Mirai Solutions

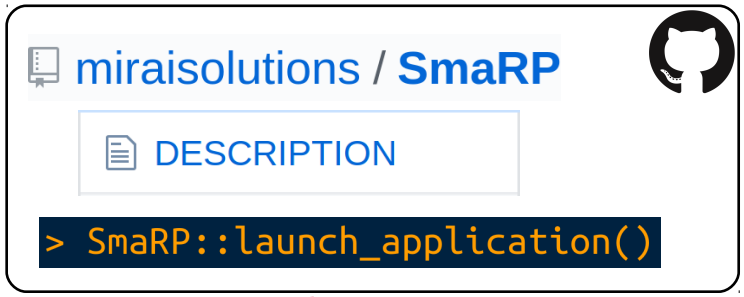
smarter analytics - better decisions

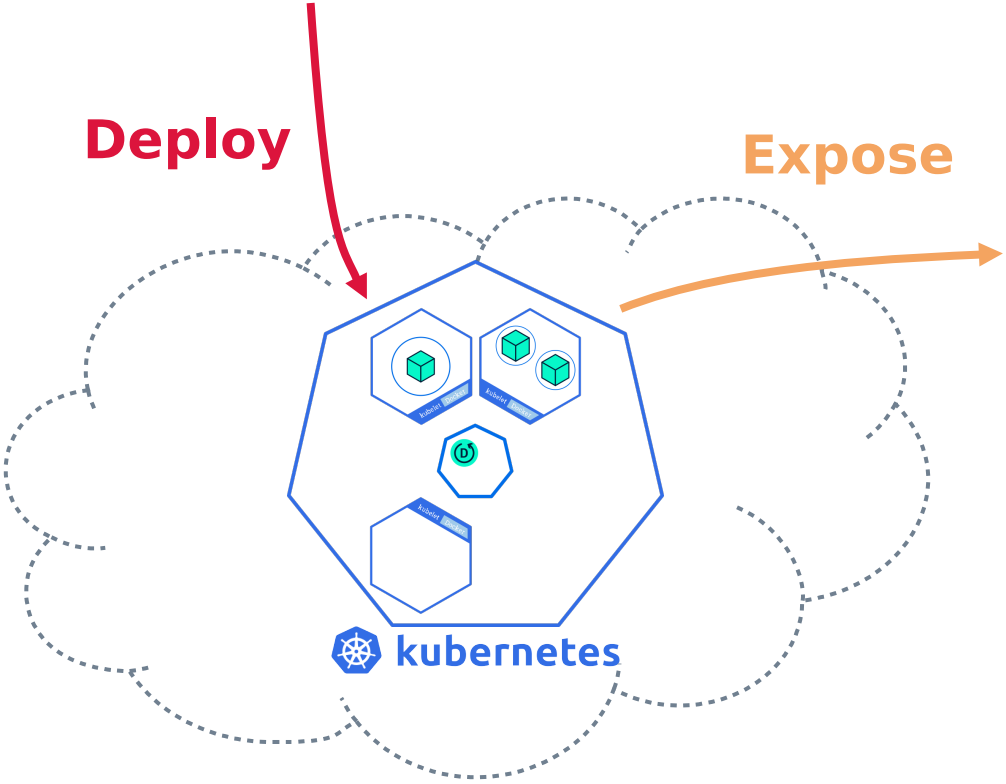
Shiny app deployment and integration into a custom website gallery

