

Stop the (password) insanity !

Juan Peredo

@JuanPeredoTech

<https://www.linkedin.com/in/juanperedotech>



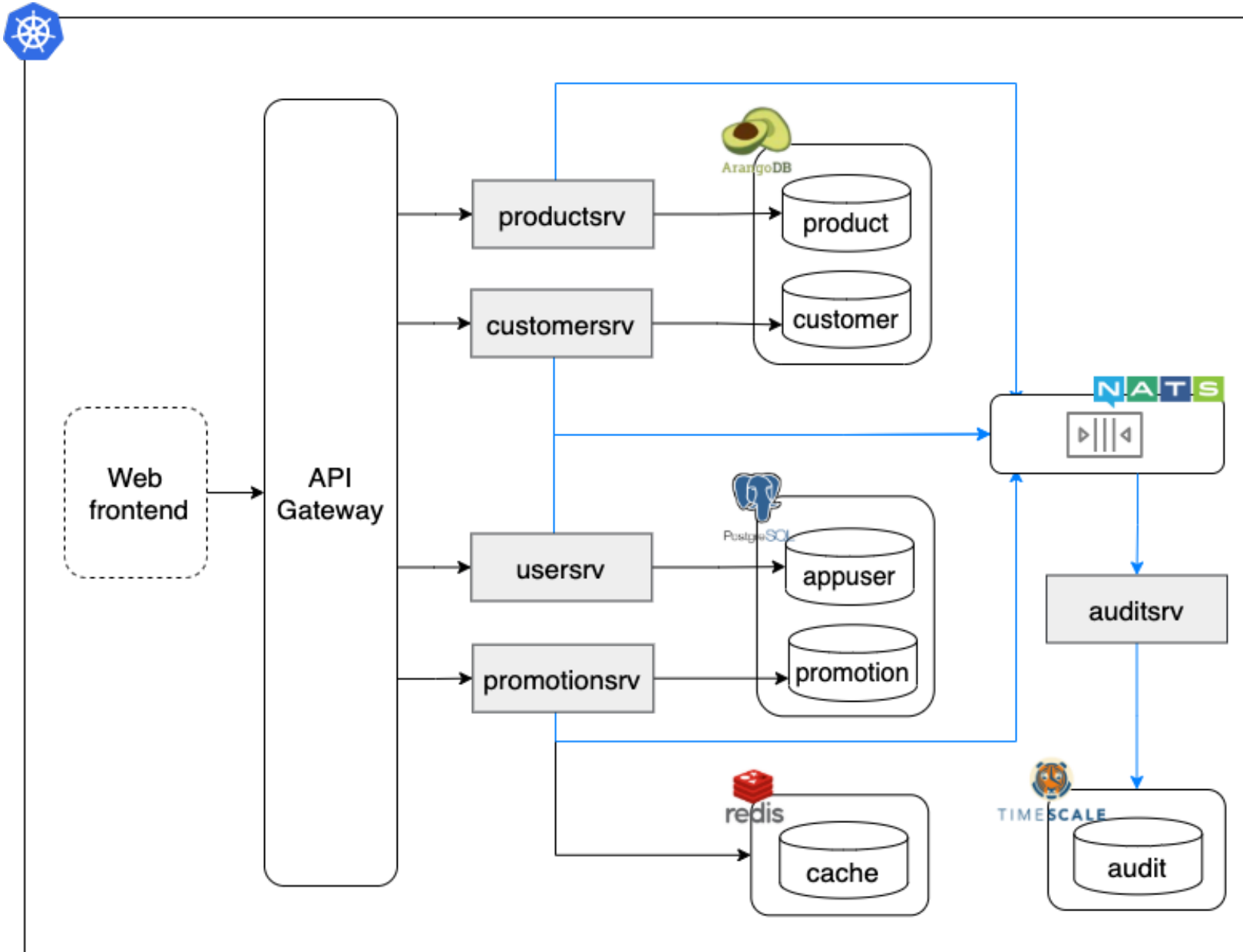


A simple assignment



- Jane is the newest member of the team
- Just assigned to rotating the application credentials
- Must be done on a regular basis

A bump on the road



- App uses microservices:
 - 5 micro services
 - 5 databases
 - All services connect to broker (NATS)
 - 1 service uses caching
 - **22** passwords & usernames

Doing a little more digging



- Services are managed by different teams
 - User service: User management team
 - Promotion service: Sales team
 - Customer and Product services: Master data team
 - Audit service: Traceability team
- Credentials need to be updated regularly

How Jane is feeling now





Our mission,



should we choose to accept it:

- Stop the (password) insanity by
 - Centralizing credentials management
 - Not changing any of the existing code
 - Bringing back Jane's work-life balance



Your guide in this mission

Independent tech architect / developer and everything in between.

Spent his extended 2020 pandemic induced "vacation":

- No going to the movies
- Not eating out
- Writing the app used our demo





Our Application

- 5 microservices: user, customer, product, promotion & audit
- Runs on Kubernetes
- Credentials read from environment variables
- We will focus initially on the user service (Usersrv)

The screenshot shows the landing page for 'goTemp'. On the left is a logo consisting of a diamond shape with 'gT' inside, followed by the text 'goTemp' and 'Microservices Demo App' in red. On the right, under the heading 'Welcome', there is a paragraph of text: 'This project provides a sample full stack microservices implementation. Try exploring the application using the menu items at the top of the page. If you are curious about the tech stack, feel free to review the repo documentation for more info. If you find any issues or problems please create an issue or a pull request. Contributions are most welcome.' Below this text is a dark button labeled 'Repo'.

The screenshot shows the user management interface for 'goTemp'. At the top left, it says 'User: Super Duck'. At the top right, there are navigation links for 'Promotions' and 'Services', and three buttons: 'Save', 'Delete', and 'Back'. The interface is divided into four sections: 'Information', 'Validity', 'Organization', and 'Security'. The 'Information' section contains fields for 'Id' (2467751939371176961), 'First Name' (Super), and 'last Name' (Duck). The 'Validity' section contains fields for 'Valid From' (1999-12-31) and 'Valid Thru' (2199-12-31). The 'Organization' section contains a field for 'Company' (Duck Inc.). The 'Security' section contains fields for 'Email' (duck@mymail.com), 'Password' (masked with dots), and an 'Active' checkbox which is checked.

