

Lien du siteweb: <http://techo.chaghen.com>

Voici les besoins du projet PoleIt détaillés :

✓ **Création d'un site web attractif pour le centre d'astronomie PoleIt :**

- Le site doit comprendre une page d'accueil présentant le centre et ses principaux atouts (Observatoire, planétarium, etc.).
- La page d'accueil doit afficher les 4 derniers articles sous forme de carrousel accessibles uniquement aux membres connectés.
- Une page décrivant les activités diurnes et nocturnes.
- Une page de contact alimentant une base de données avec une interface d'administration pour traiter les demandes reçues.

✓ **Fonctionnalité de commerce en ligne et interface d'administration :**

- Une page e-commerce permettant la présentation et l'achat de goodies via la plateforme Stripe.
- Une gestion des stocks et la possibilité de spécifier la quantité d'articles à acheter
- Un système d'administration pour visualiser les demandes de contact, ajouter/modifier des articles, changer le stock disponible, et gérer les comptes utilisateurs.

✓ **Infrastructure et déploiement :**

- Mise en place d'une infrastructure kubernetes avec un contrôleur Ingress, une architecture hyper convergée pour la haute disponibilité et la maintenabilité.
- Le cluster kubernetes doit contenir au moins 3 nœuds
- Les sessions PHP doivent être stockées de manière centralisée, par exemple dans Redis ou MySQL, au lieu d'une instance Docker PHP locale.

✓ **Pratiques DevOps et pipeline CI/CD :**

- Déploiement automatique du site web sur le cluster kubernetes via du pipeline CI/CD.
- Le pipeline doit valider le code HTML, utiliser un linter pour tous les fichiers, construire une image Docker du code, et déployer automatiquement l'application sur le cluster kubernetes lors de la création d'un tag.

✓ **Technologies utilisées :**

- Symfony et Mysql.

Pour l'application lié au projet PoleIt, un menu de header efficace et organisé aiderait les utilisateurs à naviguer facilement et à accéder aux diverses sections importantes du site. Voici une proposition pour le menu header basée sur les informations fournies :

1. **Accueil :**

- ✓ Introduction au centre d'astronomie PoleIT
- ✓ Mise en avant des installations clés comme le radiotélescope géant

2. **A propos :**

- ✓ Informations détaillées sur le centre, son histoire et sa mission
- ✓ Détails sur les infrastructures comme le planétarium et l'observatoire

3. **Activités :**

- ✓ **Activités Diurnes :** Liste et descriptions des expositions et activités disponibles pendant la journée
- ✓ **Activités Nocturnes :** Informations sur les événements spéciaux et les expositions nocturnes
- ✓ **Calendrier des Événements :** Calendrier interactif montrant les événements planifiés

4. **Blog/Actualités**

- ✓ Mises à jour régulières et articles de blog sur les dernières découvertes et événements au centre
- ✓ Articles accessibles uniquement aux membres connectés, avec un carrousel des quatre derniers articles sur la page d'accueil

5. **Boutique**

- ✓ Présentation des goodies disponibles à l'achat
- ✓ Intégration de la plateforme de paiement Stripe pour les transactions

6. **Contact**

- ✓ Formulaire de contact pour les questions générales

- ✓ Coordonnées pour les réservations de groupe ou les visites scolaires
- ✓ Interface d'administration pour gérer les requêtes reçues

## 7. Espace Membre

- ✓ Connexion/Inscription pour accéder aux contenus exclusifs et participer aux discussions
- ✓ Profil utilisateur et gestion des préférences

Ce menu est conçu pour offrir une navigation intuitive et mettre en avant les principales attractions et services du centre d'astronomie PoleIt. Il peut être ajusté en fonction des retours des utilisateurs et de l'évolution des besoins du centre.

## Infrastructure et déploiement :

Votre infrastructure est un « beau » cluster Kubernetes incluant un contrôleur Ingress(Traefik)

- ✓ Cluster Kubernetes avec 3 nœuds :

```
root@master:/home/farouckloicyann# kubectl get nodes
NAME          STATUS    ROLES          AGE      VERSION
master        Ready     control-plane,etcd,master   14m      v1.29.4+k3s1
worker1       Ready     <none>          3m36s    v1.29.4+k3s1
worker2       Ready     <none>          22s      v1.29.4+k3s1
```