UseR! Toulouse, July 10th 2019

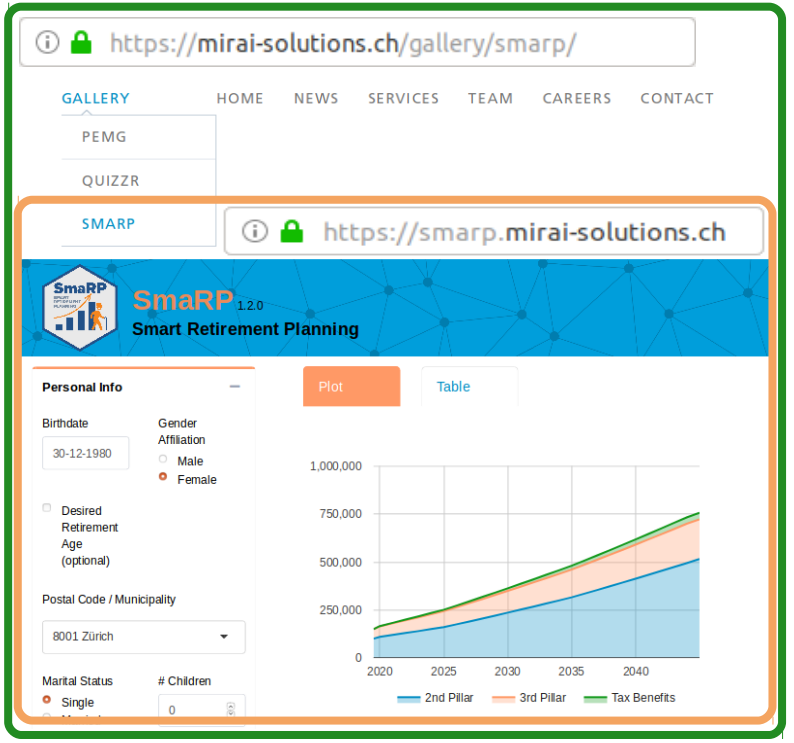
Riccardo Porreca
Roland Schmid

Shiny app: Deploy → Expose → Embed

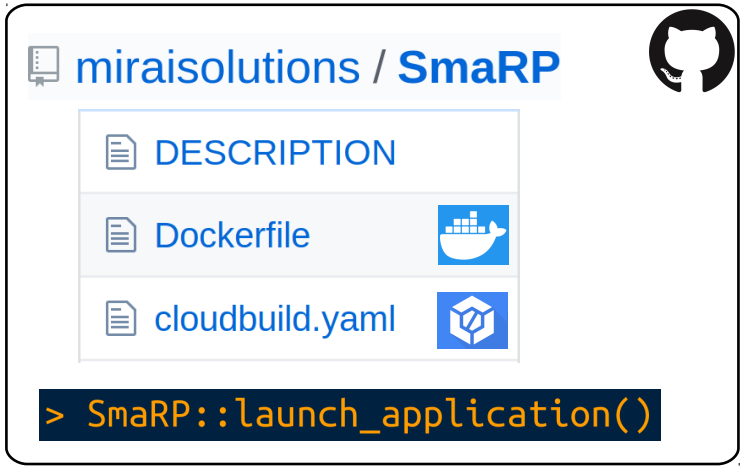


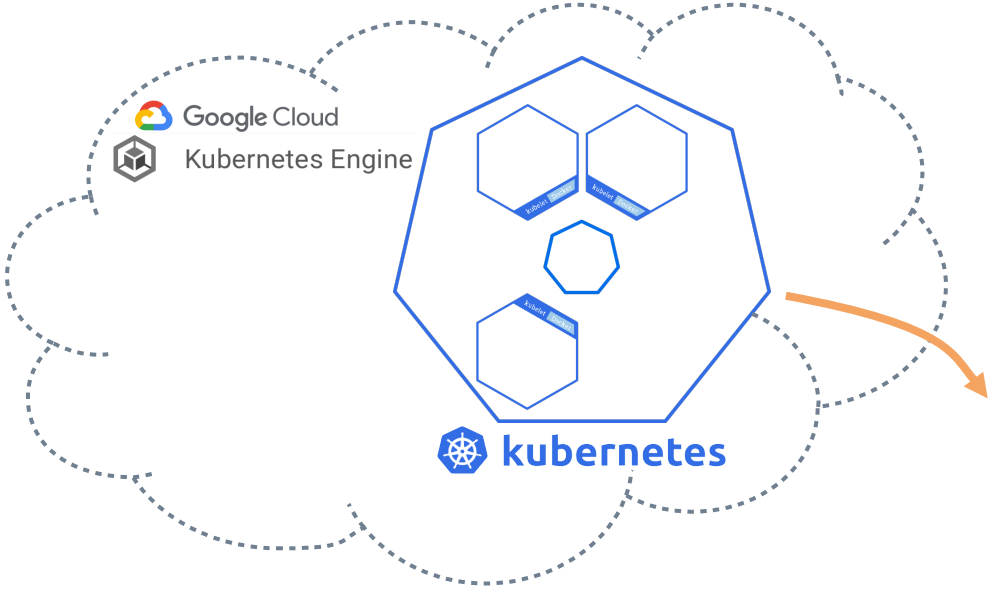


Embed

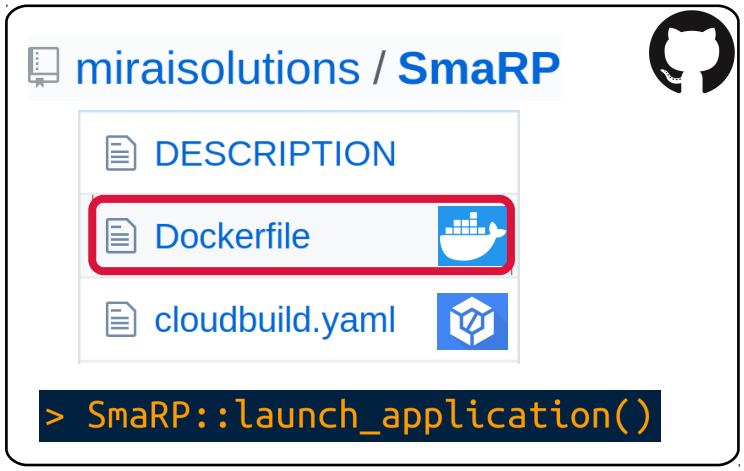


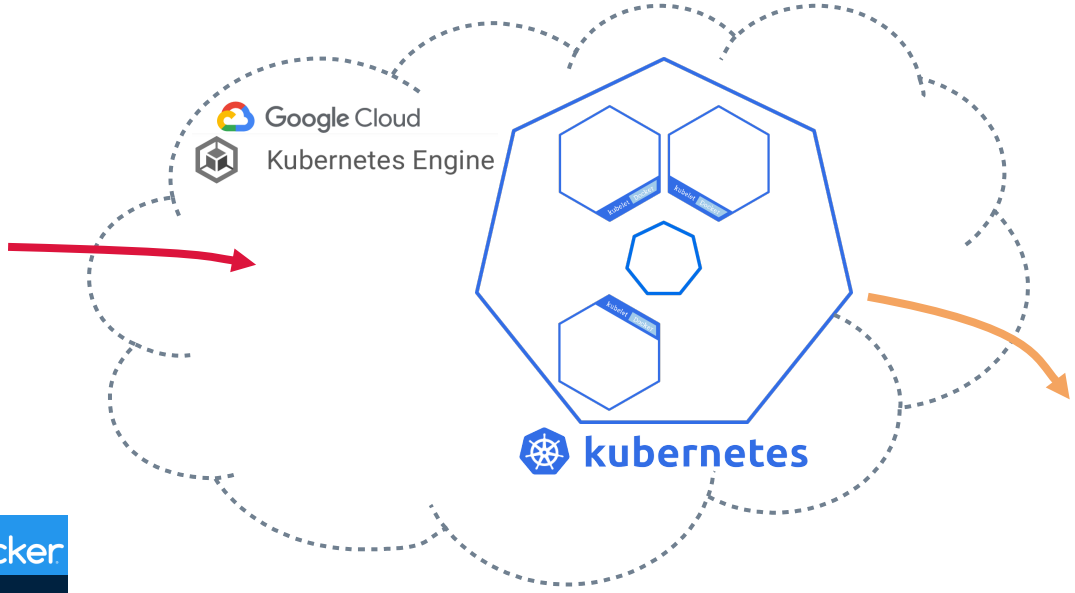
Docker-based deployment





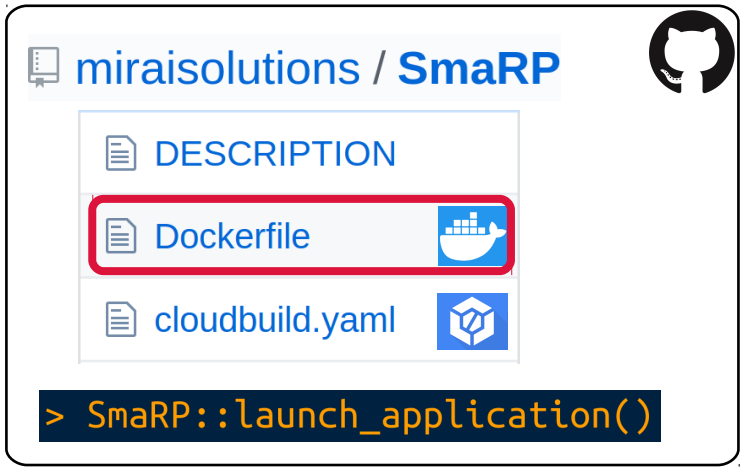
Docker-based deployment

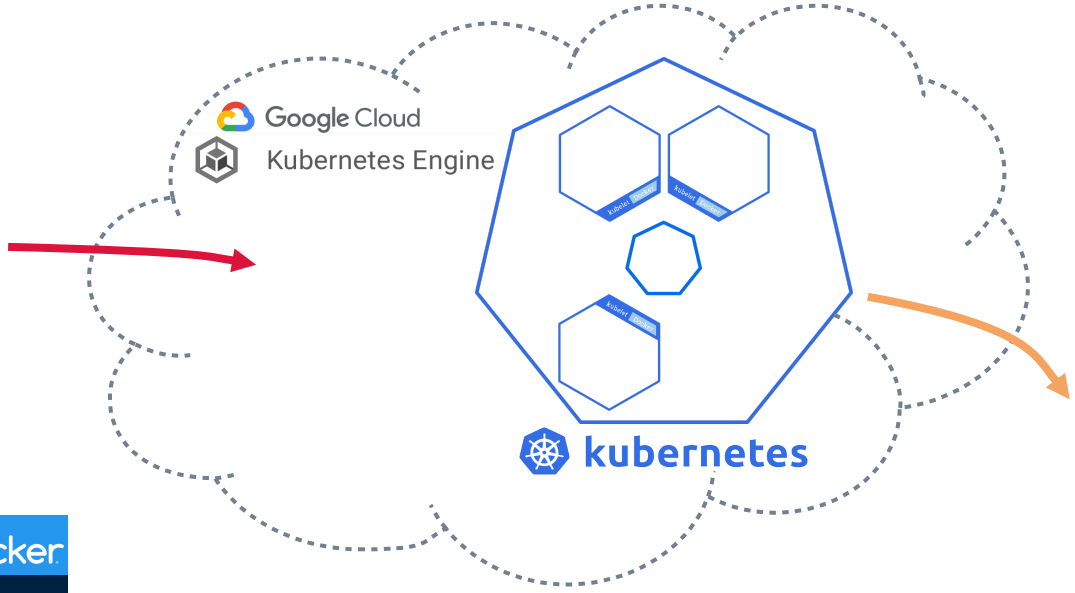





```
FROM rocker/r-ver:3.5.3
## Install required dependencies
RUN apt-get update \
  && apt-get install -y --no-install-recommends \
[...]
## Install major fixed R dependencies
RUN install2.r --error remotes shiny dplyr rmarkdown
## Copy the app to the image
ENV MARP=/tmp/SmaRP
COPY . $MARP
## Install SmaRP
RUN R -e "remotes::install_local('$MARP')" \
  && rm -rf $MARP
## Set host and port
RUN echo "options(shiny.port = 80, shiny.host = '0.0.0.0')" \
  >> /usr/local/lib/R/etc/Rprofile.site
EXPOSE 80
CMD ["R", "-e", "SmaRP::launch_application()"]
```

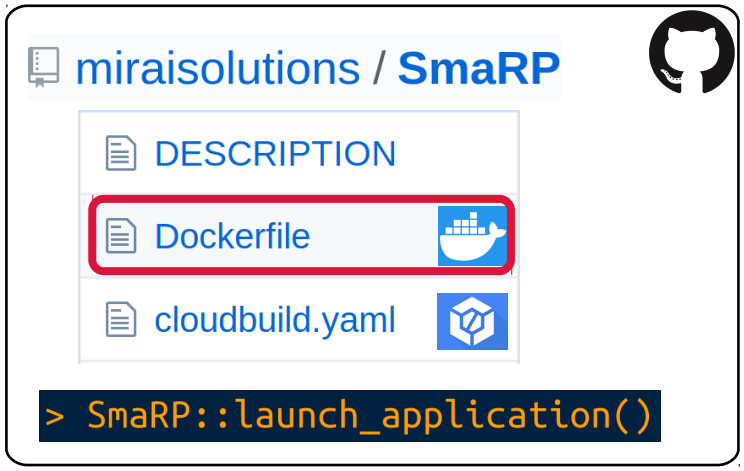
Docker-based deployment

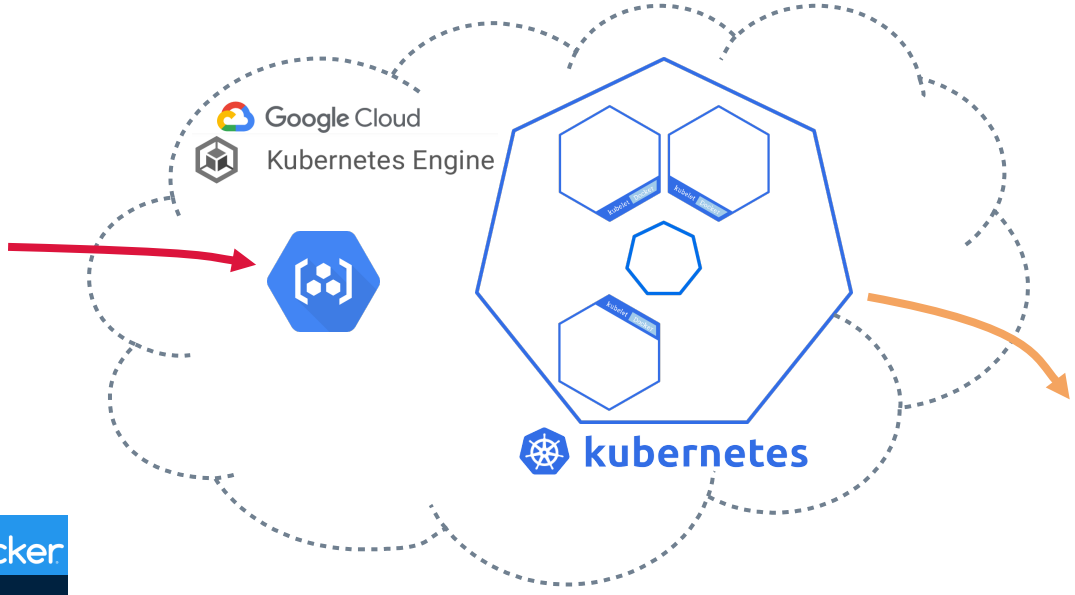





```
FROM rocker/r-ver:3.5.3
## Install required dependencies
RUN apt-get update \
  && apt-get install -y --no-install-recommends \
[...]
## Install major fixed R dependencies
RUN install2.r --error remotes shiny dplyr rmarkdown
## Copy the app to the image
ENV MARP=/tmp/SmaRP
COPY . $MARP
## Install SmaRP
RUN R -e "remotes::install_local('$MARP')" \
  && rm -rf $MARP
## Set host and port
RUN echo "options(shiny.port = 80, shiny.host = '0.0.0.0')" \
  >> /usr/local/lib/R/etc/Rprofile.site
EXPOSE 80
CMD ["R", "-e", "SmaRP::launch_application()"]
```

Docker-based deployment





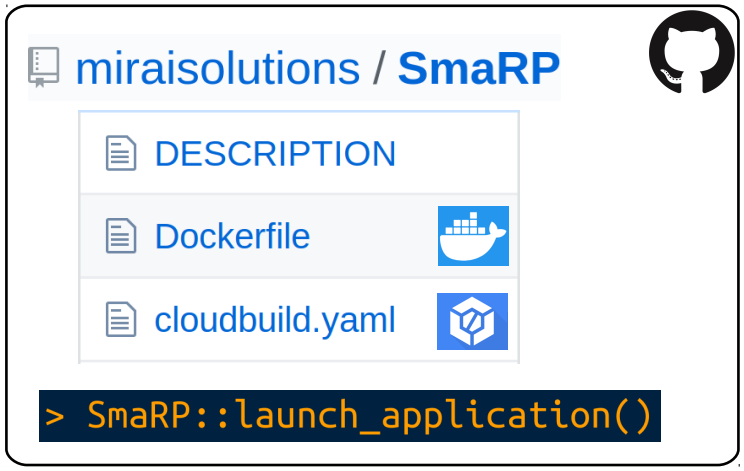

```
FROM rocker/r-ver:3.5.3
## Install required dependencies
RUN apt-get update \
  && apt-get install -y --no-install-recommends \
[...]
## Install major fixed R dependencies
RUN install2.r --error remotes shiny dplyr rmarkdown
## Copy the app to the image
ENV MARP=/tmp/SmaRP
COPY . $MARP
## Install SmaRP
RUN R -e "remotes::install_local('$MARP')" \
  && rm -rf $MARP
## Set host and port
RUN echo "options(shiny.port = 80, shiny.host = '0.0.0.0')" \
  >> /usr/local/lib/R/etc/Rprofile.site
EXPOSE 80
CMD ["R", "-e", "SmaRP::launch_application()"]
```

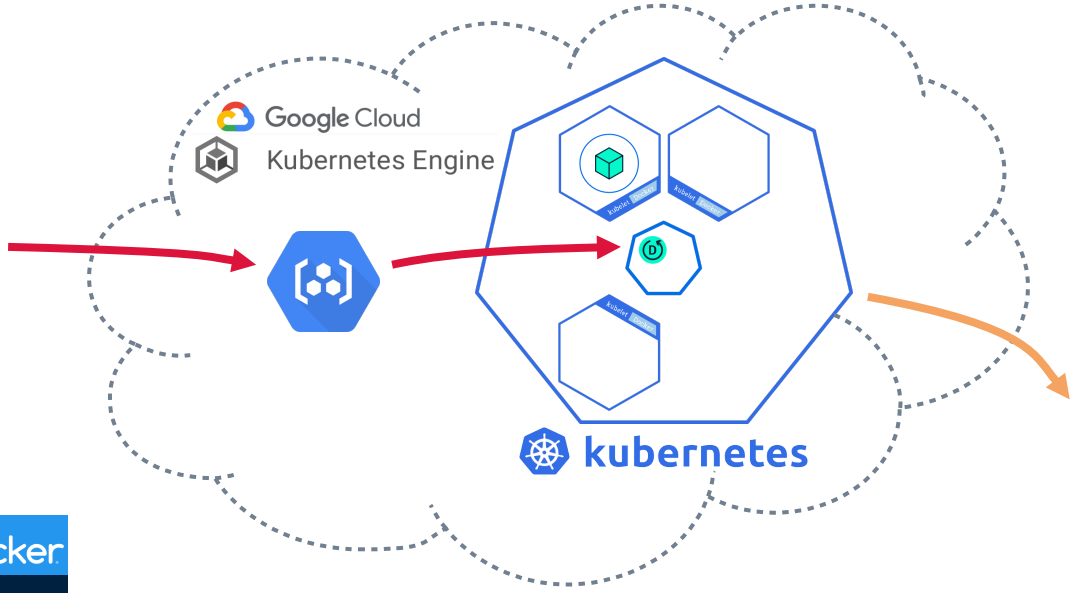
Build docker image and push to Google Container Registry

```
$ docker build -t gcr.io/<PROJ>/smarp:v0.1.0 .
$ docker push gcr.io/<PROJ>/smarp:v0.1.0
```

Docker-based deployment

Mirai Solutions
smarter analytics - better decisions






```
FROM rocker/r-ver:3.5.3
## Install required dependencies
RUN apt-get update \
  && apt-get install -y --no-install-recommends \
[...]
## Install major fixed R dependencies
RUN install2.r --error remotes shiny dplyr rmarkdown
## Copy the app to the image
ENV MARP=/tmp/SmaRP
COPY . $MARP
## Install SmaRP
RUN R -e "remotes::install_local('$MARP')" \
  && rm -rf $MARP
## Set host and port
RUN echo "options(shiny.port = 80, shiny.host = '0.0.0.0')" \
  >> /usr/local/lib/R/etc/Rprofile.site
EXPOSE 80
CMD ["R", "-e", "SmaRP::launch_application()"]
```

