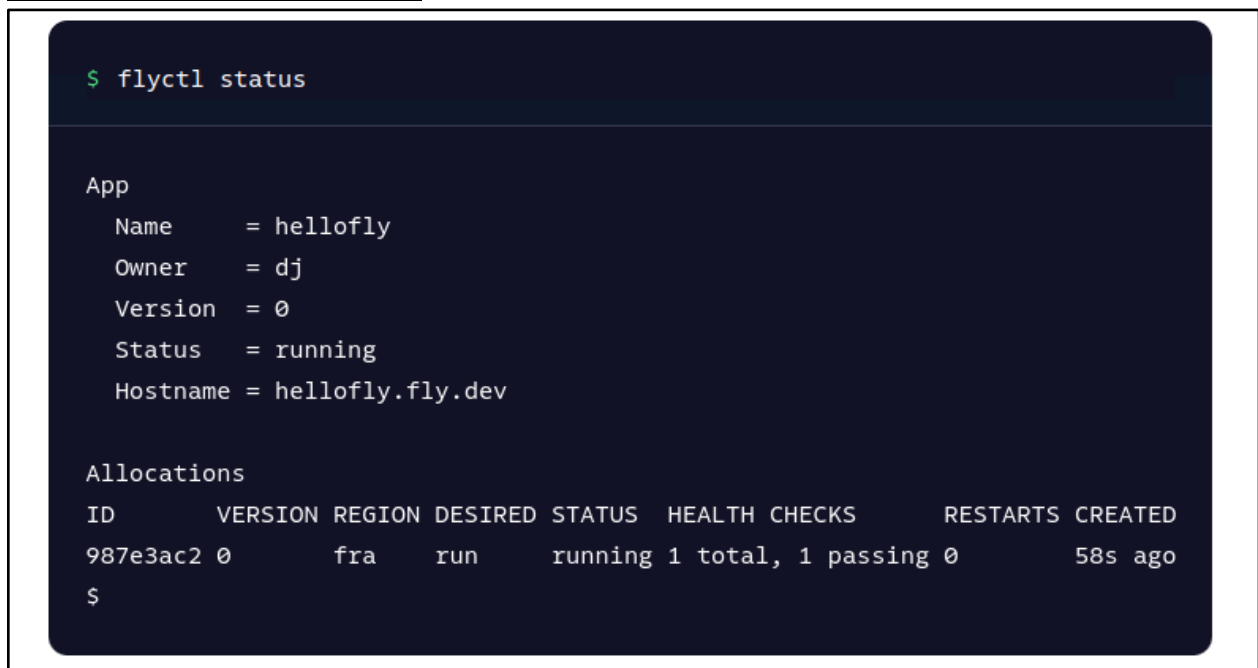


The screenshot shows the Fly.io Monitoring interface. At the top, there are filters for 'All regions' and 'All instances'. Below this is a large log viewer showing the output of the command `$ fly logs -a custom-pricing-wholesale`. The logs show the application's startup sequence, including configuring firecracker, starting a virtual machine, and the application's internal logs. Below the log viewer, there is a section titled 'Allocations' with the text 'Allocations represent your running app VMs'. Below this is a table with the following columns: ID, PROCESS, VERSION, REGION, DESIRED, STATUS, HEALTH CHECKS, RESTARTS, and CREATED.

ID	PROCESS	VERSION	REGION	DESIRED	STATUS	HEALTH CHECKS	RESTARTS	CREATED
✓ 06243e67	app	v194	maa	run	running	1 total, 1 passing	0	Mar 3, 2023 06:00 UTC

## Checking App's Status

### Command:-flyctl status



The screenshot shows the terminal output of the command `$ flyctl status`. The output is divided into two sections: 'App' and 'Allocations'.

