App Studio 4.1 Deployment Guide

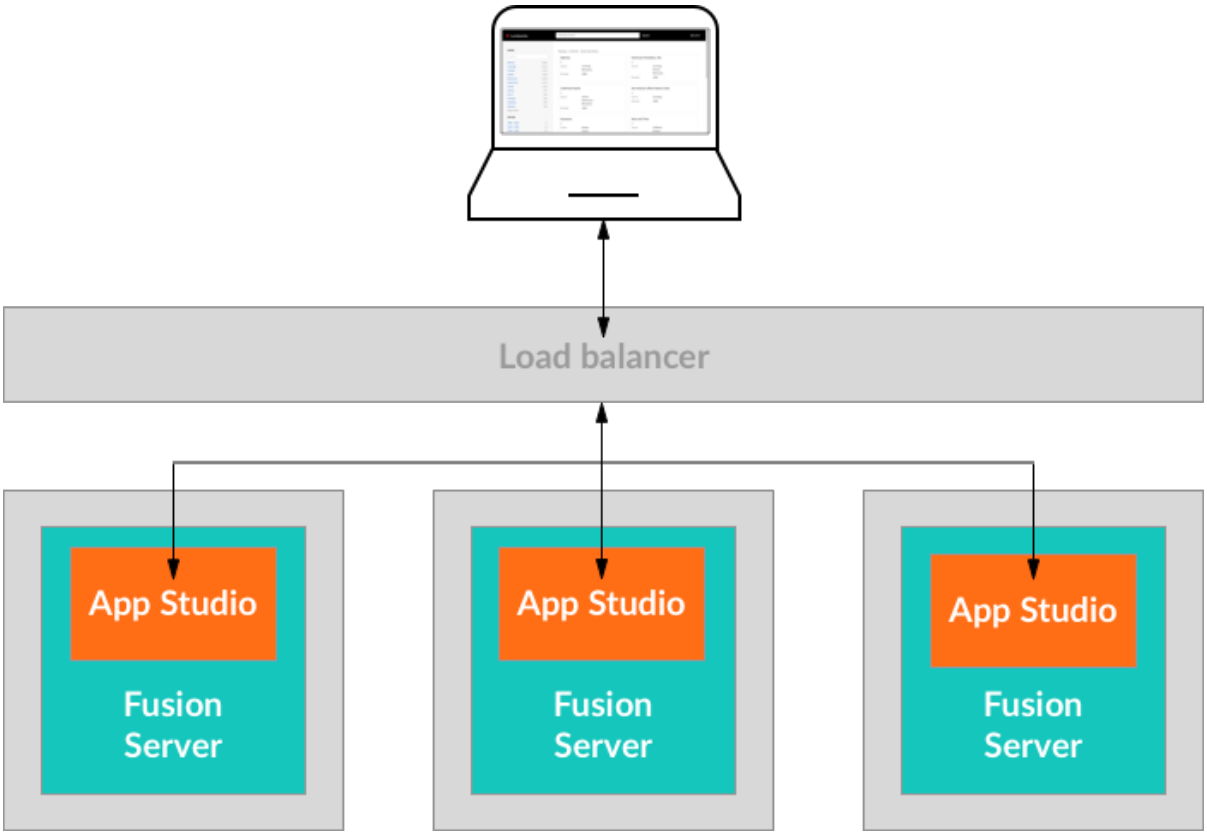
2018-12-11

Table of Contents

Fusion Web App Deployment

To deploy App Studio in Fusion Server, you develop your search interface, then publish it to Fusion's Webapps service. This deployment type leverages your existing production environment, requiring no dedicated nodes for App Studio. However, you may need to deploy additional Fusion nodes if you anticipate significant traffic to your search interface.

UI deployment overview



Here’s an overview of the steps:

