Keycloak Folder Structure

```
keycloak/
├── docker-compose.yml              # Main orchestration file
├── .env                            # Environment variables
├── entrypoint.sh                   # Custom startup script
├── nginx/
│   ├── nginx.conf                  # Reverse proxy config
│   └── ssl/                        # SSL certificates (auto-generated)
├── postgres/
│   ├── init.sql                    # Database initialization
│   └── data/                       # Database data (volume mount)
├── keycloak/
│   ├── data/                       # Keycloak data directory
│   └── providers/                  # Custom providers/extensions
├── themes/
│   └── centuries-theme/
│       ├── theme.properties
│       ├── login/
│       │   ├── theme.properties
│       │   ├── login.ftl           # Custom login template
│       │   ├── register.ftl        # Custom registration template
│       │   └── resources/
│       │       ├── css/
│       │       │   └── login.css   # Custom styling
│       │       └── img/
│       │           └── logo.png    # Company logo
│       └── account/
│           ├── theme.properties
│           └── account.ftl
├── realms/
│   ├── home-realm.json             # Home app realm export
│   ├── legend-realm.json           # Legend app realm export
│   ├── admin-realm.json            # Admin app realm export
│   └── master-realm-config.json    # Master realm configuration
├── extensions/
│   ├── hyperledger-webhook/
│   │   ├── pom.xml                 # Maven build file
│   │   └── src/
│   │       └── main/
│   │           └── java/
│   │               └── com/
│   │                   └── centuriesmutual/
│   │                       └── keycloak/
```

```
|   |                   ├── HyperledgerEventListener.java
|   |                   └── HyperledgerWebhookProvider.java
|   └── compiled/
|       └── hyperledger-webhook-1.0.jar # Compiled extension
├── scripts/
|   ├── setup.sh                  # Initial setup script
|   ├── backup.sh                 # Backup script
|   └── restore.sh                # Restore script
└── monitoring/
    ├── prometheus.yml              # Monitoring configuration
    └── docker-compose.monitoring.yml   # Optional monitoring stack
```

Core Configuration Files

docker-compose.yml

yaml

```yaml
version: '3.8'

services:
  postgres:
    image: postgres:15-alpine
    container_name: keycloak-postgres
    environment:
      POSTGRES_DB: keycloak
      POSTGRES_USER: keycloak
      POSTGRES_PASSWORD: ${POSTGRES_PASSWORD}
    volumes:
      - ./postgres/data:/var/lib/postgresql/data
      - ./postgres/init.sql:/docker-entrypoint-initdb.d/init.sql
    networks:
      - keycloak-network
    restart: unless-stopped
    healthcheck:
      test: ["CMD-SHELL", "pg_isready -U keycloak"]
      interval: 30s
      timeout: 10s
      retries: 3

  keycloak:
    image: quay.io/keycloak/keycloak:24.0.1
    container_name: keycloak-server
    depends_on:
      postgres:
        condition: service_healthy
    environment:
      KC_DB: postgres
      KC_DB_URL: jdbc:postgresql://postgres:5432/keycloak
      KC_DB_USERNAME: keycloak
      KC_DB_PASSWORD: ${POSTGRES_PASSWORD}
      KC_HOSTNAME: ${KEYCLOAK_HOSTNAME}
      KC_HOSTNAME_STRICT: false
      KC_HTTP_ENABLED: true
      KC_PROXY: edge
      KC_HEALTH_ENABLED: true
      KC_METRICS_ENABLED: true
      KEYCLOAK_ADMIN: ${KEYCLOAK_ADMIN_USER}
      KEYCLOAK_ADMIN_PASSWORD: ${KEYCLOAK_ADMIN_PASSWORD}
    volumes:
      - ./keycloak/data:/opt/keycloak/data
      - ./themes:/opt/keycloak/themes
      - ./extensions/compiled:/opt/keycloak/providers
```

```
    - ./entrypoint.sh:/opt/keycloak/bin/custom-entrypoint.sh
  ports:
    - "8080:8080"
  networks:
    - keycloak-network
  restart: unless-stopped
  command:
    - /opt/keycloak/bin/custom-entrypoint.sh
  healthcheck:
    test: ["CMD-SHELL", "curl -f http://localhost:8080/health/ready || exit 1"]
    interval: 30s
    timeout: 10s
    retries: 5
    start_period: 120s

nginx:
  image: nginx:alpine
  container_name: keycloak-nginx
  depends_on:
    keycloak:
      condition: service_healthy
  ports:
    - "80:80"
    - "443:443"
  volumes:
    - ./nginx/nginx.conf:/etc/nginx/nginx.conf
    - ./nginx/ssl:/etc/nginx/ssl
    - /etc/letsencrypt:/etc/letsencrypt:ro
  networks:
    - keycloak-network
  restart: unless-stopped
  healthcheck:
    test: ["CMD-SHELL", "curl -f http://localhost/health || exit 1"]
    interval: 30s
    timeout: 10s
    retries: 3

networks:
  keycloak-network:
    driver: bridge

volumes:
  postgres-data:
  keycloak-data:
```