**App**

- Name = hellofly
- Owner = dj
- Version = 0
- Status = running
- Hostname = hellofly.fly.dev

**Allocations**

ID	VERSION	REGION	DESIRED	STATUS	HEALTH CHECKS	RESTARTS	CREATED
987e3ac2	0	fra	run	running	1 total, 1 passing	0	58s ago

\$

## Set environment variables:- (How to add, edit ENV variable)

## Reference Document:-


<https://shopify.dev/docs/apps/deployment/web#set-env-vars>

- PORT = ""
- HOST = ""
- SCOPES = ""
- SHOPIFY\_API\_KEY = ""
- SHOPIFY\_API\_SECRET = ""

## Configure FLY.io

Update the `fly.toml` file

1. From your project root, open `fly.toml`.
2. In the `[env]` section, add the following environment variables, using the format `<VARIABLE>= "<VALUE>"`:



```
1 PORT = ""
2 HOST = ""
3 SHOPIFY_API_KEY = ""
4 SCOPES = ""
```

3. In the `[[services]]` section, change the value of `internal_port`. This value needs to match the `PORT` value.

## Set environment secrets

**Command :-** `flyctl secrets set SHOPIFY_API_SECRET=<API_SECRET>`

## To deploy Fly.io

**Command:** `-flyctl deploy --build-arg`

`SHOPIFY_API_KEY=<SHOPIFY_API_KEY> --remote-only`

**“After the process done update URL’s in the PartnerDashboard.(shopify Dashboard)”**

## Original fly.toml File:-



Here is an example of the fly.toml file. Do change parameters according to your specification but take backup before changes.

## **(Detail explanations on fly.toml file's configuration)**

# fly.toml file generated for (App Name) on 2023-02-23T17:03:31+05:30

```
app = "*****"
kill_signal = "SIGINT"
kill_timeout = 5
processes = []
```

```
[build]
  image = "flyio/hellofly:latest"
```

```
[experimental]
  auto_rollback = true
```

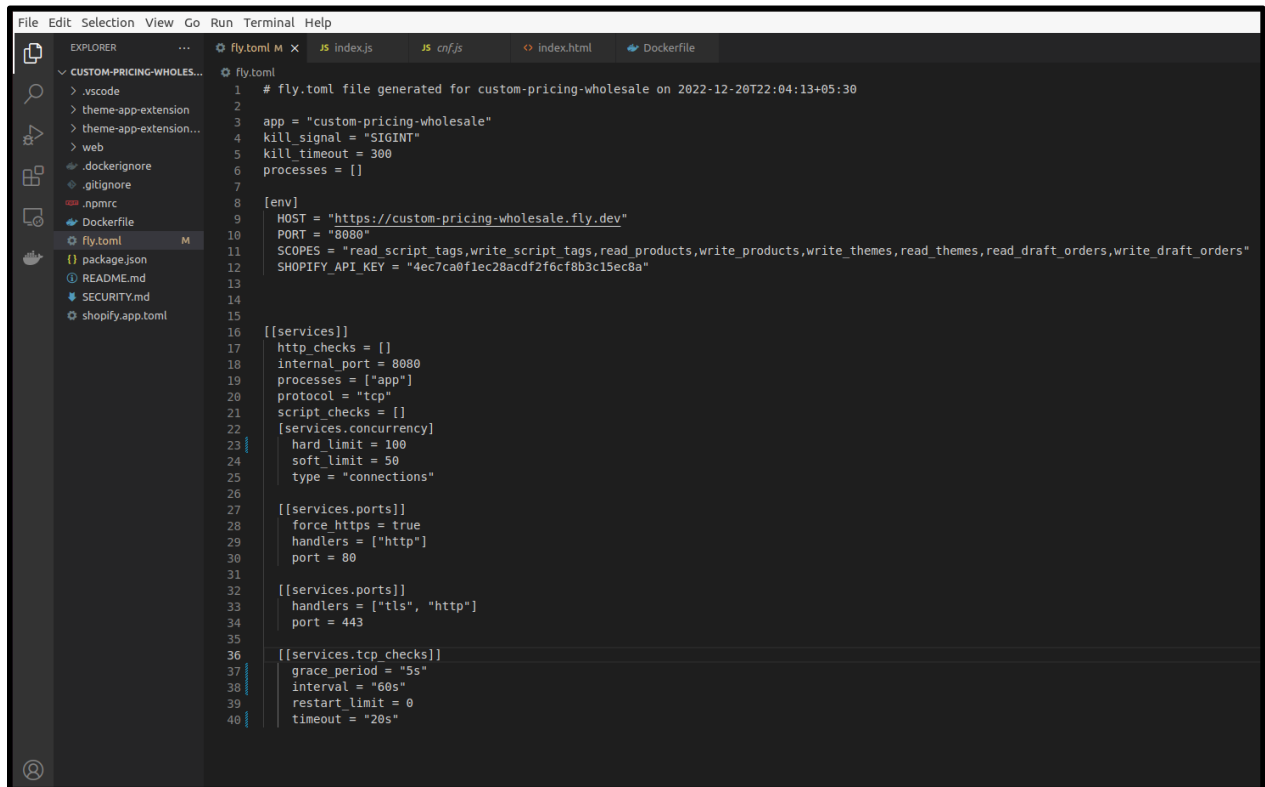
```
[[services]]
  http_checks = []
  internal_port = 8080
  processes = ["app"]
  protocol = "tcp"
  script_checks = []
  [services.concurrency]
    hard_limit = 25
    soft_limit = 20
    type = "connections"
```

```
[[services.ports]]
  force_https = true
  handlers = ["http"]
  port = 80
```

```
[[services.ports]]
  handlers = ["tls", "http"]
  port = 443
```

```
[[services.tcp_checks]]
  grace_period = "1s"
  interval = "15s"
  restart_limit = 0
  timeout = "2s"
```

