DApps Delivery Guide Documentation

Release 0.0.1

Sergii Bomko

Aug 14, 2022

Contents

The guide covers the delivery process of DApps.

CHAPTER 1

Web2 epoch

We use to use DevOps tools and Internet infrastructure for Web Apps delivery. Everybody knows a number of hosting providers and domain name providers, including CI/CD services that allow achieving our needs in Web App delivery. In the guide, we would cover how we can use Web2 infrastructure and tools for Web3 applications.

CHAPTER 2

Web3 epoch

Web3 is our nearest future which is still under construction.

2.1 Delivery tools

2.1.1 GitHub and GitHub Actions

GitHub provides hosting for software development version control using Git. GitHub is the largest host of source code in the world [1].

A decentralized application (DApp, dApp, Dapp, or dapp) is a computer application that runs on a distributed computing system. DApps have been popularized by distributed ledger technologies (DLT) such as the Ethereum Blockchain, where DApps are often referred to as smart contracts.

Many DApps are open-sourced, most of them are hosted in GitHub. That is why it is important to have proper tooling for delivering DApps.

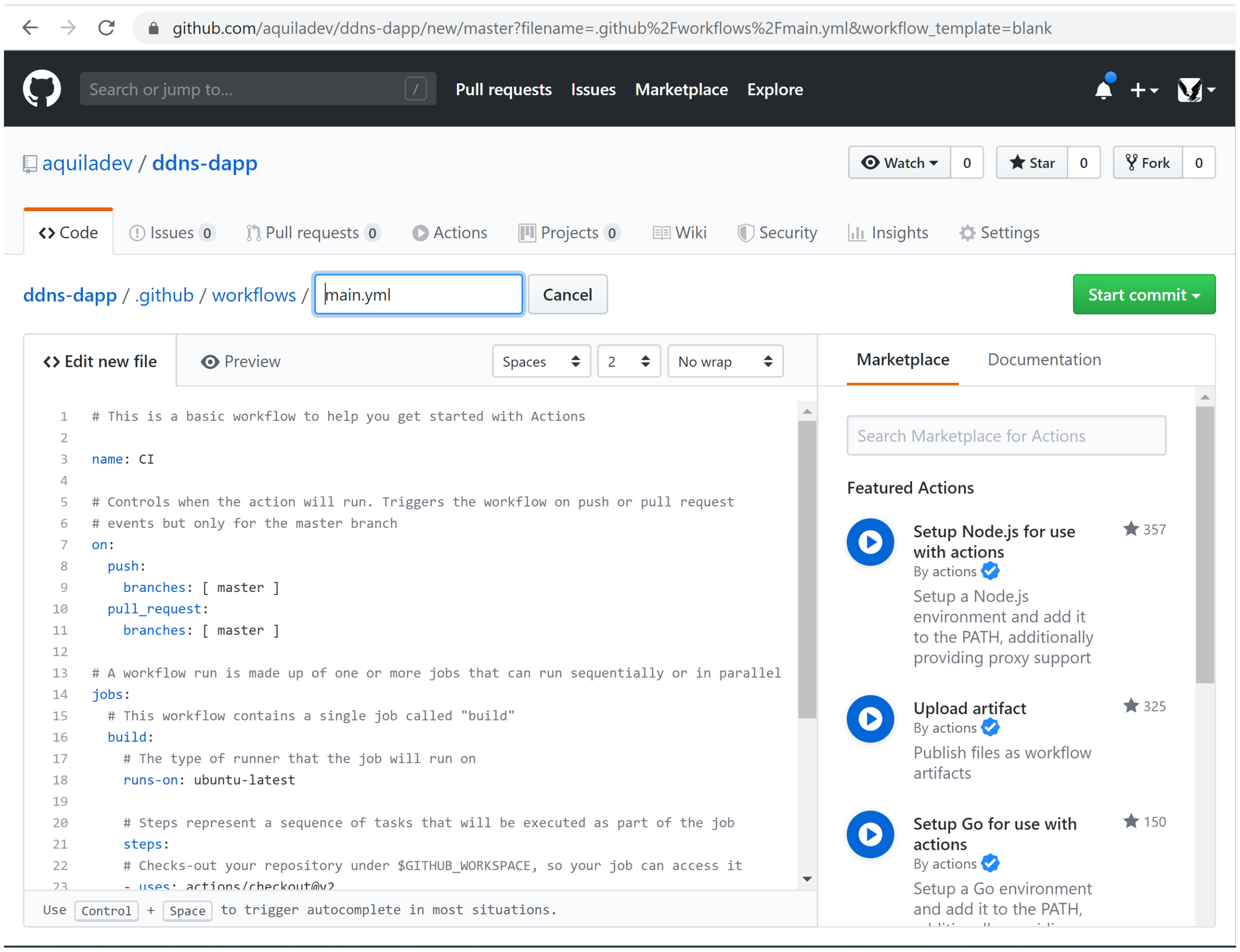
GitHub Actions enables everyone to create custom software development life cycle (SDLC) workflows directly in the GitHub repository.

Since GitHub introduced Actions, a single commit is needed to create a pipeline for a DApp. You do not need to use 3rd party services for that anymore.

Basic Pipeline for a DApp

Step 1: Create pipeline

Open GitHub repository -> Actions -> *New Workflow* -> *Set up a workflow yourself*



```
# This is a basic workflow to help you get started with Actions

name: CI

# Controls when the action will run. Triggers the workflow on push or pull request
# events but only for the master branch
on:
  push:
    branches: [ master ]
  pull_request:
    branches: [ master ]

# A workflow run is made up of one or more jobs that can run sequentially or in parallel
jobs:
  # This workflow contains a single job called "build"
  build:
    # The type of runner that the job will run on
    runs-on: ubuntu-latest

    # Steps represent a sequence of tasks that will be executed as part of the job
    steps:
    # Checks-out your repository under $GITHUB_WORKSPACE, so your job can access it
    - uses: actions/checkout@v2
```

Setup a Node.js environment and add it to the PATH, additionally providing proxy support

Publish files as workflow artifacts

Setup a Go environment and add it to the PATH,

Step 2: Modify pipeline

1. Define a trigger of the pipeline (e.g. only on *master* branch):

```
on:
  push:
    branches:
    - master
```

2. Leave the definition of the job and run environment:

```
jobs:
  build:
    runs-on: ubuntu-latest
```

3. Define prep-steps:

```
steps:
  - uses: actions/checkout@v2

  - name: Setup Node
    uses: actions/setup-node@v1
    with:
      node-version: '10.x'

  - run: npm ci

  - run: npm run build
```

Then you need to commit your *main.yml* by pressing *Start commit* button.

2.1.2 Fleek

Fleek is a service that allows launch and maintains fast, modern websites hosted on IPFS.

Take a look on documentation

2.2 Hosting

We used to deploy Web Apps in hosting services. DApps can and should be deployed in **decentralised storage**.

2.2.1 IPFS

InterPlanetary File System (IPFS) is a protocol and peer-to-peer network for storing and sharing data in a distributed file system. IPFS uses content-addressing to uniquely identify each file in a global namespace connecting all computing devices.

IPFS allows users to not only receive but host content, in a similar manner to BitTorrent. As opposed to a centrally located server, IPFS is built around a decentralized system of user-operators who hold a portion of the overall data, creating a resilient system of file storage and sharing. Any user in the network can serve a file by its content address, and other peers in the network can find and request that content from any node who has it using a distributed hash table (DHT).

Upload to IPFS

There is a GitHub Action which allows to upload a DApp to IPFS on Marketplace.

Input parameters:

Parameter	Required	Service	Description
path	Yes		Directory's path to upload.
service	No		Type of target service to upload. Supported services [*ipfs*, *pinata*, *infura*]. Default *ipfs*
timeout	No		Request timeout. Default *60000* (1 minute)
verbose	No		Level of verbosity [*false* - quiet, *true* - verbose]. Default *false*
host	No	ipfs	IPFS host. Default *ipfs.komputing.org*
port	No	ipfs	IPFS host's port. Default *443*
protocol	No	ipfs	IPFS host's protocol. Default *https*
headers	No	ipfs	IPFS headers as json object. Default *{}*
pinataKey	Yes*	pinata	Pinata Api Key. Required for pinata service.
pinataSecret	Yes*	pinata	Pinata Secret Api Key. Required for pinata service.
pinataPinName	No	pinata	Human name for pin.

In order to use it, you need to add step to *main.yml*:

```yaml
- uses: aquiladev/ipfs-action@v0.1.1
  id: upload
  with:
    path: ./build
```

There will be a build artifact on a runner after steps (usually in directory *build* or *dist*). You need to pass the directory as a *path* parameter.

The step will have *hash* output — it is needed for later use. Token *${{ steps.upload.outputs.hash }}* can be used in next steps where *upload* is the id of current step.

Upload to IPFS Pinata pinning service

The same GitHub Action allows to upload a DApp to Pinata pinning service. Pinata simplifies immutable data with simple IPFS API and toolkit.

In order to use it, you need to add step to *main.yml*:

```
- uses: aquiladev/ipfs-action@v0.1.3
  id: pinata
  with:
    path: ./build
    service: pinata
    pinataKey: ${{ secrets.PINATA_KEY }}
```

(continues on next page)

(continued from previous page)

```
    pinataSecret: ${{ secrets.PINATA_SECRET }}
    pinataPinName: {pin_name}
```

The output of the upload action is similar to the previous example.

2.2.2 Ethereum Swarm

Swarm - Decentralised data storage and distribution: Swarm is a peer to peer data sharing network in which files are addressed by the hash of their content. Similar to Bittorrent, it is possible to fetch the data from many nodes at once and as long as a single node hosts a piece of data, it will remain accessible everywhere. This approach makes it possible to distribute data without having to host any kind of server - data accessibility is location independent. Other nodes in the network can be incentivised to replicate and store the data themselves, obviating the need for hosting services when the original nodes are not connected to the network.

Upload to Swarm

There is a GitHub Action which allows to upload a DApp to Swarm on Marketplace.

In order to use it, you need to add step to *main.yml*:

```
- uses: aquiladev/swarm-action@v0.1
  id: upload
  with:
    path: ./build
```

There will be a build artifact on a runner after steps (usually in directory *build* or *dist*). You need to pass the directory as a *path* parameter.

The step will have *hash* output — it is needed for later use. Token *${{ steps.upload.outputs.hash }}* can be used in next steps where *upload* is the id of current step.

2.3 Domain names

2.3.1 ENS

Ethereum Name Service (ENS), a distributed, open, and extensible naming system based on the Ethereum blockchain. It maps human-readable names like *alice.eth* to machine-readable identifiers such as Ethereum addresses, content hashes, and metadata.

The GitHub Action allows updating *.eth* name during pipeline execution.