Build docker image and push to Google Container Registry

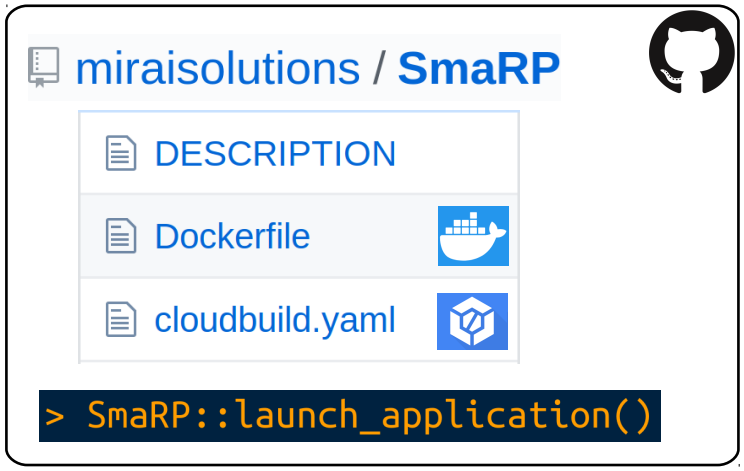
```
$ docker build -t gcr.io/<PROJ>/smarp:v0.1.0 .
$ docker push gcr.io/<PROJ>/smarp:v0.1.0
```

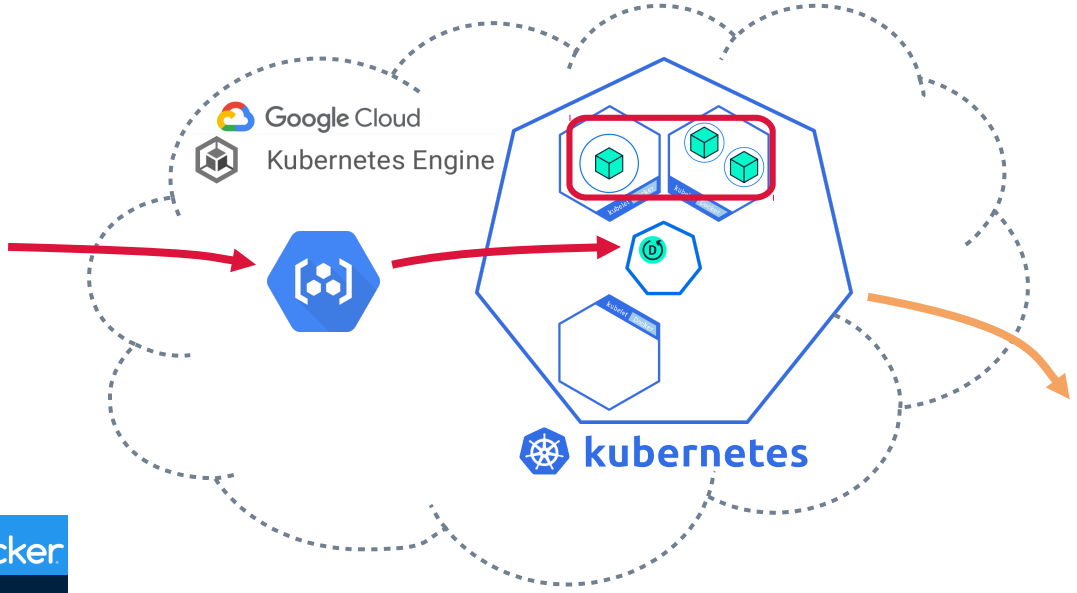
Create a deployment to the cluster

```
$ kubectl run smarp \
  --image=eu.gcr.io/<PROJ>/smarp:v0.1.0
```

Docker-based deployment

Mirai Solutions
smarter analytics - better decisions






```dockerfile
FROM rocker/r-ver:3.5.3
## Install required dependencies
RUN apt-get update \
  && apt-get install -y --no-install-recommends \
[...]
## Install major fixed R dependencies
RUN install2.r --error remotes shiny dplyr rmarkdown
## Copy the app to the image
ENV MARP=/tmp/SmaRP
COPY . $MARP
## Install SmaRP
RUN R -e "remotes::install_local('$MARP')" \
  && rm -rf $MARP
## Set host and port
RUN echo "options(shiny.port = 80, shiny.host = '0.0.0.0')" \
  >> /usr/local/lib/R/etc/Rprofile.site
EXPOSE 80
CMD ["R", "-e", "SmaRP::launch_application()"]
```

Build docker image and push to Google Container Registry

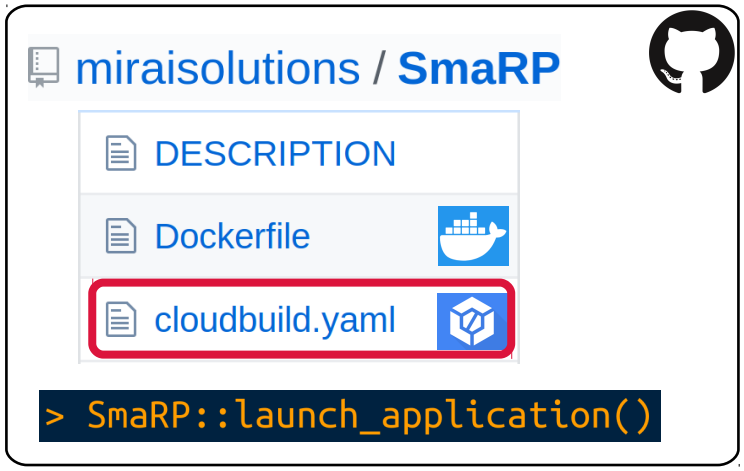
```
$ docker build -t gcr.io/<PROJ>/smarp:v0.1.0 .
$ docker push gcr.io/<PROJ>/smarp:v0.1.0
```

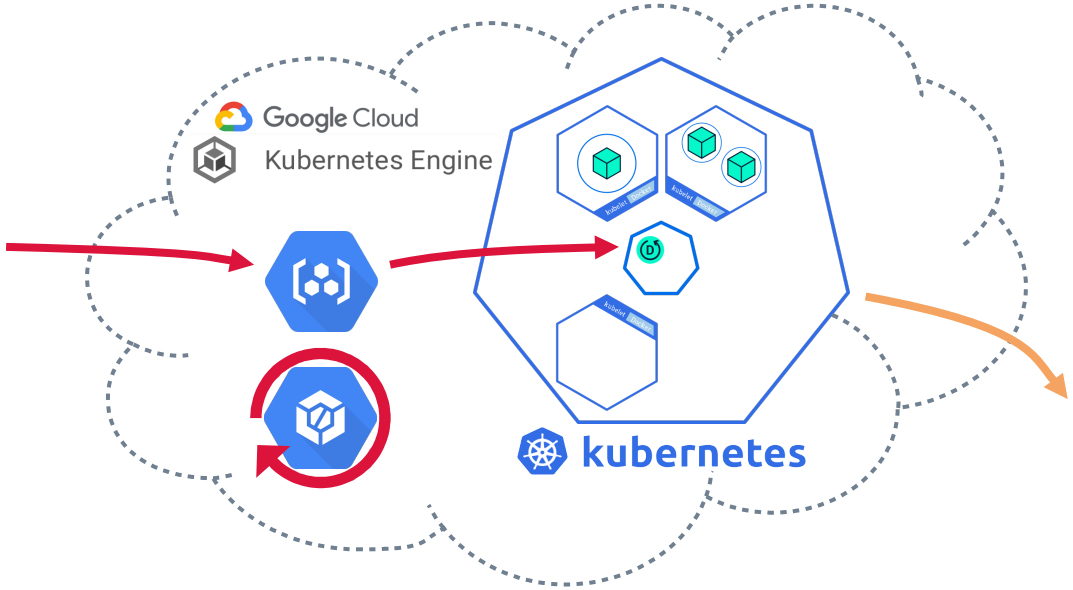
Create a deployment to the cluster

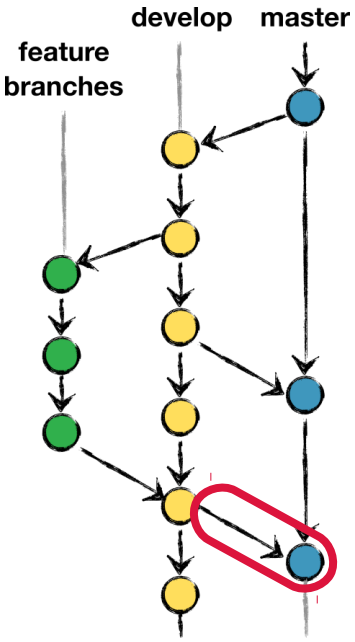
```
$ kubectl run smarp \
  --image=eu.gcr.io/<PROJ>/smarp:v0.1.0
```

Manage replicas of the app as *Pods*

Continuous delivery pipeline







GitFlow

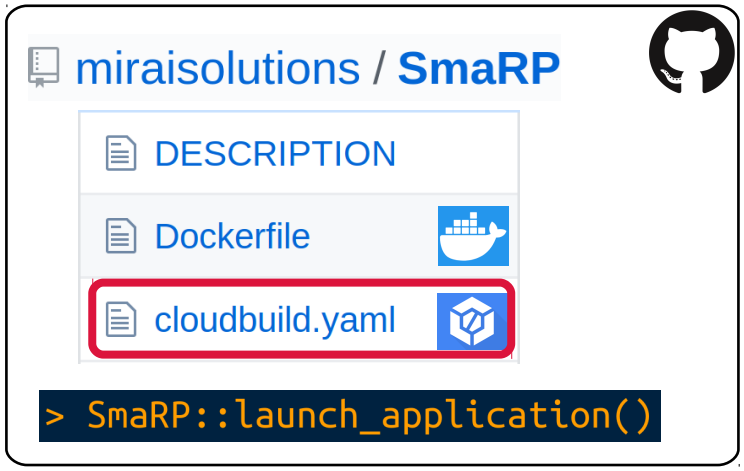
+

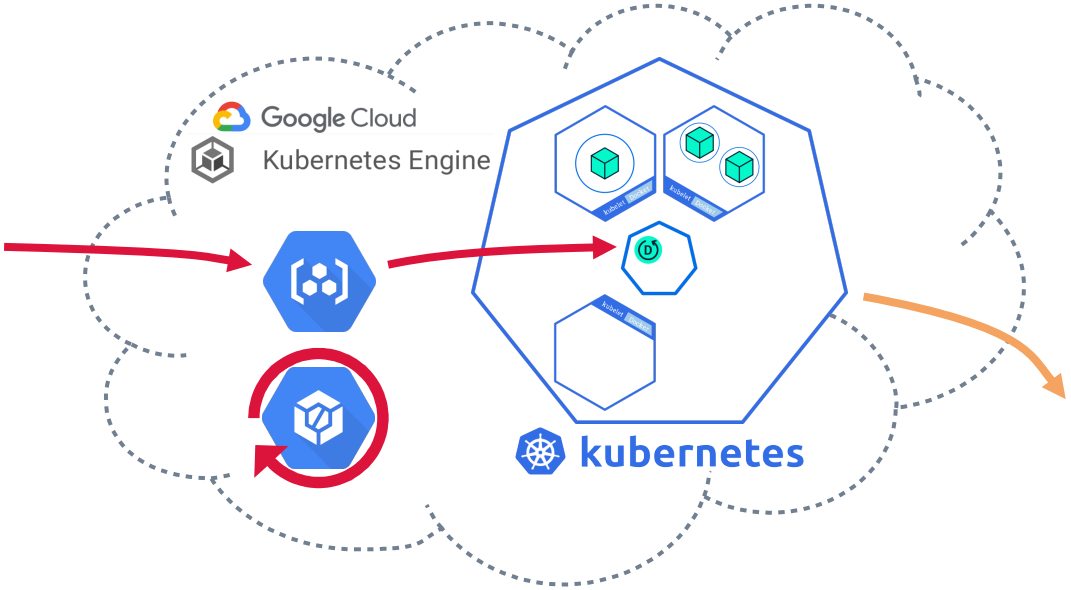
Cloud Build

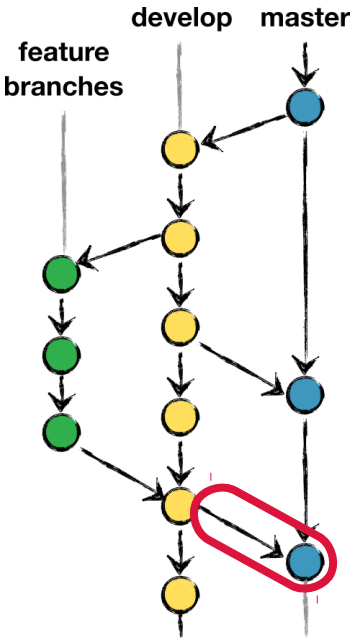
miraisolutions/SmaRP — GitHub

Name	Type	Filter	Build configuration
Push to master branch	Push to branch	master	cloudbuild.yaml

