1. backend-deployment.yaml

This file defines the Deployment for your backend application. It manages your backend Pods, sets up the necessary environment variables to connect to MongoDB, and includes Liveness and Readiness probes to ensure the application is healthy and ready to receive traffic.

Important: Before you apply this, make sure you have built your backend Docker image and pushed it to a container registry (like Docker Hub, GCR, or ECR).

```
# backend-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: backend-deployment
  labels:
    app: backend
spec:
  replicas: 1 # You can scale this up later by changing this value
  selector:
    matchLabels:
      app: backend
  template:
    metadata:
      labels:
        app: backend
    spec:
      containers:
      - name: backend
        # --- IMPORTANT ---
        # Replace 'your-docker-registry/your-backend-image:latest' with the actual path
to your image
        image: your-docker-registry/your-backend-image:latest
        ports:
        - containerPort: 8001 # The port your backend application listens on
        env:
        - name: MONGO_URL
          # This points to the internal MongoDB Service we created earlier
          value: mongodb://mongodb-service:27017
        - name: DB_NAME
```

```
        value: linkshare_db
      - name: JWT_SECRET
        # For development this is okay, but for production, use Kubernetes Secrets for
sensitive data
        value: "your-super-secure-jwt-secret-key-change-in-production"
      # --- Health Checks (Probes) ---
      livenessProbe:
        httpGet:
          path: /api/health # The endpoint Kubernetes will check
          port: 8001        # The port on the container to check
        initialDelaySeconds: 15 # Wait 15s before first check
        periodSeconds: 20       # Check every 20s
        timeoutSeconds: 5
        failureThreshold: 3     # Restart container after 3 failures
      readinessProbe:
        httpGet:
          path: /api/health
          port: 8001
        initialDelaySeconds: 5  # Start checking readiness after 5s
        periodSeconds: 10       # Check every 10s
        timeoutSeconds: 5
        failureThreshold: 3     # Stop sending traffic after 3 failures
```

2. backend-service.yaml

This file defines the Service for your backend. We use the ClusterIP type because this service only needs to be reachable from *within* the Kubernetes cluster (specifically, by your frontend). The frontend will use the stable DNS name backend-service to send API requests.

```
# backend-service.yaml
apiVersion: v1
kind: Service
metadata:
  name: backend-service # The stable, internal name for your backend service
  labels:
    app: backend
spec:
  selector:
```

```
    app: backend # This links the Service to the Pods created by the
backend-deployment
  ports:
    - protocol: TCP
      port: 8001       # The port this Service will listen on
      targetPort: 8001 # The port on the backend container to forward traffic to
  type: ClusterIP      # This service is for internal cluster communication only
```