Registering .ETH name

This shortened guide will give you instructions of how to register a new .ETH name. For more detaild take a look at this step-by-step tutorial for registering .ETH name.

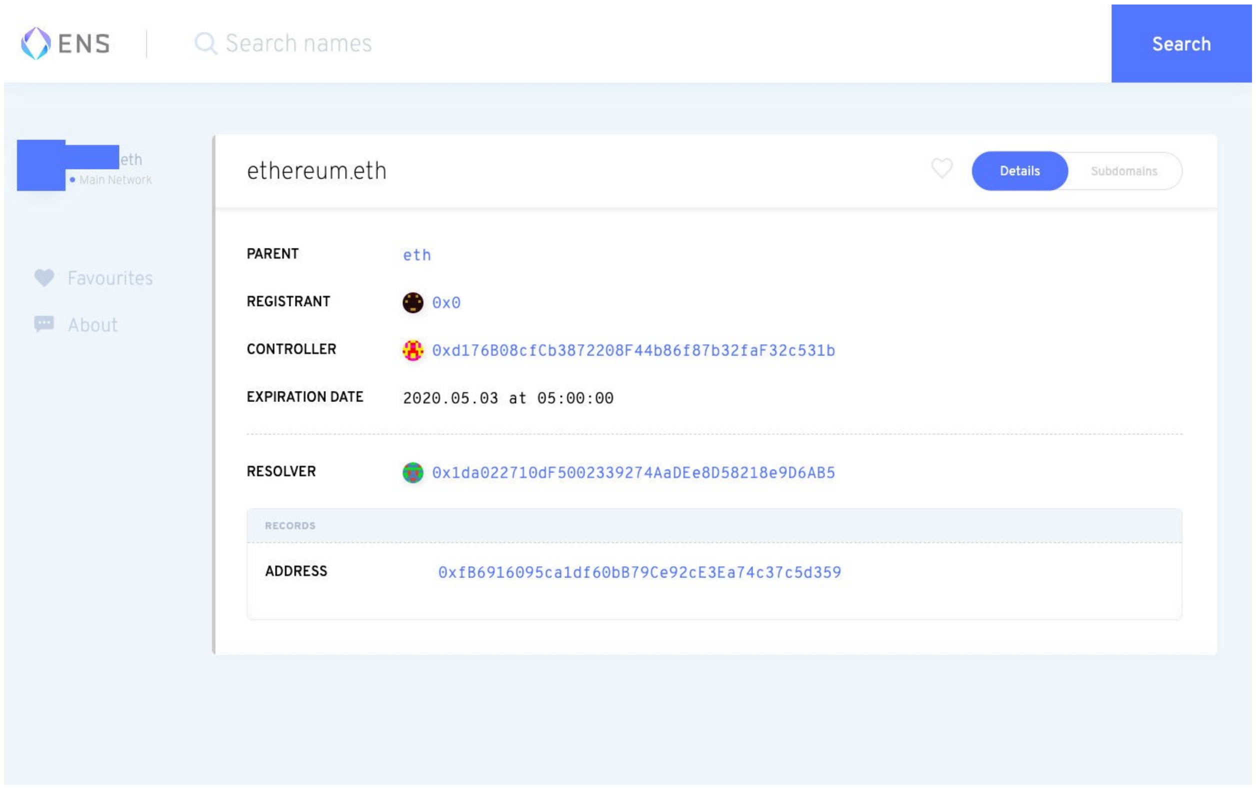
Step 1: Open your Ethereum-enabled browser

On desktop, this could be Chrome with the extension MetaMask, or the browser Brave with MetaMask (which comes built-in) enabled.

Step 2: Navigate to the ENS Manager

In your browser, go to app.ens.domains

Step 3: Search for your desired .eth name



ethereum.eth

PARENT	eth
REGISTRANT	0x0
CONTROLLER	0xd176B08cfCb3872208F44b86f87b32faF32c531b
EXPIRATION DATE	2020.05.03 at 05:00:00
RESOLVER	0x1da022710dF5002339274AaDEe8D58218e9D6AB5
ADDRESS	0xfB6916095ca1df60bB79Ce92cE3Ea74c37c5d359

Step 4: Start the registration process

Click the blue *Request To Register* button on the bottom right. A box should pop up from MetaMask asking you to confirm the transaction. Click the *Confirm* button to confirm it.

Step 5: Wait

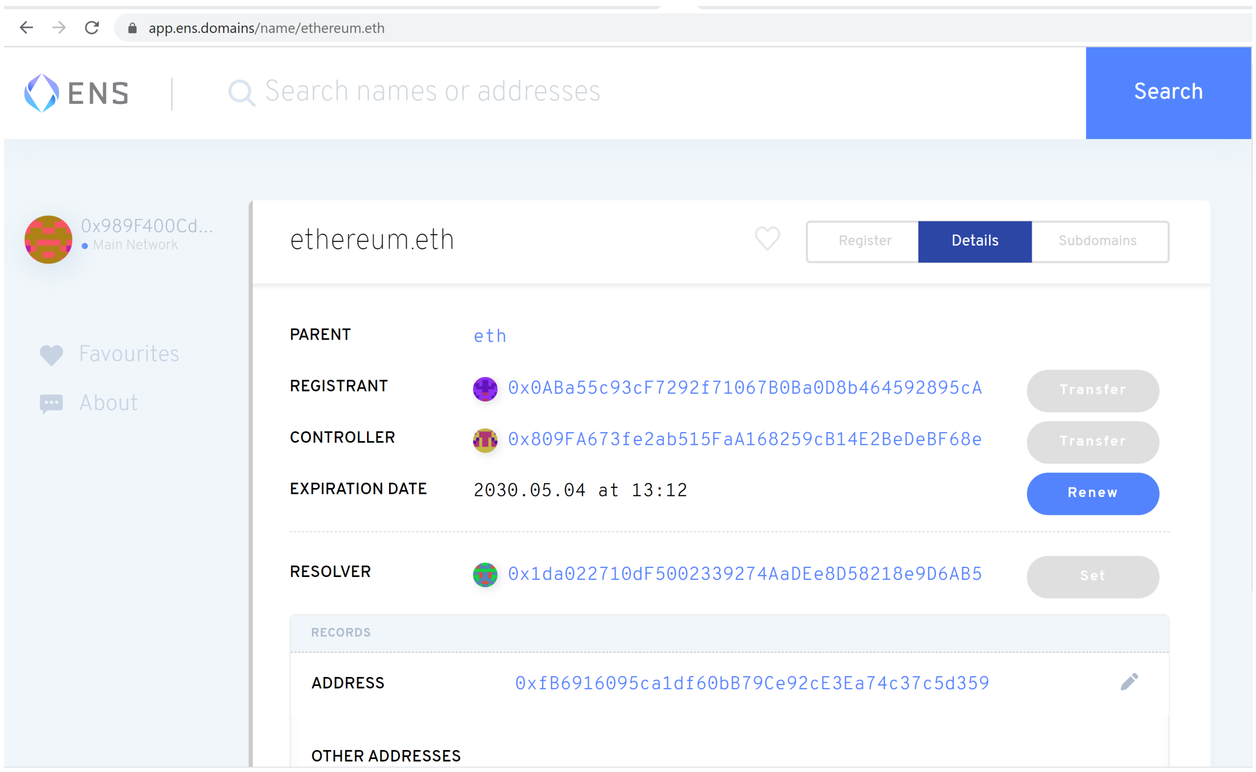
The waiting period is required to ensure another person hasn't tried to register the same name

Step 6: Finish registration

Once your wait time is over, orange text will appear that says *Click register to move to the 3rd step*. Click the blue next to it that says *Register*.

Step 7: Set Relosver

After registration is completed you need to set Resolver.



ethereum.eth

PARENT eth

REGISTRANT 0x0ABa55c93cF7292f71067B0Ba0D8b464592895cA

CONTROLLER 0x809FA673fe2ab515FaA168259cB14E2BeDeBF68e

EXPIRATION DATE 2030.05.04 at 13:12

RESOLVER 0x1da022710dF5002339274AaDEe8D58218e9D6AB5

RECORDS

ADDRESS 0xfB6916095ca1df60bB79Ce92cE3Ea74c37c5d359

OTHER ADDRESSES

Setup pipeline with .ETH update

Requirements

1. Before setting up a pipeline the ENS name should be configured, it should have a resolver. Take a look on prev section.
2. Basic pipeline should be configured with step which provides IPFS hash

Pipeline step

Open and add the step to *main.yml*:

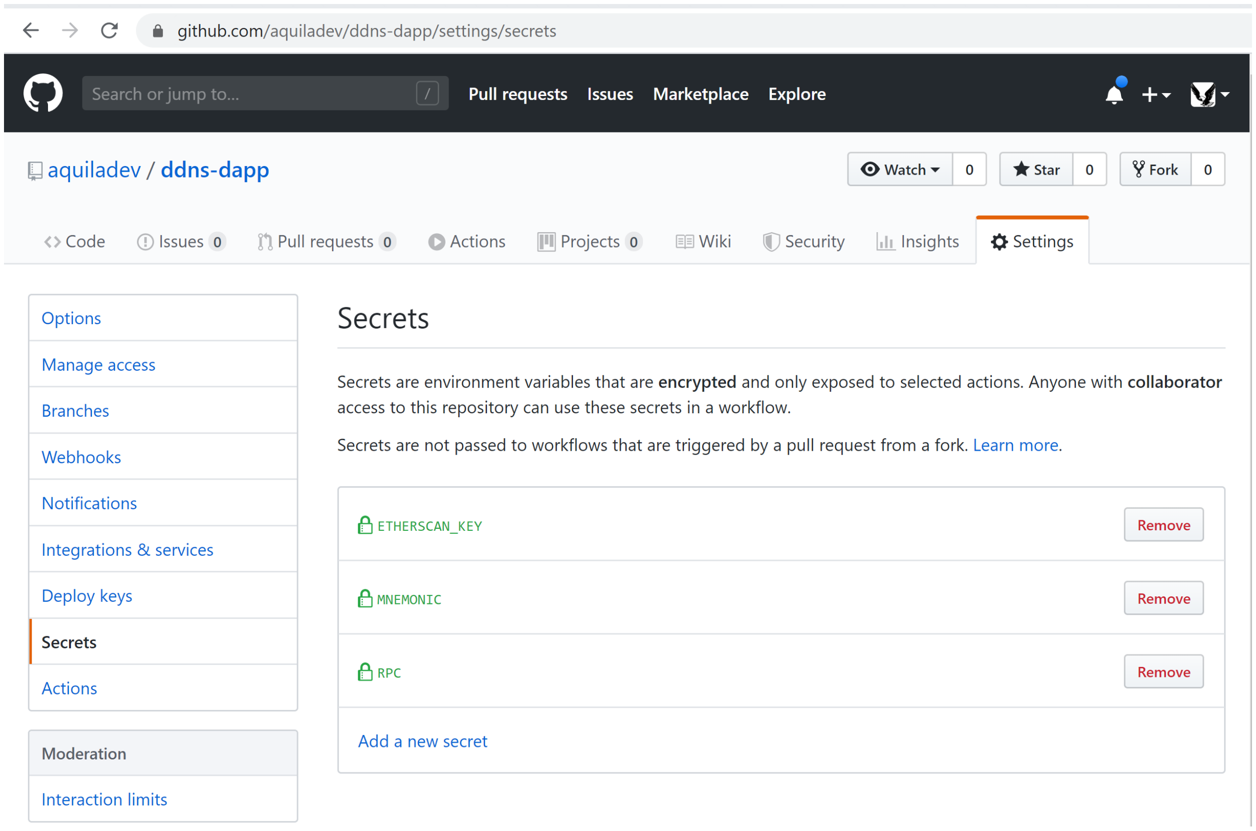
```
- uses: aquiladev/ddns-action@v0.1.1
  with:
    mnemonic: ${{ secrets.MNEMONIC }}
    rpc: ${{ secrets.RPC }}
    name: ddns-action.eth
    contentHash: ${{ steps.upload.outputs.hash }}
```