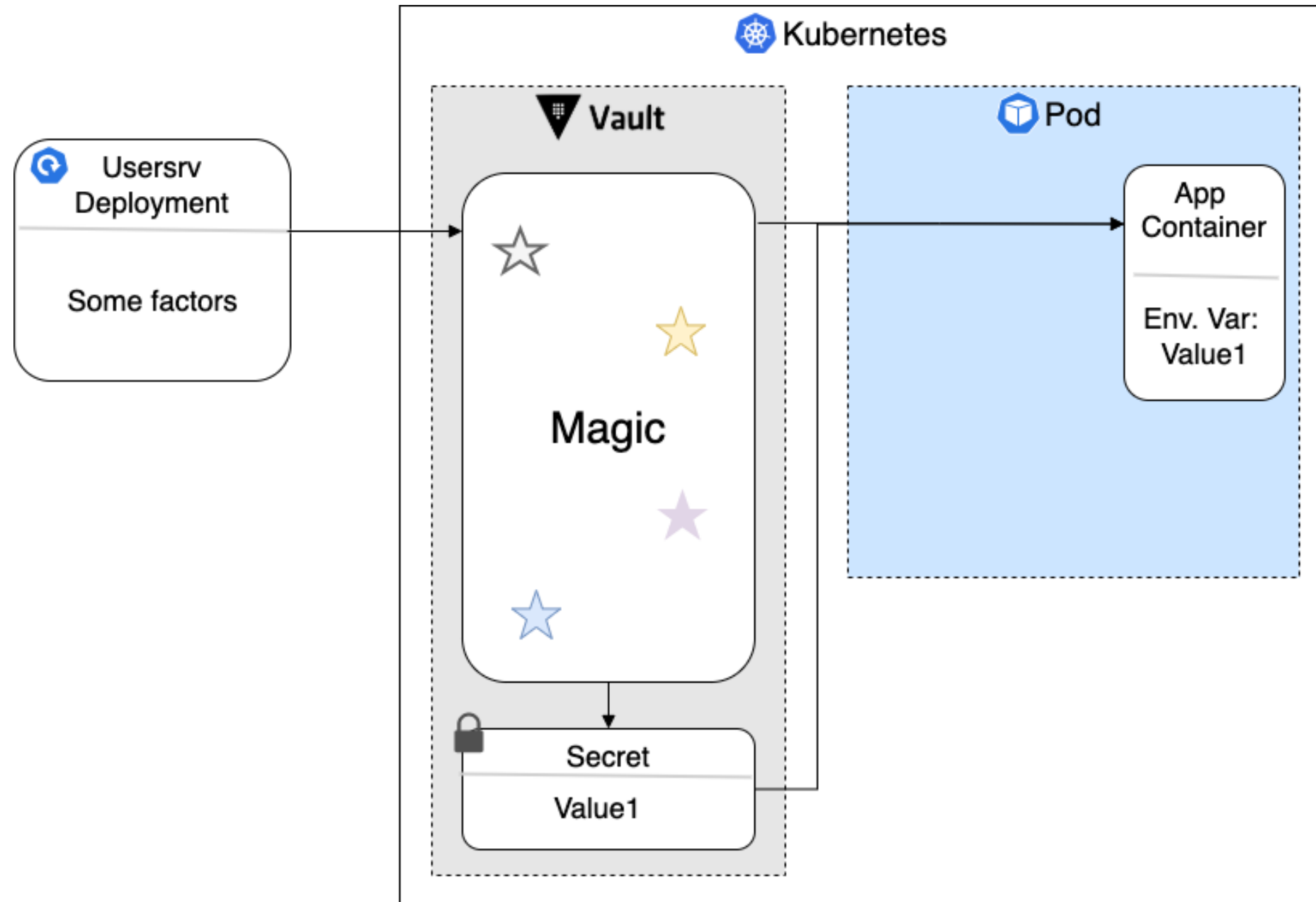


Usersrv Environment Variables

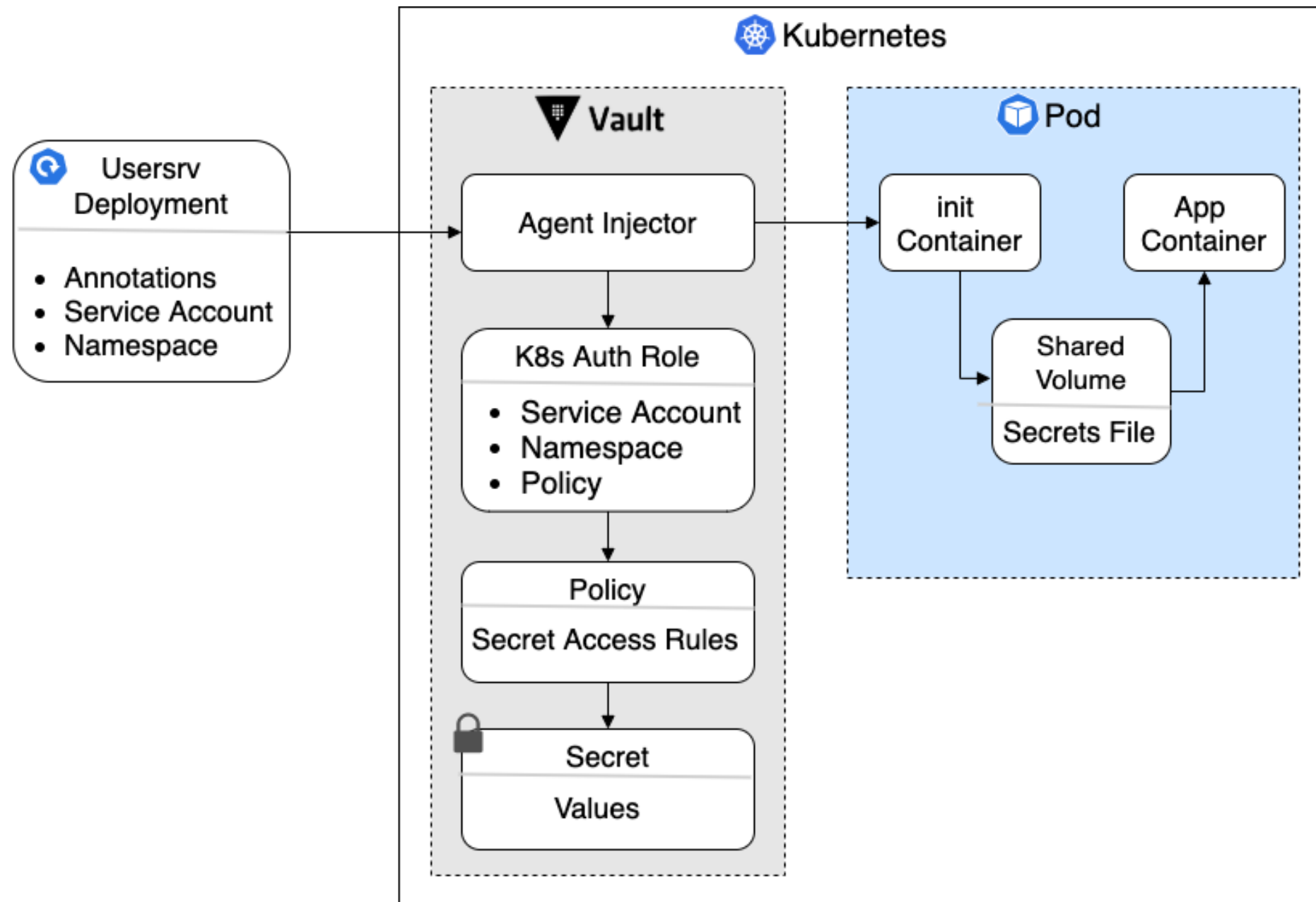


```
MICRO_SERVER_ADDRESS=:50053
POSTGRES_CONNECT=
    postgresql://postgres:Passwd@pgdb/
        appuser?application_name=userSrv
MICRO_BROKER=nats
MICRO_BROKER_ADDRESS=natsUser:Passwd@nats
DISABLE_AUDIT_RECORDS=false
```