Continuous delivery pipeline







GitFlow

+

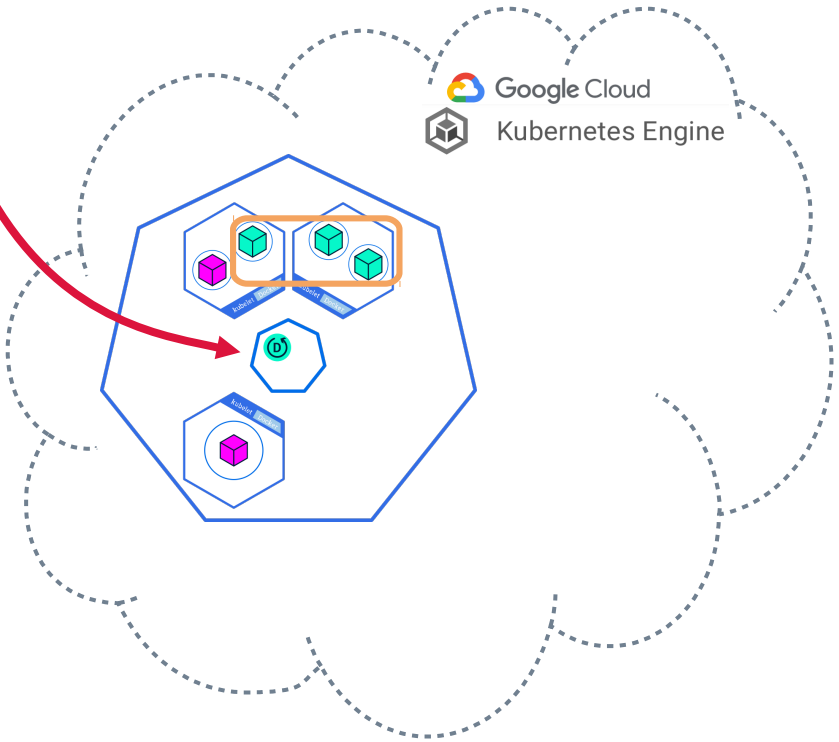
Cloud Build

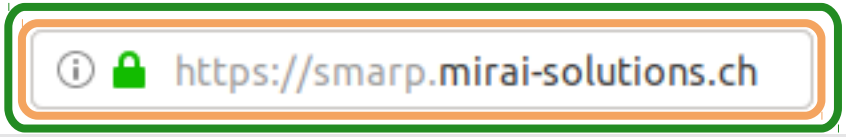
miraisolutions/SmaRP — GitHub

Name	Type	Filter	Build configuration
Push to master branch	Push to branch	master	cloudbuild.yaml

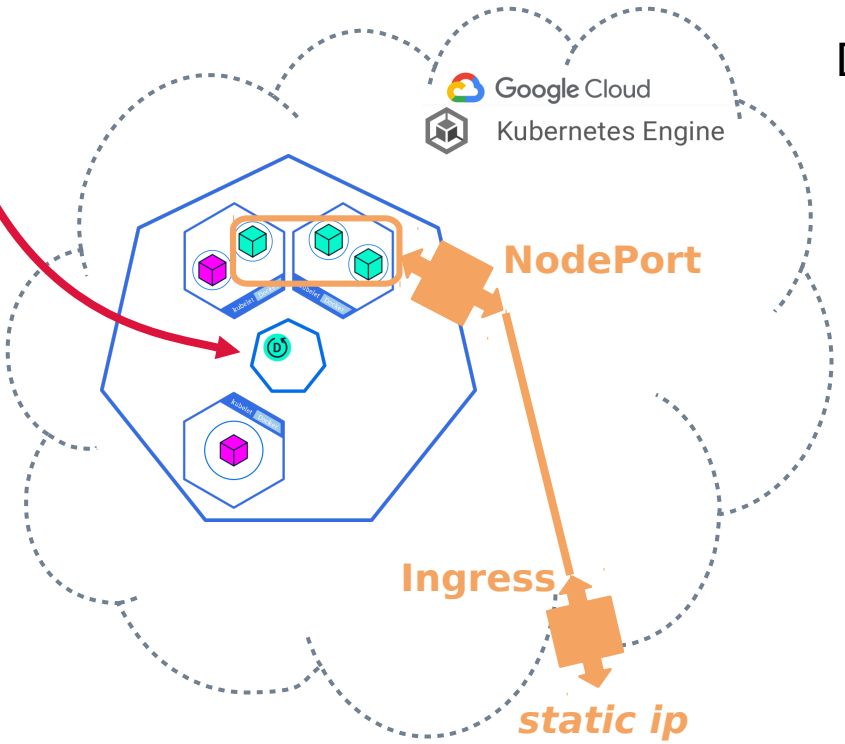
```yaml
# cloudbuild.yaml
steps:
[...]
- name: 'gcr.io/cloud-builders/docker'
  args: ['build',
         '-t', 'eu.gcr.io/<PROJ>/smarp:$COMMIT_SHA',
         '.']
- name: 'gcr.io/cloud-builders/docker'
  args: ['push', 'eu.gcr.io/<PROJ>/smarp:$COMMIT_SHA']
[...]
- name: 'gcr.io/cloud-builders/kubectl'
  args: ['set', 'image', 'deployment', 'smarp',
         'smarp=eu.gcr.io/<PROJ>/smarp:$COMMIT_SHA']
[...]
```

Expose the app to the web





Expose the app to the web



Define and create kubernetes resources

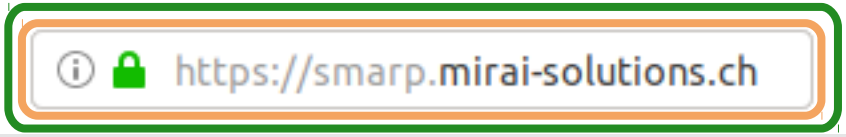
NodePort

Abstraction layer for a set of ***Pods*** (replicas of the app), ***backend*** for the app exposed as ***service***

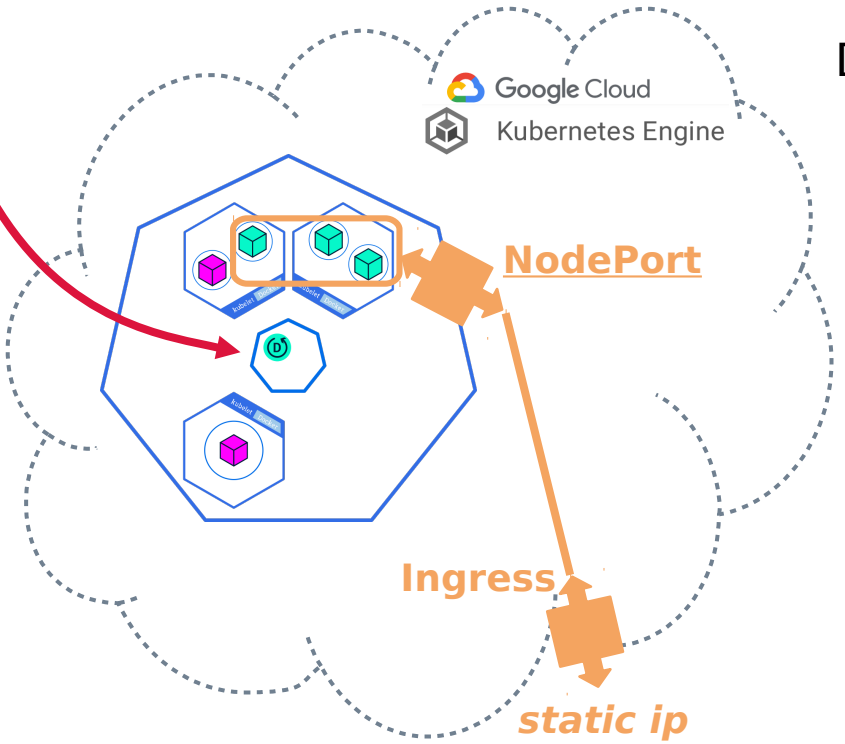
Requests from outside the cluster forwarded the running member pods

Ingress

Rules for routing external load-balanced HTTP(S) traffic to the NodePort service via an ***external ip address***



Expose the app to the web



Define and create kubernetes resources

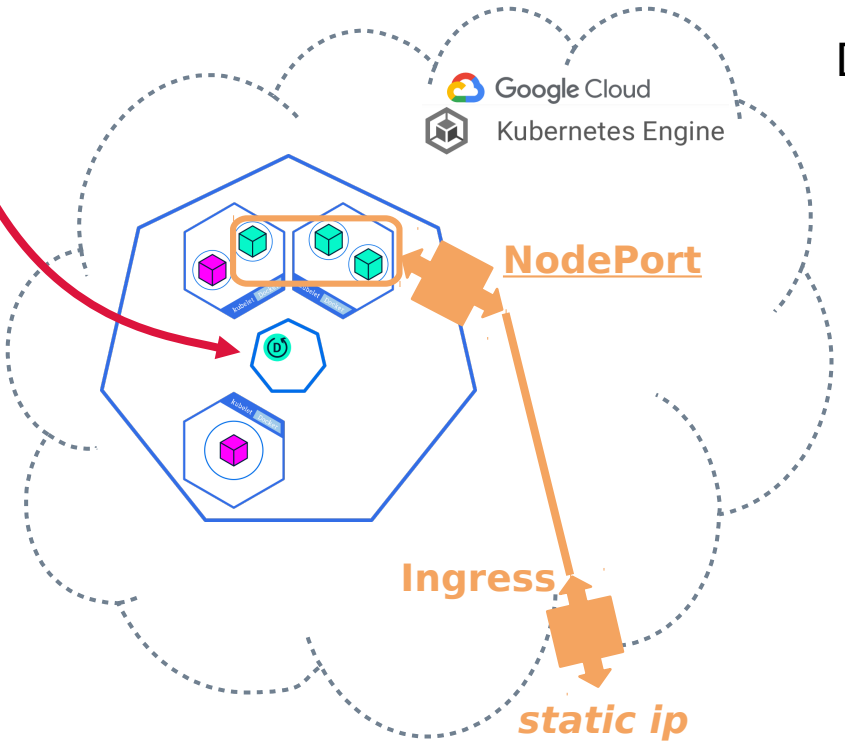
YAML manifest: NodePort service backend

```yaml
# smarp-backend.yaml
apiVersion: v1
kind: Service
metadata:
  labels:
    run: smarp
  name: smarp-backend
  annotations:
    beta.cloud.google.com/backend-config:
spec:
  type: NodePort
  selector:
    run: smarp
  ports:
  - port: 80
    targetPort: 80
```

```
$ kubectl apply -f smarp-backend.yaml
```

