Homework Assignment: Helm-based GitOps-Friendly Deployment for Internal Services

📖 Scenario

Our development teams are not Kubernetes experts. To enable them to easily deploy their containerized applications, we want to provide a **Helm chart** that abstracts away Kubernetes complexity while offering:

- GitOps-friendly workflows
- Clear config separation for `dev` and `prod`
- Secure-by-default deployments
- Easy configuration for images, resources, and environment variables

---

🧪 Your Task

You are to create a **public GitHub repository** that satisfies the following requirements.

✅ Functional Requirements

1. **Simple Web Application**
   A containerized web application that:
   - Serves a web page on port `8080`
   - Outputs all environment variables to the page in JSON or text
2. **Dockerfile**
   - Create a `Dockerfile` that builds the application
   - The image must be **publicly available** (e.g., Docker Hub)
3. **Helm Chart**
   - Helm chart called `internal-service`
   - Values must allow easy configuration of the following:

   -
     ```
     hub:        # registry hostname
     image:      # image name
     tag:        # image tag
     prod:       # true or false
     env:        # additional environment variables
     resources:  # CPU and memory requests/limits
     ```
   - Default to a **replicated deployment (min 3 pods)** with:
     - HPA autoscaling
     - Pod anti-affinity (to spread pods across nodes)
     - SecurityContext with best practices (readOnlyRootFilesystem, runAsNonRoot, etc.)

4. **Environment Support**
   - `dev` and `prod` deployments differ **only** by the `PROD=true|false` environment variable.
   - No Ingress is required (internal service).
   - No persistent volume is needed.
5. **Run it on `kind`**
   - Provide easy-to-follow instructions in `README.md` to:
     - Set up a local `kind` cluster
     - Install the chart for both `dev` and `prod`

⭐ Bonus Points

- Use **Kustomize** with Helm output to store rendered manifests under GitOps folders:

```
gitops/
├── base/
│   ├── kustomization.yaml
├── overlays/
│   ├── dev/
│   │   ├── kustomization.yaml
│   │   └── patch-env.yaml   # sets PROD=false
│   └── prod/
│       ├── kustomization.yaml
│       └── patch-env.yaml   # sets PROD=true

```

- Separate overlays for `dev` and `prod` (with `PROD=true|false`)
- Clear documentation for how to render and commit changes

📝 Submission Guidelines

- Host your solution in a **public GitHub repository**
- Include:
  - A working Helm chart
  - Docker image link (public)
  - Clear `README.md` with:
    - Setup instructions
    - How to deploy to `kind`
    - How to render and apply with Kustomize (if included)
- Provide GitHub repo link in your submission

📌 Evaluation Criteria

Area	Description
✅ Functionality	Meets all functional requirements
📦 Helm Best Practices	Templates, values.yaml, and reuse are clean and modular
🔐 Security	Secure Pod configuration (runAsNonRoot, etc.)
📚 Documentation	Clarity and usability of `README.md`
🧠 Bonus	Kustomize support and GitOps structure
💡 Thoughtfulness	Are default values reasonable? Is structure intuitive?