From our self-build authentication to Ory

David ALEXANDRE
Co-founder and CTO at Wildcard
david.alexandre@w6d.io

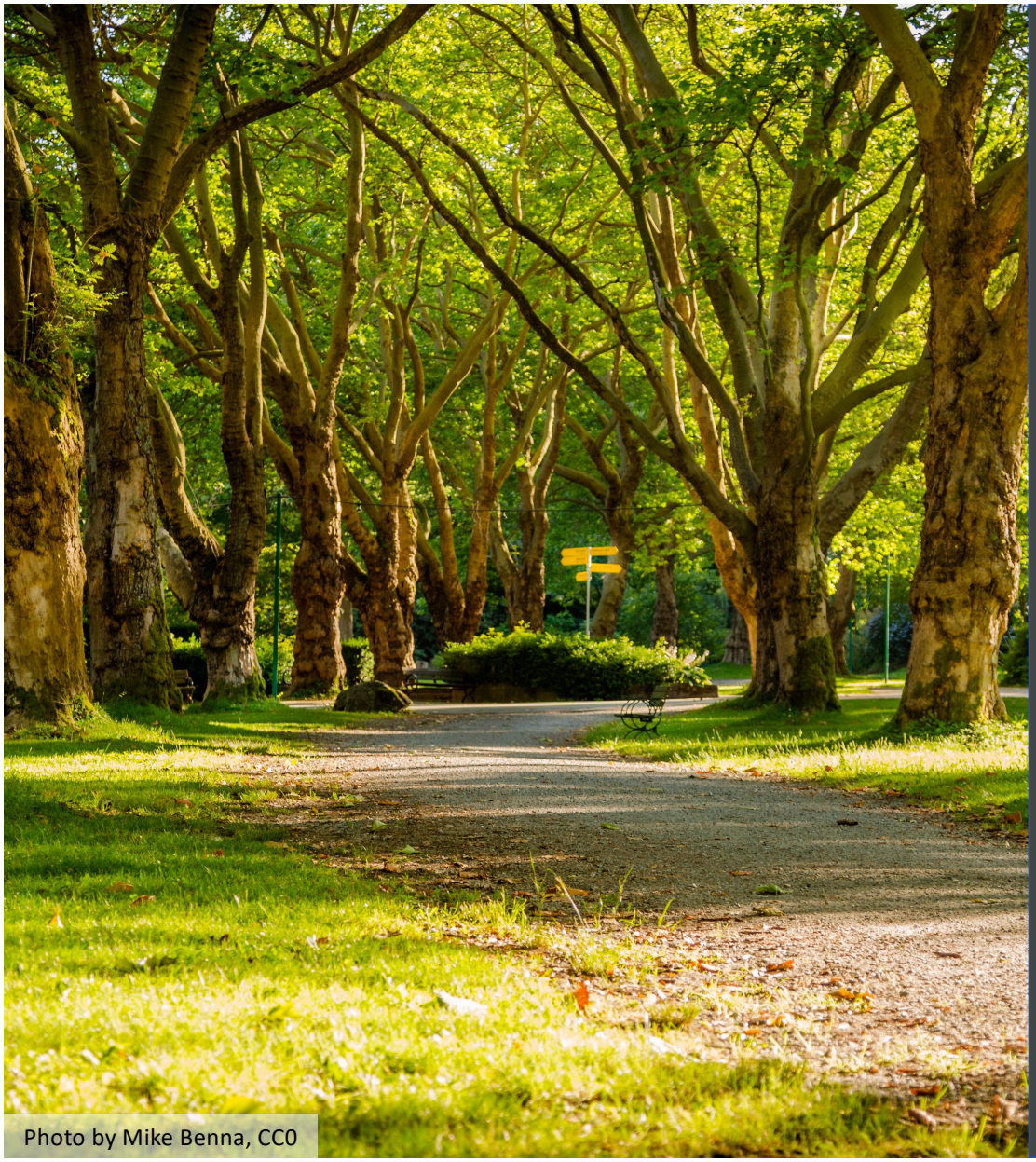


Photo by Mike Benna, CC0

Our use case

About us

Why Ory

The future user experience

How we do it

Q&A


CI/CD V2.0

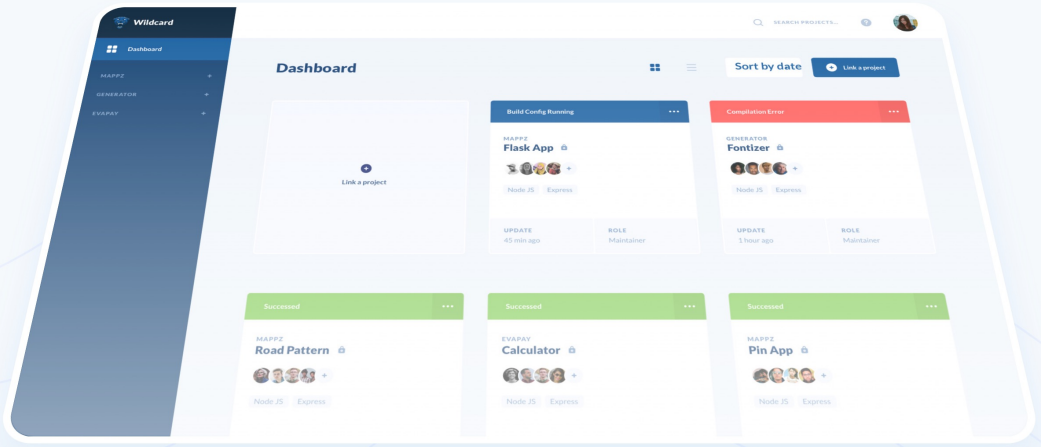
Build, test & deploy
app with NoCode

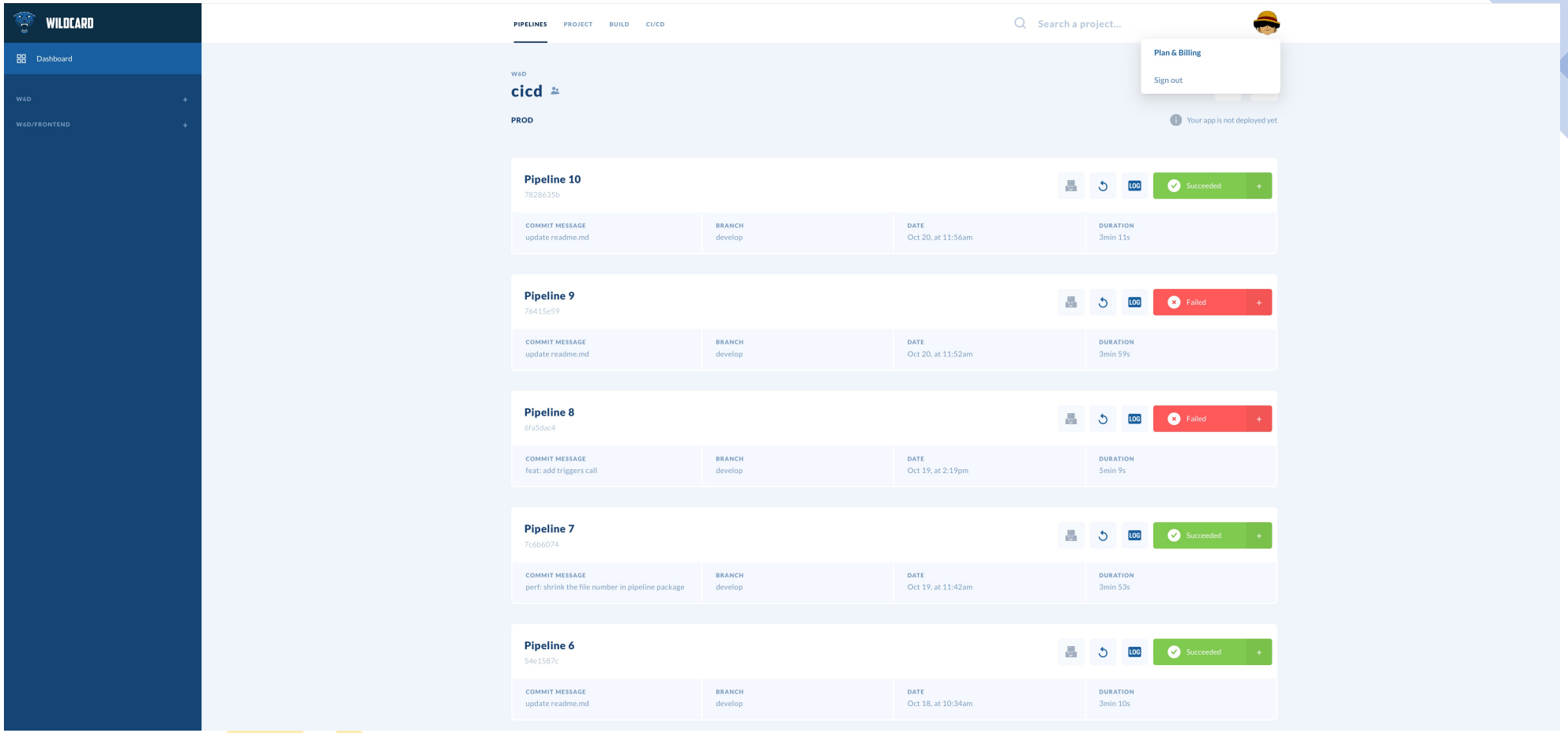
The hassle-free NoCode Platform
to deliver your cloud native apps