https://smarp.mirai-solutions.ch

Expose the app to the web



Define and create kubernetes resources

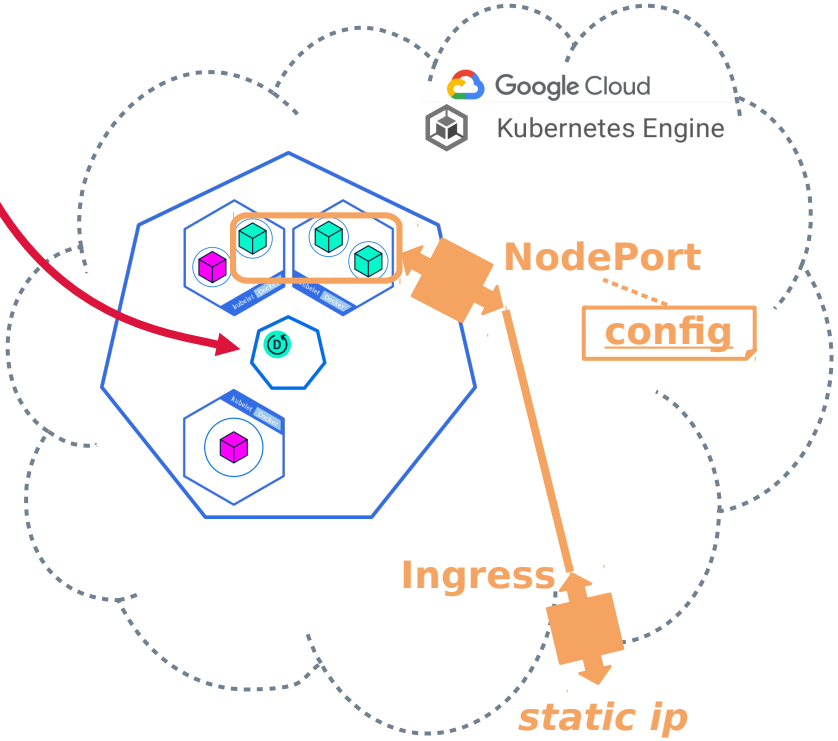
YAML manifest: NodePort service backend

```yaml
# smarp-backend.yaml
apiVersion: v1
kind: Service
metadata:
  labels:
    run: smarp
  name: smarp-backend
  annotations:
    [...]'{"ports": {"80":"smarp-backendconfig"}}'
spec:
  type: NodePort
  selector:
    run: smarp
  ports:
  - port: 80
    targetPort: 80
```

```
$ kubectl apply -f smarp-backend.yaml
```

https://smarp.mirai-solutions.ch

Expose the app to the web



Define and create kubernetes resources

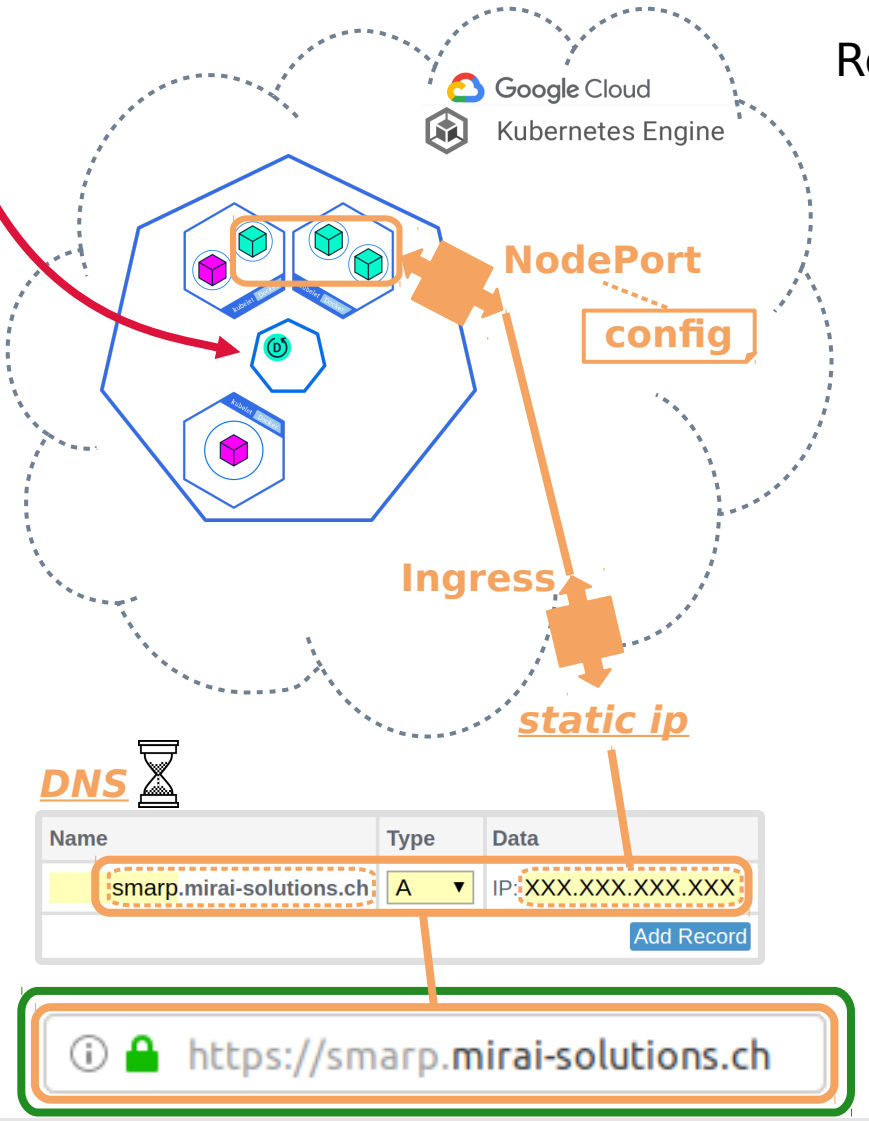
YAML manifest: **BackendConfig**

```yaml
# smarp-backendconfig.yaml
apiVersion: cloud.google.com/v1beta1
kind: BackendConfig
metadata:
  name: smarp-backendconfig
spec:
  # Shiny uses WebSockets, for which the default
  # max time a connection can live is only 30s,
  # not suitable to interactive apps
  timeoutSec: 10800 # 3h
  sessionAffinity:
    affinityType: "CLIENT_IP"
```

```
$ kubectl apply -f smarp-backendconfig.yaml
```

https://smarp.mirai-solutions.ch

Expose the app to the web



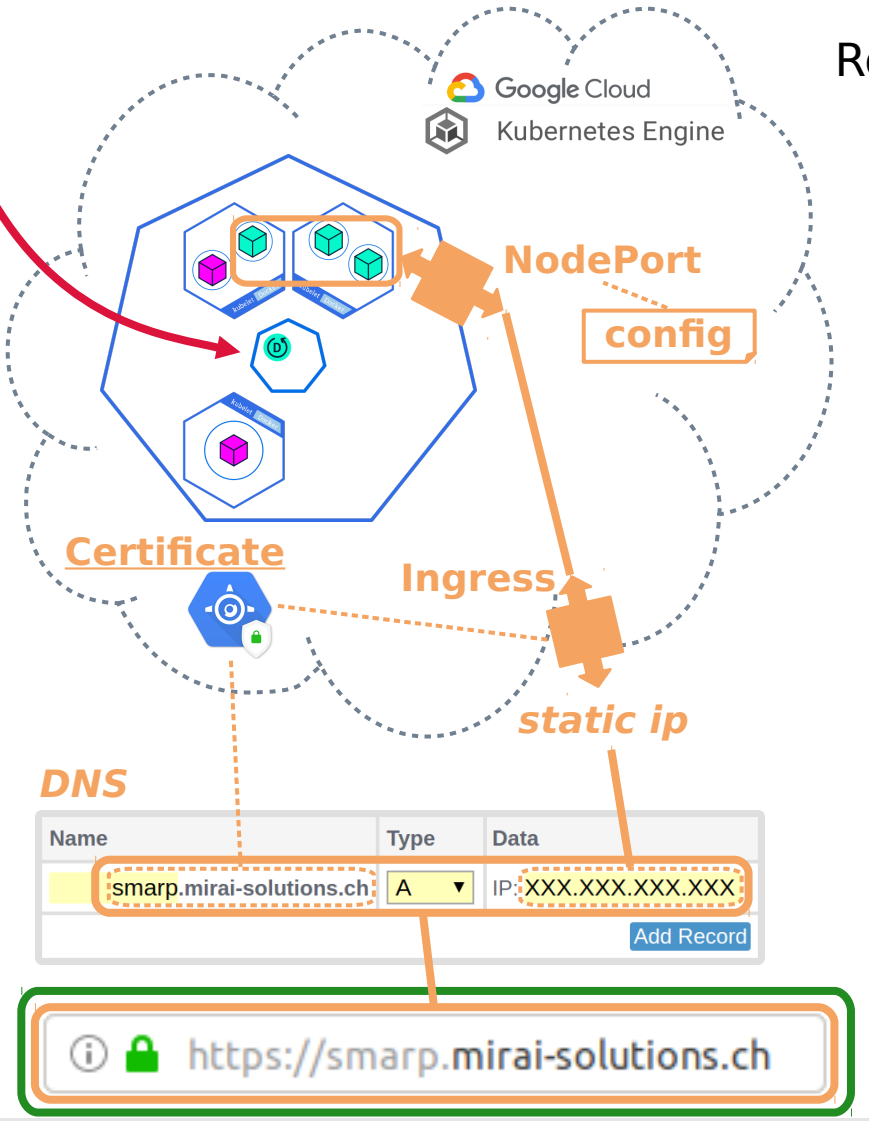
Route **HTTPS** traffic for a (sub)-**domain**

Static external **ip address**

```
$ gcloud compute addresses create \
    smarp-ip --global
$ gcloud compute addresses list smarp-ip
```

A-type **DNS record** for the domain →

Expose the app to the web



Route **HTTPS** traffic for a (sub)-**domain**

Static external **ip address**

```
$ gcloud compute addresses create \
    smarp-ip --global
$ gcloud compute addresses list smarp-ip
```

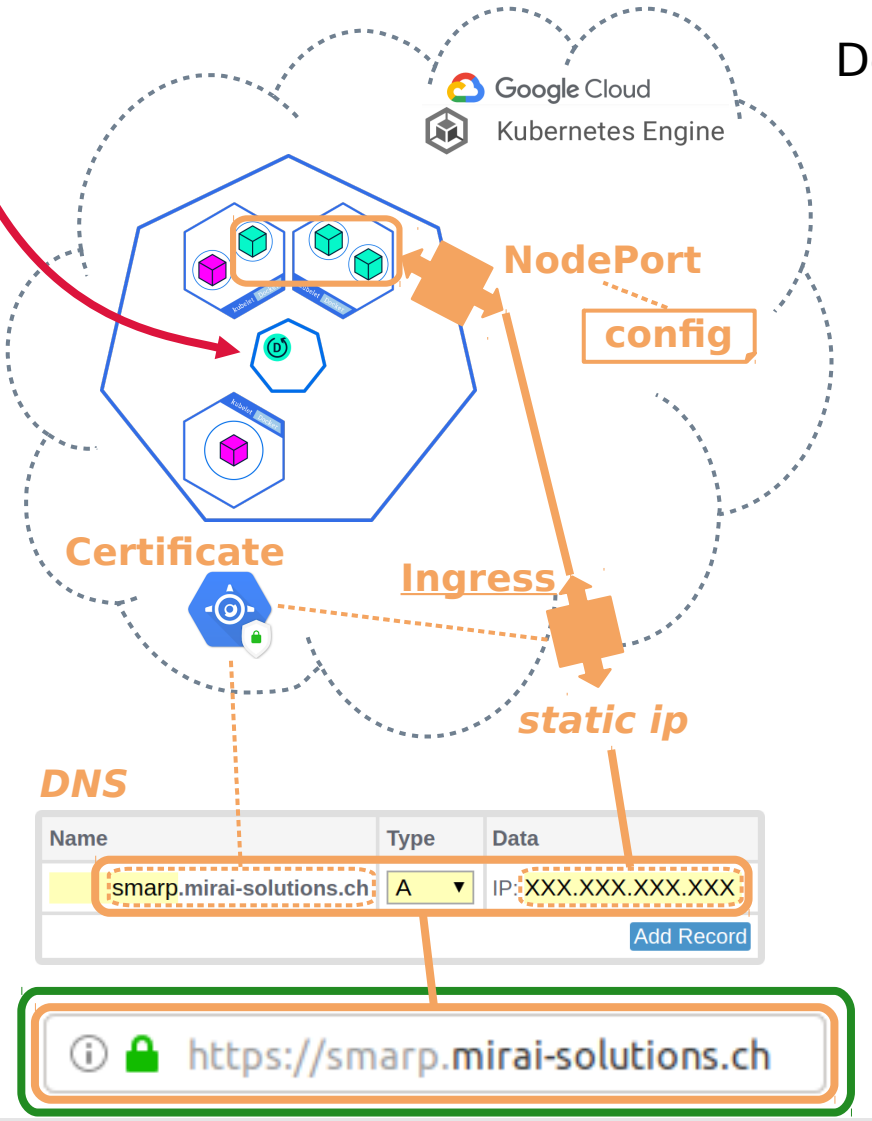
A-type **DNS record** for the domain →

Google-managed **TLS certificate** to enable HTTPS traffic

```
# smarp-certificate.yaml
apiVersion: networking.gke.io/v1beta1
kind: ManagedCertificate
metadata:
  name: smarp-certificate
spec:
  domains:
  - smarp.mirai-solutions.ch
```

```
$ kubectl apply -f smarp-certificate.yaml
```