Parameters

- *${{ secrets.MNEMONIC }}* is a secret. The mnemonic phrase is needed for wallet recovery of an account which owns ENS name. It can be a private key of the account as well
- *${{ secrets.RPC }}* is a secret. RPC is a URL of Ethereum Mainnet node
- *ddns-action.eth* - ENS name which you want to update
- *${{ steps.upload.outputs.hash }}* is content hash. It came from upload to IPFS step

Secrets

In order to manage sercerts in a repository you need to open Settings -> Secrets



Secrets

Secrets are environment variables that are **encrypted** and only exposed to selected actions. Anyone with **collaborator** access to this repository can use these secrets in a workflow.

Secrets are not passed to workflows that are triggered by a pull request from a fork. Learn more.

ETHERSCAN_KEY Remove

MNEMONIC Remove

RPC Remove

Add a new secret

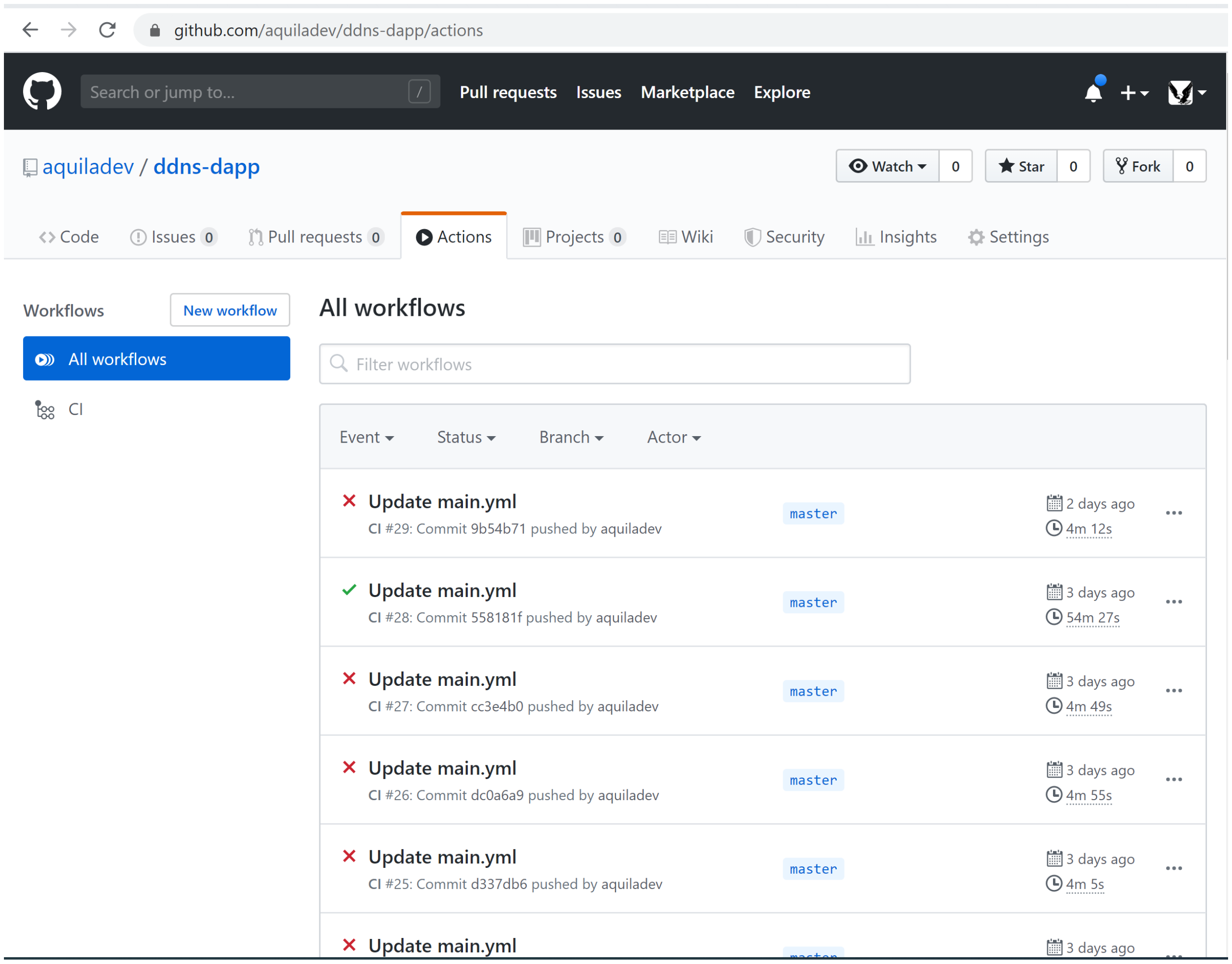
Pipeline

Eventually pipeline should look like:

```
name: CI
on:
  push:
    branches:
    - master

jobs:
  build:
    runs-on: ubuntu-latest

steps:
  - uses: actions/checkout@v2

  - name: Setup Node
    uses: actions/setup-node@v1
    with:
      node-version: '10.x'

  - run: npm ci

  - run: npm run build

  - name: Upload to IPFS
    uses: aquiladev/ipfs-action@v0.1.2-alpha
    id: upload
    with:
      path: ./build

  - name: Update ENS
    uses: aquiladev/ddns-action@v0.1.1
    with:
      mnemonic: ${{ secrets.MNEMONIC }}
      rpc: ${{ secrets.RPC }}
      name: ddns-action.eth
      contentHash: ${{ steps.upload.outputs.hash }}
```

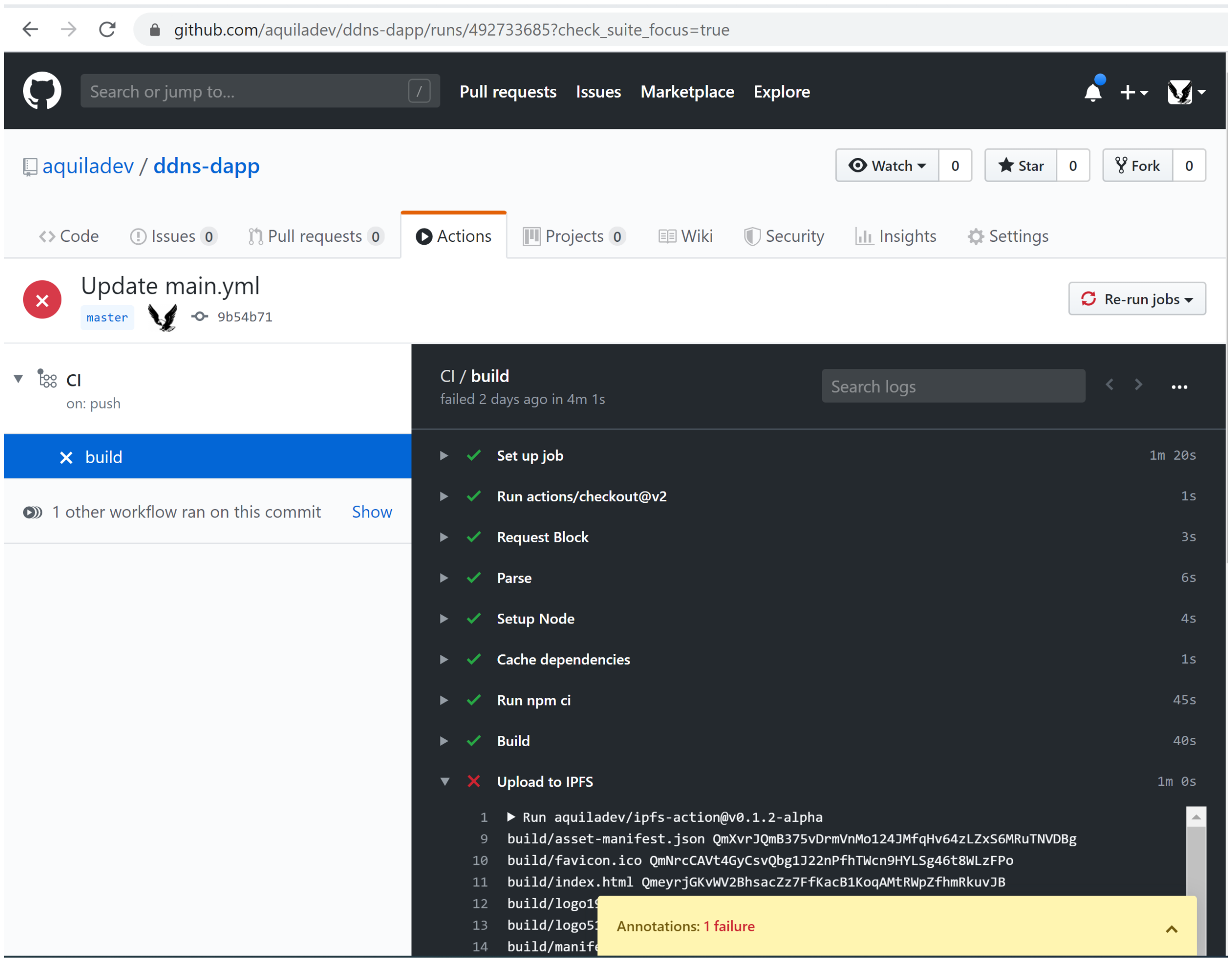
Run pipeline

The pipeline will run immediately after commit (if you committed to master branch)



✗ Update main.yml CI #29: Commit 9b54b71 pushed by aquiladev	master	2 days ago 4m 12s	...
✓ Update main.yml CI #28: Commit 558181f pushed by aquiladev	master	3 days ago 54m 27s	...
✗ Update main.yml CI #27: Commit cc3e4b0 pushed by aquiladev	master	3 days ago 4m 49s	...
✗ Update main.yml CI #26: Commit dc0a6a9 pushed by aquiladev	master	3 days ago 4m 55s	...
✗ Update main.yml CI #25: Commit d337db6 pushed by aquiladev	master	3 days ago 4m 5s	...
✗ Update main.yml	master	3 days ago	...