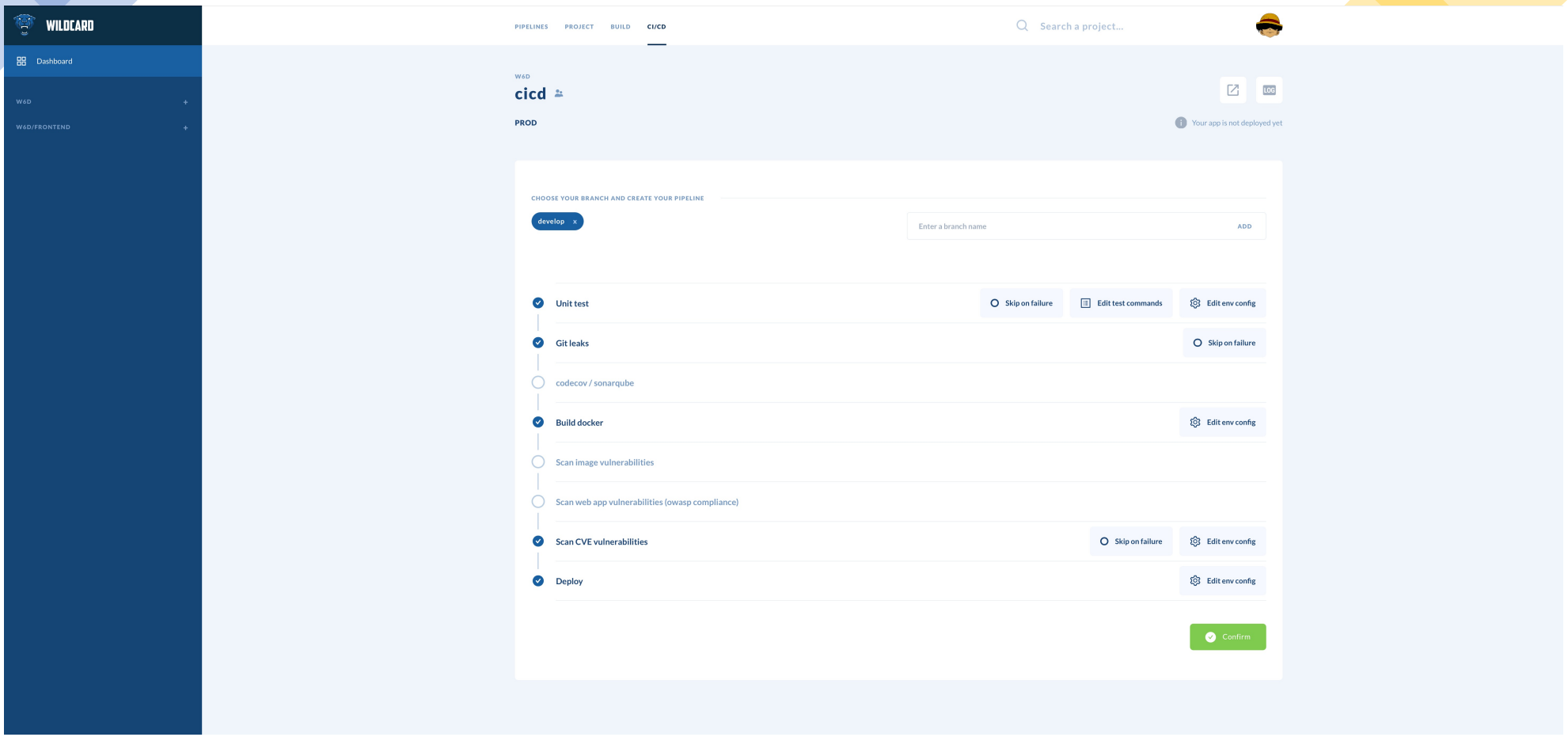
TRY FOR FREE NOW

Github

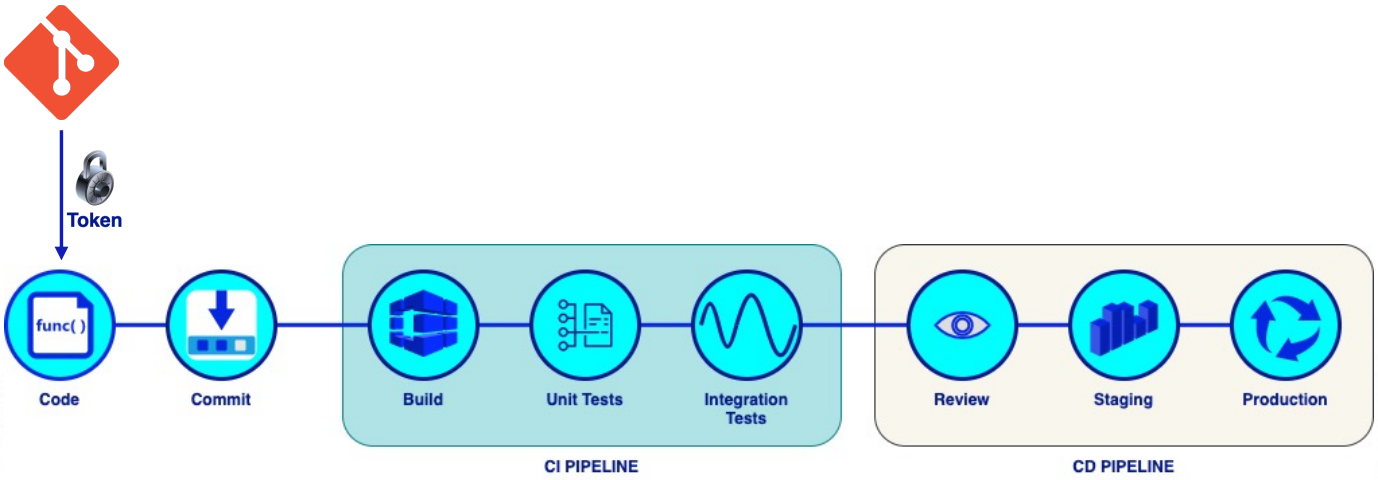
Gitlab




WILDCARD
Dashboard
W6D
W6D/FRONTEND
PIPELINES
PROJECT
BUILD
CI/CD
Search a project...
Plan & Billing
Sign out
W6D
cicd
PROD
Your app is not deployed yet
Pipeline 10
7828635b
Succeeded
COMMIT MESSAGE
update readme.md
BRANCH
develop
DATE
Oct 20, at 11:56am
DURATION
3min 11s
Pipeline 9
76415e59
Failed
COMMIT MESSAGE
update readme.md
BRANCH
develop
DATE
Oct 20, at 11:52am
DURATION
3min 59s
Pipeline 8
6fa5dac4
Failed
COMMIT MESSAGE
feat: add triggers call
BRANCH
develop
DATE
Oct 19, at 2:19pm
DURATION
5min 9s
Pipeline 7
7c6b6074
Succeeded
COMMIT MESSAGE
perf: shrink the file number in pipeline package
BRANCH
develop
DATE
Oct 19, at 11:42am
DURATION
3min 53s
Pipeline 6
54e1587c
Succeeded
COMMIT MESSAGE
update readme.md
BRANCH
develop
DATE
Oct 18, at 10:34am
DURATION
3min 10s