The Goal



The plan





Demo

Scripts can be found at:

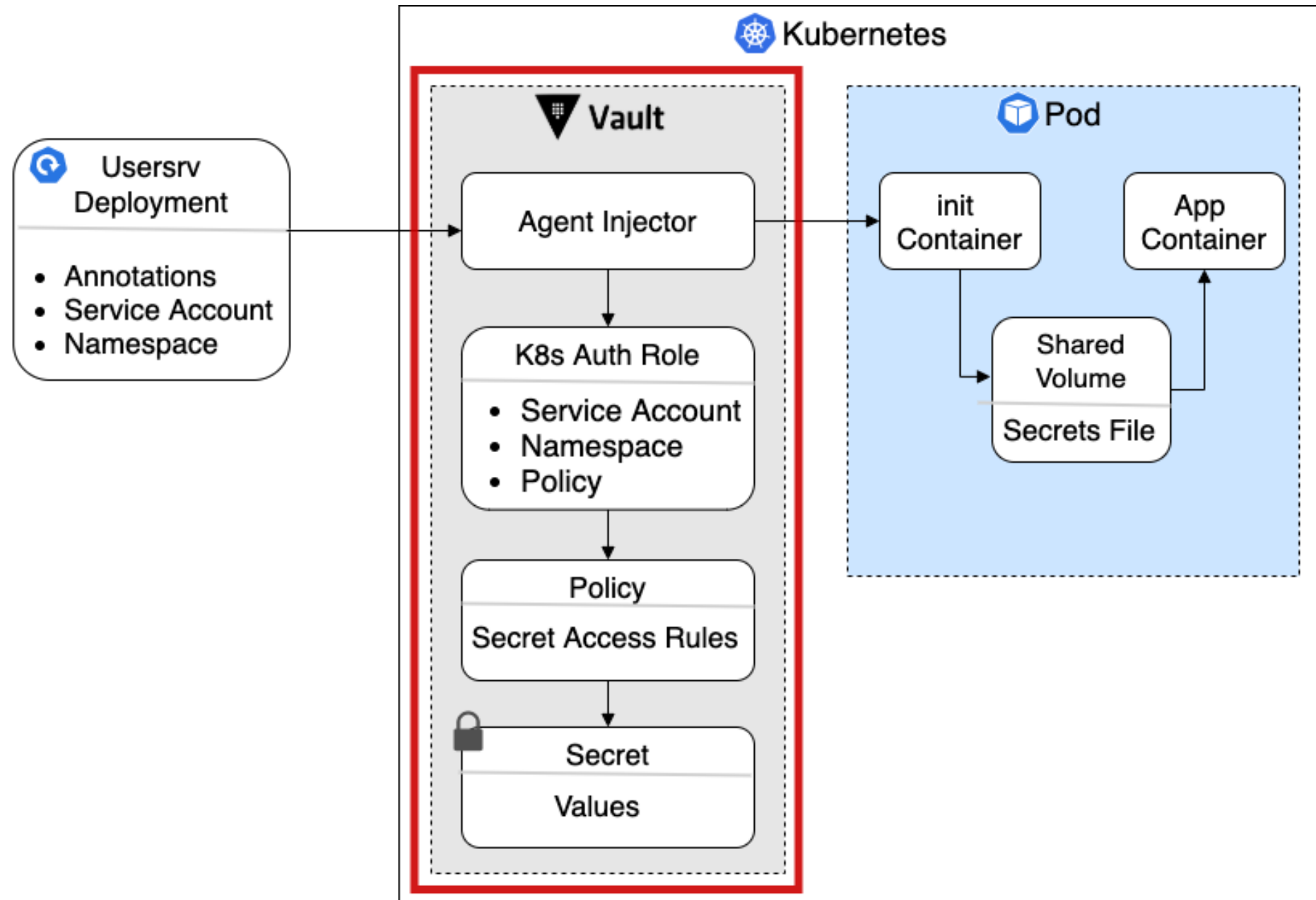
<https://github.com/camba1/gotemp/tree/master/vault>

Application found at:

<https://github.com/camba1/gotemp>

```
57 t.appeared = false;
58 return;
59 }
60 //is the element inside the visible window?
61 var a = w.scrollLeft();
62 var b = w.scrollTop();
63 var o = t.offset();
64 var x = o.left;
65 var y = o.top;
66
67 var ax = settings.accX;
68 var ay = settings.accY;
69 var th = t.height();
70 var wh = w.height();
71 var tw = t.width();
72 var ww = w.width();
73
74 if (y + th + ay >= b &&
75     y <= b + wh + ay &&
76     x + tw + ax >= a &&
77     x <= a + ww + ax) {
78     //trigger the custom event
79     if (!t.appeared) t.trigger('appear', settings.data);
80 } else {
81     //it scrolled out of view
82     t.appeared = false;
83 }
84 };
85
86 //create a modified fn with some additional logic
87 var modifiedFn = function() {
88
89     //mark the element as visible
90     t.appeared = true;
91
92     //is this supposed to happen only once?
93     if (settings.one) {
94
95         //remove the check
96         w.unbind('scroll', check);
97         var i = $.inArray(check, $.fn.appear.checks);
98         if (i >= 0) $.fn.appear.checks.splice(i, 1);
99     }
100 }
```