- ✓ Implémentation du controller Traefik : lien de la documentation : <https://doc.traefik.io/traefik/getting-started/quick-startwith-kubernetes/>

```
root@master:/home/farouckloicyann# touch 00-role.yaml
root@master:/home/farouckloicyann# touch 00-account.yaml
root@master:/home/farouckloicyann# touch 01-role-binding.yaml
root@master:/home/farouckloicyann# touch 02-traefik.yaml
root@master:/home/farouckloicyann# touch 02-traefik-services.yaml
```

```
root@master:/home/farouckloicyann# kubectl apply -f 00-role.yaml
clusterrole.rbac.authorization.k8s.io/traefik-role created
root@master:/home/farouckloicyann# kubectl apply -f 00-account.yaml
serviceaccount/traefik-account created
root@master:/home/farouckloicyann# kubectl apply -f 01-role-binding.yaml
clusterrolebinding.rbac.authorization.k8s.io/traefik-role-binding created
root@master:/home/farouckloicyann# kubectl apply -f 02-traefik.yaml
deployment.apps/traefik-deployment created
root@master:/home/farouckloicyann# kubectl apply -f 02-traefik-services.yaml
service/traefik-dashboard-service created
service/traefik-web-service created
```

- ✓ Le stockage doit être hautement disponible et il faut le penser comme une « infrastructure hyperconvergée » ( cf Annexes ) pour simplifier l'architecture et la maintenabilité.

- Volume persistant pour mysql :

```

apiVersion: v1
kind: PersistentVolume
metadata:
  name: mysql-pv
spec:
  capacity:
    storage: 2Gi
  volumeMode: Filesystem
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/home/farouckloicyann/mysql-data"
  storageClassName: hostpath
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mysql-pvc
spec:
  resources:
    requests:
      storage: 1Gi
  volumeMode: Filesystem
  accessModes:
    - ReadWriteOnce
  storageClassName: hostpath
  volumeName: mysql-pv

```

- Déploiement Mysql et service :

```

apiVersion: apps/v1
kind: Deployment
metadata:
  name: mysql
spec:
  selector:
    matchLabels:
      app: mysql
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
        - image: mariadb:latest
          name: mysql
          env:
            - name: MYSQL_ROOT_PASSWORD
              value: ""
            - name: MYSQL_DATABASE
              value: poleit
          ports:
            - containerPort: 3306
              name: mysql
          volumeMounts:
            - name: mysql-persistent-storage
              mountPath: /var/lib/mysql
      volumes:
        - name: mysql-persistent-storage
          persistentVolumeClaim:
            claimName: mysql-pvc

```

```

apiVersion: v1
kind: Service
metadata:
  name: mariadb-service
spec:
  selector:
    app: mysql
  ports:
    - protocol: TCP
      port: 3306
      targetPort: 3306
  type: NodePort

```

✓ Déploiement de l'application symphony

Création du fichier DockerFile :

```
FROM php:8.1-fpm

# Install dependencies
RUN apt-get update && apt-get install -y \
    libc-dev \
    libonig-dev \
    libzip-dev \
    unzip \
    git \
    && docker-php-ext-install pdo pdo_mysql intl zip opcache

# Set working directory
WORKDIR /var/www

# Install Composer
COPY --from=composer:latest /usr/bin/composer /usr/bin/composer

# Set environment variables for Composer
ENV COMPOSER_ALLOW_SUPERUSER=1
ENV COMPOSER_HOME=/composer

# Copy application source
COPY . .

# Install PHP dependencies
RUN composer install --no-dev --optimize-autoloader --no-scripts

# Expose port 9000 and start php-fpm server
EXPOSE 9000
CMD ["php-fpm"]
```

- **Dockerfile pour le serveur web Nginx :**

```
1 # Utiliser une image Nginx
2 >> FROM nginx:latest
3
4 # Copier le fichier de configuration Nginx
5 COPY nginx.conf /etc/nginx/nginx.conf
6
7 # Exposer le port 80
8 EXPOSE 80
9
```

- **docker-compose.yml pour l'orchestration des conteneurs :**

```
1  version: '3'
2
3  services:
4    php:
5      build:
6        context: .
7        dockerfile: Dockerfile
8      volumes:
9        - ./var/www/html
10     depends_on:
11       - db
12     environment:
13       DATABASE_URL: mysql://user:password@db:3306/poleit
14     networks:
15       - app-network
16
17   db:
18     image: mysql:latest
19     restart: always
20     environment:
21       MYSQL_DATABASE: dbname
22       MYSQL_USER: user
23       MYSQL_PASSWORD: password
24       MYSQL_ROOT_PASSWORD: rootpassword
25     ports:
26       - "3306:3306"
27     volumes:
28       - db-data:/var/lib/mysql
29     networks:
30
31
32   volumes:
33     - db-data:/var/lib/mysql
34     networks:
35       - app-network
36
37   nginx:
38     build:
39       context: .
40       dockerfile: nginx.Dockerfile
41     ports:
42       - "8080:80"
43     depends_on:
44       - php
45     volumes:
46       - ./var/www/html
47     networks:
48       - app-network
49
50   networks:
51     app-network:
52       driver: bridge
53
54   volumes:
55     db-data:
56       driver: local
```