You can open all pipeline runs and check outputs



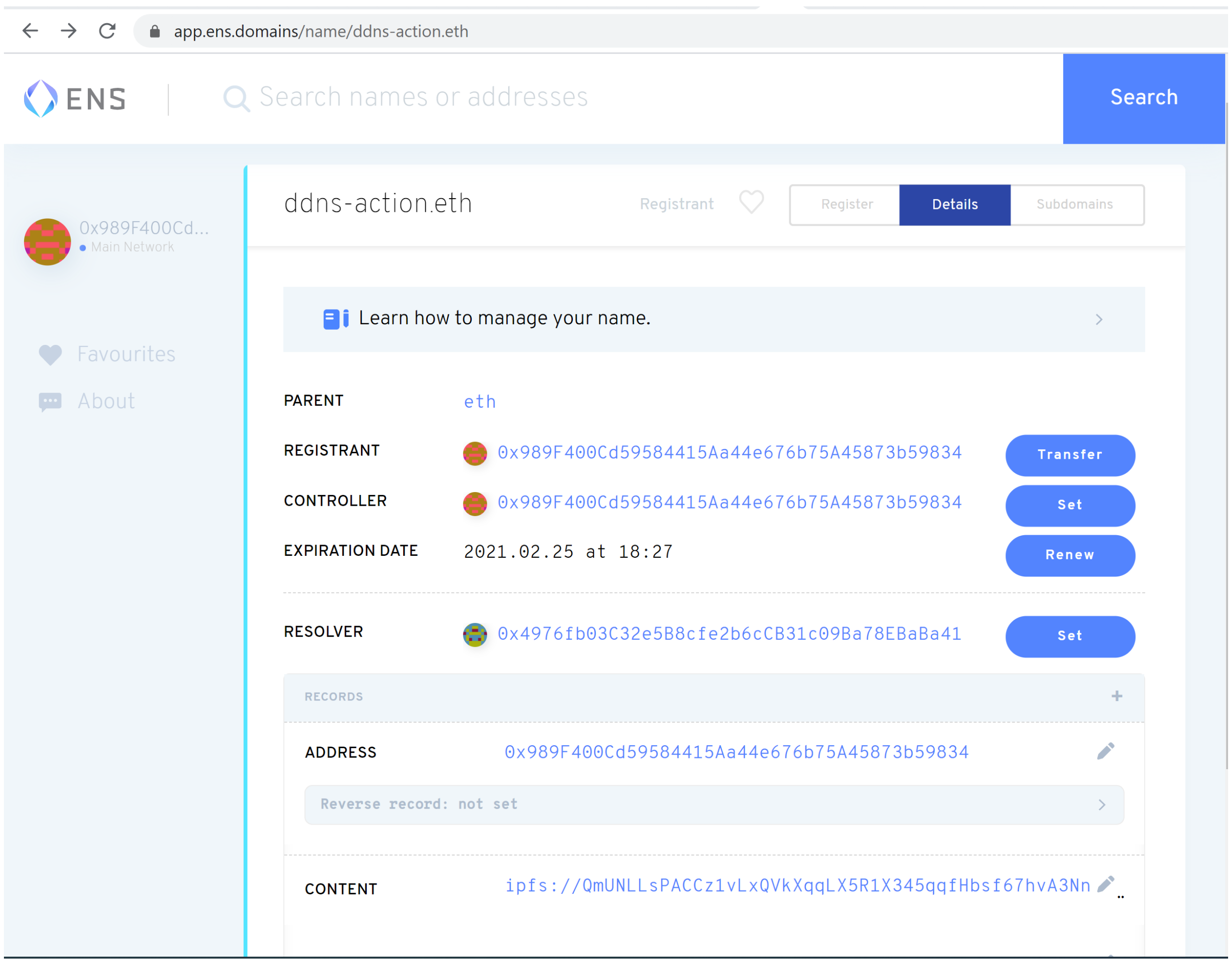
```
Set up job                    1m 20s
Run actions/checkout@v2       1s
Request Block                 3s
Parse                         6s
Setup Node                    4s
Cache dependencies            1s
Run npm ci                    45s
Build                         40s
Upload to IPFS                1m 0s
▶ Run aquiladev/ipfs-action@v0.1.2-alpha
build/asset-manifest.json QmXvrJQmB375vDrmVnMo124JMfqHv64zLZxS6MRuTNVDBg
build/favicon.ico QmNrcCAVt4GyCsvQbg1J22nPfhTWcn9HYLSg46t8WLzFPo
build/index.html QmeyrjGKvWV2BhsacZz7FfKacB1KoqAMtRWpZfhmRkuvJB
build/logo19
build/logo51
build/manife
```

How to access a DApp with .ETH name?

After a successful run of the pipeline, you should be able to access your updated DApp, but it takes some time on IPFS side to resolve newly uploaded content.

Check ENS record

In order to check updated IPFS hash in ENS, you need to open https://app.ens.domains



ddns-action.eth

Learn how to manage your name.

PARENT eth

REGISTRANT 0x989F400Cd59584415Aa44e676b75A45873b59834

CONTROLLER 0x989F400Cd59584415Aa44e676b75A45873b59834

EXPIRATION DATE 2021.02.25 at 18:27

RESOLVER 0x4976fb03C32e5B8cfe2b6cCB31c09Ba78EBaBa41

RECORDS

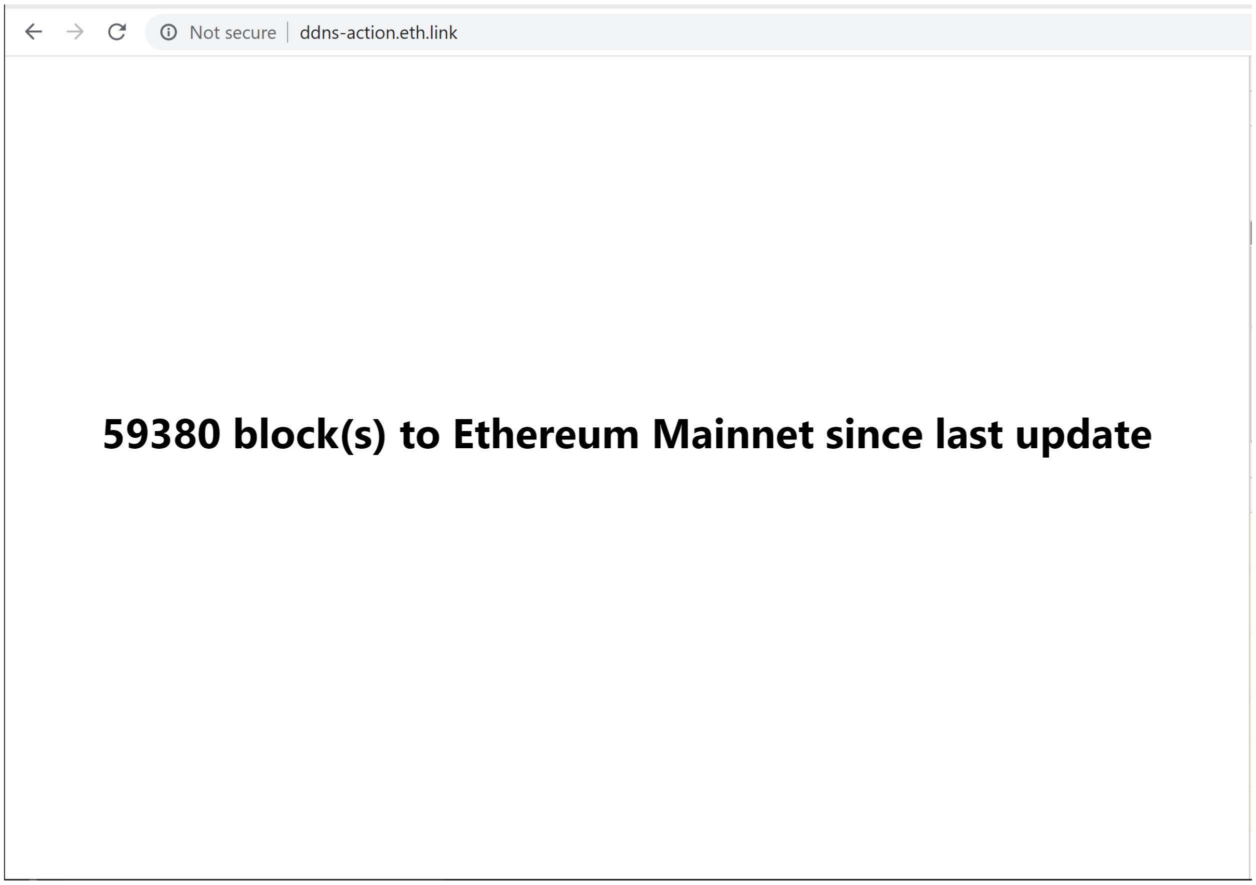
ADDRESS 0x989F400Cd59584415Aa44e676b75A45873b59834

Reverse record: not set

CONTENT ipfs://QmUNLLsPACCz1vLxQVkXqqLX5R1X345qqfHbsf67hvA3Nn

Access DApp

The easiest way to open your DApp through http://eth.link



Not secure | ddns-action.eth.link

59380 block(s) to Ethereum Mainnet since last update

2.3.2 CNS

Unstoppable Domains launching domain names on blockchain technology. The domain names can be used for both payments and websites. A domain name can't be taken down by the players that typically participating in the domain name system. Hence, the name Unstoppable Domains. Another value is that It can replace your cryptocurrency addresses with one single human-readable name.

Unstoppable Domains provides *.crypto* (aka CNS) domain names for the entire crypto community.

The GitHub Action allows updating *.crypto* name during pipeline execution.

Registering a .CRYPTO Name

This shortened guide will give you instructions of how to register a new .CRYPTO name. For more detaild take a look at this step-by-step tutorial for registering .CRYPTO name.