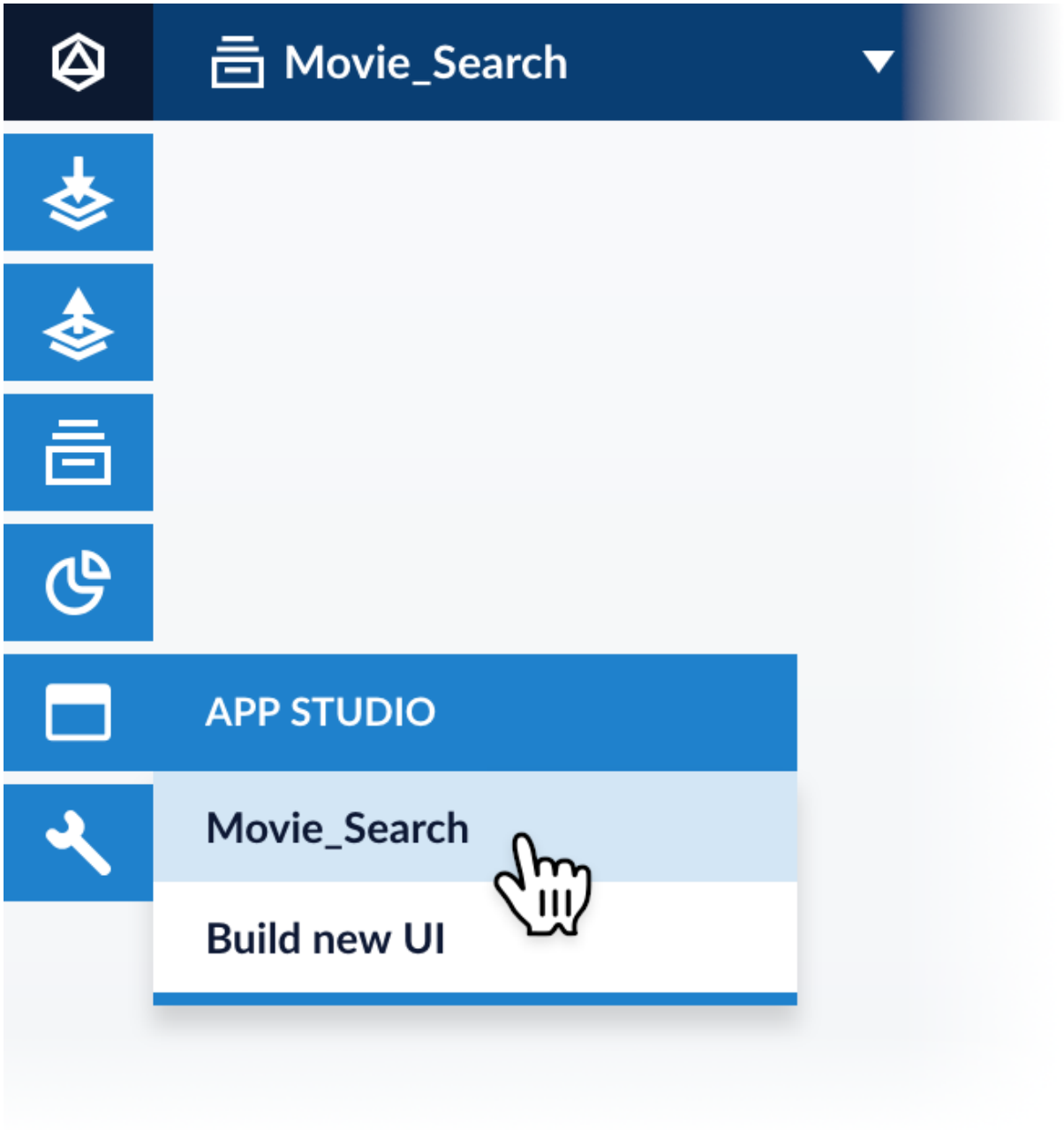
1. Develop your search interface.
2. Publish your UI to Fusion Server using the instructions below.

Your Web app then becomes accessible at <code><a href="http://localhost:8764/webapps/<project-name&gt" class="bare">http://localhost:8764/webapps/<project-name&gt;</code>.

Publishing a search interface to Fusion Server

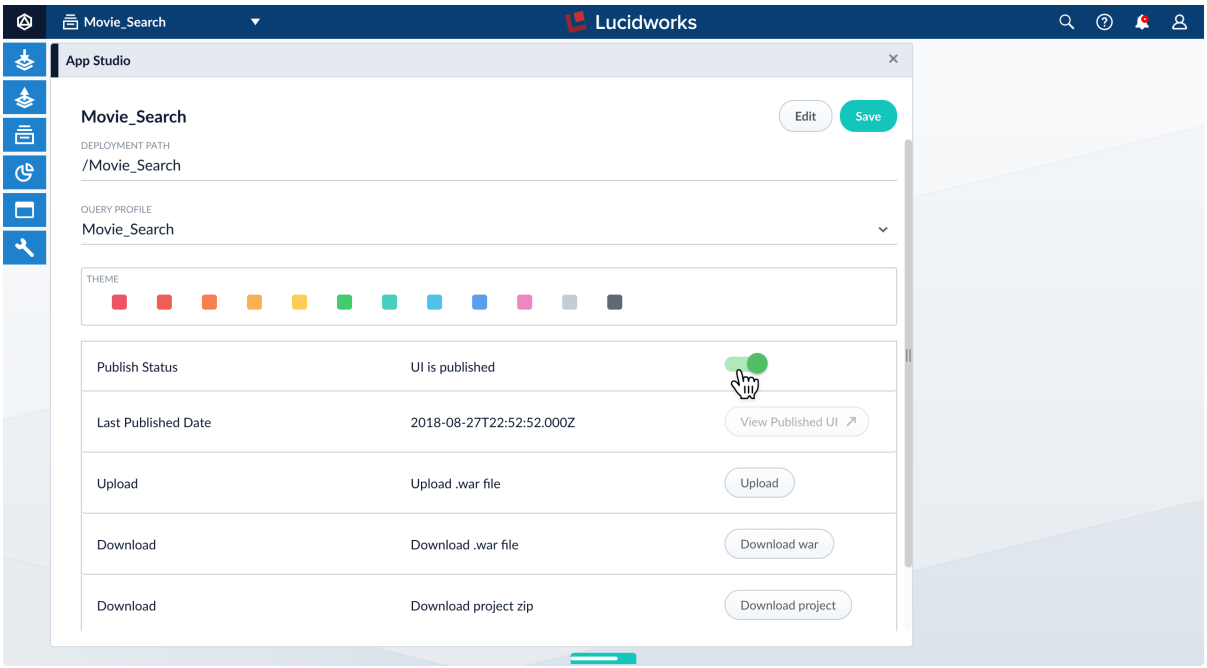
These steps publish your search interface to a Fusion instance (and its cluster) where the App Studio project already exists.