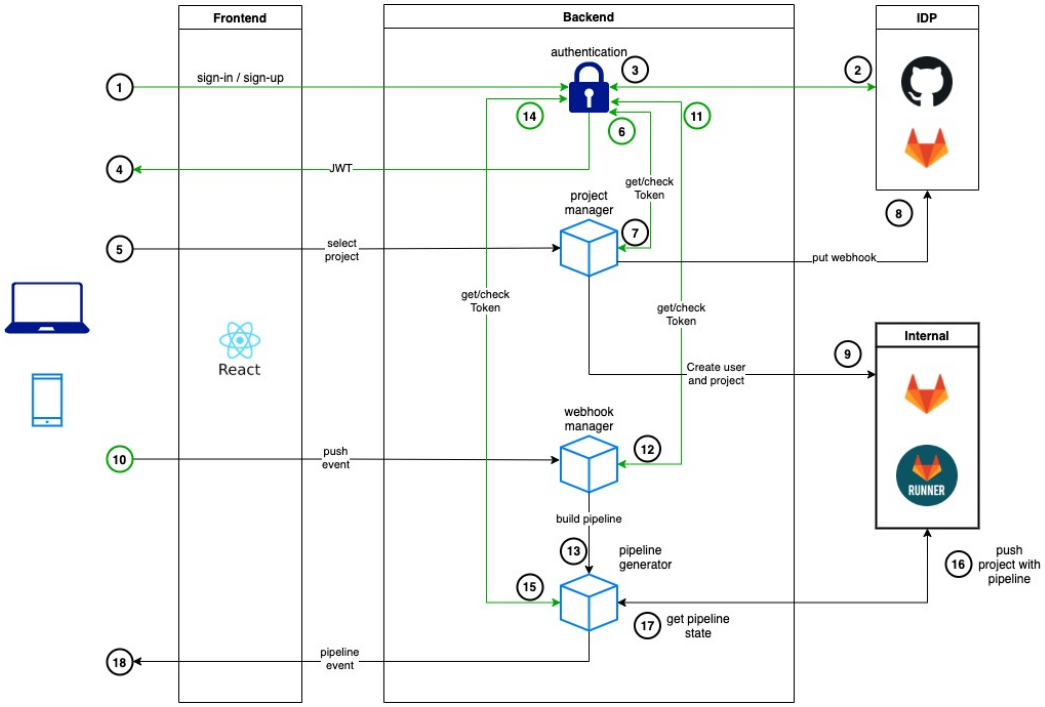

WILDCARD
Dashboard
W6D
W6D/FRONTEND
PIPELINES
PROJECT
BUILD
CI/CD
Search a project...
W6D
cicd
PROD
Your app is not deployed yet
CHOOSE YOUR BRANCH AND CREATE YOUR PIPELINE
develop
Enter a branch name
ADD
Unit test
Skip on failure
Edit test commands
Edit env config
Git leaks
Skip on failure
codecov / sonarqube
Build docker
Edit env config
Scan image vulnerabilities
Scan web app vulnerabilities (owasp compliance)
Scan CVE vulnerabilities
Skip on failure
Edit env config
Deploy
Edit env config
Confirm

ORY

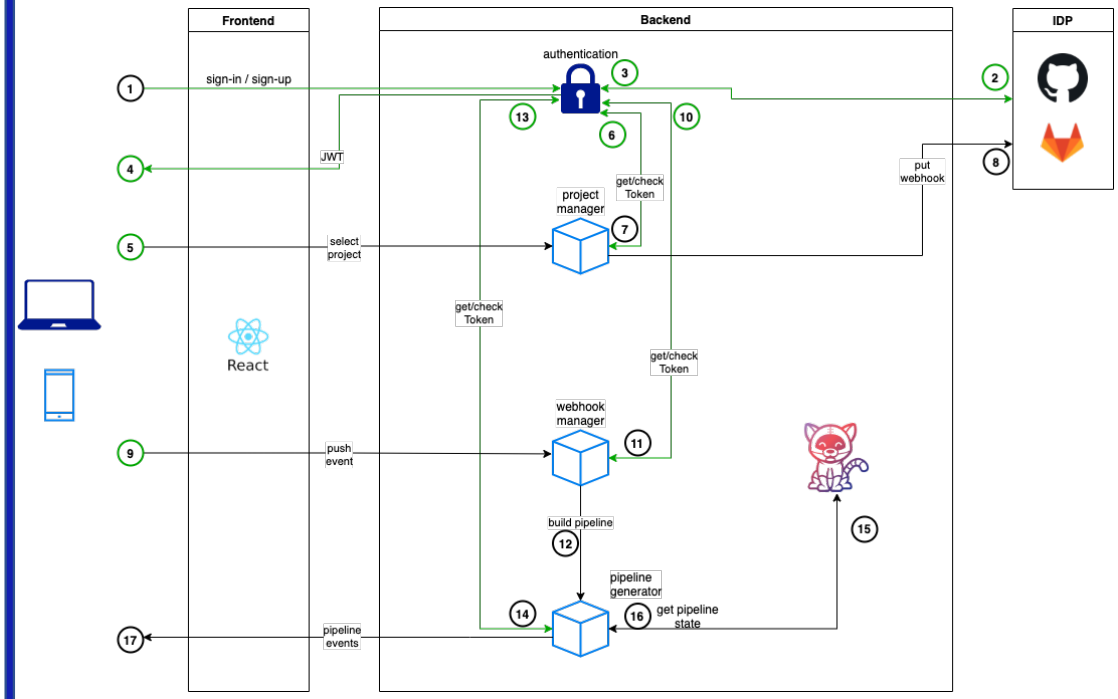

Token
func( )
Code
Commit
Build
Unit Tests
Integration Tests
Review
Staging
Production
CI PIPELINE
CD PIPELINE

Our history

BEFORE



NOW



Why Ory ?