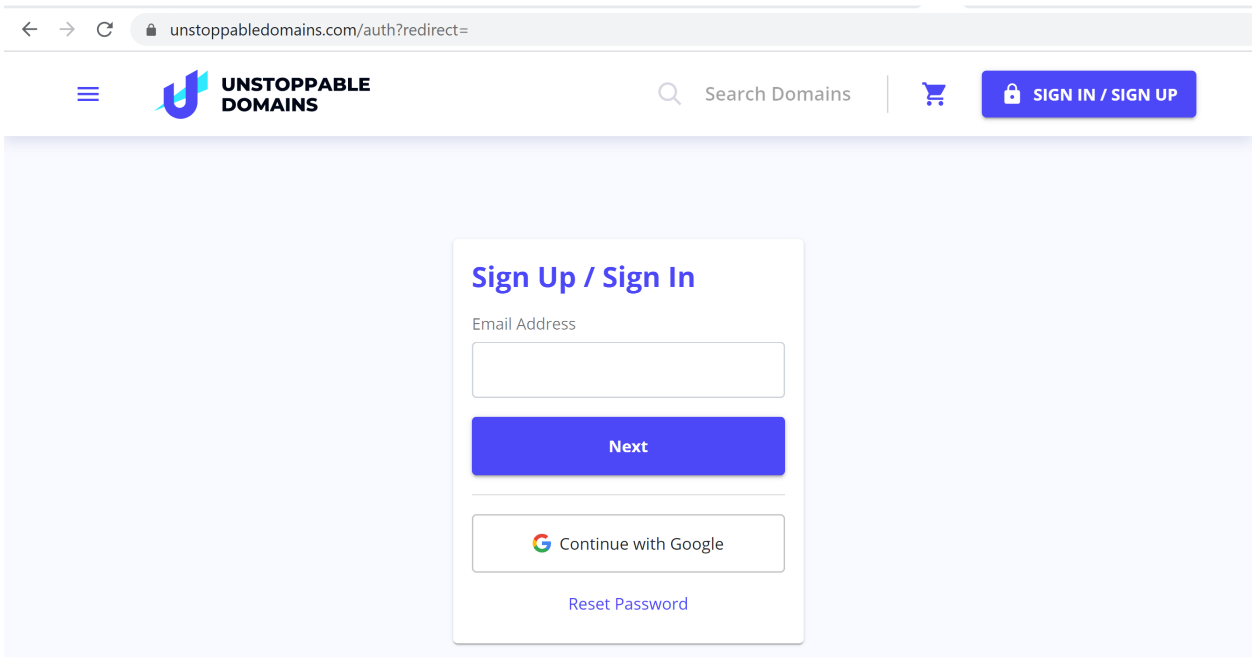
Step 1: Open your Ethereum-enabled browser

On desktop, this could be Chrome with the extension MetaMask, or the browser Brave with MetaMask (which comes built-in) enabled.

Step 2: Navigate to the Unstoppable Domains web site

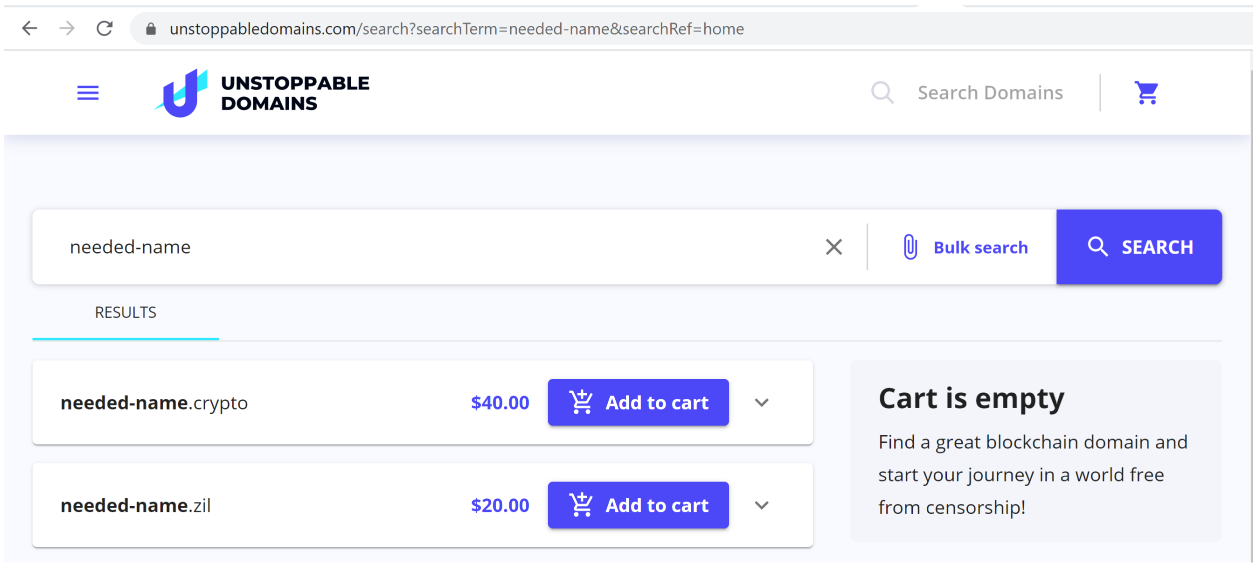
In your browser, go to unstoppabledomains.com

Step 3: Pass sign up process



Step 4: Search for your desired .crypto name

Login and search for *.crypto* name



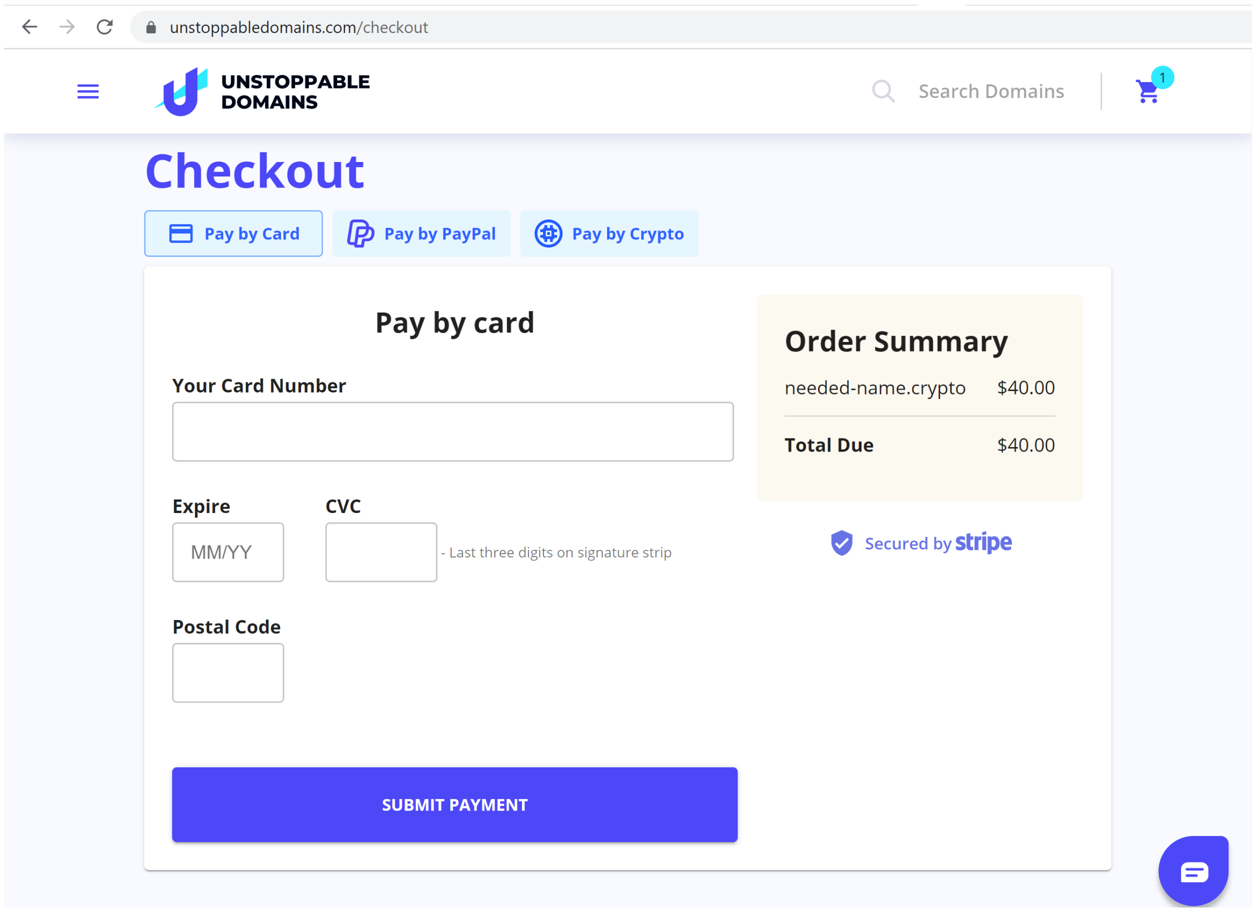
needed-name.crypto	$40.00	Add to cart
needed-name.zil	$20.00	Add to cart

Cart is empty

Find a great blockchain domain and start your journey in a world free from censorship!

Step 5: Checkout

Add *.crypto* name to a cart and go to Checkout page. Choose a suitable payment type and complete payment

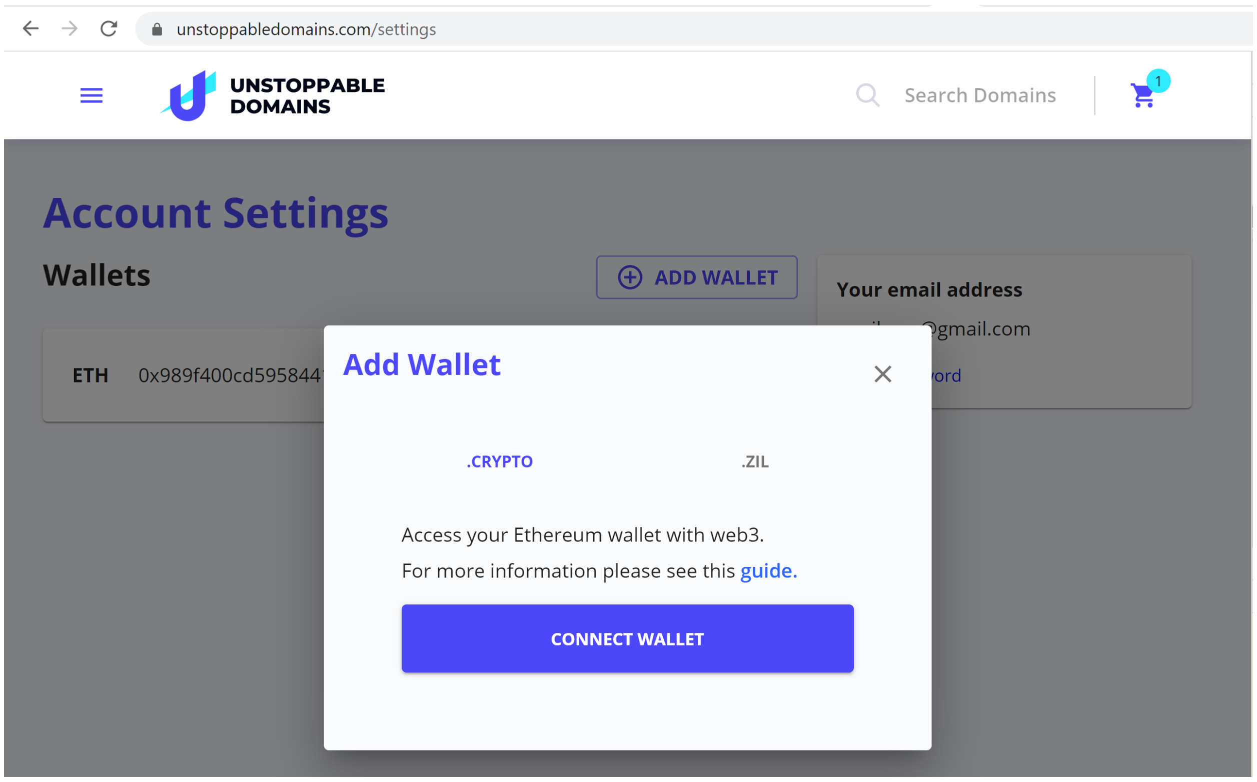


Order Summary

needed-name.crypto	$40.00
Total Due	$40.00

Step 6: Connect Wallet to your account

Go to Account Settings page and click *Add wallet* button. You need to connect your MetaMask account to the web site in an appeared popup.

