Hylur Knowledge Foundation - Deployment Guide

Prerequisites

Before starting the deployment, you’ll need:

1. **GitHub Personal Access Token** with repo permissions
2. **Google Cloud Platform account** with billing enabled
3. **Domain ownership** of hylur.net

Step 1: GitHub Repository Setup

1. Create a GitHub Personal Access Token:
   - Go to GitHub Settings > Developer settings > Personal access tokens > Tokens (classic)
   - Generate new token with `repo`, `workflow`, and `admin:repo_hook` permissions
   - Copy the token

2. Set the GitHub token and create repository:
```
bash
   cd /home/ubuntu/hylur-knowledge-foundation
   export GITHUB_TOKEN="your_github_token_here"
   gh auth login --with-token <<< "$GITHUB_TOKEN"
   gh repo create hylur-knowledge-foundation --public --description "Hylur Knowledge Foundation - Document management and AI chatbot platform" --source . --push
```

Step 2: Google Cloud Setup

1. **Create a new Google Cloud Project:**
   ```bash
   # Authenticate with Google Cloud
   gcloud auth login

# Create a new project (replace PROJECT_ID with your desired project ID)
export PROJECT_ID=”hylur-knowledge-foundation”
gcloud projects create $PROJECT_ID –name=”Hylur Knowledge Foundation”
gcloud config set project $PROJECT_ID

# Enable billing (you’ll need to do this in the console)
echo “Please enable billing for project $PROJECT_ID in the Google Cloud Console”
```

1. **Enable required APIs:**
```
bash
   gcloud services enable cloudbuild.googleapis.com
   gcloud services enable run.googleapis.com
   gcloud services enable sql-component.googleapis.com
   gcloud services enable sqladmin.googleapis.com
```

```
   gcloud services enable secretmanager.googleapis.com
   gcloud services enable containerregistry.googleapis.com
```

2. **Create Cloud SQL PostgreSQL instance:**

```bash
# Create the SQL instance
gcloud sql instances create hylur-postgres \
–database-version=POSTGRES_14 \
–tier=db-f1-micro \
–region=us-central1 \
–storage-type=SSD \
–storage-size=10GB \
–authorized-networks=0.0.0.0/0

# Create database
gcloud sql databases create hylur_db –instance=hylur-postgres

# Create user
gcloud sql users create hylur_user –instance=hylur-postgres –password=your_secure_password_here

# Get connection name
export INSTANCE_CONNECTION_NAME=$(gcloud sql instances describe hylur-postgres –format=”value(connectionName)”)
echo “Instance connection name: $INSTANCE_CONNECTION_NAME”
```

1. **Store secrets in Secret Manager:**

```bash
# Database URL
export DATABASE_URL=”postgresql://hylur_user:your_secure_password_here@/hylur_db?host=/cloudsql/$INSTANCE_CONNECTION_NAME”
echo -n “$DATABASE_URL” | gcloud secrets create DATABASE_URL –data-file=-

# NextAuth Secret (generate a random secret)
export NEXTAUTH_SECRET=$(openssl rand -base64 32)
echo -n “$NEXTAUTH_SECRET” | gcloud secrets create NEXTAUTH_SECRET –data-file=-

# AbacusAI API Key (use the existing one)
echo -n “bf597d4ba720483199a6149f02065c67” | gcloud secrets create ABACUSAI_API_KEY –data-file=-
```