Photo by Andy Hall, CC0

auth0

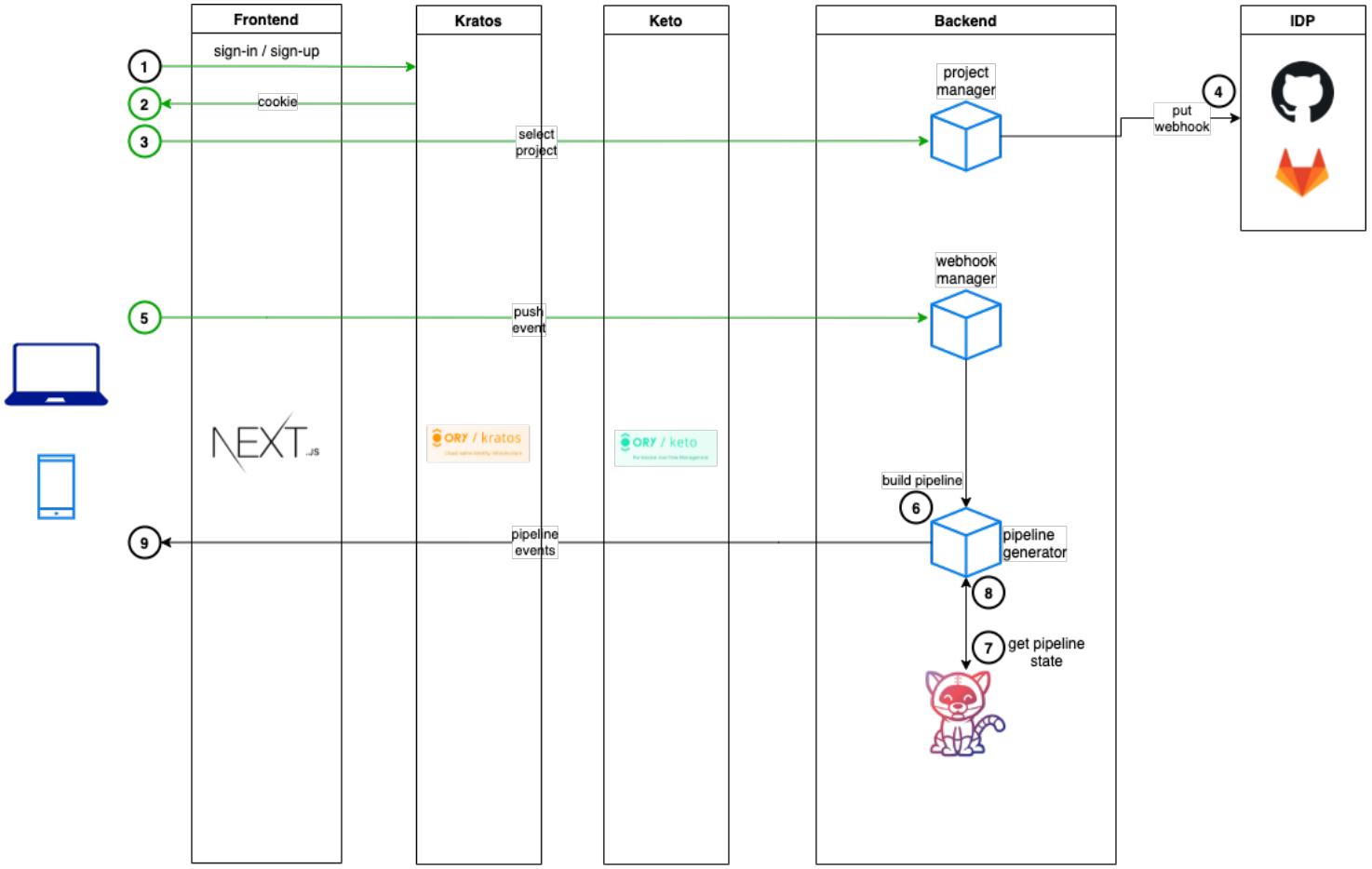
okta

ORY

KEYCLOAK

ZITADEL

The future user experience



The future user experience

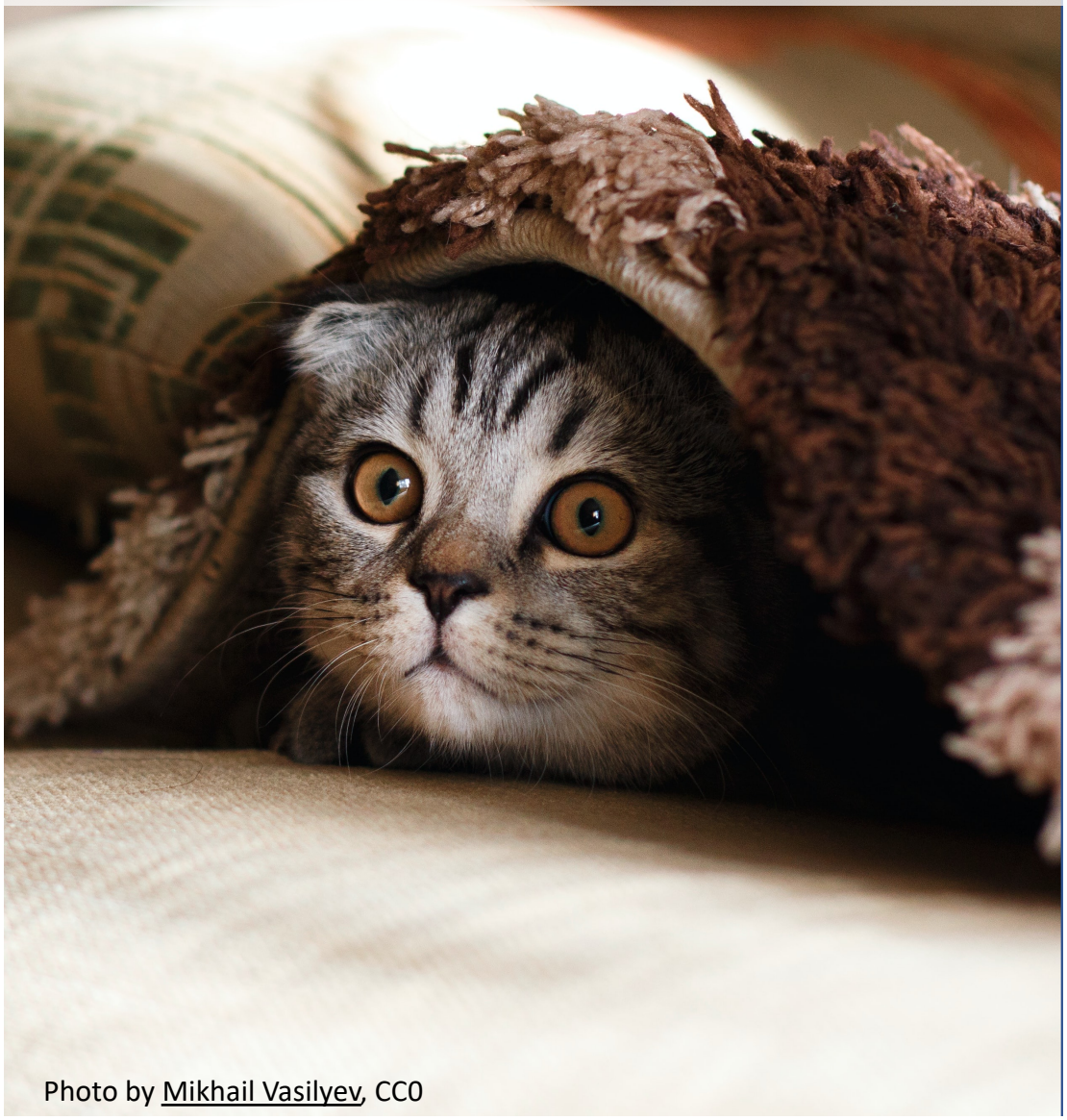
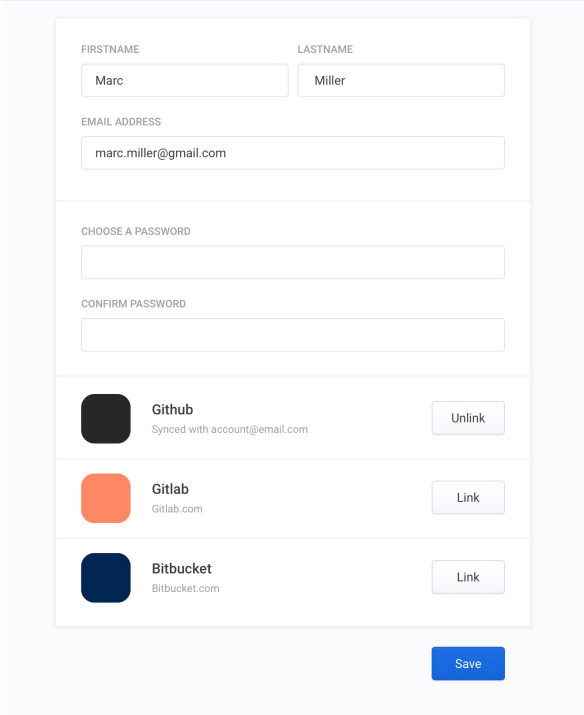


