# Title: Microservices Architecture Deployment for OCR and Translation Application Using Kubernetes

---

### 1. Overview

The objective of this assignment was to containerize an OCR and Translation application, break it down from a monolith into microservices, and deploy it on a Kubernetes cluster.

---

### 2. Monolith Architecture (Before)

- **Frontend and Backend together**: The application initially had a single service that handled OCR functionality and translation combined.
- **Limitations**: Scalability issues and difficulty maintaining separate concerns such as OCR processing and translation.

---

### 3. Microservices Architecture (After)

The application was restructured into two distinct services:

- **OCR Service**: Handles text extraction from images.
- **Translation Service**: Translates extracted text into a target language.

Each service was containerized and managed independently in the Kubernetes environment.

---

### 4. Architecture Diagram

- **Monolith**: Depicted as a single block handling OCR and translation.
- **Microservices**: Shows two services (OCR and Translation) interacting via HTTP requests.

---

### 5. Kubernetes Deployment

**YAML Files Overview:**

- **OCR Service:** Handles text extraction from images.
  - Deployment and service definition in YAML (`ocr-deployment.yaml`).
- **Translation Service:** Manages translation of text into different languages.
  - YAML deployment and service definition (`translation-deployment.yaml`).
- **Frontend Service:** A React-based frontend hosted separately, interacts with both services.
  - Defined in `frontend-deployment.yaml`.

---

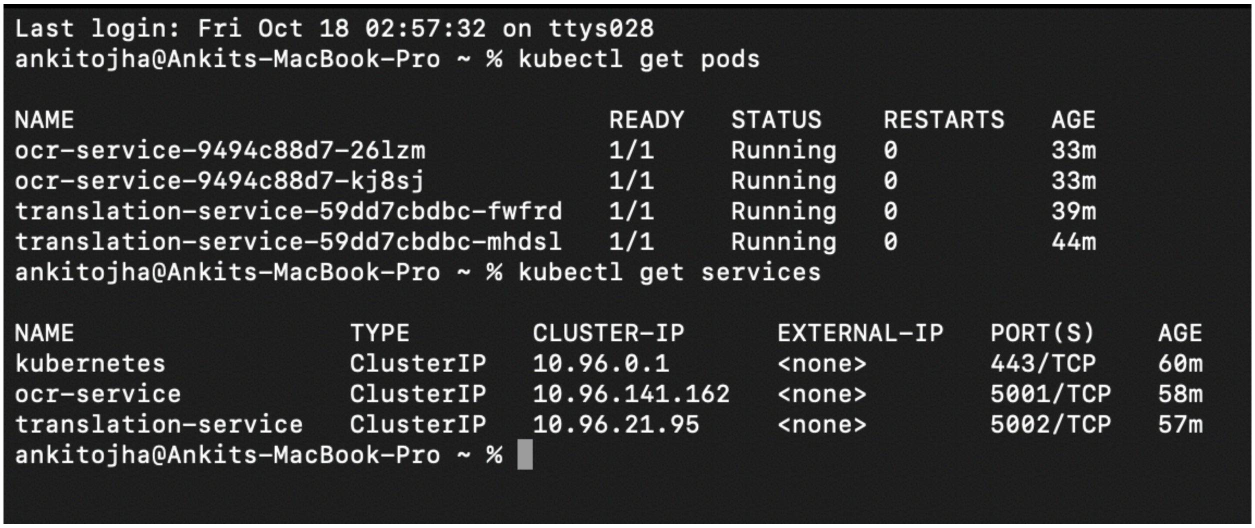
### 6. Docker and Kubernetes Commands

1. **Docker Image Creation:** For each service:
   - `docker build -t ocr-service .`
   - `docker build -t translation-service .`
   - `docker build -t frontend-app .`
2. **Loading into Kubernetes:**
   - `kind load docker-image ocr-service`
   - `kind load docker-image translation-service`
   - `kind load docker-image frontend-app`
3. **Deployment on Kubernetes:**
   - `kubectl apply -f ocr-deployment.yaml`
   - `kubectl apply -f translation-deployment.yaml`
   - `kubectl apply -f frontend-deployment.yaml`

---

### 7. Results and Screenshots

- **Pods and Services:** Screenshots of `kubectl get pods` and `kubectl get services` showing running services.