## Changing Parameter fly.toml File:- (we can change parameter according to trial basis error)



The image shows a screenshot of the Visual Studio Code editor interface. The Explorer sidebar on the left displays the file structure of a project named 'CUSTOM-PRICING-WHOLES...'. The file 'fly.toml' is selected and highlighted. The main editor window displays the content of 'fly.toml', which is a TOML configuration file for the Fly.io platform. The file contains metadata, environment variables, and service configurations.

```
1 # fly.toml file generated for custom-pricing-wholesale on 2022-12-20T22:04:13+05:30
2
3 app = "custom-pricing-wholesale"
4 kill_signal = "SIGINT"
5 kill_timeout = 300
6 processes = []
7
8 [env]
9 HOST = "https://custom-pricing-wholesale.fly.dev"
10 PORT = "8080"
11 SCOPES = "read_script_tags,write_script_tags,read_products,write_products,write_themes,read_themes,read_draft_orders,write_draft_orders"
12 SHOPIFY_API_KEY = "4ec7ca0f1ec28acdf2f6cf8b3c15ec8a"
13
14
15
16 [[services]]
17 http_checks = []
18 internal_port = 8080
19 processes = ["app"]
20 protocol = "tcp"
21 script_checks = []
22 [services.concurrency]
23   hard_limit = 100
24   soft_limit = 50
25   type = "connections"
26
27 [[services.ports]]
28   force_https = true
29   handlers = ["http"]
30   port = 80
31
32 [[services.ports]]
33   handlers = ["tls", "http"]
34   port = 443
35
36 [[services.tcp_checks]]
37   grace_period = "5s"
38   interval = "60s"
39   restart_limit = 0
40   timeout = "20s"
```

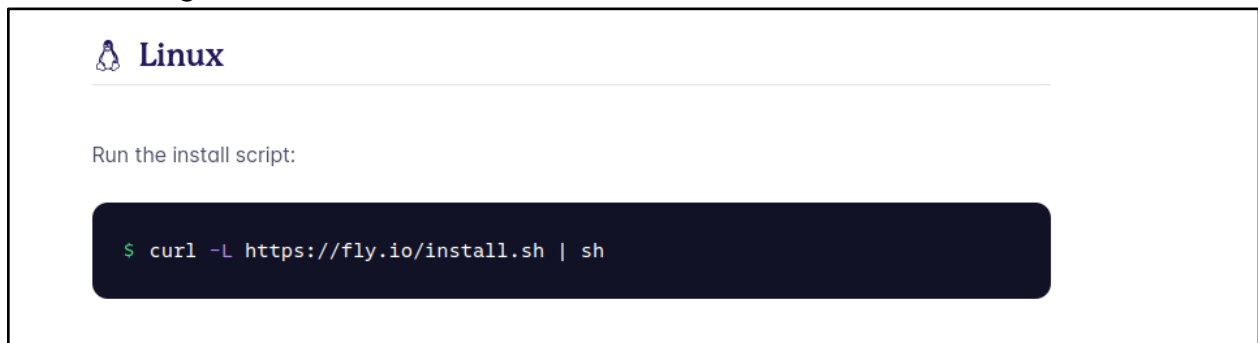
# MySQL Database Configuration

(How to setup MySQL on fly)

