**Deploying ASP.NET Core MVC Application to Kubernetes**

**1. Framework Selection**  
I selected ASP.NET Core MVC for this task because:

* It represents a fundamentally different architecture from NodeJS (C# vs JavaScript, integrated middleware pipeline vs Express middleware)
* Microsoft's official documentation provides comprehensive deployment guides
* The framework includes built-in support for containerization
* It enforces strict MVC separation (Controllers/Views/Models folders structure)

Unlike NodeJS which uses package.json for dependencies, ASP.NET Core uses NuGet packages and csproj files. The routing system is also more structured, with explicit controller-action mapping rather than NodeJS's callback-based routes.

**2. Learning Process**  
Learning steps:

1. Followed Microsoft's "Get started with ASP.NET Core MVC" tutorial
2. Created basic BookController with Index action
3. Added simple Book model class
4. Scaffolded basic CRUD views

Key challenges:

* Understanding Razor view syntax (transitioning from Handlebars/EJS)
* Configuring Dockerfile for .NET Core (different from NodeJS multi-stage builds)
* Runtime differences between development and production modes

Solutions:

* Used dotnet new mvc template as foundation
* Referenced Microsoft's containerization guidelines
* Enabled detailed error pages during development

**3. Deployment Process**   
Deployment steps:

1. Created optimized Dockerfile using SDK/Runtime stages
2. Built and tested container locally
3. Set up local registry on Kubernetes cluster
4. Created Kubernetes manifests with:
   * Deployment (2 replicas)
   * NodePort service
5. Verified operation through:
   * kubectl get endpoints
   * Load testing with kubectl port-forward
   * Log inspection

Critical fixes:

* Adjusted containerPort to match ASP.NET's default 8080
* Added proper liveness probes
* Configured proper environment variables for production

**4. Reflection**   
Key learnings:

* Containerizing .NET apps requires understanding of runtime vs SDK images
* Kubernetes deployments need proper readiness checks for MVC apps
* ASP.NET's configuration system differs significantly from NodeJS

Challenges:

* Debugging containerized applications requires different approaches
* Kubernetes networking for local development needs careful planning

For future work:

* Implement proper health checks earlier
* Use ConfigMaps for environment variables
* Consider ingress controllers from the start

**Appendices**

A screenshot of a computer

AI-generated content may be incorrect.

Website is now able to run on Ubuntu via “dotnet run” command.

A screenshot of a computer program

AI-generated content may be incorrect.

Successfully create a Docker Image through Dockerfile

A computer screen shot of a program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

The basic Bookstore website is now accessible through “http://localhost:8080”

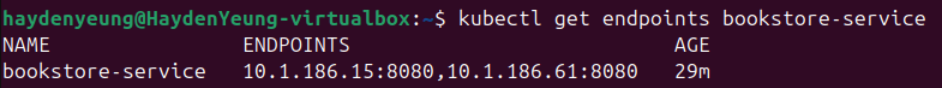
A computer screen shot of a program

AI-generated content may be incorrect.

A screenshot of a computer screen

AI-generated content may be incorrect.

Bookstore Web Application got deployed with Kubernetes



Checking with command “kubectl get endpoints bookstore-service”

GitHub Link:

https://github.com/HaydenDuong/SIT226\_Cloud\_Automation\_Technologies/tree/main/Coding%20Tasks/2.3D%20-%20Bookstore%20Web%20App