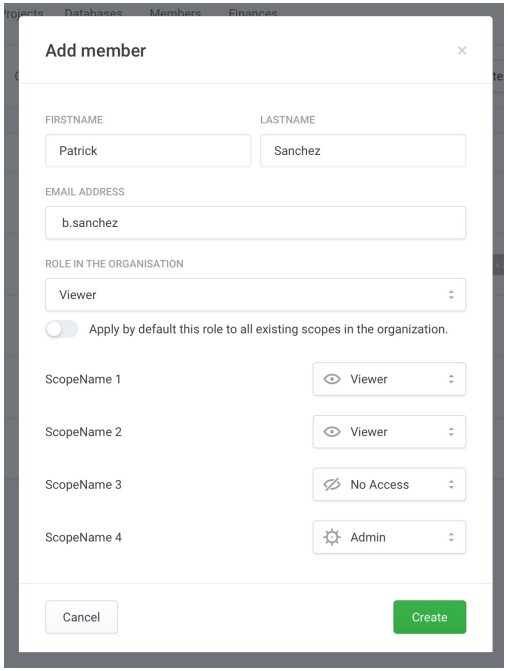
Photo by Mikhail Vasilyev, CC0

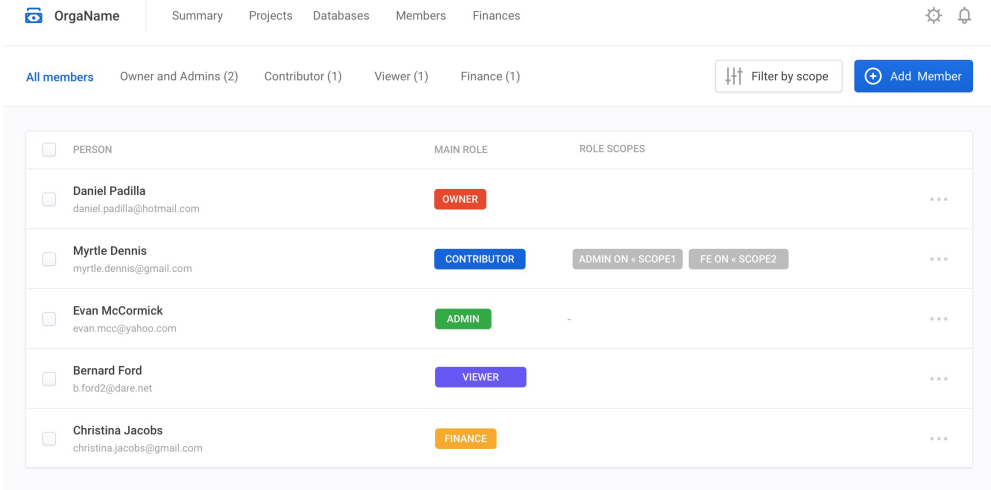
Provider Management



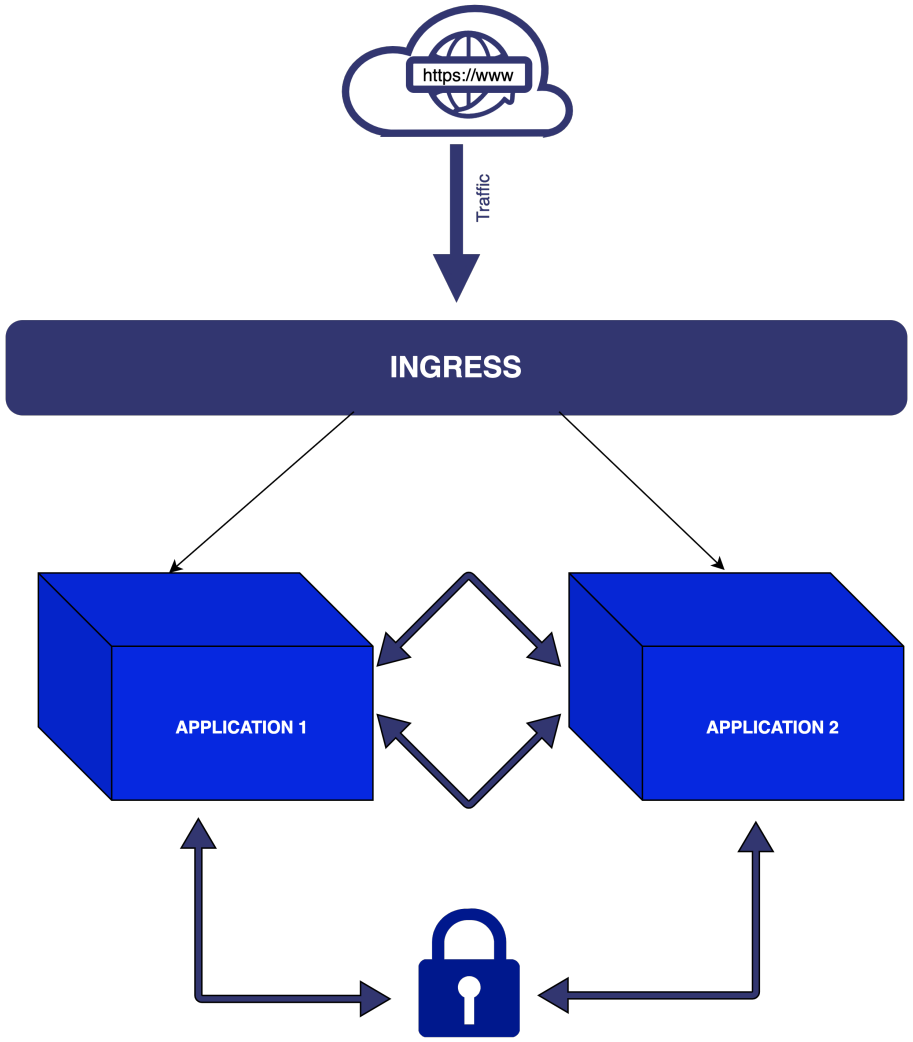
The future user experience

- Access management
- Team management





How we do it



How we do it – Step by step

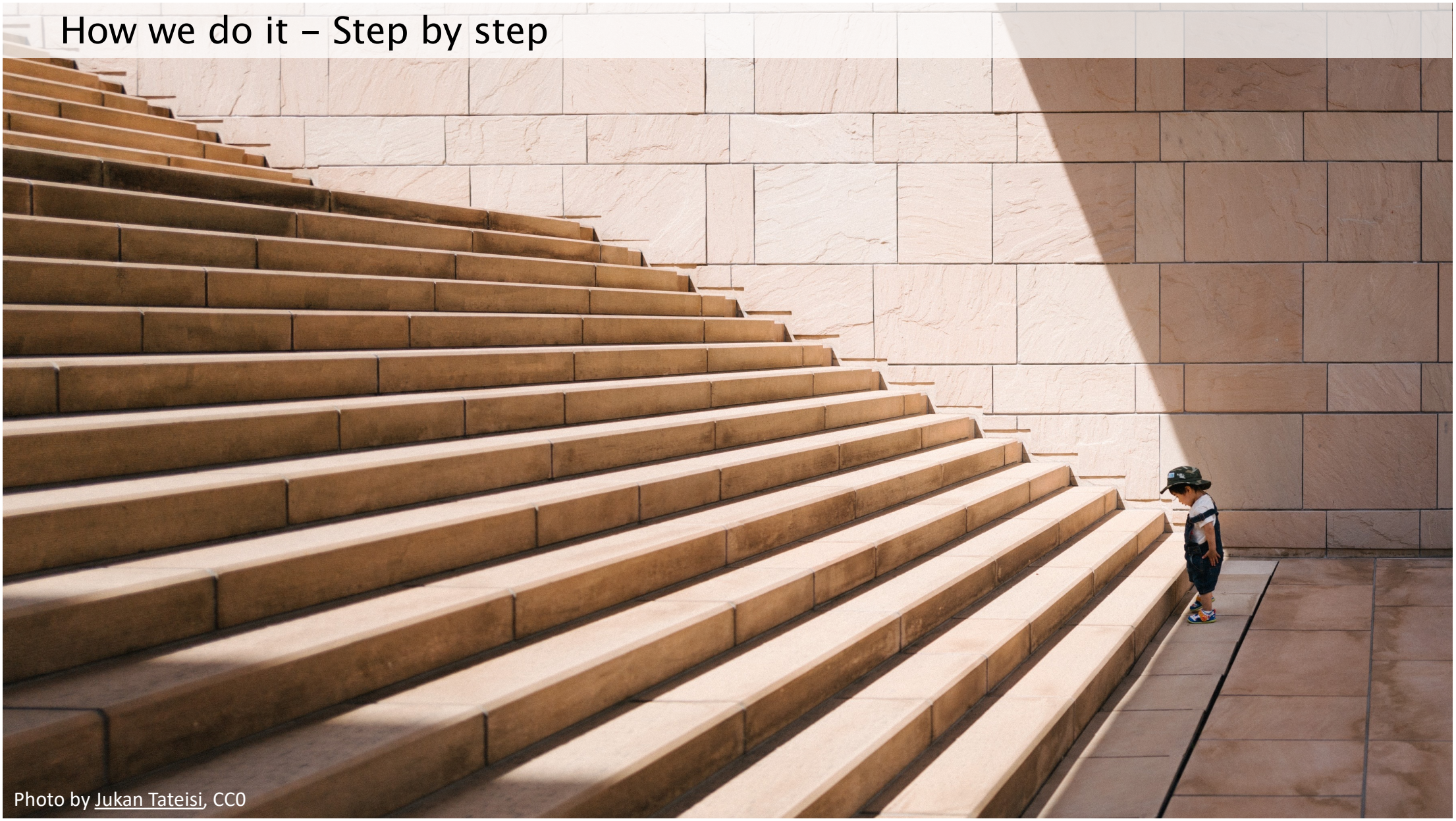
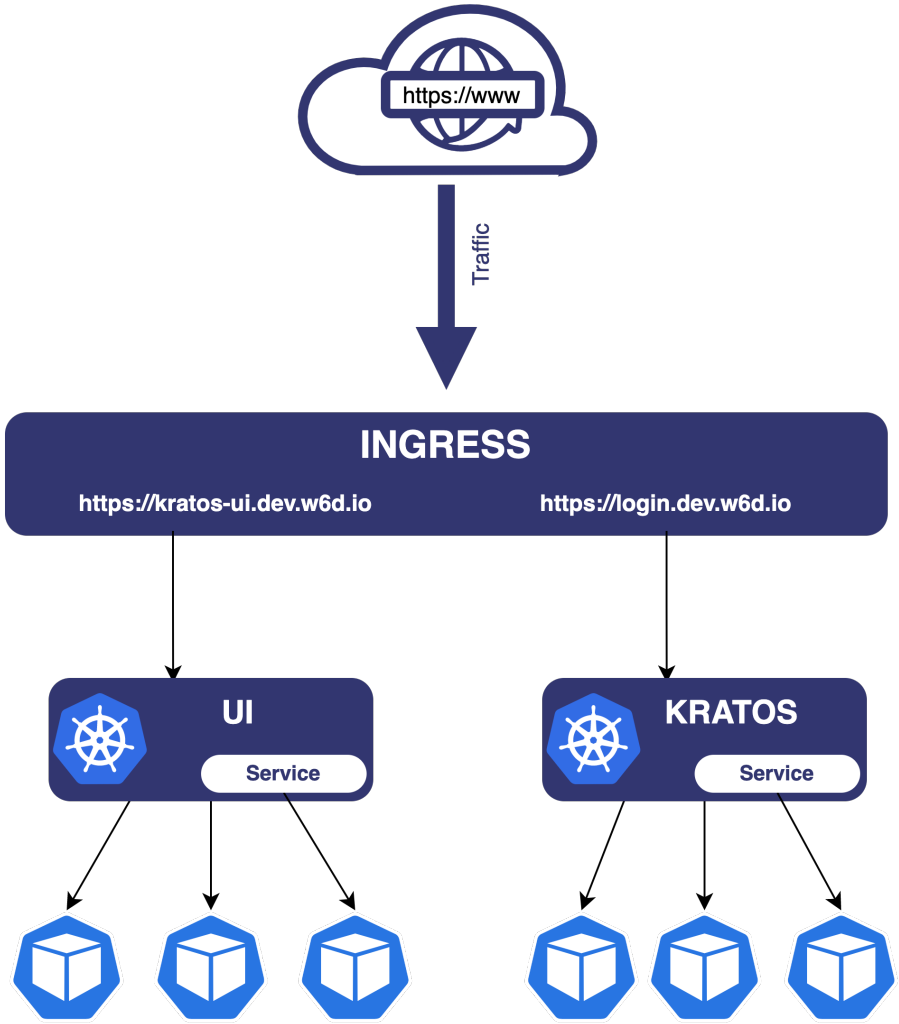
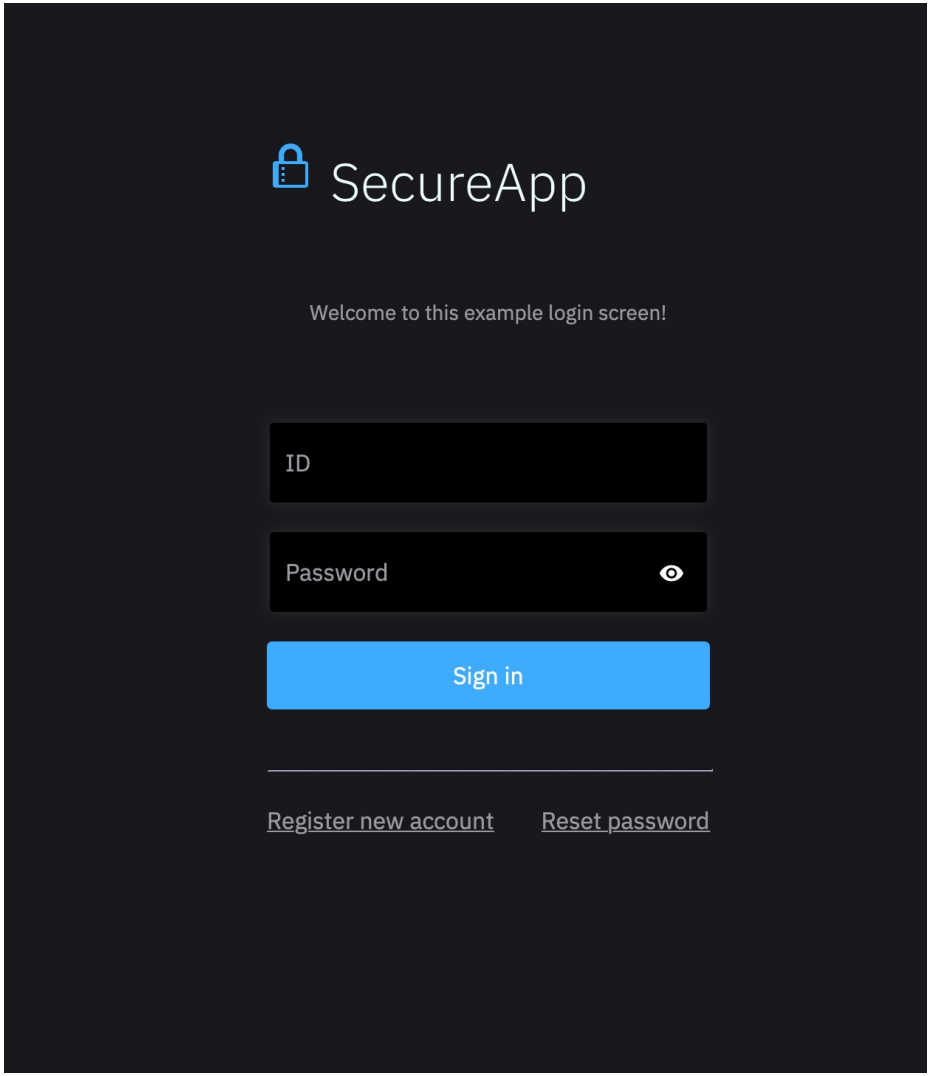


