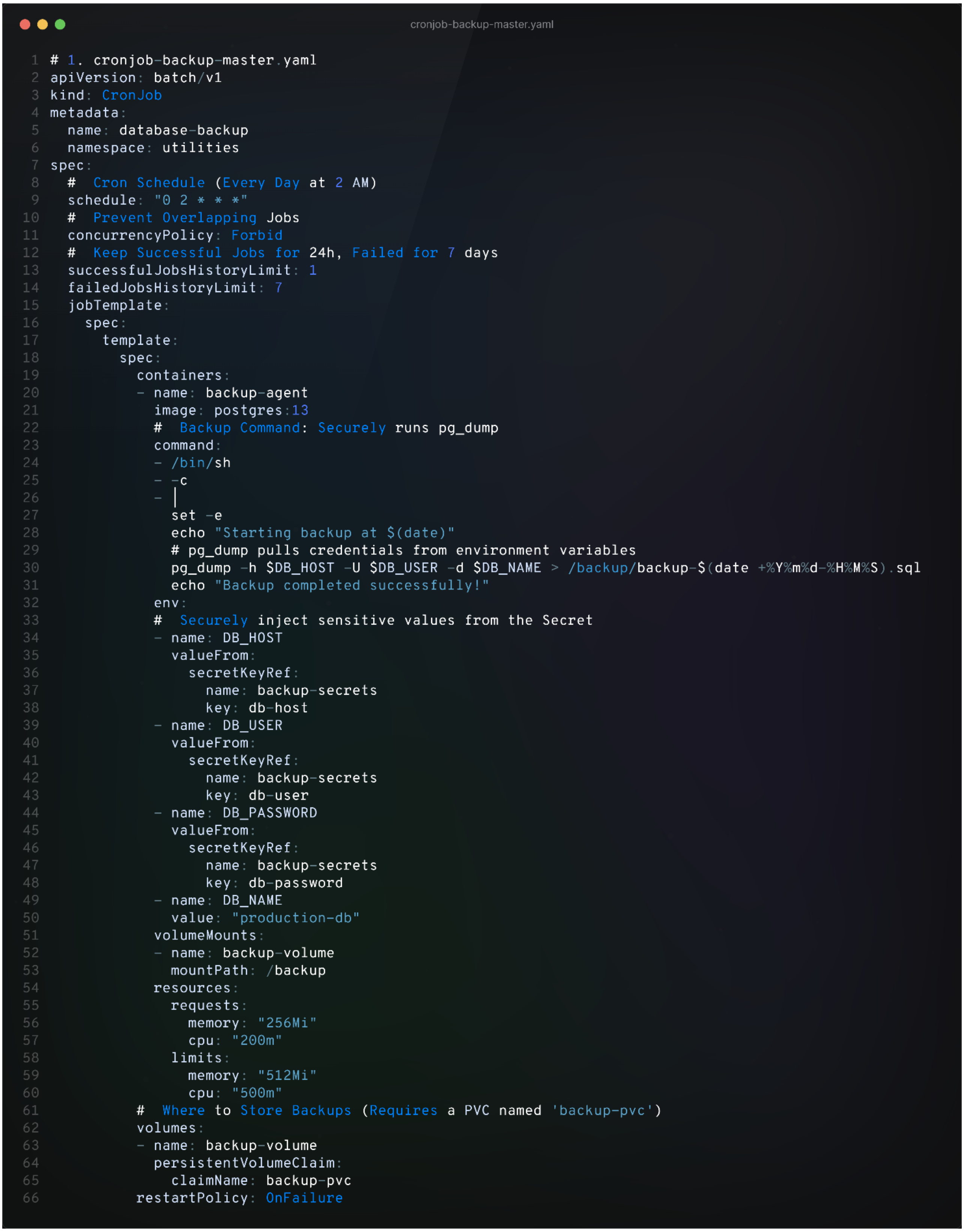
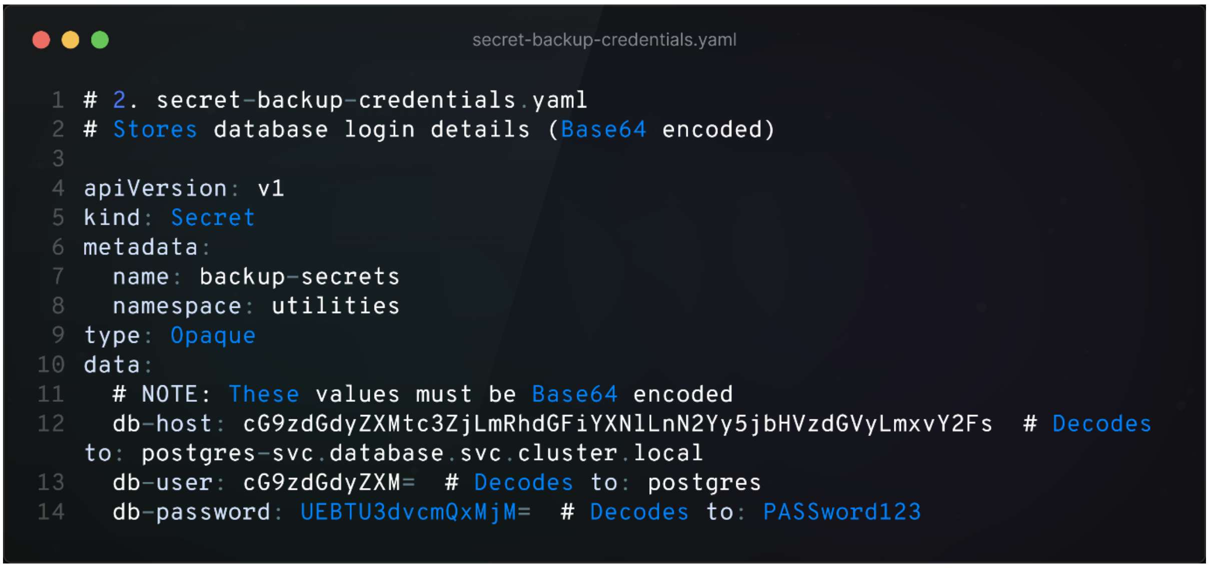
1. The Scheduled Task Master (*`cronjob-backup-master.yaml`*) - This is the engine. It defines the schedule, the container (using `postgres:13` for the *pg_dump* utility), and critical safety nets like `concurrencyPolicy: Forbid` and history limits to prevent runaways and keep your logs clean.



```yaml
# 1. cronjob-backup-master.yaml
apiVersion: batch/v1
kind: CronJob
metadata:
  name: database-backup
  namespace: utilities
spec:
  #  Cron Schedule (Every Day at 2 AM)
  schedule: "0 2 * * *"
  #  Prevent Overlapping Jobs
  concurrencyPolicy: Forbid
  #  Keep Successful Jobs for 24h, Failed for 7 days
  successfulJobsHistoryLimit: 1
  failedJobsHistoryLimit: 7
  jobTemplate:
    spec:
      template:
        spec:
          containers:
          - name: backup-agent
            image: postgres:13
            #  Backup Command: Securely runs pg_dump
            command:
            - /bin/sh
            - -c
            - |
              set -e
              echo "Starting backup at $(date)"
              # pg_dump pulls credentials from environment variables
              pg_dump -h $DB_HOST -U $DB_USER -d $DB_NAME > /backup/backup-$(date +%Y%m%d-%H%M%S).sql
              echo "Backup completed successfully!"
            env:
            #  Securely inject sensitive values from the Secret
            - name: DB_HOST
              valueFrom:
                secretKeyRef:
                  name: backup-secrets
                  key: db-host
            - name: DB_USER
              valueFrom:
                secretKeyRef:
                  name: backup-secrets
                  key: db-user
            - name: DB_PASSWORD
              valueFrom:
                secretKeyRef:
                  name: backup-secrets
                  key: db-password
            - name: DB_NAME
              value: "production-db"
            volumeMounts:
            - name: backup-volume
              mountPath: /backup
            resources:
              requests:
                memory: "256Mi"
                cpu: "200m"
              limits:
                memory: "512Mi"
                cpu: "500m"
          #  Where to Store Backups (Requires a PVC named 'backup-pvc')
          volumes:
          - name: backup-volume
            persistentVolumeClaim:
              claimName: backup-pvc
          restartPolicy: OnFailure
```