.env

```bash
# Database Configuration
POSTGRES_PASSWORD=your_super_secure_postgres_password_here_min_32_chars

# Keycloak Configuration
KEYCLOAK_HOSTNAME=auth.centuriesmutual.com
KEYCLOAK_ADMIN_USER=admin
KEYCLOAK_ADMIN_PASSWORD=your_super_secure_admin_password_here_min_16_chars

# SSL Configuration
SSL_EMAIL=admin@centuriesmutual.com

# Hyperledger Configuration
HYPERLEDGER_WEBHOOK_URL=https://api.centuriesmutual.com/webhooks/auth
HYPERLEDGER_API_KEY=your_hyperledger_api_key_here

# Monitoring (Optional)
PROMETHEUS_ENABLED=true
```

entrypoint.sh

```bash
```

```bash
#!/bin/bash

set -e

# Import custom themes and configurations
echo "Starting Keycloak with custom configuration..."

# Start Keycloak in development mode first to build
echo "Building Keycloak..."
/opt/keycloak/bin/kc.sh build

# Check if realm import is needed
if [ ! -f "/opt/keycloak/data/realm-imported" ]; then
    echo "Importing realms on first startup..."

    # Start Keycloak in background for realm import
    /opt/keycloak/bin/kc.sh start --optimized &
    KEYCLOAK_PID=$!

    # Wait for Keycloak to be ready
    echo "Waiting for Keycloak to start..."
    timeout=300
    while ! curl -f http://localhost:8080/health/ready > /dev/null 2>&1; do
        sleep 5
        timeout=$((timeout - 5))
        if [ $timeout -le 0 ]; then
            echo "Timeout waiting for Keycloak to start"
            exit 1
        fi
    done

    # Import realms
    echo "Importing realms..."

    # Get admin token
    ADMIN_TOKEN=$(curl -s -X POST \
        "http://localhost:8080/realms/master/protocol/openid-connect/token" \
        -H "Content-Type: application/x-www-form-urlencoded" \
        -d "username=${KEYCLOAK_ADMIN}" \
        -d "password=${KEYCLOAK_ADMIN_PASSWORD}" \
        -d "grant_type=password" \
        -d "client_id=admin-cli" | jq -r '.access_token')

    # Import each realm
    for realm_file in /opt/keycloak/themes/centuries-theme/realms/*.json; do
```

```
    if [ -f "$realm_file" ]; then
      echo "Importing $(basename "$realm_file")..."
      curl -X POST \
        "http://localhost:8080/admin/realms" \
        -H "Authorization: Bearer ${ADMIN_TOKEN}" \
        -H "Content-Type: application/json" \
        -d @"$realm_file"
    fi
  done

  # Mark import as completed
  touch /opt/keycloak/data/realm-imported

  # Stop the background process
  kill $KEYCLOAK_PID
  wait $KEYCLOAK_PID 2>/dev/null || true
fi

# Start Keycloak normally
echo "Starting Keycloak in production mode..."
exec /opt/keycloak/bin/kc.sh start --optimized
```