Vault Configuration





Configure Vault

Enable secret Engine

Enable Kubernetes
Auth

Configure Kubernetes
Auth

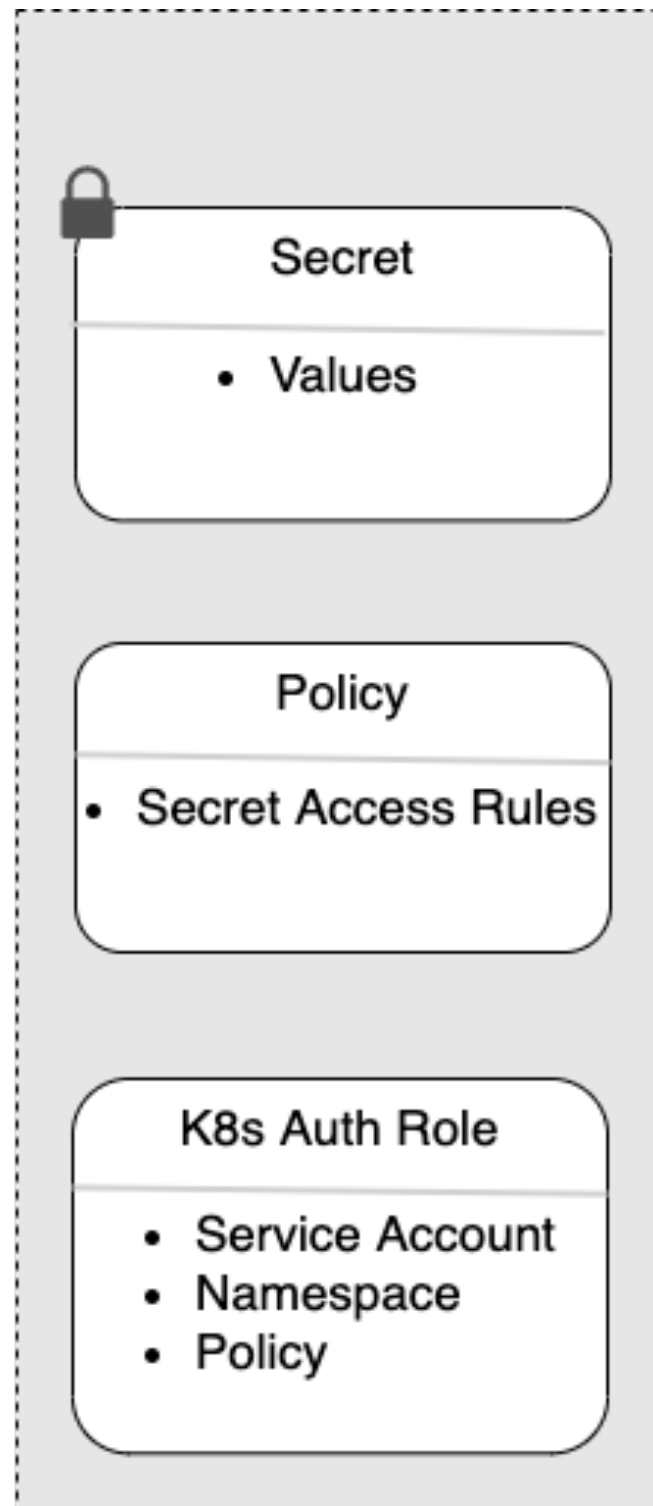
```
vault secrets enable -path=gotempkv kv-v2
```

```
vault auth enable kubernetes
```

```
vault write auth/kubernetes/config \  
  token_reviewer_jwt=  
    "$(cat /var/run/secrets/  
      kubernetes.io/serviceaccount/token)" \  
  kubernetes_host=  
    "https://  
      $KUBERNETES_PORT_443_TCP_ADDR:443" \  
  kubernetes_ca_cert=  
    @/var/run/secrets/kubernetes.io/  
      serviceaccount/ca.crt
```



Setup Secrets

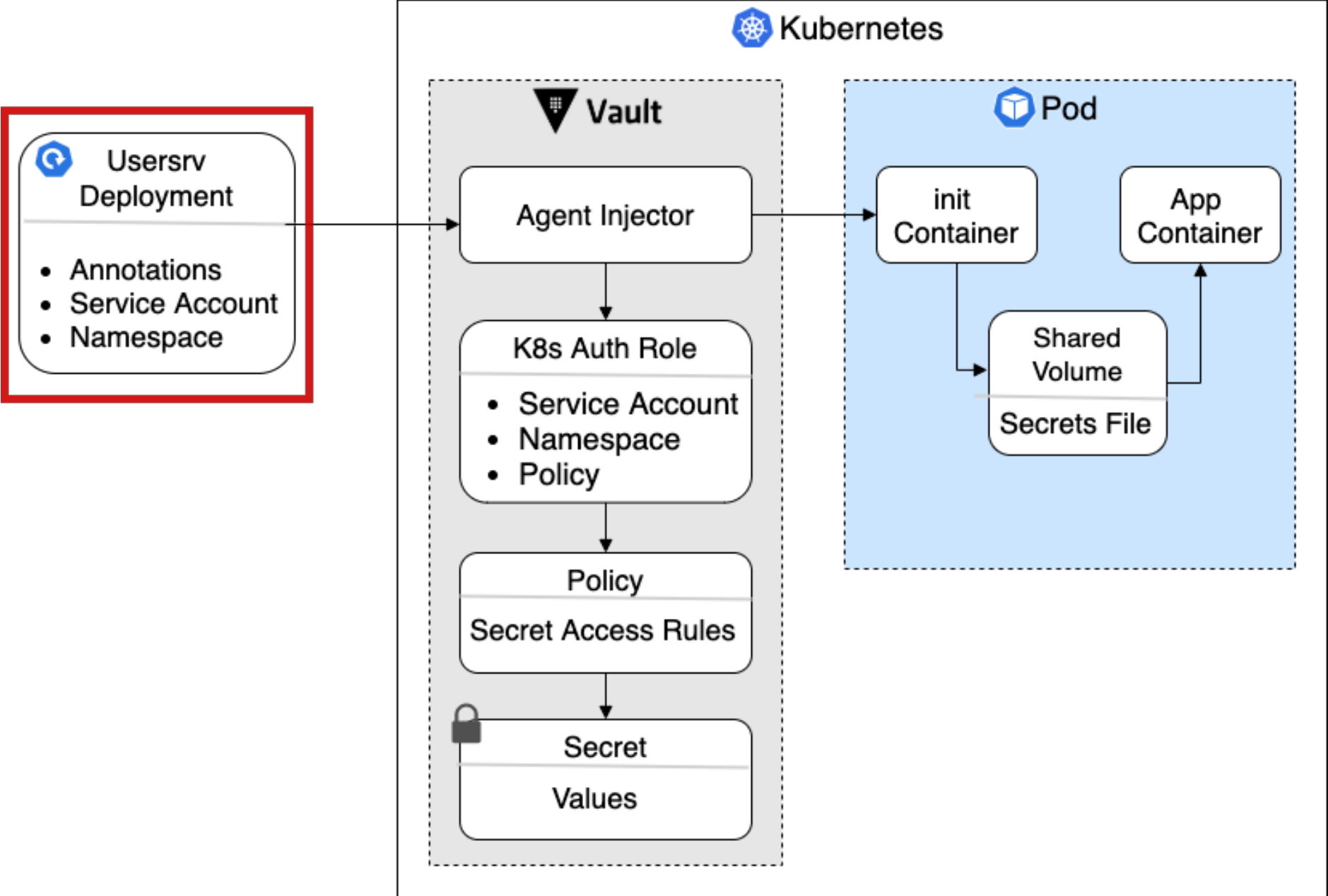


```
vault kv put \  
  gotempkv/broker/nats/usersrv \  
  username="natsUser" \  
  password="Passwd" \  
  server="nats"
```

```
vault policy write gotemp-usersrv - <<EOF  
path "gotempkv/data/broker/nats/usersrv" {  
  capabilities = ["read"]  
}  
EOF
```

```
vault write \  
  auth/kubernetes/role/gotemp-usersrv \  
  bound_service_account_names=gotemp-usersrv \  
  bound_service_account_namespaces=default \  
  policies=gotemp-usersrv \  
  ttl=24h
```