Photo by Jukan Tateisi, CC0

STEP 1
Rolling out QuickStart



STEP 2
Tests


SecureApp
Welcome to this example login screen!
ID
Password
Sign in
Register new account
Reset password

STEP 3
OIDC Configuration

```yaml
whitelisted_return_urls:
  - http://127.0.0.1:8080
  - http://127.0.0.1:4433
  - http://localhost:8080
  - "*"

methods:
  password:
    enabled: true
  oidc:
    enabled: true
    config:
      providers:
        - id: gitlab
          provider: gitlab
          client_id: 'xxxxxxxxxxxxxxxxxxxxxxxxxxxx'
          client_secret: 'xxxxxxxxxxxxxxxxxxxxxxxxxxxx'
          mapper_url: file:///kratos/mapper/gitlab-w6d.data-mapper.jsonnet
          scope:
            - api
            - read_user
            - read_repository
            - openid
            - profile
            - email
        - id: github
          provider: github
          client_id: 'xxxxxxxxxxxxxxxxxxxxxxxxxxxx'
          client_secret: 'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx'
          mapper_url: file:///etc/config/kratos/github.data-mapper.jsonnet
          scope:
            - user:email
            - repo
flows:
  error:
    ui_url: http://127.0.0.1:8080/error

  settings:
    ui_url: http://127.0.0.1:8080/settings
    privileged_session_max_age: 15m
```

STEP 4
Tests

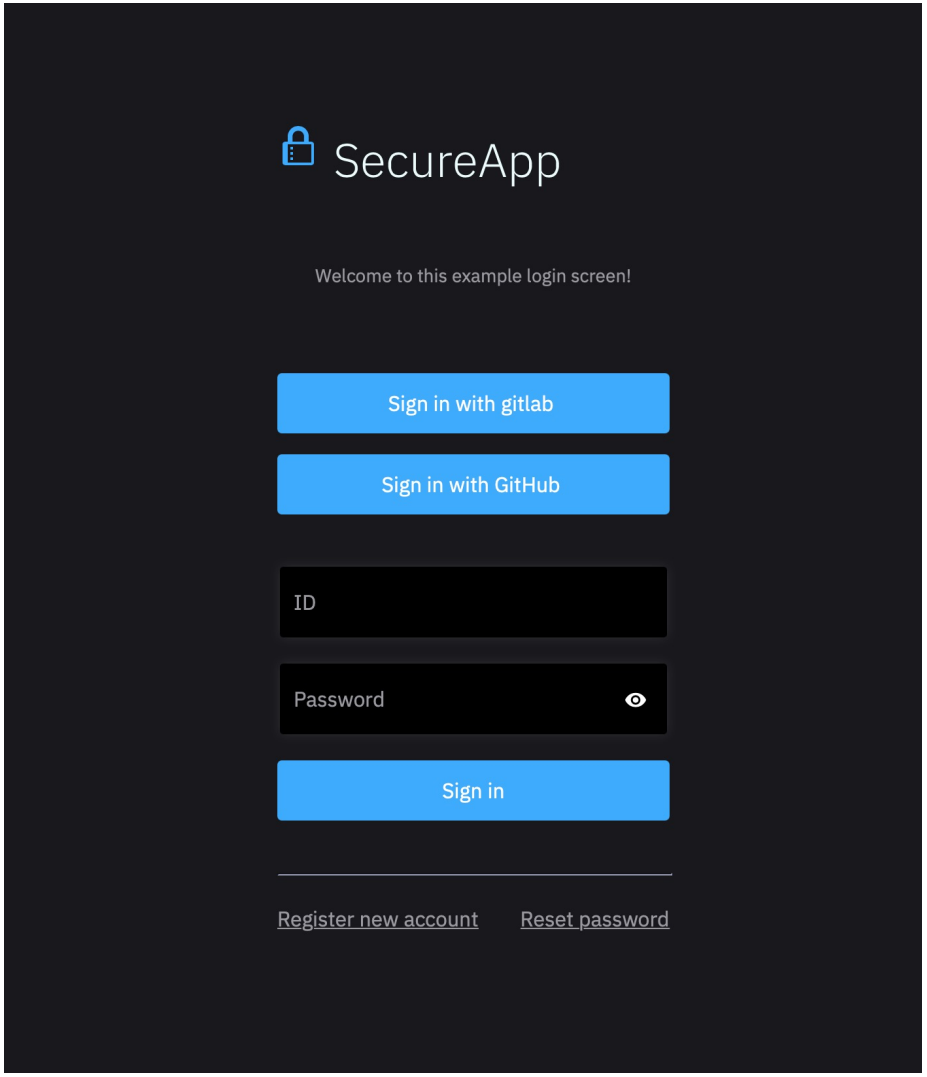
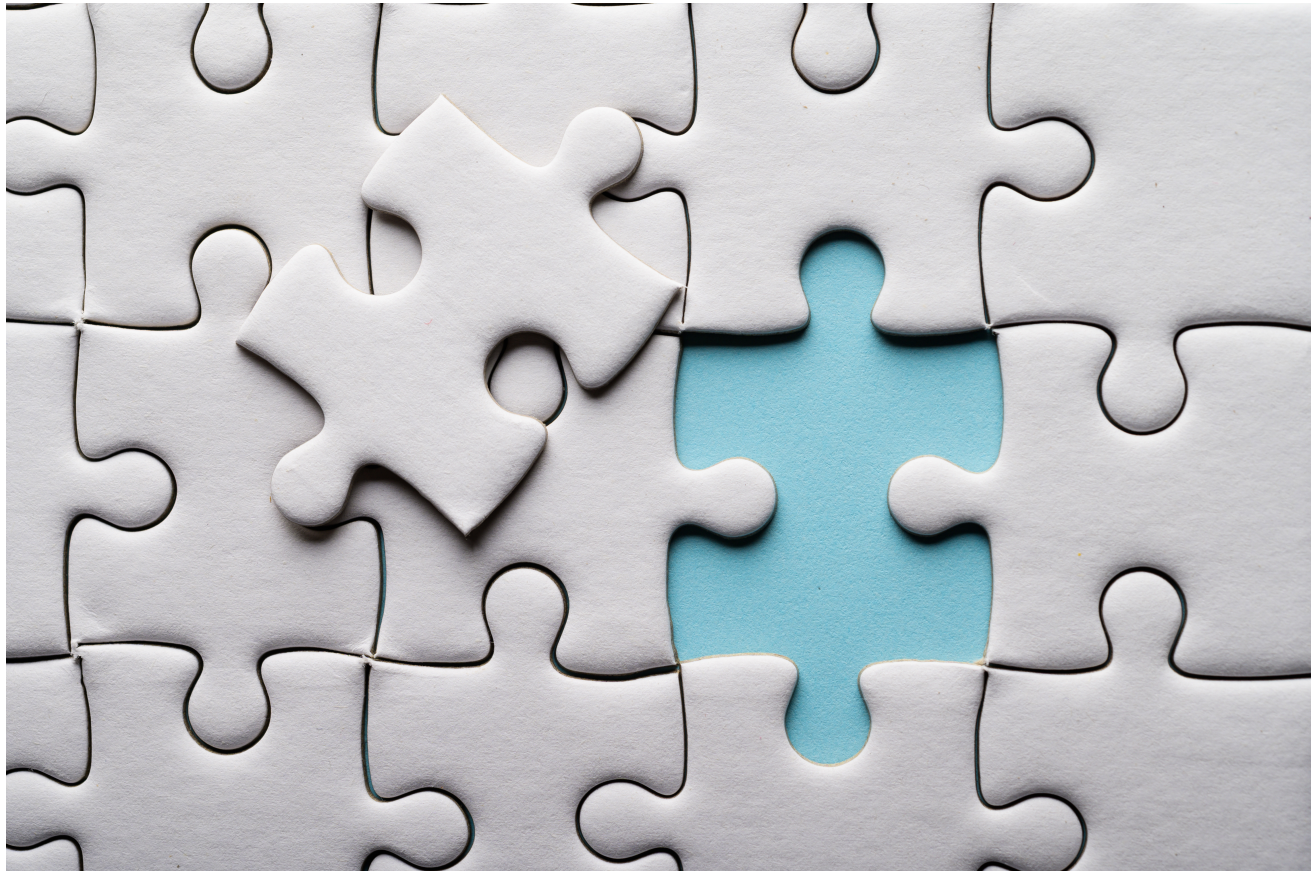