1. **Create service account for GitHub Actions:**

```bash
# Create service account
gcloud iam service-accounts create github-actions \
–description=”Service account for GitHub Actions” \
–display-name=”GitHub Actions”

# Grant necessary roles
gcloud projects add-iam-policy-binding $PROJECT_ID \
–member=”serviceAccount:github-actions@$PROJECT_ID.iam.gserviceaccount.com” \
–role=”roles/cloudbuild.builds.editor”
```

```
gcloud projects add-iam-policy-binding $PROJECT_ID \
–member=”serviceAccount:github-actions@$PROJECT_ID.iam.gserviceaccount.com” \
–role=”roles/run.admin”

gcloud projects add-iam-policy-binding $PROJECT_ID \
–member=”serviceAccount:github-actions@$PROJECT_ID.iam.gserviceaccount.com” \
–role=”roles/storage.admin”

gcloud projects add-iam-policy-binding $PROJECT_ID \
–member=”serviceAccount:github-actions@$PROJECT_ID.iam.gserviceaccount.com” \
–role=”roles/secretmanager.secretAccessor”

# Create and download service account key
gcloud iam service-accounts keys create github-actions-key.json \
–iam-account=github-actions@$PROJECT_ID.iam.gserviceaccount.com
```

Step 3: GitHub Secrets Configuration

Add the following secrets to your GitHub repository:

1. Go to your GitHub repository > Settings > Secrets and variables > Actions
2. Add these repository secrets:
   - `GCP_PROJECT_ID`: Your Google Cloud project ID
   - `GCP_SA_KEY`: Contents of the `github-actions-key.json` file

Step 4: Initial Deployment

1. **Deploy the application:**
   `bash`
   ```
      # Build and deploy using Cloud Build
      gcloud builds submit --config cloudbuild.yaml
   ```

2. **Set up custom domain:**
   ```bash
   # Create domain mapping
   gcloud run domain-mappings create –service=hylur-knowledge-foundation –domain=hylur.net –region=us-central1

# Get the DNS records to configure
gcloud run domain-mappings describe –domain=hylur.net –region=us-central1
```

Step 5: DNS Configuration

Configure your DNS provider (where hylur.net is registered) with the records provided by the previous command. Typically:

1. Add an A record pointing to the IP address provided
2. Add AAAA record for IPv6 (if provided)
3. Add CNAME record for www subdomain (if desired)

Step 6: Database Migration and Seeding

The Cloud Build process will automatically run database migrations and seed the database with founder accounts.

Step 7: Verification

1. **Test the deployment:**
   - Visit https://hylur.net (after DNS propagation)
   - Test authentication with founder accounts:
     - haukur@hylur.net
     - leif@hylur.net
     - Test file uploads
     - Test AI chatbot functionality

Environment Variables Summary

The application uses these environment variables:

- `DATABASE_URL` : PostgreSQL connection string
- `NEXTAUTH_URL` : https://hylur.net
- `NEXTAUTH_SECRET` : Random secret for NextAuth.js
- `ABACUSAI_API_KEY` : API key for AbacusAI chatbot integration

Monitoring and Maintenance

1. **View logs:**
   ```
   bash
      gcloud run services logs read hylur-knowledge-foundation --region=us-central1
   ```

2. **Update the application:**
   - Push changes to the main/master branch
   - GitHub Actions will automatically trigger deployment

3. **Scale the application:**
   ```
   bash
      gcloud run services update hylur-knowledge-foundation \
        --region=us-central1 \
        --min-instances=1 \
        --max-instances=10
   ```

Security Considerations

1. **Database Security:**
   - Remove the `0.0.0.0/0` authorized network after deployment
   - Use Cloud SQL Proxy for secure connections

2. **Secrets Management:**
   - All sensitive data is stored in Google Secret Manager
   - Service accounts follow principle of least privilege

3. **HTTPS:**
   - Cloud Run automatically provides SSL certificates
   - All traffic is encrypted in transit

Troubleshooting

1. **Build failures:**
   - Check Cloud Build logs in Google Cloud Console
   - Verify all secrets are properly configured

2. **Database connection issues:**
   - Verify Cloud SQL instance is running
   - Check database credentials in Secret Manager

3. **Domain mapping issues:**
   - Verify DNS records are correctly configured
   - Allow time for DNS propagation (up to 48 hours)

Cost Optimization

1. **Cloud Run:** Pay per request, scales to zero
2. **Cloud SQL:** Use smallest instance size for development
3. **Storage:** Clean up old container images periodically

For production, consider upgrading to larger Cloud SQL instances and enabling high availability.