nginx/nginx.conf

```
nginx
```

```
events {
    worker_connections 1024;
}

http {
    upstream keycloak {
        server keycloak:8080;
    }

    # Rate limiting
    limit_req_zone $binary_remote_addr zone=auth:10m rate=10r/m;
    limit_req_zone $binary_remote_addr zone=api:10m rate=100r/m;

    # Security headers
    add_header X-Frame-Options DENY always;
    add_header X-Content-Type-Options nosniff always;
    add_header X-XSS-Protection "1; mode=block" always;
    add_header Referrer-Policy "strict-origin-when-cross-origin" always;
    add_header Content-Security-Policy "default-src 'self'; script-src 'self' 'unsafe-inline'; style-src 'self' 'unsafe-ir

    # SSL Configuration
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384;
    ssl_prefer_server_ciphers off;
    ssl_session_cache shared:SSL:10m;
    ssl_session_tickets off;

    # Redirect HTTP to HTTPS
    server {
        listen 80;
        server_name _;
        return 301 https://$host$request_uri;
    }

    # Health check endpoint (HTTP only)
    server {
        listen 80;
        server_name localhost;
        location /health {
            access_log off;
            return 200 "healthy\n";
            add_header Content-Type text/plain;
        }
    }
```

```
# Main HTTPS server
server {
    listen 443 ssl http2;
    server_name auth.centuriesmutual.com;

    ssl_certificate /etc/letsencrypt/live/auth.centuriesmutual.com/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/auth.centuriesmutual.com/privkey.pem;

    # Security
    client_max_body_size 10M;

    # Auth endpoints with rate limiting
    location ~ ^/(realms/.*/protocol/openid-connect/(auth|token|logout)) {
        limit_req zone=auth burst=20 nodelay;
        proxy_pass http://keycloak;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto https;
        proxy_set_header X-Forwarded-Port 443;

        # CORS headers for mobile apps
        add_header 'Access-Control-Allow-Origin' '*' always;
        add_header 'Access-Control-Allow-Methods' 'GET, POST, OPTIONS' always;
        add_header 'Access-Control-Allow-Headers' 'DNT,User-Agent,X-Requested-With,If-Modified-Since,Cache

        if ($request_method = 'OPTIONS') {
            add_header 'Access-Control-Max-Age' 1728000;
            add_header 'Content-Type' 'text/plain; charset=utf-8';
            add_header 'Content-Length' 0;
            return 204;
        }
    }

    # Admin endpoints with stricter rate limiting
    location /admin {
        limit_req zone=api burst=10 nodelay;
        proxy_pass http://keycloak;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto https;
        proxy_set_header X-Forwarded-Port 443;
    }

    # All other endpoints
```

```
    location / {
        limit_req zone=api burst=50 nodelay;
        proxy_pass http://keycloak;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto https;
        proxy_set_header X-Forwarded-Port 443;

        # Additional headers for Keycloak
        proxy_buffer_size 128k;
        proxy_buffers 4 256k;
        proxy_busy_buffers_size 256k;
    }
  }
}
```

postgres/init.sql

sql

```sql
-- Create additional databases for future expansion
CREATE DATABASE hyperledger_audit;
CREATE DATABASE user_analytics;

-- Create users with limited permissions
CREATE USER keycloak_readonly WITH PASSWORD 'readonly_secure_password';
CREATE USER analytics_user WITH PASSWORD 'analytics_secure_password';

-- Grant permissions
GRANT SELECT ON ALL TABLES IN SCHEMA public TO keycloak_readonly;
GRANT ALL PRIVILEGES ON DATABASE user_analytics TO analytics_user;

-- Create audit table for login events
\c hyperledger_audit;

CREATE TABLE auth_events (
    id SERIAL PRIMARY KEY,
    user_id VARCHAR(255) NOT NULL,
    realm VARCHAR(100) NOT NULL,
    client_id VARCHAR(100) NOT NULL,
    event_type VARCHAR(50) NOT NULL,
    ip_address INET,
    user_agent TEXT,
    session_id VARCHAR(255),
    timestamp TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
    details JSONB
);

CREATE INDEX idx_auth_events_user_id ON auth_events(user_id);
CREATE INDEX idx_auth_events_timestamp ON auth_events(timestamp);
CREATE INDEX idx_auth_events_event_type ON auth_events(event_type);
```