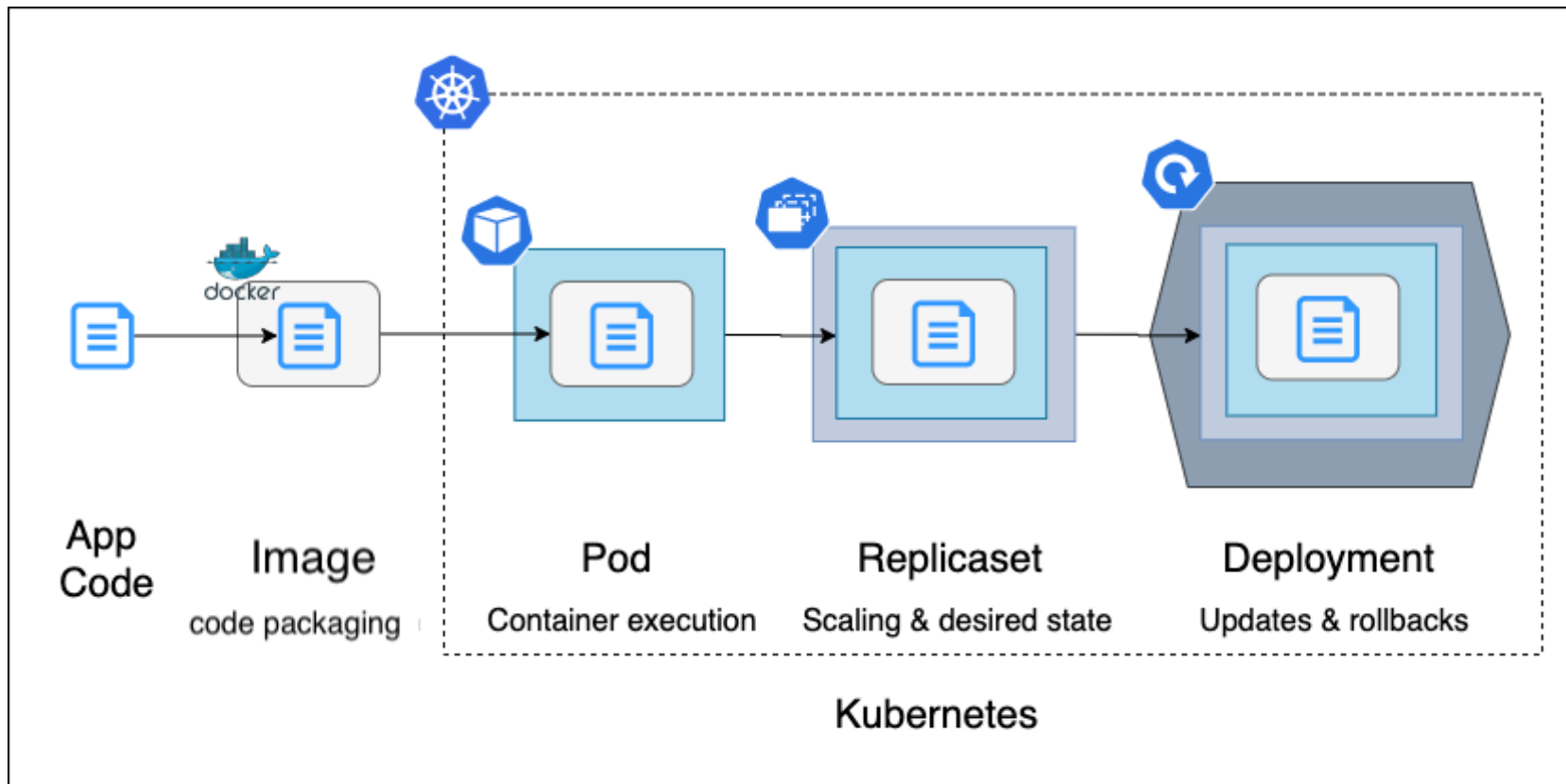
Usersrv Configuration



Apps in Kubernetes

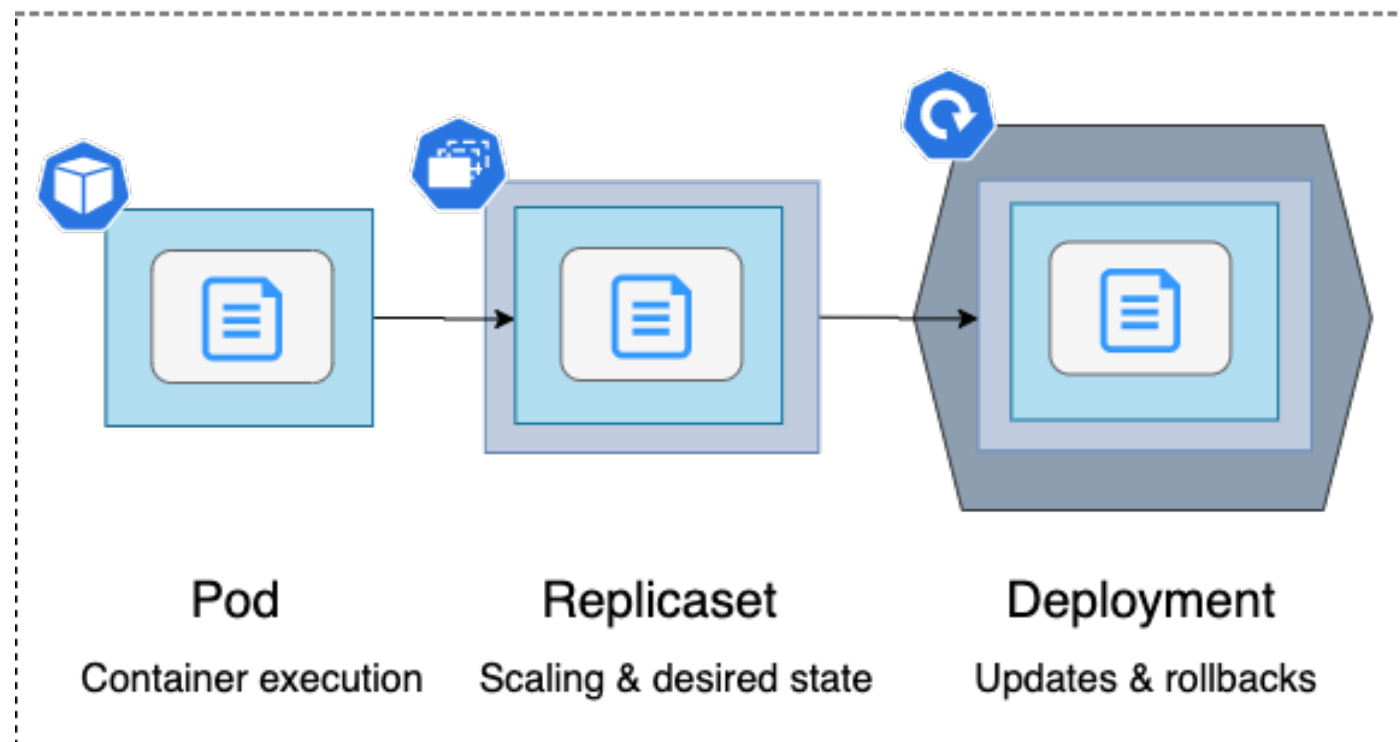


- Pods: Smallest deployment unit
- Replicaset: Controls number of pod instances
- Deployment: Where all the magic happens





Usersrv Deployment YAML



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: usersrv
spec:
  replicas: 1
  template:
    metadata:
      annotations:
        kompose.version: 1.21.0 ()
    spec:
      containers:
      - env:
        - name: MICRO_BROKER_ADDRESS
          valueFrom: ...
        - name: POSTGRES_CONNECT
          valueFrom: ....
        image: bolbeck/gotemp_usersrv
        name: usersrvcont
        serviceAccountName: "" ...
```

Service Account