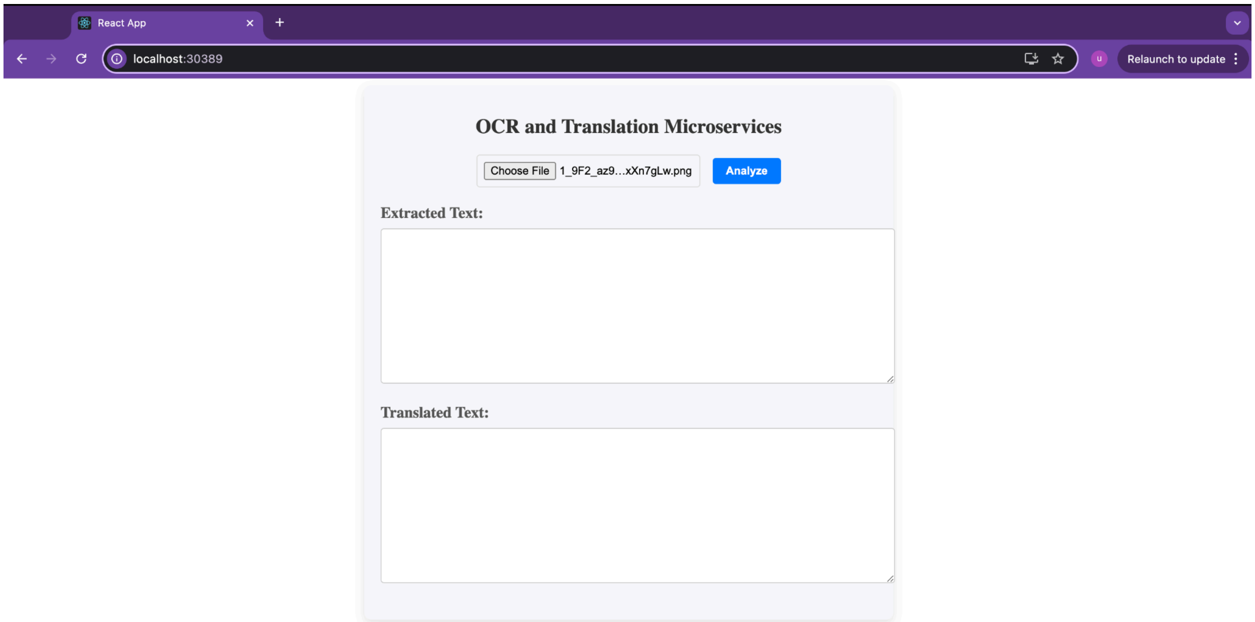


```
Last login: Fri Oct 18 02:57:32 on ttys028
ankitojha@Ankits-MacBook-Pro ~ % kubectl get pods

NAME                                   READY   STATUS    RESTARTS   AGE
ocr-service-9494c88d7-26lzm            1/1     Running   0          33m
ocr-service-9494c88d7-kj8sj            1/1     Running   0          33m
translation-service-59dd7cbdbc-fwfrd   1/1     Running   0          39m
translation-service-59dd7cbdbc-mhdsl   1/1     Running   0          44m
ankitojha@Ankits-MacBook-Pro ~ % kubectl get services

NAME                  TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)    AGE
kubernetes            ClusterIP   10.96.0.1       <none>        443/TCP    60m
ocr-service           ClusterIP   10.96.141.162   <none>        5001/TCP   58m
translation-service   ClusterIP   10.96.21.95     <none>        5002/TCP   57m
ankitojha@Ankits-MacBook-Pro ~ %
```