Reference Document:- <https://fly.io/docs/app-guides/mysql-on-fly/>

We are operating on Ubuntu 20.04 so we can run below command

After Adding cURL it



will give export command on desktop.

Add export in command line.

Example:- `export FLYCTL_INSTALL="/home/****/.fly"`  
`export PATH="$FLYCTL_INSTALL/bin:$PATH"`

```
set channel to shell
flyctl was installed successfully to /home/tumon/.fly/bin/flyctl
Manually add the directory to your $HOME/.bash_profile (or similar)
  export FLYCTL_INSTALL="/home/tumon/.fly"
  export PATH="$FLYCTL_INSTALL/bin:$PATH"
Run '/home/tumon/.fly/bin/flyctl --help' to get started
```

## Sign In Process

### Sign In

If you already have a Fly.io account, all you need to do is sign in with flyctl. Simply run:

```
$ fly auth login
```

Your browser will open up with the Fly.io sign-in screen, enter your user name and password to sign in. If you signed up with GitHub, use the [Sign in with GitHub](#) button to sign in.

Whichever route you take you will be returned to your command line, ready to use Fly.io.

Making Directory For The Mysql App and after that we can run

### Command:-fly launch

```
# Make a directory for the mysql app
mkdir my-mysql
cd my-mysql

# Run `fly launch` to create an app
fly launch
```

## Configure the App

After Launching we can create volume for app.

Please Add this command in console and we set the size in 4 gb according to our app.

```
# Create a volume named "mysqldata" within our app "my-mysql"
fly volumes create mysqldata --size 10 # gb
```

## Set the PASSWORD and USER ID :-

```
# Set secrets:
# MYSQL_PASSWORD      - password set for user $MYSQL_USER
# MYSQL_ROOT_PASSWORD - password set for user "root"
fly secrets set MYSQL_PASSWORD=password MYSQL_ROOT_PASSWORD=password
```

- After first Deployment fly.toml file created as like as App deployment
- Please Do Not change anything From File.
- We Dont Require to create Docker File For Mysql.
- We deleted the [[services]] block and everything under it. We don't need it!
- Don't Use mysql user as "root" it Will Give Error.Please change Name.

```
app = "my-mysql"
kill_signal = "SIGINT"
kill_timeout = 5

[mounts]
  source="mysqldata"
  destination="/data"

[env]
  MYSQL_DATABASE = "some_db"
  MYSQL_USER = "non_root_user"

[build]
  image = "mysql:8"

[experimental]
  cmd = [
    "--default-authentication-plugin",
    "mysql_native_password",
    "--datadir",
    "/data/mysql"
  ]
```

- If you use MySql 5.7 version so you Dont Need to Change VM RAM.

- No need to change we set 2 GB=2048 MB RAM

```
# Give the vm 2GB of ram
fly scale memory 2048
```

- No need to change

```
# Add the performance schema and buffer pool size flags
# Note that --performance-schema=OFF is all one string
[experimental]
cmd = [
  "--default-authentication-plugin", "mysql_native_password",
  "--datadir", "/data/mysql",
  "--performance-schema=OFF",
  "--innodb-buffer-pool-size", "64M"
]
```

## Deploy the App

```
fly deploy
```

## Access the Database From Outside

## (How to access MySQL using proxy connection)

We Didn't Use this Command Because MySql Port is Already Used Default Port.