```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: gotemp-usersrv
```




Annotations

```
vault.hashicorp.com/agent-inject: "true"
vault.hashicorp.com/agent-pre-populate-only: "true"
vault.hashicorp.com/role: "gotemp-usersrv"
vault.hashicorp.com/agent-inject-secret-nats.txt: "gotempkv/data/broker/nats/usersrv"
vault.hashicorp.com/agent-inject-template-nats.txt: |
    {{- with secret "gotempkv/data/broker/nats/usersrv" -}}
        export MICRO_BROKER_ADDRESS="{{ .Data.data.username }}:
            {{ .Data.data.password }}@{{ .Data.data.server }}"
    {{- end -}}
```



Annotations

```
vault.hashicorp.com/agent-inject: "true"
vault.hashicorp.com/agent-pre-populate-only: "true"
vault.hashicorp.com/role: "gotemp-usersrv"
vault.hashicorp.com/agent-inject-secret-nats.txt: "gotempkv/data/broker/nats/usersrv"
vault.hashicorp.com/agent-inject-template-nats.txt: |
    {{- with secret "gotempkv/data/broker/nats/usersrv" -}}
        export MICRO_BROKER_ADDRESS="{{ .Data.data.username }}:
            {{ .Data.data.password }}@{{ .Data.data.server }}"
    {{- end -}}
```



Annotations

```
vault.hashicorp.com/agent-inject: "true"
vault.hashicorp.com/agent-pre-populate-only: "true"
vault.hashicorp.com/role: "gotemp-usersrv"
vault.hashicorp.com/agent-inject-secret-nats.txt: "gotempkv/data/broker/nats/usersrv"
vault.hashicorp.com/agent-inject-template-nats.txt: |
    {{- with secret "gotempkv/data/broker/nats/usersrv" -}}
        export MICRO_BROKER_ADDRESS="{{ .Data.data.username }}:
            {{ .Data.data.password }}@{{ .Data.data.server }}"
    {{- end -}}
```



Annotations

```
vault.hashicorp.com/agent-inject: "true"
vault.hashicorp.com/agent-pre-populate-only: "true"
vault.hashicorp.com/role: "gotemp-usersrv"
vault.hashicorp.com/agent-inject-secret-nats.txt: "gotempkv/data/broker/nats/usersrv"
vault.hashicorp.com/agent-inject-template-nats.txt: |
    {{- with secret "gotempkv/data/broker/nats/usersrv" -}}
        export MICRO_BROKER_ADDRESS="{{ .Data.data.username }}:
            {{ .Data.data.password }}@{{ .Data.data.server }}"
    {{- end -}}
```

Pull secret from Vault:

- Using auth role: "gotemp-usersrv"
- From end point: "gotempkv/data/broker/nats/usersrv"
- Place it in the file: /vault/secret/secret-nats.txt



Annotations

```
vault.hashicorp.com/agent-inject: "true"
vault.hashicorp.com/agent-pre-populate-only: "true"
vault.hashicorp.com/role: "gotemp-usersrv"
vault.hashicorp.com/agent-inject-secret-nats.txt: "gotempkv/data/broker/nats/usersrv"
vault.hashicorp.com/agent-inject-template-nats.txt: |
  {{- with secret "gotempkv/data/broker/nats/usersrv" -}}
    export MICRO_BROKER_ADDRESS="{{ .Data.data.username }}:
      {{ .Data.data.password }}@{{ .Data.data.server }}"
  {{- end -}}
```

Format secret as :

```
export MICRO_BROKER_ADDRESS = "username:password@server"
```




Annotations

```
vault.hashicorp.com/agent-inject: "true"
vault.hashicorp.com/agent-pre-populate-only: "true"
vault.hashicorp.com/role: "gotemp-usersrv"
vault.hashicorp.com/agent-inject-secret-nats.txt: "gotempkv/data/broker/nats/usersrv"
vault.hashicorp.com/agent-inject-template-nats.txt: |
    {{- with secret "gotempkv/data/broker/nats/usersrv" -}}
        export MICRO_BROKER_ADDRESS="{{ .Data.data.username }}:
            {{ .Data.data.password }}@{{ .Data.data.server }}"
    {{- end -}}
```




Patch the deployment

 Usersrv
Deployment


annotations:
 kompose.version: 1.21.0 ()
spec:
 serviceAccountName: ""
 containers:
 - name: usersrvcont



 Usersrv
Patch

annotations:
 vault.hashicorp.com/agent-inject
spec:
 serviceAccountName: gotemp-usersrv
 containers:
 - name: usersrvcont
 command: newCommand



 Usersrv
Deployment

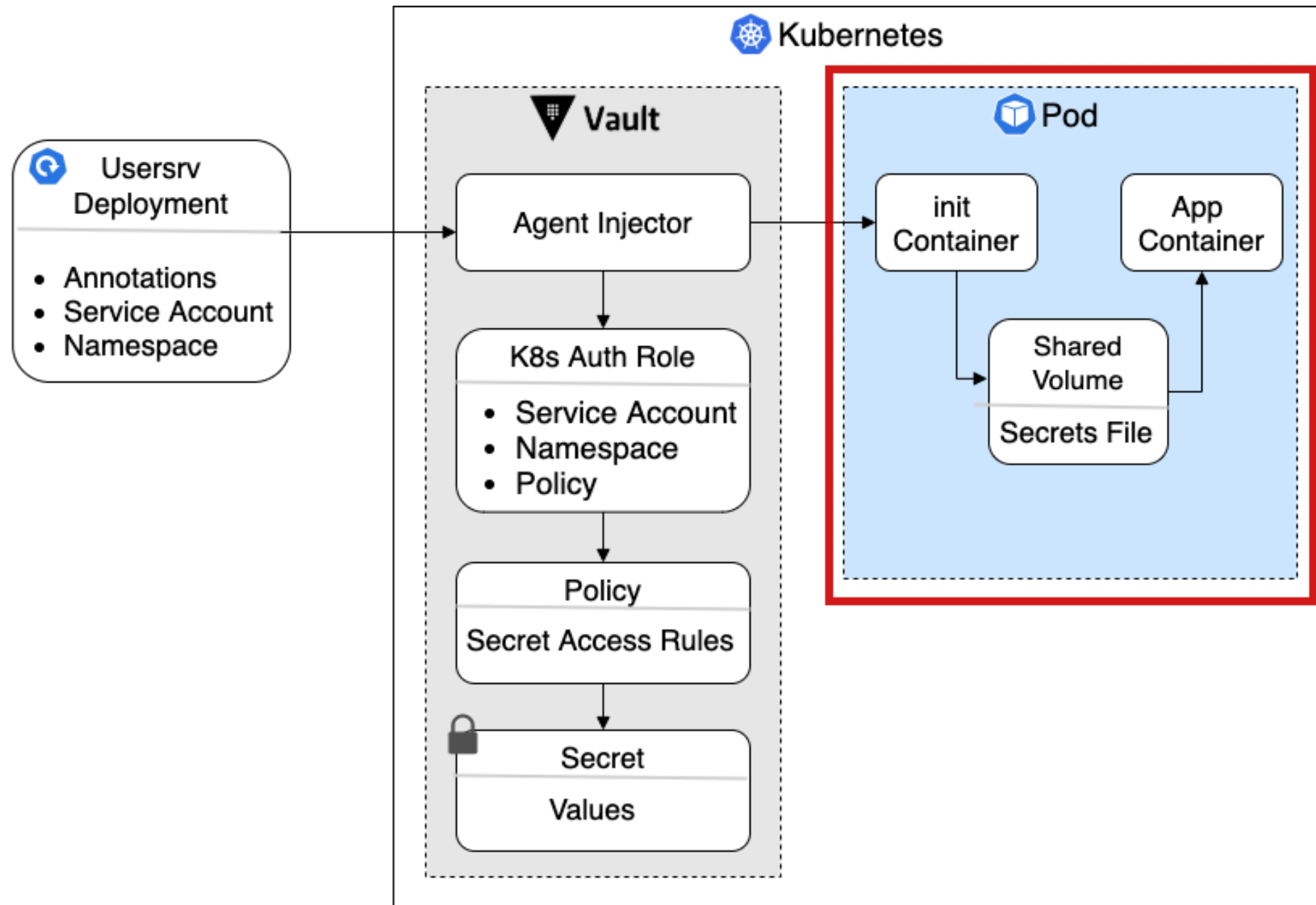
annotations:
 kompose.version: 1.21.0 ()
 vault.hashicorp.com/agent-inject
spec:
 serviceAccountName: gotemp-usersrv
 containers:
 - name: usersrvcont
 command: newCommand

The Patch