2. The Credential Vault (`secret-backup-credentials.yaml`)

NEVER hardcode credentials. Use a Kubernetes **`Secret`** to inject them securely as environment variables into your `CronJob` container.

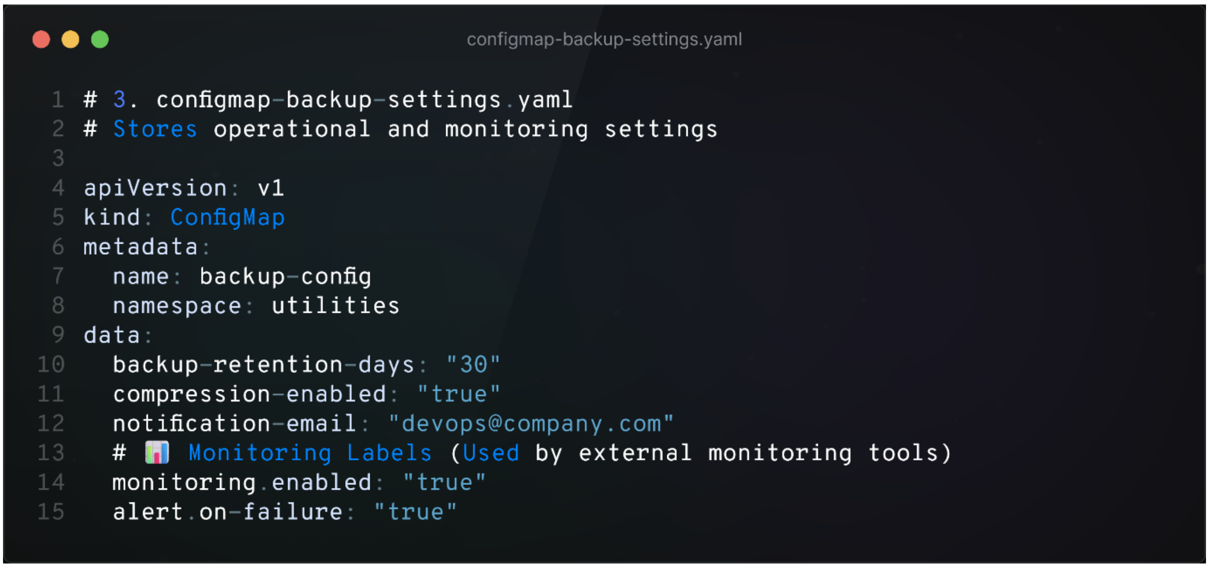


```yaml
# 2. secret-backup-credentials.yaml
# Stores database login details (Base64 encoded)

apiVersion: v1
kind: Secret
metadata:
  name: backup-secrets
  namespace: utilities
type: Opaque
data:
  # NOTE: These values must be Base64 encoded
  db-host: cG9zdGdyZXMtc3ZjLmRhdGFiYXNlLnN2Yy5jbHVzdGVyLmxvY2Fs  # Decodes to: postgres-svc.database.svc.cluster.local
  db-user: cG9zdGdyZXM=  # Decodes to: postgres
  db-password: UEBTU3dvcmQxMjM=  # Decodes to: PASSword123
```

3. The Configuration Hub (`configmap-backup-settings.yaml`)

While the CronJob doesn't *directly* use these settings yet, this **ConfigMap** demonstrates best practice. It provides a central place for non-sensitive, operational settings (like retention policies or monitoring labels) that can be easily updated without touching the core CronJob logic.



```yaml
# 3. configmap-backup-settings.yaml
# Stores operational and monitoring settings

apiVersion: v1
kind: ConfigMap
metadata:
  name: backup-config
  namespace: utilities
data:
  backup-retention-days: "30"
  compression-enabled: "true"
  notification-email: "devops@company.com"
  # 📊 Monitoring Labels (Used by external monitoring tools)
  monitoring.enabled: "true"
  alert.on-failure: "true"
```