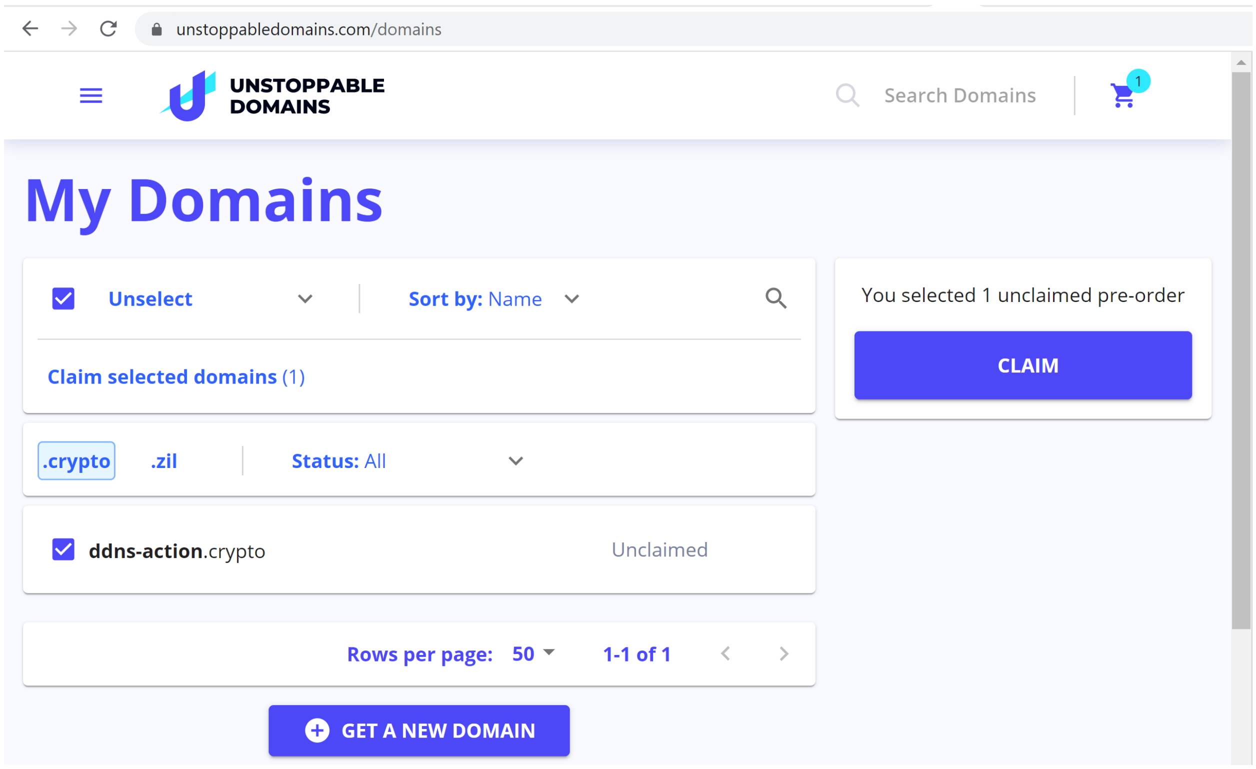


Access your Ethereum wallet with web3.

For more information please see this guide.

Step 7: Claim your domain name

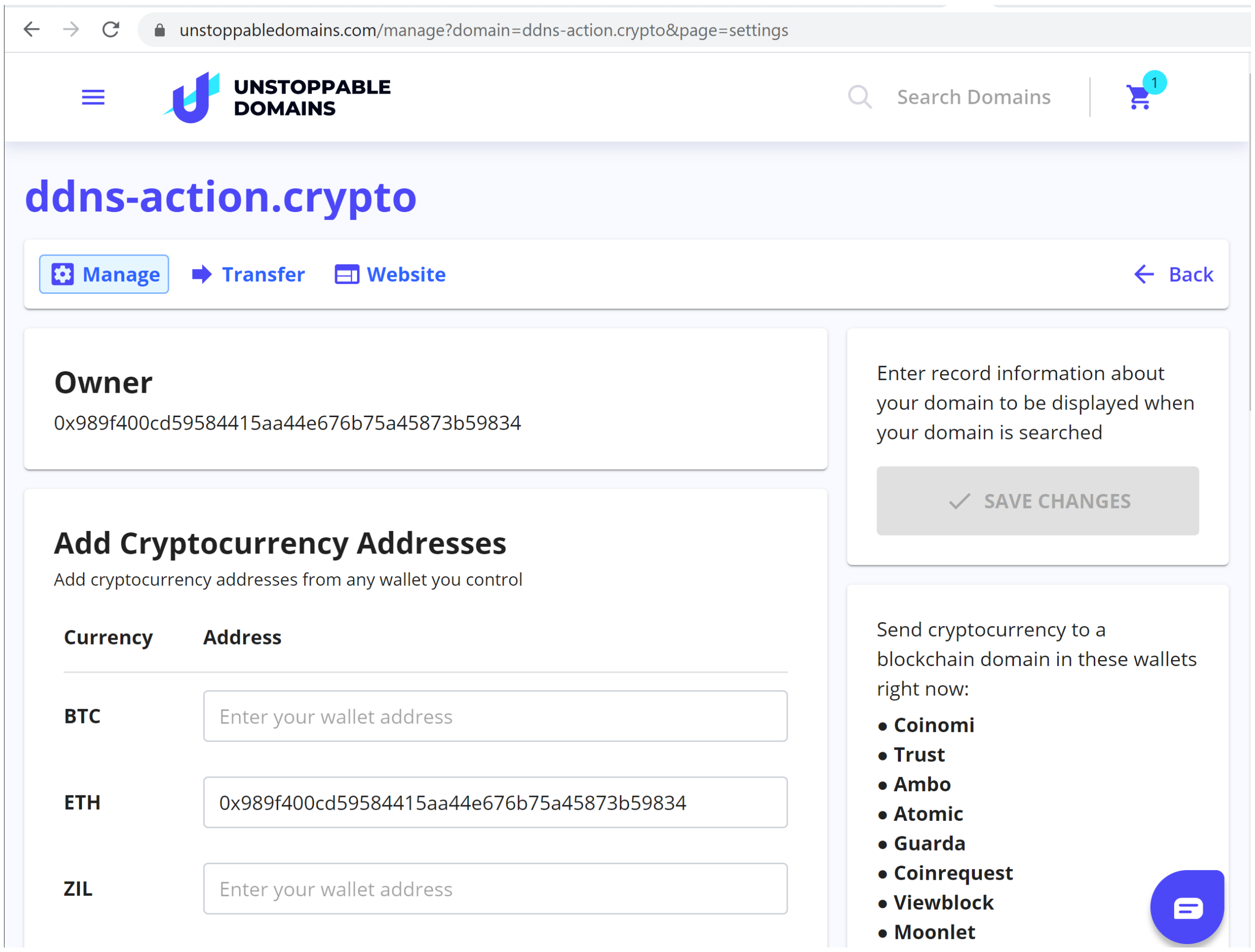
Go to My Domains page, select domain name and click *Claim*



The process takes time, there is manual approval. You will need to wait for a while.

Step 8: Manage domain name

After all you will be able to manage your domains. Go to My Domains page and click *Manage* button for the domain name.


unstoppabledomains.com/manage?domain=ddns-action.crypto&page=settings

UNSTOPPABLE DOMAINS

Search Domains

ddns-action.crypto

Manage Transfer Website Back

Owner

0x989f400cd59584415aa44e676b75a45873b59834

Add Cryptocurrency Addresses

Add cryptocurrency addresses from any wallet you control

Currency	Address
BTC	Enter your wallet address
ETH	0x989f400cd59584415aa44e676b75a45873b59834
ZIL	Enter your wallet address

Enter record information about your domain to be displayed when your domain is searched

SAVE CHANGES

Send cryptocurrency to a blockchain domain in these wallets right now:

- Coinomi
- Trust
- Ambo
- Atomic
- Guarda
- Coinrequest
- Viewblock
- Moonlet

Step 9: Set Resolver

In order to do this, you need to add at least one cryptocurrency address (e.g. owner of the name).

