Documentation of security improvements made

Application Dockerfile

Use multi-stage to reduce the size of Docker image and specify SHA version

```
# SECURITY ISSUE: Using latest tag instead of specific version
FROM
python:3.9@sha256:bc2e05bca883473050fc3b7c134c28ab822be73126ba1ce29517d9e8b7f3703
b AS build
```

```
# use slim version
FROM python:3.9-slim-
bullseye@sha256:5ece09c3d27b302ce8a3d87c6c7e33ee144329b757e68ae7b6ed2fc807dc44d5
```

Build stage

Use full image version with specify SHA version

Prepare application dependencies for next stage

```
FROM
python:3.9@sha256:bc2e05bca883473050fc3b7c134c28ab822be73126ba1ce29517d9e8b7f3703
b AS build

WORKDIR /app

# Install dependencies
COPY requirements.txt .
RUN pip install --no-cache-dir --user -r requirements.txt
```

Image stage

```
# use slim version
FROM python:3.9-slim-
bullseye@sha256:5ece09c3d27b302ce8a3d87c6c7e33ee144329b757e68ae7b6ed2fc807dc44d5

# set working directory
WORKDIR /app

# SECURITY ISSUE: Running as root
# add non-root user
RUN groupadd -r nonroot && useradd -r -g nonroot nonroot

# use non-root user
USER nonroot

# copy dependency from build
COPY --from=build /root/.local /home/nonroot/.local

# copy application code and change owner
COPY --chown=nonroot:nonroot . .

# SECURITY ISSUE: Using environment variables for sensitive information
# ENV DB_PASSWORD="supersecretpassword"
# pass DB_PASSWORD through docker cli command
ENV DEBUG=True

# SECURITY ISSUE: Exposing unnecessary ports
# expose application used port
EXPOSE 5000

# SECURITY ISSUE: Running with high privileges
CMD ["python", "app.py"]
```

Use slim image version

Create new user which non-root account name “nonroot” and use it

Copy prepared dependencies from build stage to current image

Copy application code and change owner to current user

Remove passing sensitive information through ENV in Dockerfile use Docker cli instead

Expose necessary port and run application with current user permission

Kubernetes manifests

Add securityContext to pod behavior. User id and group id will be 1000 instead of 0 (root user id) and fsGroup to 2000 which is share storage files for group 2000 instead of relying on container's default group

```
    spec:
      # SECURITY ISSUE: No security context
      securityContext:
        runAsUser: 1000
        runAsGroup: 1000
        fsGroup: 2000
```

Add securityContext for container to ensure that container run as non-root and not allow privilege escalation

```
        # SECURITY ISSUE: Running as root by default
        securityContext:
          allowPrivilegeEscalation: false
          runAsNonRoot: true
```

Set container's resource limit

```
        # SECURITY ISSUE: No resource limits
        resources:
          limits:
            cpu: "500m"
            memory: "128Mi"
```

Use Kubernetes ConfigMap and Secret to store environment data which sensitive data should store in Secret and non-sensitive data store in ConfigMap

```
        env:
        - name: DB_PASSWORD  # SECURITY ISSUE: Secret as environment variable
          valueFrom:
            secretKeyRef:
              name: app-secret
              key: DB_PASSWORD
        - name: DEBUG
          # value: "True"  # SECURITY ISSUE: Debug enabled in production
          valueFrom:
            configMapKeyRef:
              name: app-configmap
              key: DEBUG
```

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: app-secret
data:
  DB_PASSWORD: "c3VwZXJzZWNyZXRwYXNzd29yZA==" # "supersecretpassword"
---
apiVersion: v1
kind: ConfigMap
metadata:
  name: app-configmap
data:
  DEBUG: "True" # SECURITY ISSUE: Debug enabled in production
```

Add readiness and liveness probes

```yaml
        # SECURITY ISSUE: No liveness/readiness probes
        readinessProbe:
          httpGet:
            path: /
            port: 5000
          initialDelaySeconds: 10
          periodSeconds: 10
        livenessProbe:
          httpGet:
            path: /
            port: 5000
          initialDelaySeconds: 10
          periodSeconds: 10
```

Evidence

```
Administrator@AkaruIxRiN MINGW64 /f/work/devops-interview-exams/scripts (main)
$ ./docker_security_report.sh user-management
Generating security report for Docker image: user-management
==================================
DOCKER SECURITY REPORT
Image: user-management
Date: Mon Mar 31 23:16:08 SEAST 2025
==================================
[PASS] Not using 'latest' tag
[PASS] Image runs as non-root user: nonroot
[INFO] Exposed ports: 5000/tcp
[INFO] Environment variables found
[HIGH] Potentially sensitive information in environment variables
[PASS] Image size is reasonable: 136 MB
[INFO] Basic package check (simulated)
[INFO] A real implementation would use tools like Trivy, Clair, or Anchore
==================================
Security scan completed
Consider using dedicated tools like Trivy, Clair, or Anchore for comprehensive scanning
==================================
```