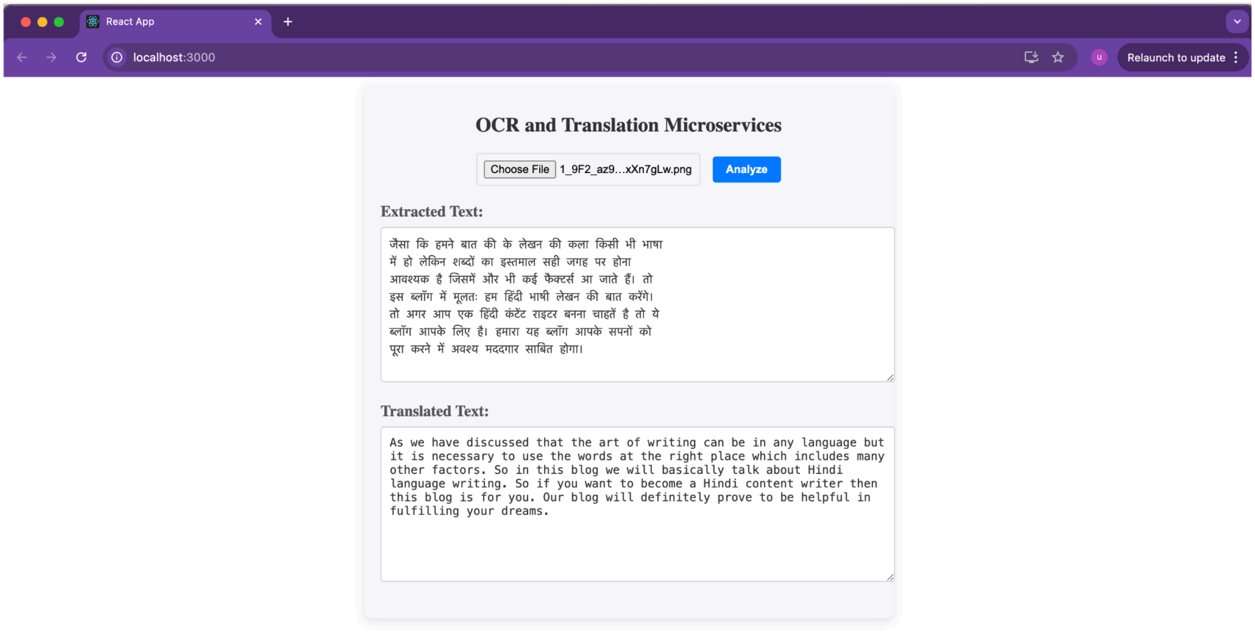
- **Application UI:** UI screenshots after successful deployment showing OCR results and translations.



OCR and Translation Microservices

Extracted Text:

Translated Text:



OCR and Translation Microservices

Extracted Text:

```
जैसा कि हमने बात की के लेखन की कला किसी भी भाषा
में हो लेकिन शब्दों का इस्तमाल सही जगह पर होना
आवश्यक है जिसमें और भी कई फैक्टर्स आ जाते हैं। तो
इस ब्लॉग में मूलतः हम हिंदी भाषी लेखन की बात करेंगे।
तो अगर आप एक हिंदी कंटेंट राइटर बनना चाहतें है तो ये
ब्लॉग आपके लिए है। हमारा यह ब्लॉग आपके सपनों को
पूरा करने में अवश्य मददगार साबित होगा।
```

Translated Text:

```
As we have discussed that the art of writing can be in any language but
it is necessary to use the words at the right place which includes many
other factors. So in this blog we will basically talk about Hindi
language writing. So if you want to become a Hindi content writer then
this blog is for you. Our blog will definitely prove to be helpful in
fulfilling your dreams.
```

## 8. Challenges and Lessons Learned

- **Networking Issues:** Initial issues with service connectivity between frontend and backend were resolved through proper service mapping and port configuration.
- **Scalability:** The microservices architecture proved to be more scalable and manageable.

---

## 9. Conclusion

This project demonstrates the transition from a monolithic to microservices architecture, utilizing Docker and Kubernetes for deployment, and emphasizes the advantages in scalability, maintainability, and service isolation.