Setup pipeline with .CRYPTO update

Requirements

1. Before setting up a pipeline the CNS name should be configured, it should have a resolver. Take a look on prev section
2. Basic pipeline should be configured with step which provides IPFS hash

Pipeline step

Open and add the step to *main.yml*:

```
- uses: aquiladev/ddns-action@v0.1.1
  with:
    mnemonic: ${{ secrets.MNEMONIC }}
```

(continues on next page)

(continued from previous page)

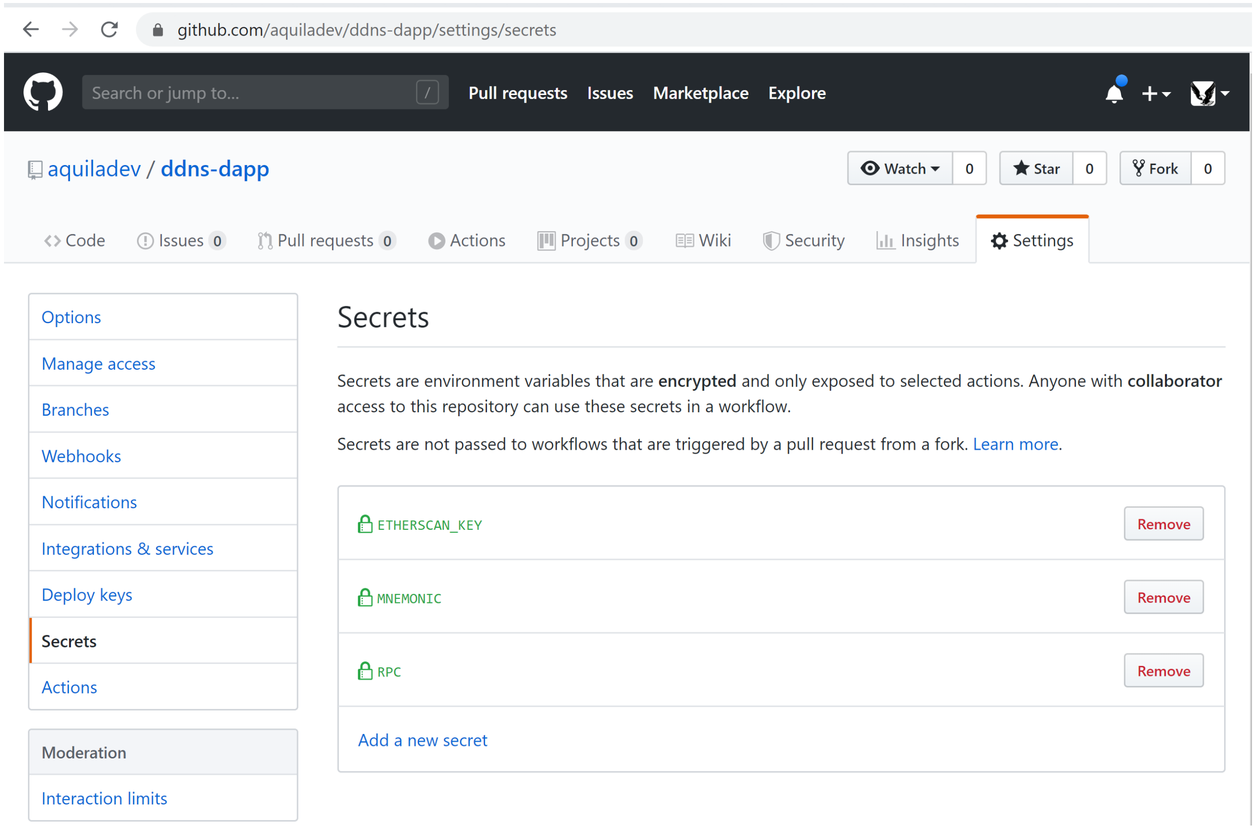
```
    rpc: ${{ secrets.RPC }}
    name: ddns-action.crypto
    contentHash: ${{ steps.upload.outputs.hash }}
```

Parameters

- *${{ secrets.MNEMONIC }}* is a secret. The mnemonic phrase is needed for wallet recovery of an account which owns CNS name. It can be a private key of the account as well
- *${{ secrets.RPC }}* is a secret. RPC is a URL of Ethereum Mainnet node
- *ddns-action.crypto* - CNS name which you want to update
- *${{ steps.upload.outputs.hash }}* is content hash. It came from upload to IPFS step

Secrets

In order to manage sercerts in a repository you need to open Settings -> Secrets



Secrets

Secrets are environment variables that are **encrypted** and only exposed to selected actions. Anyone with **collaborator** access to this repository can use these secrets in a workflow.

Secrets are not passed to workflows that are triggered by a pull request from a fork. Learn more.

ETHERSCAN_KEY

MNEMONIC

RPC

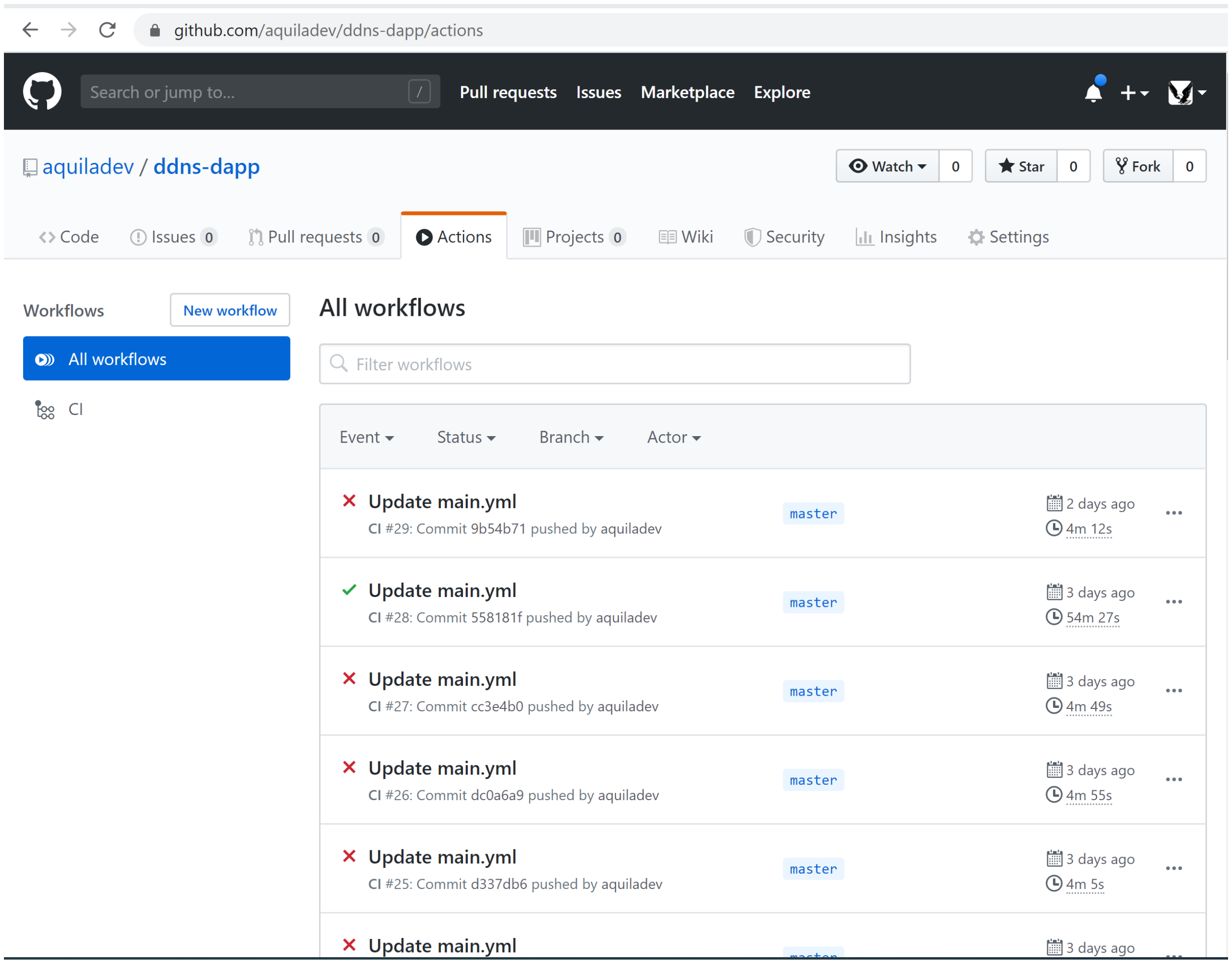
Pipeline

Eventually pipeline should look like:

```
name: CI
on:
  push:
    branches:
    - master

jobs:
  build:
    runs-on: ubuntu-latest

steps:
  - uses: actions/checkout@v2

  - name: Setup Node
    uses: actions/setup-node@v1
    with:
      node-version: '10.x'

  - run: npm ci

  - run: npm run build

  - name: Upload to IPFS
    uses: aquiladev/ipfs-action@v0.1.2-alpha
    id: upload
    with:
      path: ./build

  - name: Update CNS
    uses: aquiladev/ddns-action@v0.1.1
    with:
      mnemonic: ${{ secrets.MNEMONIC }}
      rpc: ${{ secrets.RPC }}
      name: ddns-action.crypto
      contentHash: ${{ steps.upload.outputs.hash }}
```

Run pipeline

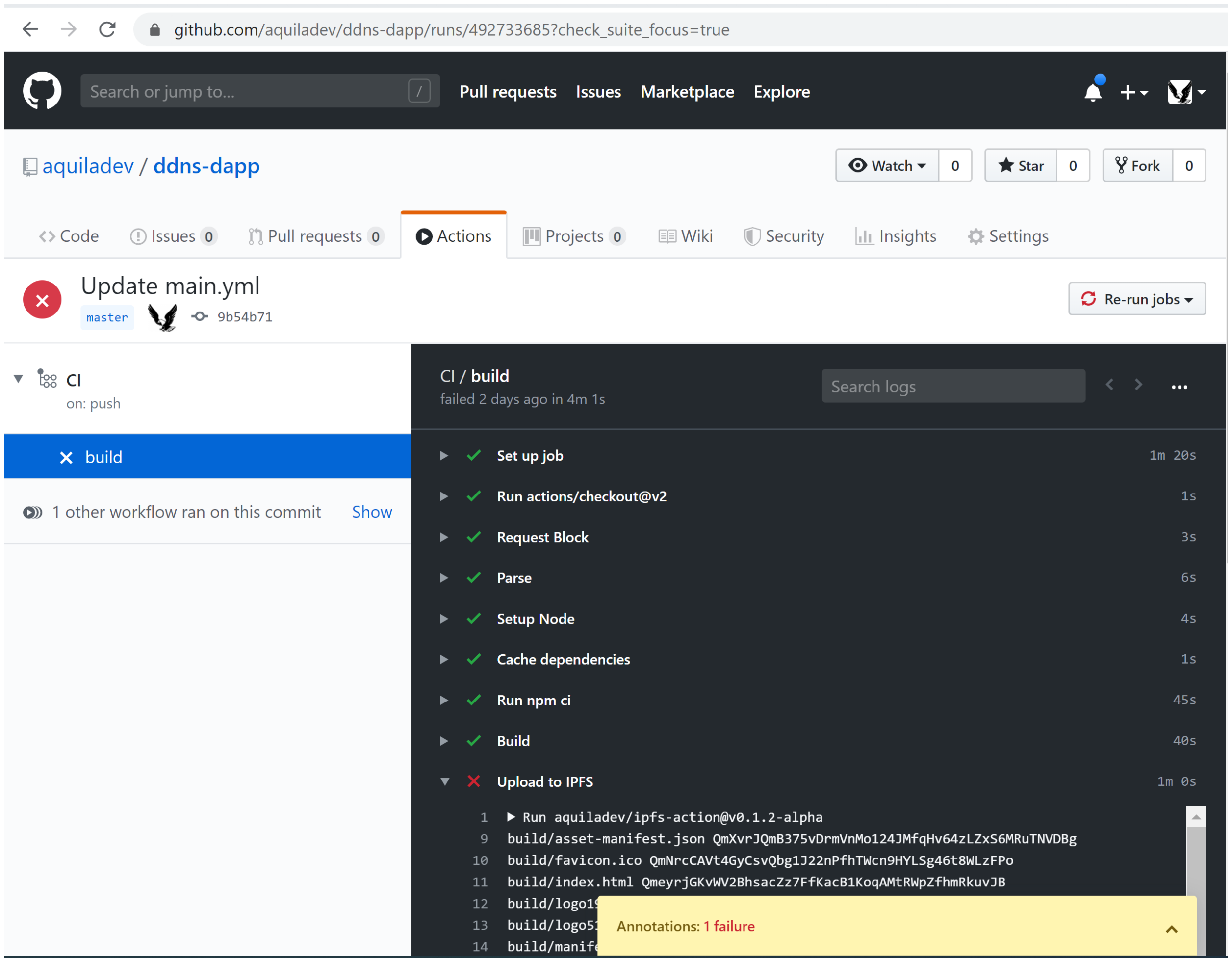
The pipeline will run immediately after commit (if you committed to master branch)