You can forward the MySQL server port to your local machine using [fly\\_proxy](#):

```
$ flyctl proxy 3306 -a my-mysql
```

You can also set a different local port, if your [3306](#) port is already in use:

```
$ flyctl proxy 13306:3306 -a my-mysql
```

Then connect to your MySQL server at [localhost:3306](#) and the username and password credentials from above:

```
$ mysql -h localhost -P 3306 -u non_root_user -ppassword some_db
```

**Original MySql Database File**

# fly.toml file generated for cpw2mysql on 2023-01-06T12:35:13+05:30

```
app = "*****"  
kill_signal = "SIGINT"  
kill_timeout = 5  
processes = []
```

```
[mounts]  
  source="mysqldata"  
  destination="/data"
```

```
[env]  
  MYSQL_DATABASE = "*****"  
  MYSQL_USER = "*****"
```

```
[build]  
  image = "mysql:8"
```

```
[experimental]  
  cmd = [  
    "--default-authentication-plugin",  
    "mysql_native_password",  
    "--datadir",  
    "/data/mysql"  
  ]
```



# Fly.io Commands

(List of useful command with explanation )

Reference Doc:- <https://shopify.dev/docs/apps/deployment/web#deploy>  
<https://fly.io/docs/hands-on/install-flyctl/>

## **Deploying apps and machines:**

- 1) fly help (To get fly commands list)
- 2) fly apps (To manage apps)
- 3) fly apps list (To get the list of apps hosted on server)
- 4) fly apps open "App\_name" (To open a particular App)
- 5) fly apps restart (To restart an application)
- 6) fly apps create (Create a new application)
- 7) fly apps destroy (Permanently destroys an app)
- 8) fly apps move (Move an app to another organization)
- 9) fly apps releases (List app releases)

## **Scaling and configuring:**

- 1) fly scale (Scale app resources)
- 2) fly regions (Manage regions)
- 3) fly secrets (Manage application secrets with the set and unset commands)

## **Provisioning storage:**

- 1) fly volumes (Volume management commands)
- 2) fly postgres (Manage Postgres clusters)
- 3) fly redis (Launch and manage Redis databases managed by Upstash.com)

## **Networking configuration:**

- 1) fly ips (Manage IP addresses for apps)
- 2) fly wireguard (Commands that manage WireGuard peer connections)
- 3) fly proxy (Proxies connections to a fly VM)
- 4) fly certs (Manage certificates)

### **Monitoring and managing things:**

- 1) fly logs (View app logs)
- 2) fly status (Show app status)
- 3) fly dashboard (Open web browser on Fly Web UI for this app)
- 4) fly dig (Make DNS requests against Fly.io's internal DNS server)
- 5) fly ping (Test connectivity with ICMP ping messages)
- 6) fly ssh (Use SSH to login to or run commands on VMs)
- 7) fly sftp (Get or put files from a remote VM).

#### **Access control:**

- 1) fly orgs (Commands for managing Fly organizations)
- 2) fly auth (Manage authentication)
- 3) fly move ( Move an app to another organization)

#### **More help:**

- 1) fly docs (View Fly documentation)
- 2) fly doctor (The DOCTOR command allows you to debug your Fly environment)
- 3) fly help commands (A complete list of commands (there are a bunch more))

## **List of issues we faced and how we resolve it**

### **Please Refer Following LINK:-**

- [https://docs.google.com/document/d/1qSY-UwZVpQoFoH-iOMT2nIPI\\_-2P2nezlEmzLQvQNYg/edit](https://docs.google.com/document/d/1qSY-UwZVpQoFoH-iOMT2nIPI_-2P2nezlEmzLQvQNYg/edit)
- <https://docs.google.com/document/d/1IWBxgAkXJoEO07iHRj1kdV-cQli72B7r0GSDgHP4FQ0/edit>
- <https://docs.google.com/document/d/1DwncoYDITVGjQipHcLHyyV0iQrltdQlrm58mb00Sy4/edit>

## **Detail explanations on fly.toml file's configuration**

### **Please Refer Following LINK:-**

- [https://docs.google.com/document/d/1MD2ONvCtEPvlkzsv2f7Ayf1Qd\\_hYwt7G0uNAWgS45ow/edit#](https://docs.google.com/document/d/1MD2ONvCtEPvlkzsv2f7Ayf1Qd_hYwt7G0uNAWgS45ow/edit#)