SecureApp
Welcome to this example login screen!
Sign in with gitlab
Sign in with GitHub
ID
Password
Sign in
Register new account
Reset password

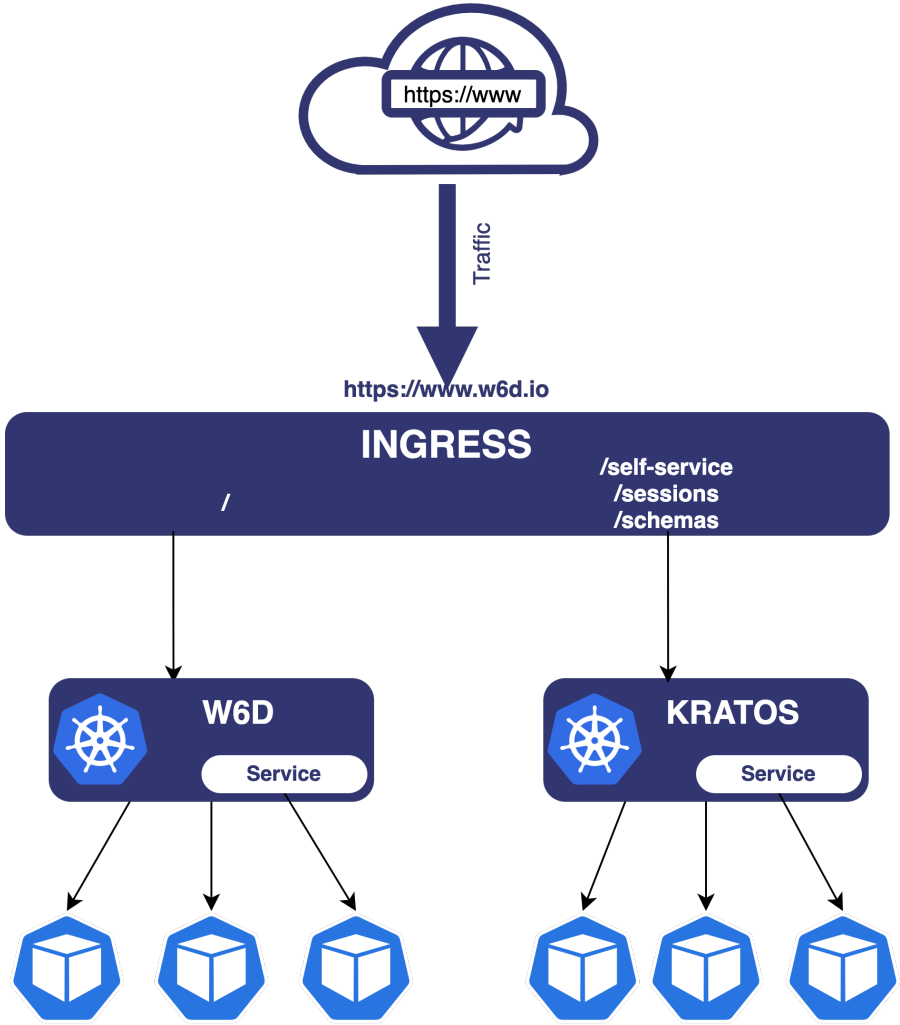
Photo by Aaron Burden, CC0

Something missing !