```
spec:
  template:
    metadata:
      annotations:
        vault.hashicorp.com/agent-inject: "true"
        vault.hashicorp.com/role: "gotemp-usersrv"
        vault.hashicorp.com/agent-pre-populate-only: "true"
        vault.hashicorp.com/agent-inject-secret-nats.txt: "gotempkv/data/broker/nats/usersrv"
        vault.hashicorp.com/agent-inject-template-nats.txt: |
          {{- with secret "gotempkv/data/broker/nats/usersrv" -}}
            export MICRO_BROKER_ADDRESS="{{ .Data.data.username }}:
              {{ .Data.data.password }}@{{ .Data.data.server }}"
          {{- end -}}
spec:
  serviceAccountName: gotemp-usersrv
  containers:
  - name: usersrvcont
    command: ['sh', '-c', 'source /vault/secrets/nats.txt && ./userServerAlp' ]
```


Deploy Changes

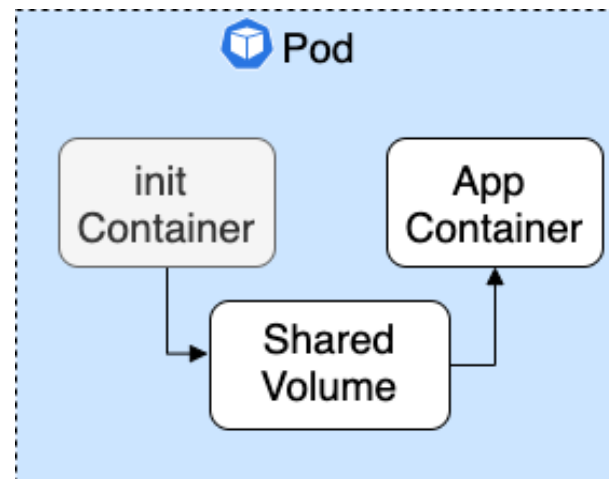




Putting it all together

Action	Result	Usersrv	 PostgreSQL
Application Startup	All services and DBs started up	Environment Variables: Username: postgres Pasword: XXXXXXXX	DB Credentials: Username: postgres Pasword: XXXXXXXX
Change DB credentials	DB credentials changed		DB Credentials: Username: postgres Pasword: YYYYYYYY