- Fichier de configuration Nginx (nginx.conf)

```

1  user  nginx;
2  worker_processes  1;
3
4  events {
5      worker_connections  1024;
6  }
7
8  http {
9      include        /etc/nginx/mime.types;
10     default_type    application/octet-stream;
11
12     sendfile        on;
13     keepalive_timeout  65;
14
15     server {
16         listen       80;
17         server_name  localhost;
18
19         root         /var/www/html/public;
20         index        index.php;
21
22         location / {
23             try_files $uri /index.php$is_args$args;
24         }
25
26         location ~ ^/index\.php(/|$) {
27             fastcgi_pass   php:9000;
28             fastcgi_split_path_info ^(.+\.php)(/.*)$;

```

```

        location ~ ^/index\.php(/|$) {
            fastcgi_pass   php:9000;
            fastcgi_split_path_info ^(.+\.php)(/.*)$;
            include         fastcgi_params;
            fastcgi_param   SCRIPT_FILENAME $document_root$fastcgi_script_name;
            fastcgi_param   PATH_INFO $fastcgi_path_info;
            internal;
        }

        location ~ \.php$ {
            return 404;
        }

        error_log  /var/log/nginx/error.log;
        access_log /var/log/nginx/access.log;
    }
}

```

- Construire et pousser l'image docker

Commandes:

### **docker login**

s'authentifier : nom d'utilisateur : farouckloicyann motdepasse : Adechina2024@

Construire l'image docker :

**docker build -t farouckloicyann/symfony-app:1.0 .**

Pousser l'image docker

**docker push farouckloicyann/symfony-app:1.0**

**Preuve de réussite des commandes docker :**

farouckloicyann

Search by repository name

All Content

Create repository

farouckloicyann / symfony-app

Security unknown

☆ 0

↓ 5

Public

Contains: Image • Last pushed: about 9 hours ago

- **Stockage persistant pour l'application symfony**

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: symfony-pv
spec:
  capacity:
    storage: 2Gi
  volumeMode: Filesystem
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/home/farouckloicyann/symfony-app-data"
    storageClassName: hostpath
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: symfony-pvc
spec:
  resources:
    requests:
      storage: 1Gi
  volumeMode: Filesystem
  accessModes:
    - ReadWriteOnce
  storageClassName: hostpath
  volumeName: symfony-pv
```

- **Création du fichier Yaml pour le déploiement et le de l'application symfony :**



```

apiVersion: apps/v1
kind: Deployment
metadata:
  name: symfony
spec:
  replicas: 2
  selector:
    matchLabels:
      app: symfony
  template:
    metadata:
      labels:
        app: symfony
    spec:
      containers:
        - image: farouckloicyann/symfony-app:1.0
          name: symfony
          env:
            - name: DATABASE_URL
              value: "mysql://root:@mysql:3306/poleit?serverVersion=mariadb-10.4.11"
          ports:
            - containerPort: 9000
              name: symfony
          volumeMounts:
            - mountPath: "/app/data"
              name: symfony-data
      volumes:
        - name: symfony-data
          persistentVolumeClaim:
            claimName: symfony-pvc

```

```

apiVersion: v1
kind: Service
metadata:
  name: symfony
spec:
  type: NodePort
  selector:
    app: symfony
  ports:
    - protocol: TCP
      port: 9000
      targetPort: 9000

```

- **Création du fichier Yaml pour l'ingress:**

```

• apiVersion: networking.k8s.io/v1
  kind: Ingress
  metadata:
    name: memos-ingress
  spec:
    rules:
      - host: poleit.localhost.com
        http:

```

```
paths:
  - pathType: Prefix
    path: "/"
    backend:
      service:
        name: memos-services
      port:
        number: 9000
```

Modifier le fichier /etc/hosts dans le master dans la machine virtuelle debian et enregistrer

```
127.0.0.1    localhost
127.0.1.1    master
192.168.1.69 poleit.localhost.com
# The following lines are desirable for IPv6 capable hosts
::1          localhost ip6-localhost ip6-loopback
ff02::1      ip6-allnodes
ff02::2      ip6-allrouters
```

Modifier Le fichier C:\Windows\System32\drivers\etc\hosts sur windows en tant qu'administrateur

```
-----
192.168.1.69 poleit.localhost.com|
```