Expose the app to the web



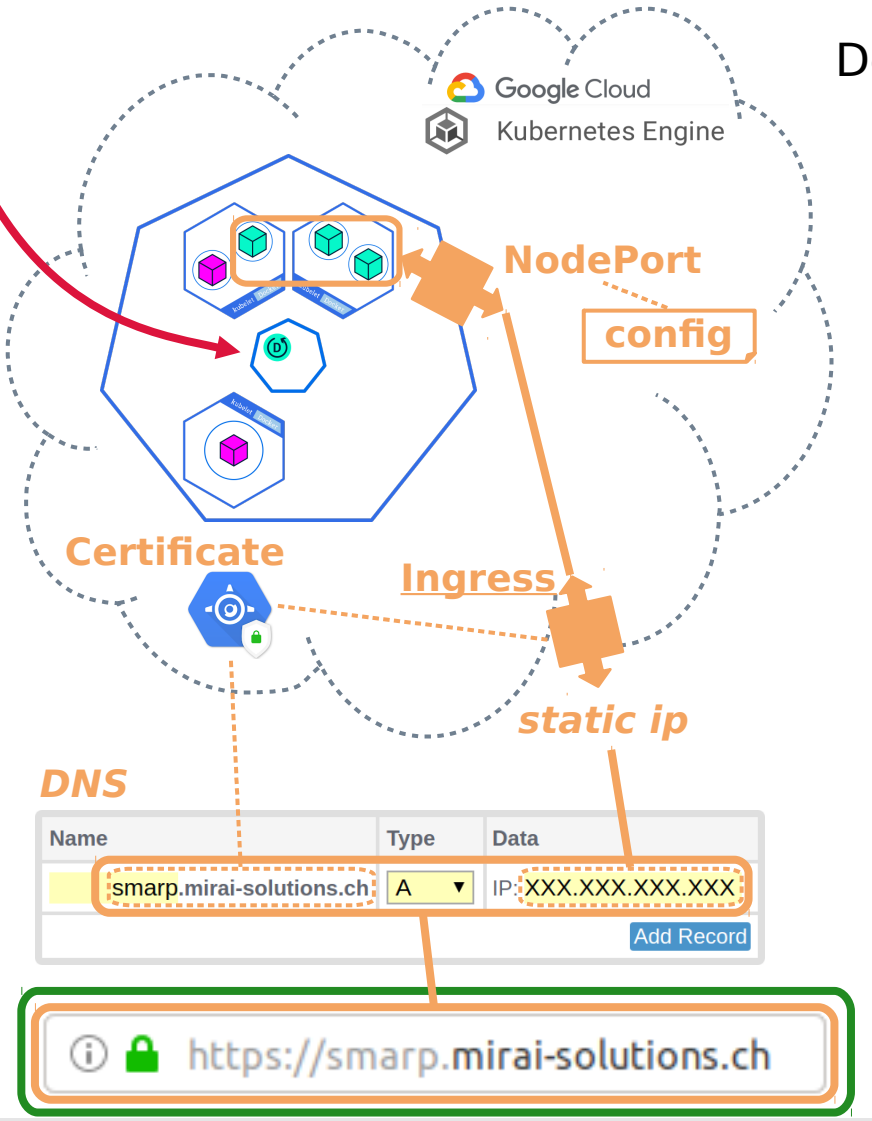
Define and create kubernetes resources

YAML manifest: **Ingress**

```yaml
# smarp-ingress.yaml
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: smarp-ingress
  annotations:
    kubernetes.io/ingress.global-static-ip-name:
    networking.gke.io/managed-certificates:
    kubernetes.io/ingress.allow-http:
spec:
  backend:
    serviceName: smarp-backend
    servicePort: 80
```

```
$ kubectl apply -f smarp-ingress.yaml
```

Expose the app to the web



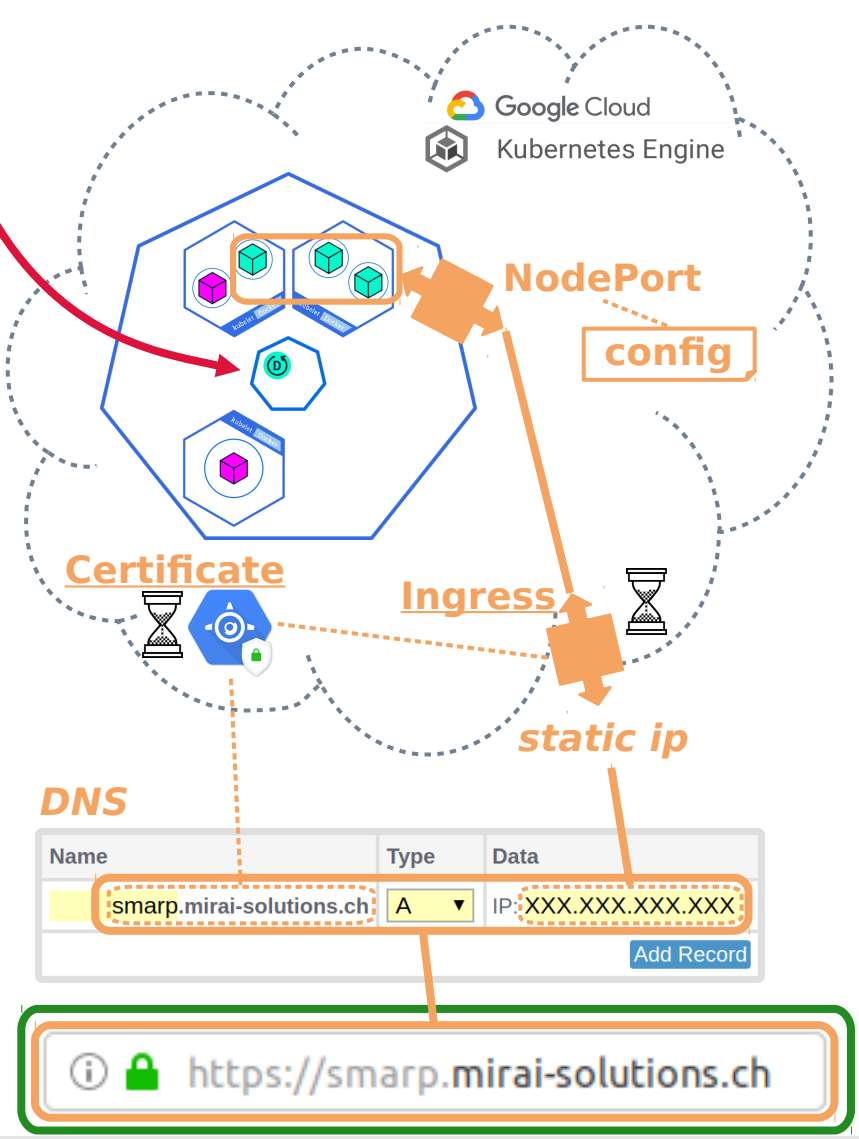
Define and create kubernetes resources

YAML manifest: **Ingress**

```yaml
# smarp-ingress.yaml
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: smarp-ingress
  annotations:
    [...]/ingress.global-static-ip-name: smarp-ip
    [...]/managed-certificates: smarp-certificate
    [...]/ingress.allow-http: "false"
spec:
  backend:
    serviceName: smarp-backend
    servicePort: 80
```

```
$ kubectl apply -f smarp-ingress.yaml
```

Expose the app to the web



Provisioning the **Certificate** and setting up forwarding rules for the **Ingress** takes **time** (tens of minutes)

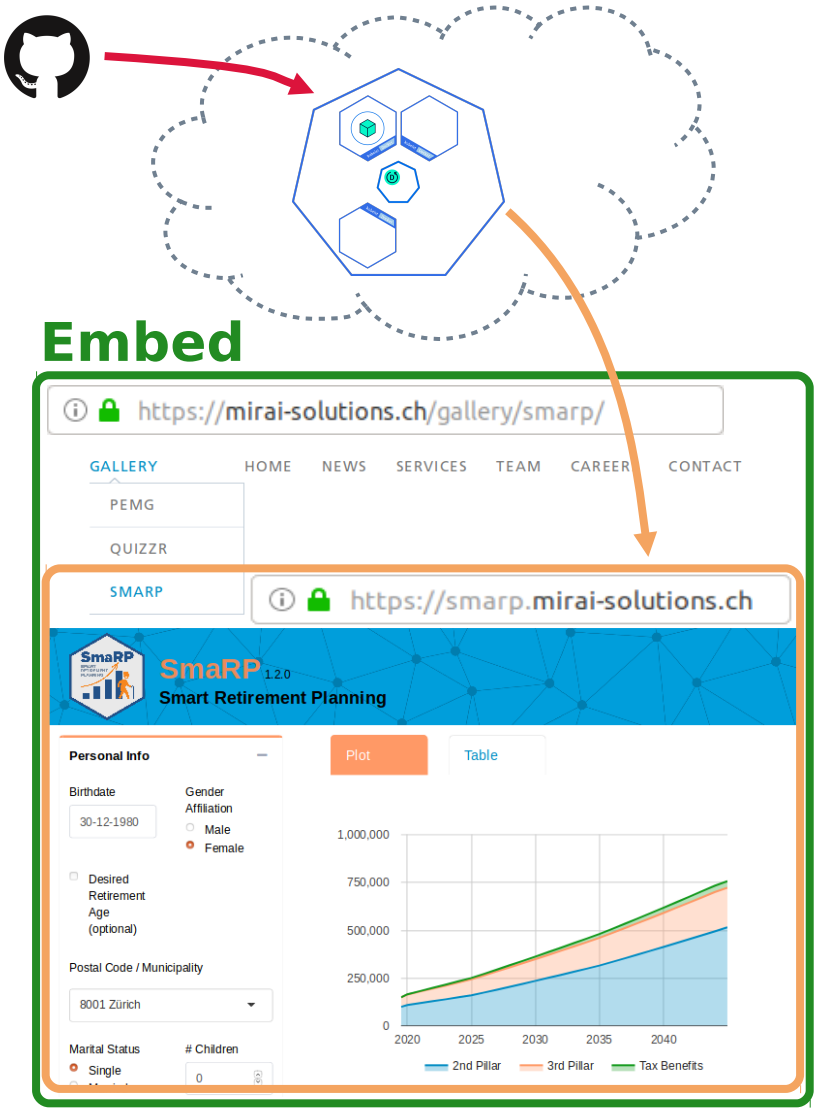
Monitor the progress

```
$ watch kubectl describe \
    ingress smarp-ingress
```