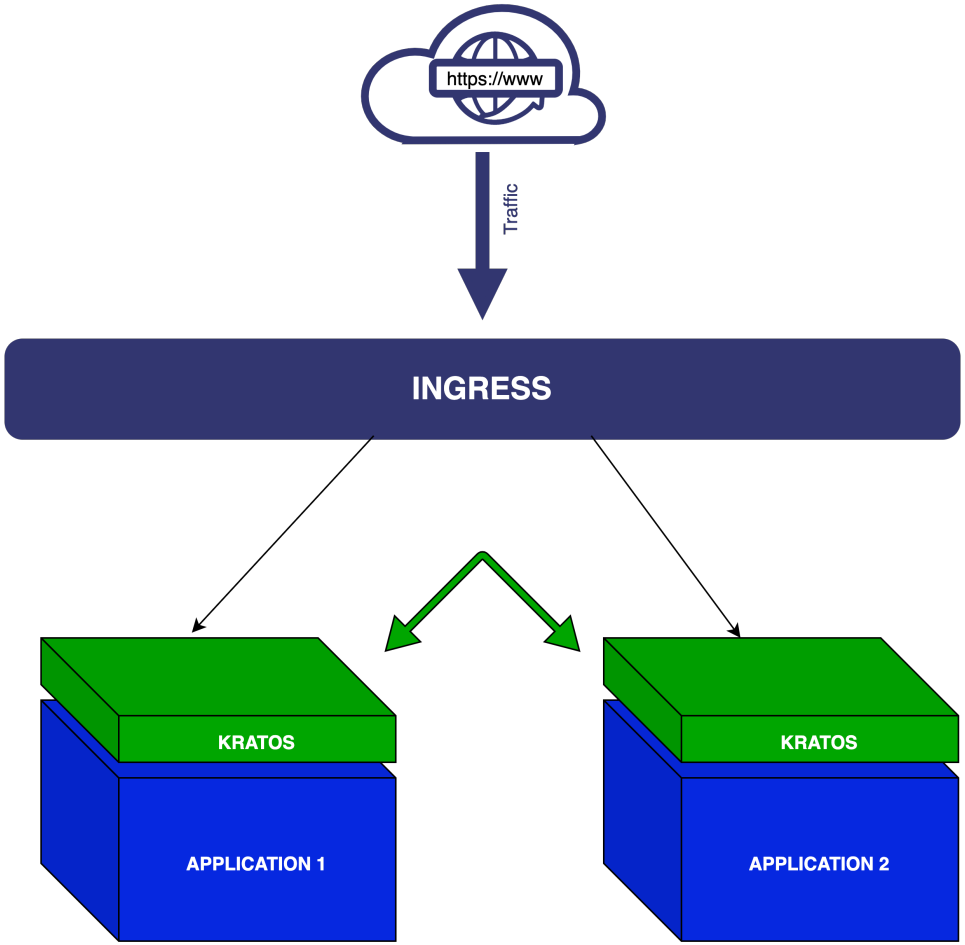
Our contribution



Last Kratos step implementation

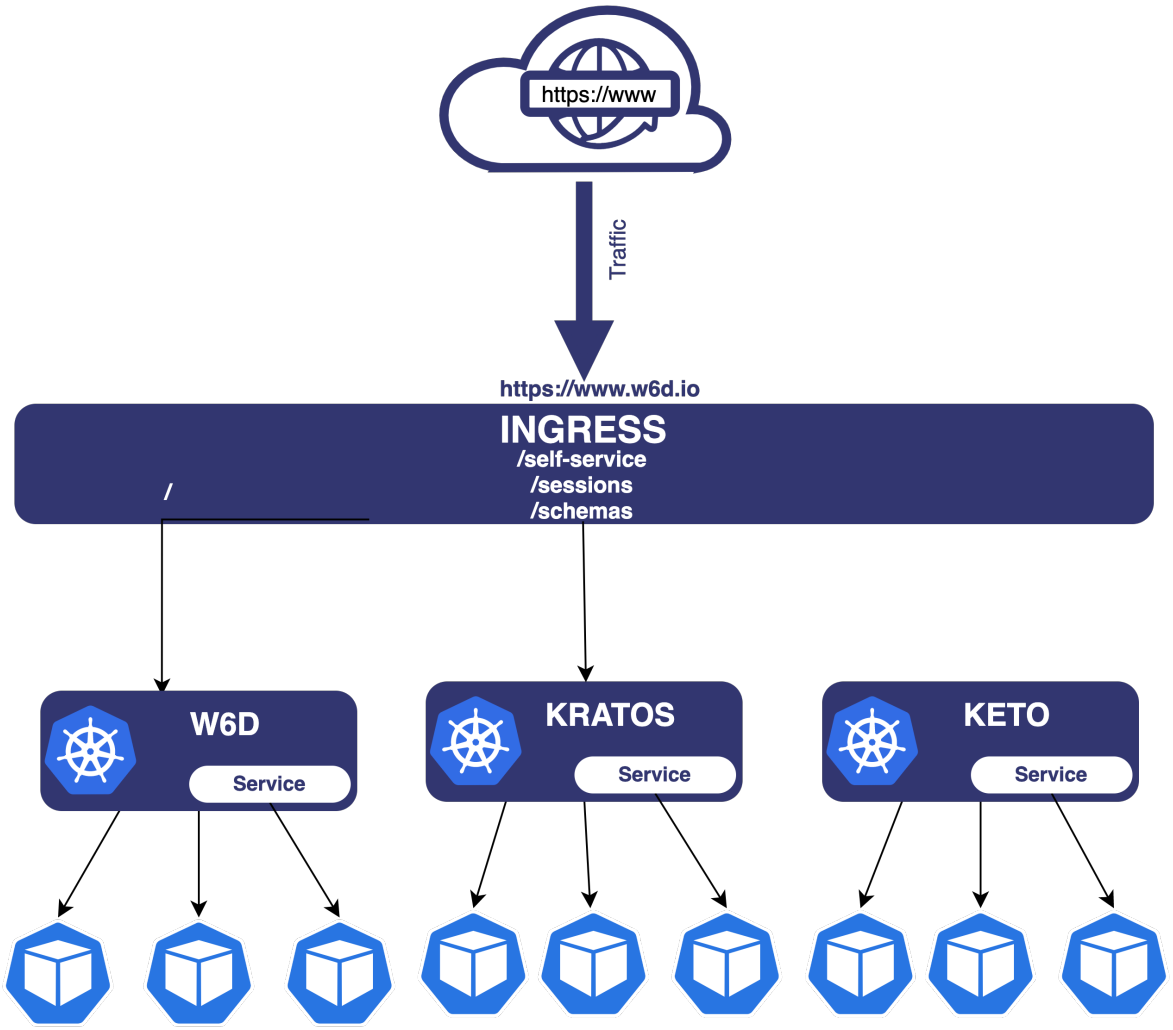


Now




```go
package kratos

import (
	"context"
	"net/http"
	"net/url"

	"github.com/w6d-io/x/errorx"

	client "github.com/ory/kratos-client-go"
	"github.com/ory/kratos/identity"
)

type Conn struct {
	// Address is the the address to the kratos micro service
	Address string `json:"address" mapstructure:"address"`
}

type Helper interface {

	// GetSessionFromHTTP is used to check if the session cookie is active ( ex: session.GetActive() )
	// and also return user information
	// if session is not set, return a nil session with StatusBadRequest and error
	// if kratos is unreachable or an other issues, return nil session with statusCode of the call and error-go
	GetSessionFromHTTP(ctx context.Context, req *http.Request) (*client.Session, error)

	// GetSessionFromGRPCCtx is used to forward a session stock into a context.
	// It checks if session on context is present
	// if session is not set, return a nil session with StatusBadRequest and error
	// if kratos is unreachable or an other issues, return nil session with statusCode of the call and error-go
	GetSessionFromGRPCCtx(ctx context.Context) (*client.Session, error)

	// GetIdentityFromHTTP is used to get the identity who correspond to the user id on kratos service
	// if kratos is unreachable or an other issues, return nil session with statusCode of the call and error-go
	GetIdentityFromHTTP(ctx context.Context, id string) (*identity.Identity, error)

	// GetIdentityFromAPI is used to get the identity who correspond to the user id on kratos service
	// if kratos is unreachable or an other issues, return nil session with statusCode of the call and error-go
	GetIdentityFromAPI(ctx context.Context, id string) (*client.Identity, error)

	// GetIdentityFromCtxHTTP gets the session from context and retrieve the identity ID
	// to make the http call
	GetIdentityFromCtxHTTP(ctx context.Context) (*identity.Identity, error)

	// GetIdentityFromCtxApi gets the session from context and retrieve the identity ID
	// to make the api call
	GetIdentityFromCtxApi(ctx context.Context) (*client.Identity, error)

	// GetTokenByHttp returns all tokens linked with the provider
	GetTokenByHttp(ctx context.Context, provider string) (*Provider, error)

	// GetTokensByHttp returns all tokens linked with the provider
	GetTokensByHttp(ctx context.Context) ([]Provider, error)
}

type Provider struct {
	TokenID      string `json:"initial_id_token"`
	Subject      string `json:"subject"`
	Provider     string `json:"provider"`
	AccessToken  string `json:"initial_access_token"`
	RefreshToken string `json:"initial_refresh_token"`
}
```

Our next move



Our next move

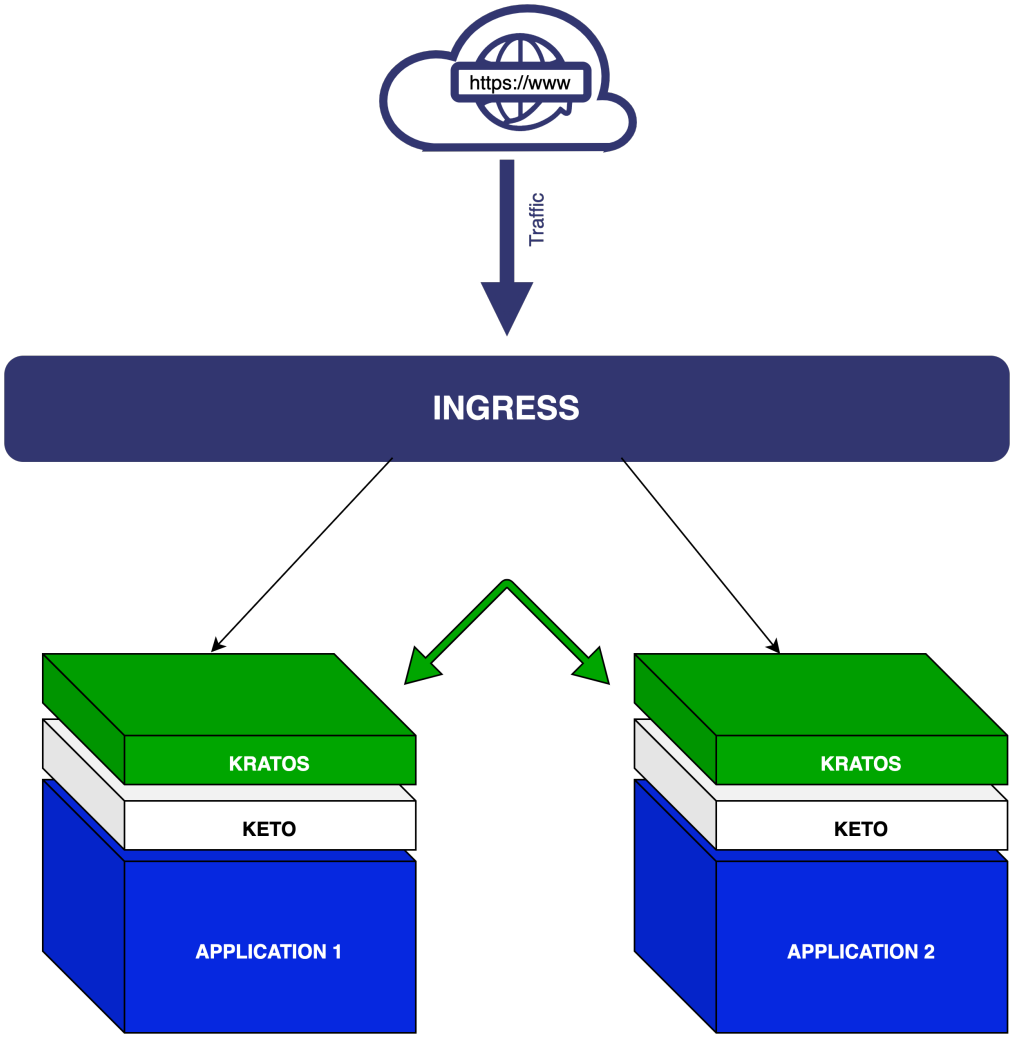


Photo by Alvan Nee, CC0

Thank you

Questions ?