GitHub Secrets Setup Guide

This guide will help you set up the required GitHub secrets for your Azure deployment.

Required Secrets

Your deployment needs two critical secrets:

1. DATABASE_URL

- **Purpose**: PostgreSQL connection string for Azure Database
- **Format**: `postgres://username:password@host:port/dbname`
- **Example**: `postgres://myuser:mypassword@myserver.postgres.database.azure.com:5432/mydatabase?sslmode=require`

2. SECRET_KEY

- **Purpose**: Application secret key for JWT tokens and session management
- **Requirements**: Should be a secure, random string (at least 32 characters)

Step-by-Step Setup

Step 1: Access Repository Settings

1. Go to your GitHub repository: https://github.com/d1hawkins/AI-EventPlanner
2. Click on **"Settings"** (top navigation bar)
3. In the left sidebar, click **"Secrets and variables"**
4. Click **"Actions"**

Step 2: Add DATABASE_URL Secret

1. Click **"New repository secret"**

2. Name: `DATABASE_URL`

3. Value: Your Azure PostgreSQL connection string

   ```
   postgres://username:password@hostname:5432/database_name?sslmode=require
   ```

   To get your Azure PostgreSQL connection string:

   - Go to Azure Portal → PostgreSQL servers
   - Select your server
   - Click "Connection strings" in the left menu

- Copy the connection string and replace `{your_password}` with your actual password

Step 3: Add SECRET_KEY Secret

1. Click **"New repository secret"**
2. Name: `SECRET_KEY`
3. Value: A secure random string (see generation methods below)

Secret Key Generation Methods

Method 1: Python (Recommended)

```
import secrets
print(secrets.token_urlsafe(32))
```

Method 2: OpenSSL

```
openssl rand -base64 32
```

Method 3: Online Generator

Use a secure password generator to create a 32+ character random string.

Verification

After adding both secrets:

1. Go to **Settings** → **Secrets and variables** → **Actions**
2. You should see both `DATABASE_URL` and `SECRET_KEY` listed
3. Trigger a new deployment by pushing a commit or manually running the workflow

Security Best Practices

- ✅ Never commit secrets to your repository
- ✅ Use different SECRET_KEY values for different environments
- ✅ Rotate secrets periodically
- ✅ Ensure DATABASE_URL includes `sslmode=require` for Azure PostgreSQL
- ✅ Keep your Azure PostgreSQL password secure and complex

Troubleshooting

Common Issues:

1. **Invalid DATABASE_URL format**

- Ensure the format is exactly: `postgres://user:pass@host:port/db?sslmode=require`
- Check for special characters in password that need URL encoding

2. **SECRET_KEY too short**
   - Use at least 32 characters for security
   - Avoid simple passwords or predictable patterns
3. **Connection refused errors**
   - Verify your Azure PostgreSQL server is running
   - Check firewall rules allow GitHub Actions IP ranges
   - Confirm the database name exists

Next Steps

1. Add both secrets using the steps above
2. Go to **Actions** tab in your repository
3. Re-run the failed deployment workflow
4. Monitor the deployment logs for success

Additional Resources

- GitHub Secrets Documentation
- Azure PostgreSQL Connection Strings
- AZURE_DEPLOYMENT_SETUP_GUIDE.md for complete Azure setup