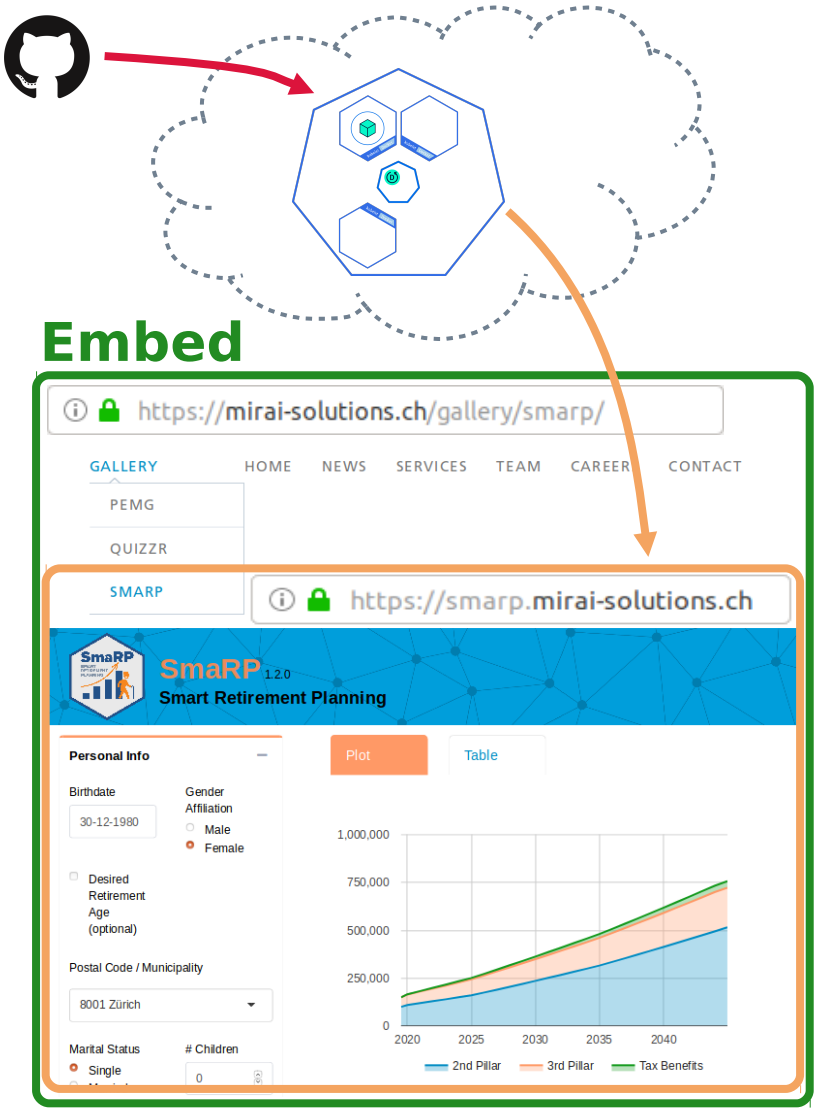
```
$ watch kubectl describe \
    managedcertificate smarp-certificate
```

Visit the **exposed app**
→ https://smarp.mirai-solutions.ch

Embed the app in a website page



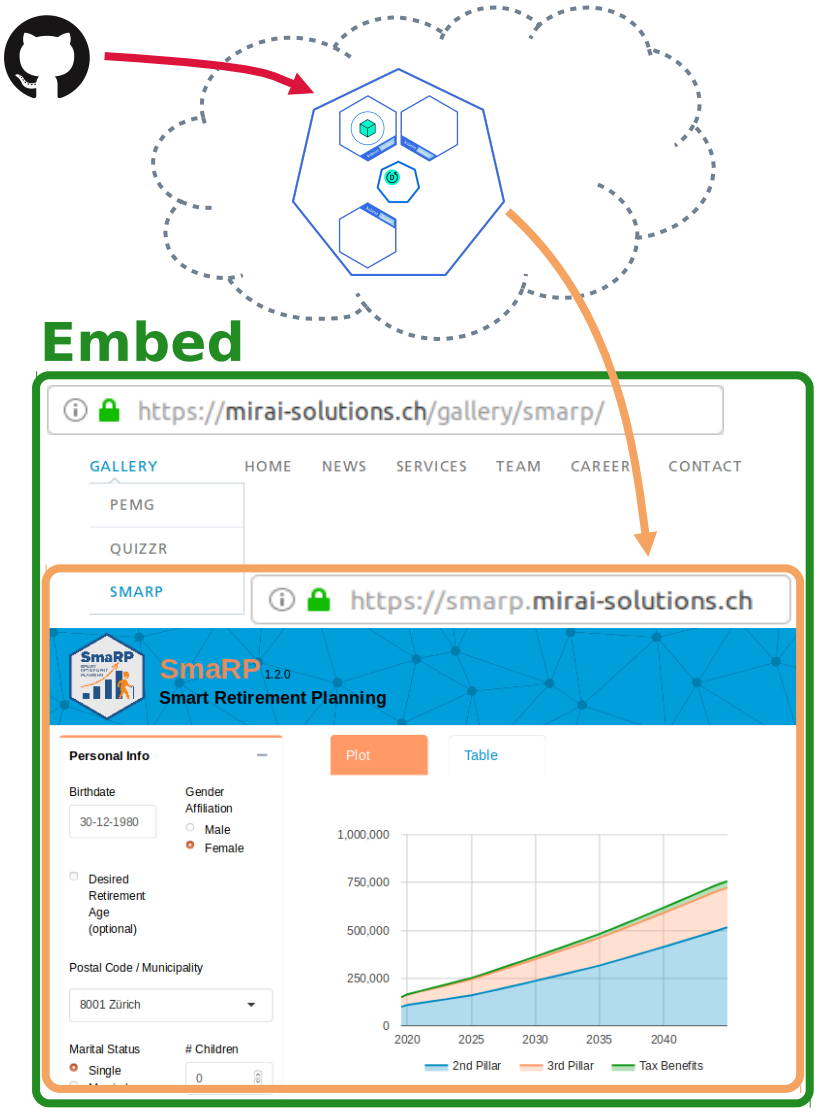
Embed the app in a website page



<iframe> tag to embed another HTML document within a hosting document

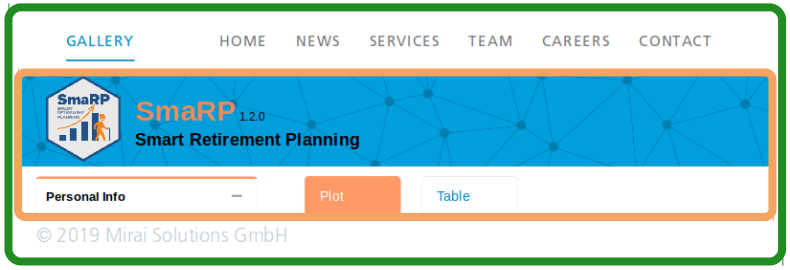
```
<iframe
 src="https://smarp.mirai-solutions.ch"
 scrolling="no" frameborder="0"
 style="min-width: 100%;">
</iframe>
```

Embed the app in a website page



<iframe> tag to embed another HTML document within a hosting document

```
<iframe
 src="https://smarp.mirai-solutions.ch"
 scrolling="no" frameborder="0"
 style="min-width: 100%;">
</iframe>
```



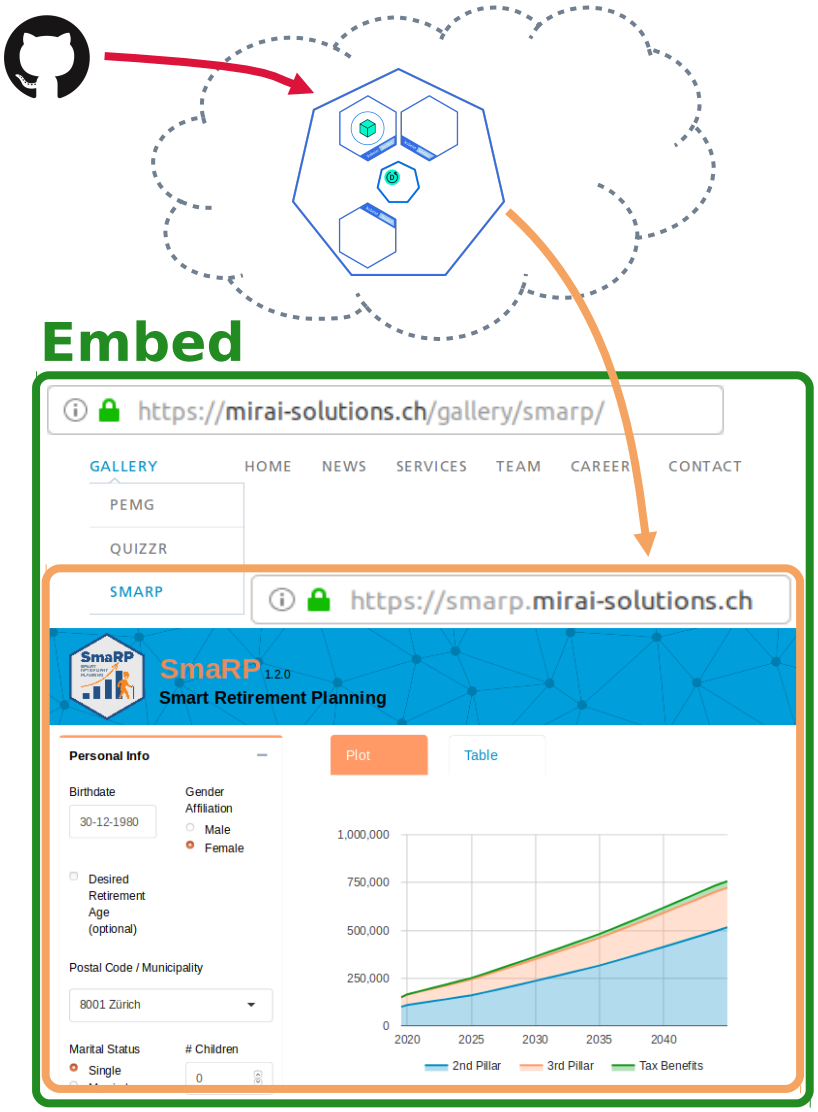
The **<iframe>** is not responsive to the height of its content

=> **iFrame Resizer** JavaScript library

https://github.com/davidjbradshaw/iframe-resizer

https://cdnjs.com/libraries/iframe-resizer

Embed the app in a website page



iFrame Resizer JavaScript library
<https://cdnjs.com/libraries/iframe-resizer>

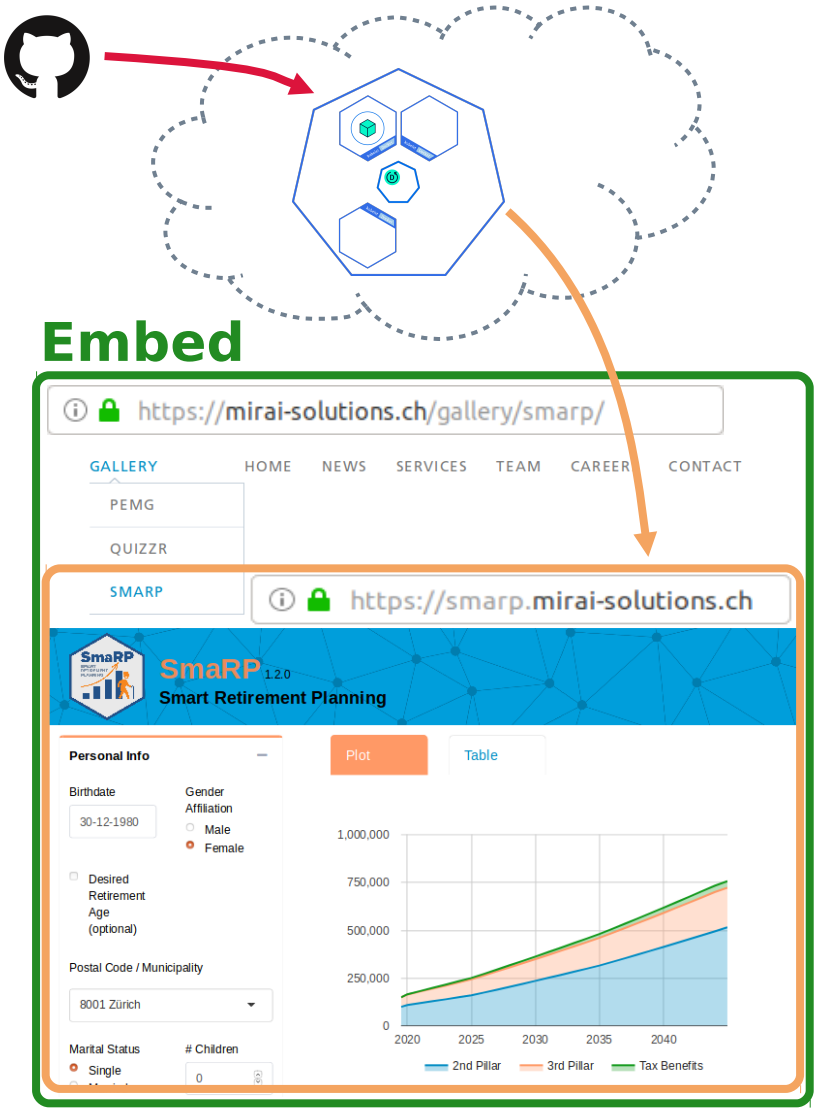
Hosting website page
=> **iframeResizer.min.js**

```
<script type="text/javascript"
 src="[...]/iframeResizer.min.js"></script>
<iframe id="gallery-iframe"
 src="https://smarp.mirai-solutions.ch"
 scrolling="no" frameborder="0"
 style="min-width: 100%;">
</iframe>
<script>
  iFrameResize({/* options */}, '#gallery-iframe');
</script>
```