Patch usersrv deployment	Updated usersrv deployed ----- Agent Injector triggered ----- Init Container created in usersrv pod ----- Secrets files created ----- Init container ends ----- Service sources secret files & updates environment variables	Environment Variables: Username: postgres Pasword: YYYYYYYY	

Keeping application in sync

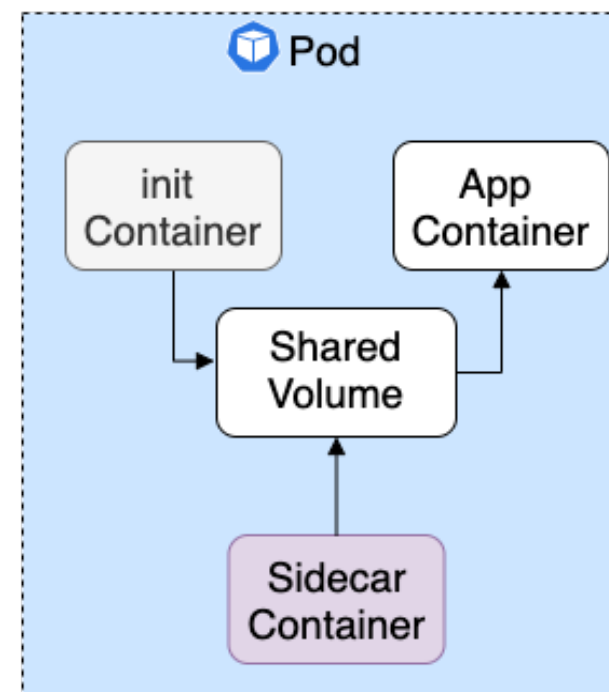


Init container great for set up secrets on startup

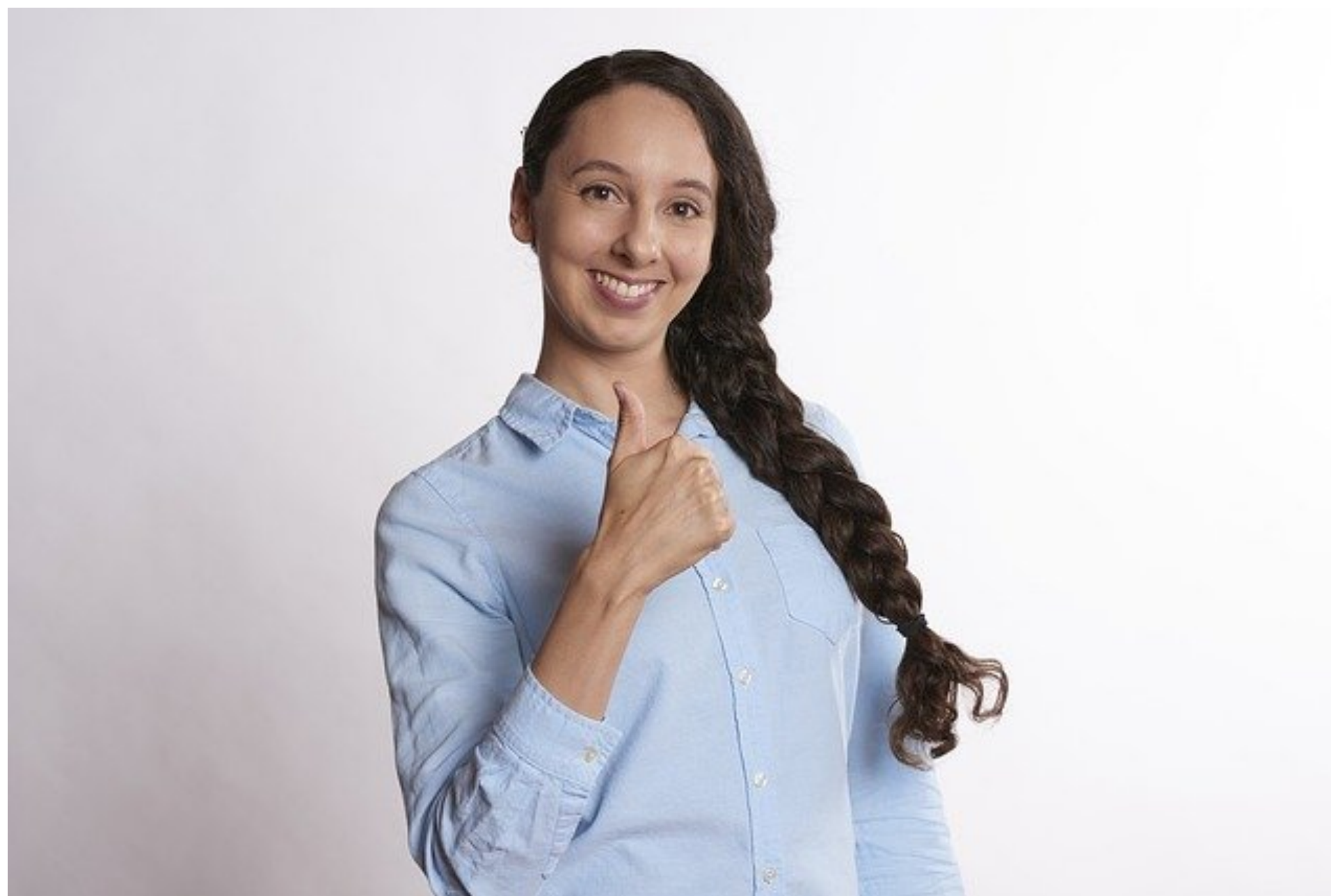
`agent-pre-populate-only: "true"`

Sidecar container used to update secrets overtime

`agent-pre-populate-only: "false"`



Jane after the enhancements





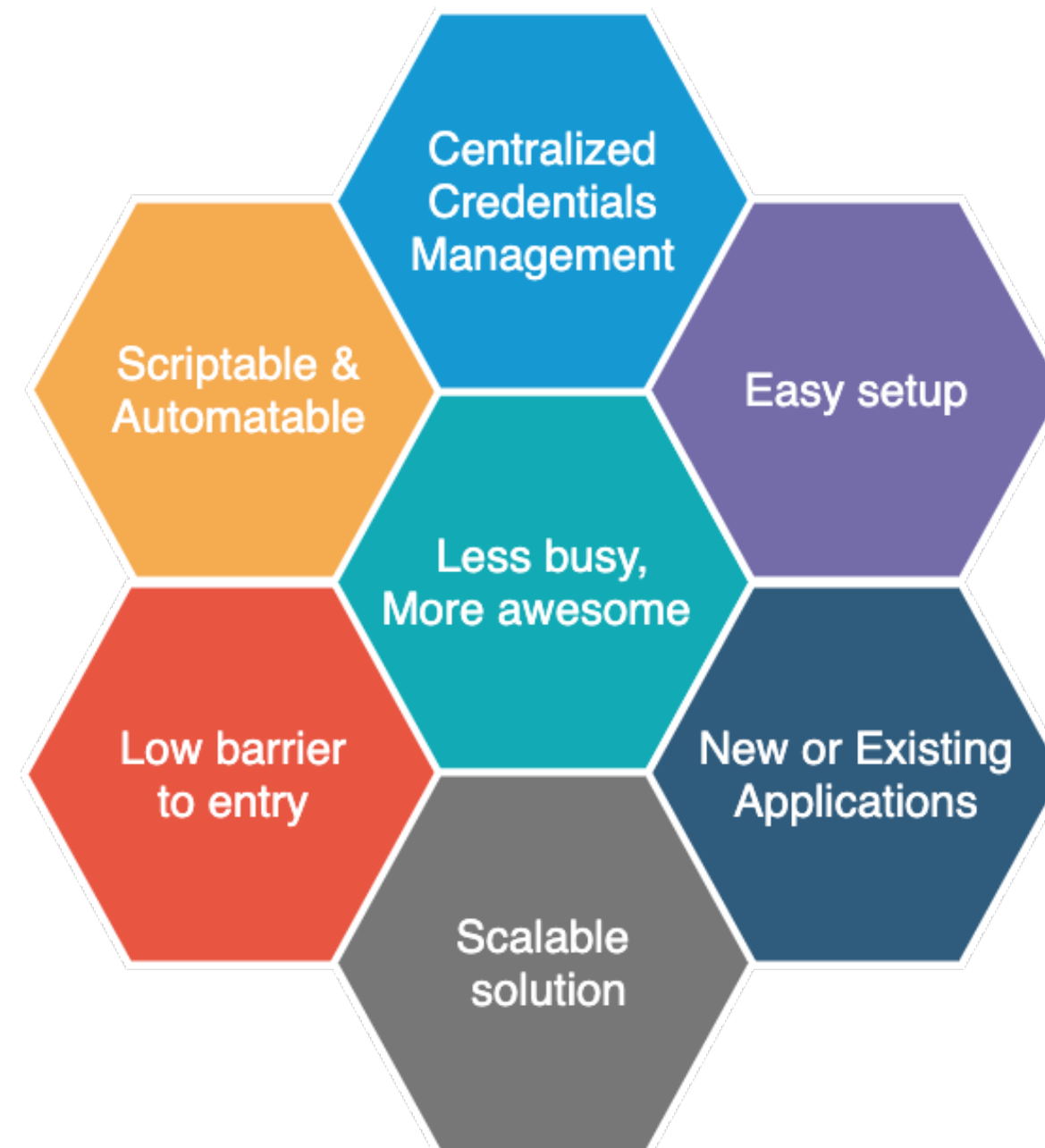
What else could Jane implement ?



- Automating with the CI/CD pipeline
- Integrate patch code into application deployments
- Database credentials management



Key take aways



Questions ?



Thanks to the Hashicorp for organizing this conference!



Appendix



Photos

 Image by [Alberto Galvis](#) from [Pixabay](#)

 Image by [Robin Higgins](#) from [Pixabay](#)

 Image by [Robin Higgins](#) from [Pixabay](#)

 Image by [Robin Higgins](#) from [Pixabay](#)

 Photo by [Campaign Creators](#) on [Unsplash](#)

 Image by [Mohamed Hassan](#) from [Pixabay](#)

Photos



Photo by [AbsolutVision](#) on [Unsplash](#)



Photo by [Markus Spiske](#) on [Unsplash](#)