All workflows

✗ Update main.yml CI #29: Commit 9b54b71 pushed by aquiladev	master	2 days ago 4m 12s
✓ Update main.yml CI #28: Commit 558181f pushed by aquiladev	master	3 days ago 54m 27s
✗ Update main.yml CI #27: Commit cc3e4b0 pushed by aquiladev	master	3 days ago 4m 49s
✗ Update main.yml CI #26: Commit dc0a6a9 pushed by aquiladev	master	3 days ago 4m 55s
✗ Update main.yml CI #25: Commit d337db6 pushed by aquiladev	master	3 days ago 4m 5s
✗ Update main.yml		3 days ago

You can open all pipeline runs and check outputs

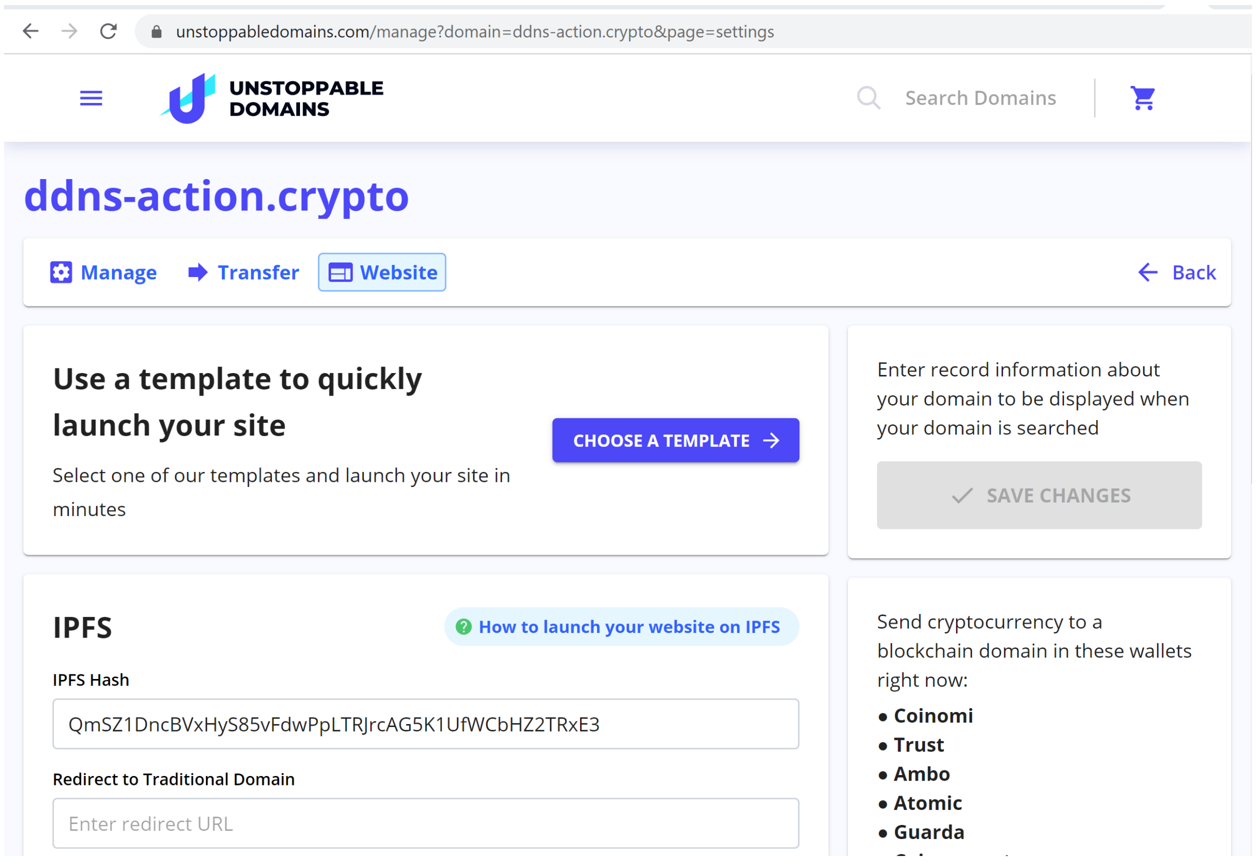


```
▶ Run aquiladev/ipfs-action@v0.1.2-alpha
build/asset-manifest.json QmXvrJQmB375vDrmVnMo124JMfqHv64zLZxS6MRuTNVDBg
build/favicon.ico QmNrcCAVt4GyCsvQbg1J22nPfhTWcn9HYLSg46t8WLzFPo
build/index.html QmeyrjGKvWV2BhsacZz7FfKacB1KoqAMtRWpZfhmRkuvJB
build/logo1
build/logo5
build/manif
```

How to access a DApp with .CRYPTO name?

Check CNS record

In order to check updated IPFS hash in CNS, you need to open My Domains page, click *Manage* button for the domain name. Then you need to open *Website* tab.



Use a template to quickly launch your site

Select one of our templates and launch your site in minutes

Enter record information about your domain to be displayed when your domain is searched

IPFS

IPFS Hash: QmSZ1DncBVxHyS85vFdwPpLTRJrcAG5K1UfWCbHZ2TRxE3

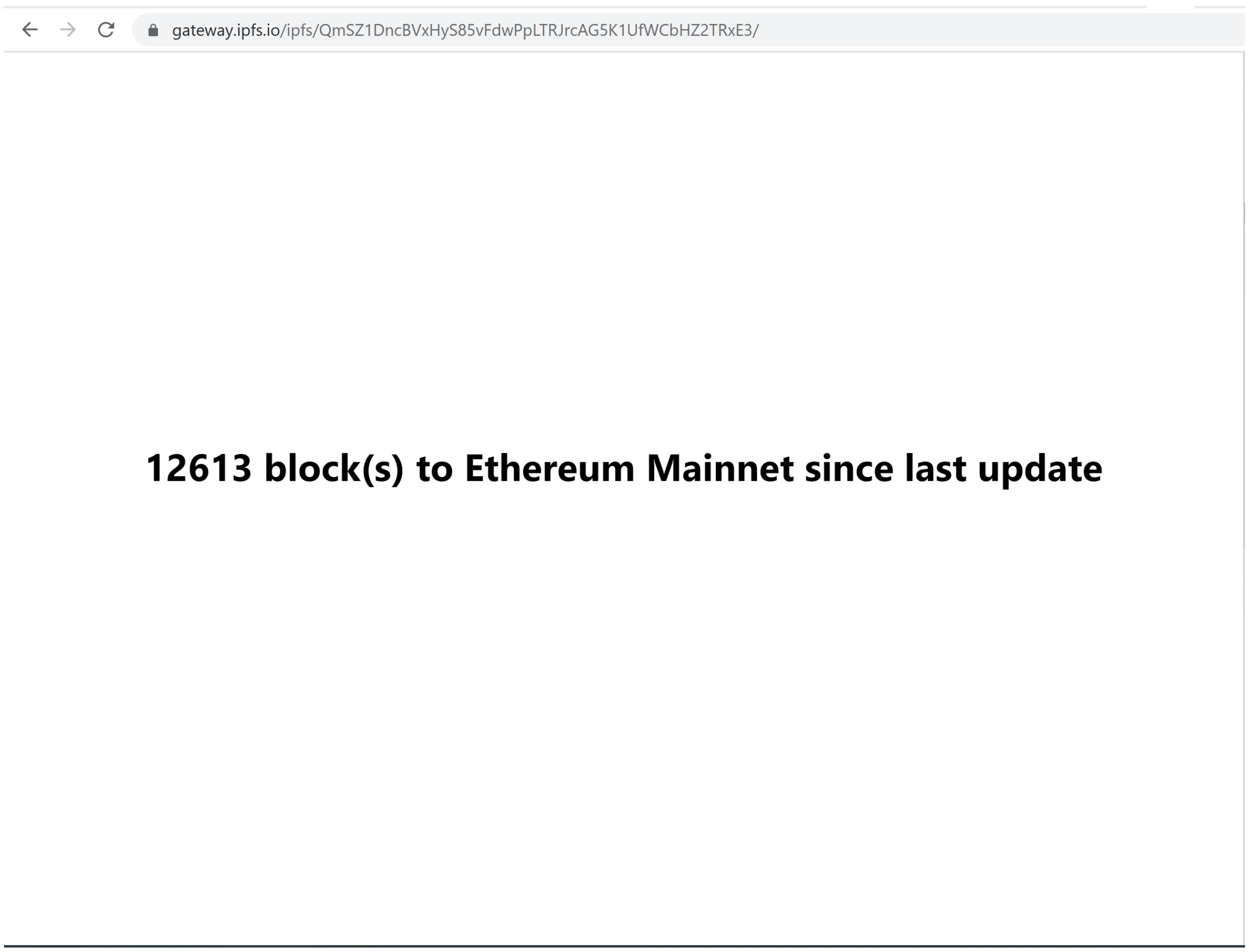
Redirect to Traditional Domain: Enter redirect URL

Send cryptocurrency to a blockchain domain in these wallets right now:

- Coinomi
- Trust
- Ambo
- Atomic
- Guarda

Access DApp

1. You should install Chrome Extension. The Extension allows Chrome browser to handle *.crypto* domain names.
2. Type *.crypto* (e.g. ddns-action.crypto) domain name in Chrome browser and you will be redirected to the DApp


gateway.ipfs.io/ipfs/QmSZ1DncBVxHyS85vFdwPpLTRJrcAG5K1UfWCbHZ2TRxE3/

12613 block(s) to Ethereum Mainnet since last update

2.4 References

1. ENS update automation
2. CNS update automation
3. DApp's pipeline (IPFS + ENS)

Contents:

Keyword Index, Search Page