Embedded Shiny app
=> **iframeResizer.contentWindow.min.js**

```
tags$head(tags$script(
  type = "text/javascript",
  src = "[...]/iframeResizer.contentWindow.min.js"
)),
```

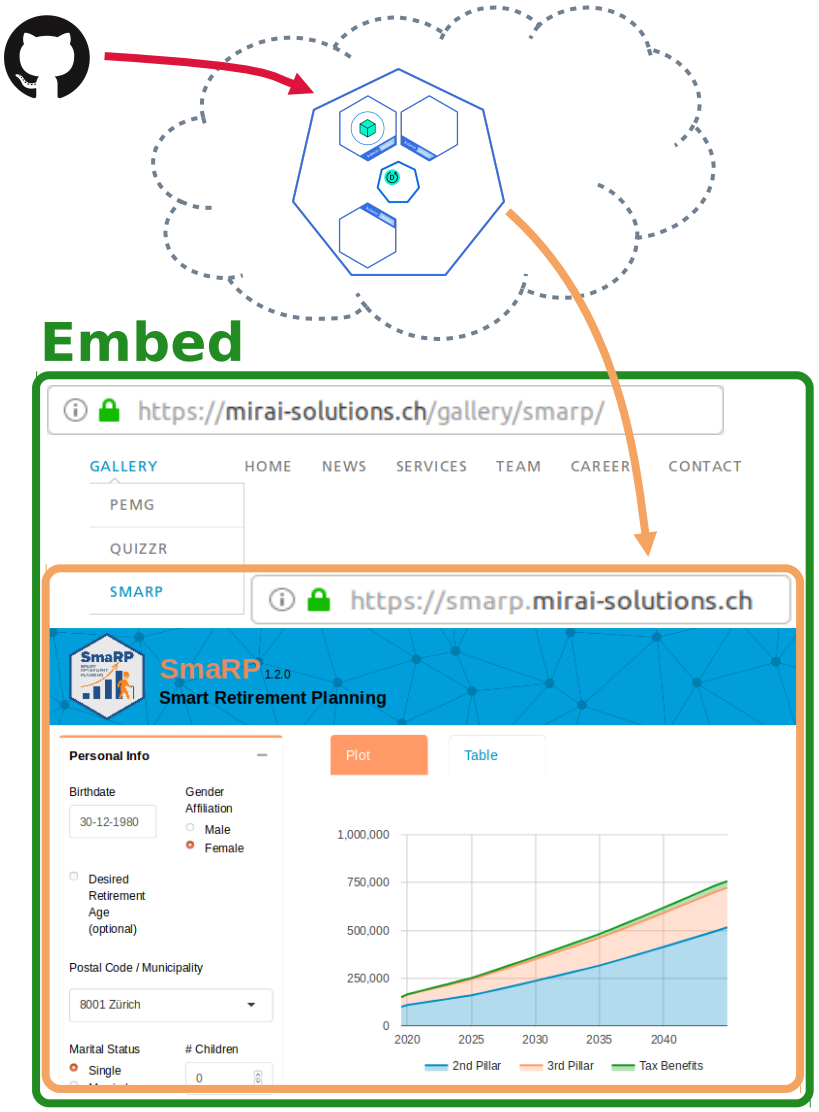
Website gallery for Shiny with Jekyll



Jekyll <https://jekyllrb.com> is a simple, blog-aware, static site generator, supporting **Markdown** and the **Liquid** template language

- **GitHub Pages**: Jekyll-based websites hosted from GitHub repositories

Website gallery for Shiny with Jekyll



Jekyll <https://jekyllrb.com> is a simple, blog-aware, static site generator, supporting **Markdown** and the **Liquid** template language

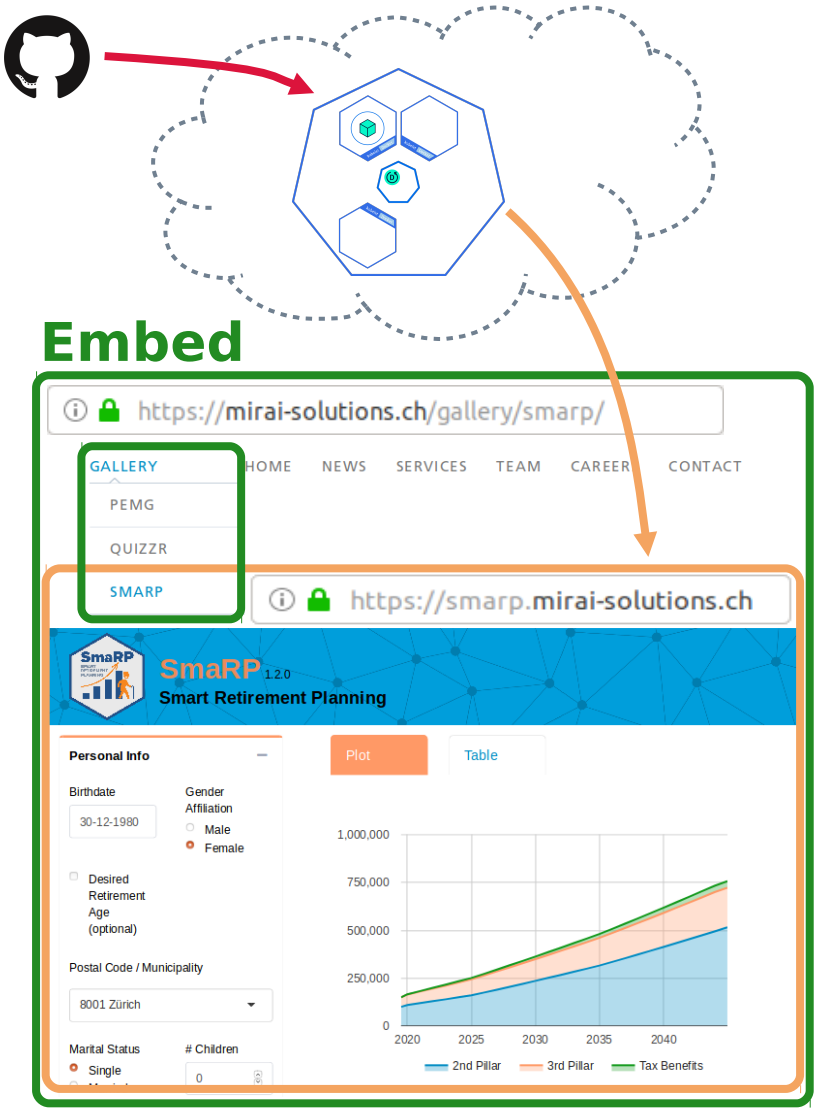
- **YAML metadata** for each app page
- Liquid **template**

```
# gallery/smarp.md
---
title: SmaRP - Smart Retirement Planning
active: smarp
parent: gallery
menu_entry: SmaRP
embed_url: https://smarp.mirai-solutions.ch
---

{% include _gallery-embed.html %}
```

```
<!--_gallery-embed.html-->
<script type="text/javascript"
 src="[...]/iframeResizer.min.js"></script>
<iframe id="gallery-iframe"
 src="{{ page.embed_url }}" [...]></iframe>
<script>
  iFrameResize({/* options */}, '#gallery-iframe');
</script>
```

Website gallery for Shiny with Jekyll



Jekyll <https://jekyllrb.com> is a simple, blog-aware, static site generator, supporting **Markdown** and the **Liquid** template language

- Automated **gallery menu** with Liquid template "programming"

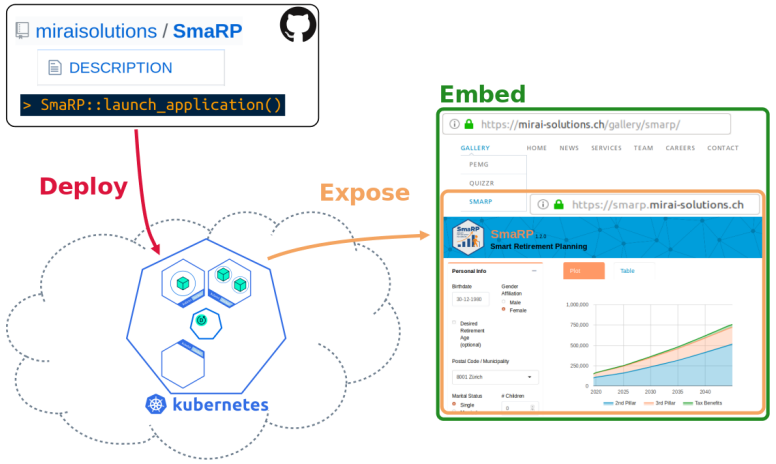
```
# gallery/smarp.md
---
title: SmaRP - Smart Retirement Planning
active: smarp
parent: gallery
menu_entry: SmaRP
```

```
<a href="#" class="dropdown-toggle" [...]>Gallery</a>
<ul class="dropdown-menu">
{% for p in site.pages %}
  {% if p.parent == 'gallery' %}
    <li>
      <a href="/gallery/{{ p.active }}/">
        {{ p.menu_entry }}
      </a>
    </li>
  {% endif %}
{% endfor %}
</ul>
```

Summary

Full workflow

- Shiny app as R package on GitHub
- Docker / Kubernetes deployment in GKE
- GitFlow approach / Continuous Delivery
- App exposed with custom domain via HTTPS
- Responsive embedding; Nice fit to existing Jekyll-based GitHub Pages website



Ground-up approach

- Flexibility and customization
- Understanding of / insights into: Docker, Kubernetes, GKE, exposing and embedding Shiny apps as services
- Technology stack relevant to other “off-the-shelf” tools (Shinyapps.io, ShinyProxy, RStudio Connect, …)

Links and references

Mirai Solutions
smarter analytics - better decisions

Detailed resources for SmaRP

https://github.com/miraisolutions/SmaRP/blob/develop/Dockerfile
https://github.com/miraisolutions/SmaRP/blob/develop/cloudbuild.yaml
https://github.com/miraisolutions/SmaRP/blob/develop/gke#readme

Kubernetes https://kubernetes.io

- Kubernetes documentation https://kubernetes.io/docs/
- Kubernetes basics tutorial https://kubernetes.io/docs/tutorials/kubernetes-basics/

Google Cloud docs, tutorials, how-to-s

- GKE overview
  https://cloud.google.com/kubernetes-engine/docs/concepts/kubernetes-engine-overview
- Continuous delivery with Cloud Build
  https://cloud.google.com/kubernetes-engine/docs/tutorials/gitops-cloud-build
- HTTPS load balancing with NodePort and Ingress
  https://cloud.google.com/kubernetes-engine/docs/concepts/ingress
  https://cloud.google.com/kubernetes-engine/docs/tutorials/http-balancer
  https://cloud.google.com/kubernetes-engine/docs/how-to/load-balance-ingress
- Backend service configuration
  https://cloud.google.com/kubernetes-engine/docs/how-to/configure-backend-service
- Google-managed TLS certificates
  https://cloud.google.com/load-balancing/docs/ssl-certificates#managed-certs

Mirai Solutions

smarter analytics - better decisions

Shiny app deployment and integration into a custom website gallery

UseR! Toulouse, July 10th 2019

Riccardo Porreca
Roland Schmid