1. In the Fusion workspace, navigate to **App Studio** and select your search interface.



   The configuration panel appears.

2. Next to **Publish Status**, click the toggle:



3. Click **Save**.

   The interface is only published after you click **Save**, which also enables the **View Published UI** button.

4. Click **View Published UI** to launch a new window where you can view the interface as an end user.

   Your published interface is available at <code><a href="http://localhost:8764/webapps/<project-name&gt" class="bare">http://localhost:8764/webapps/<project-name&gt;</code>.

Migrating a search interface to other Fusion hosts

If you've published your search interface on one Fusion Host in a cluster, then it is published throughout the cluster.

There are two ways to migrate a search interface to another Fusion cluster:

Export and import the Fusion app	**Download and upload the .war file**
This migrates all objects in the Fusion app, such as datasources, query profiles, schedules, and so on, in addition to the search interface. The search interface will be editable on the target hosts. For instructions, see Working With Apps in Fusion Server's Getting Started guide.	This migrates only the search interface. The search interface will not be editable on the target hosts. If you are uploading the file into a Fusion app other than the one in which it was created, the new Fusion app must include a query profile and data fields whose names are identical to the ones already configured in the search interface. For instructions on download or building a `.war` file and uploading it to Fusion, see below.

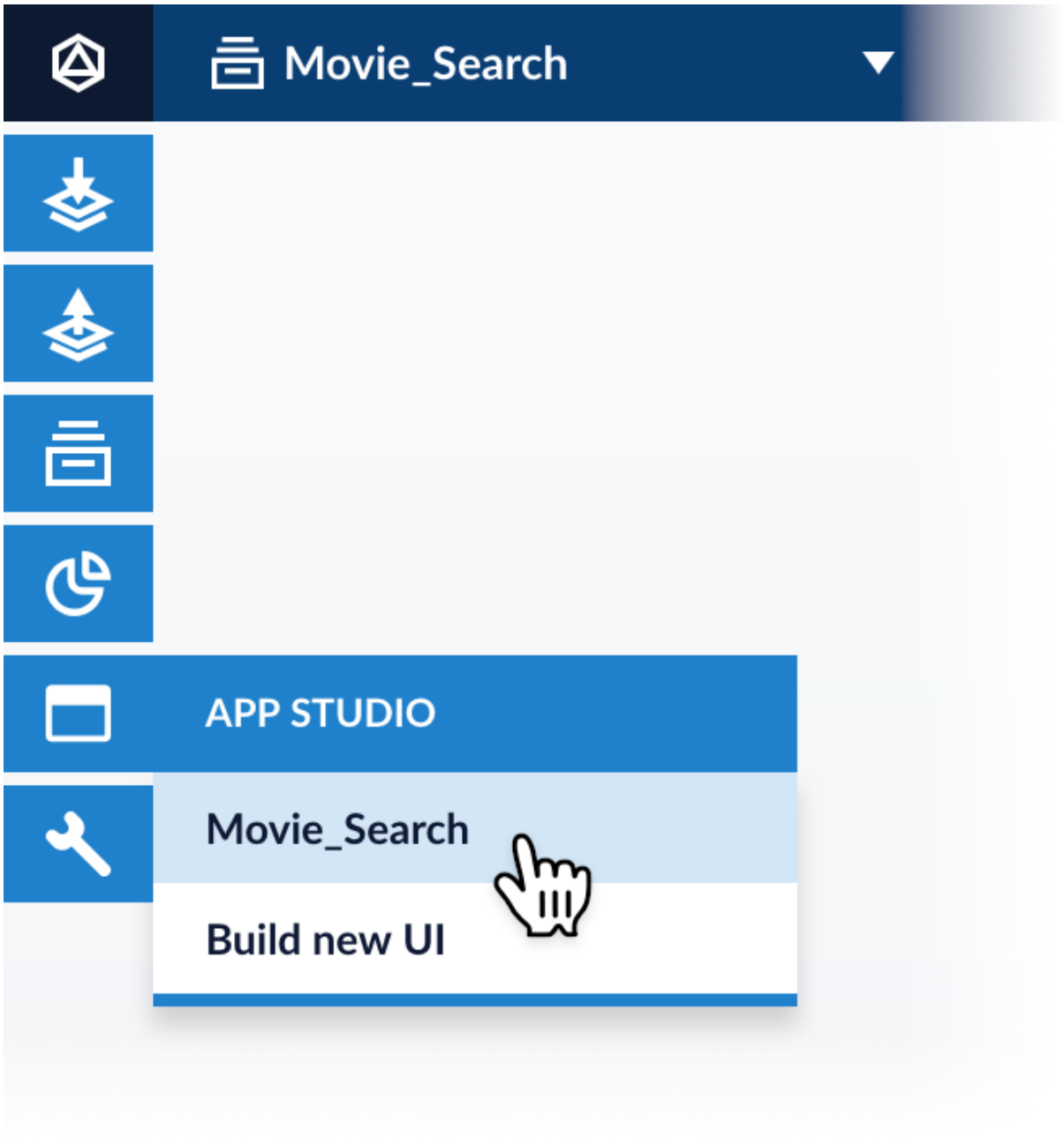
Download or build a `.war` file

You distribute a search interface as a `.war` file.

You can download a `.war` file from the Fusion UI or create one from a downloaded project.

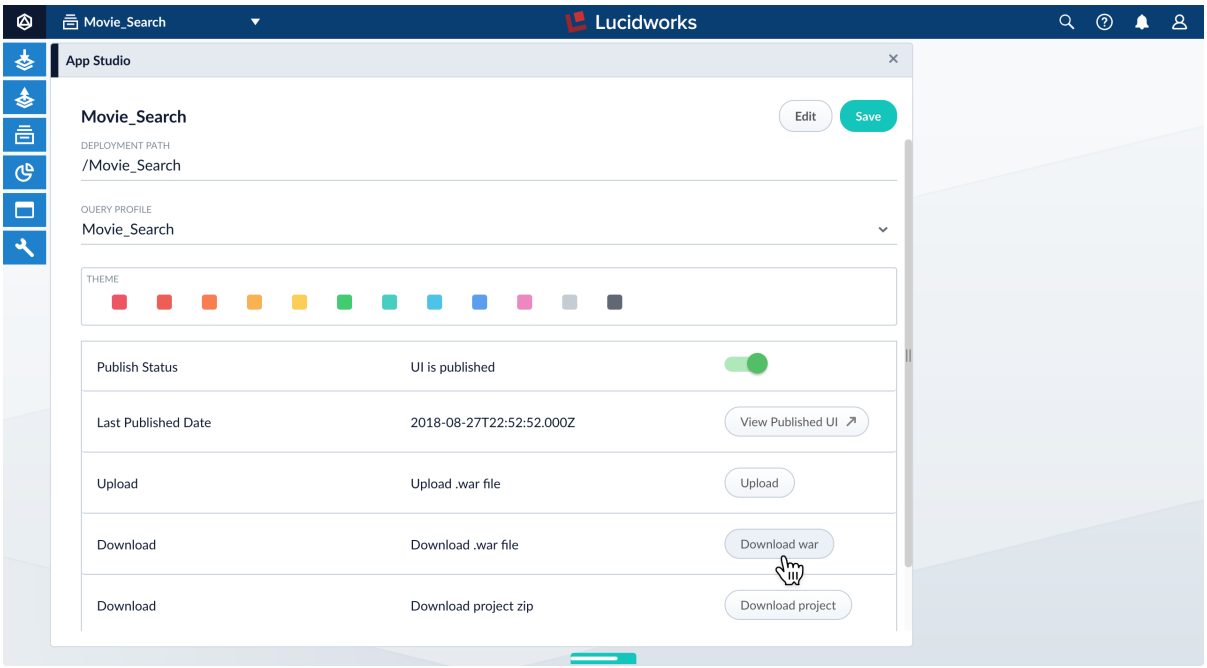
Downloading a .war file

1. In the Fusion workspace, navigate to **App Studio** and select your search interface.



The configuration panel appears.

2. Click **Download war**:



Building a .war file

A downloaded project comes with an `app-studio` script that creates a `.war` file.

1. In a shell window, switch to your project directory:

   ```
   cd /path/to/Project_Directory
   ```

2. Run the following command:

   ```
   ./app-studio package
   ```

The script creates the following files in the `dist` directory of your project:

- `search-app-4.1.x-y.y.y.jar`

  This is a self-executing application file.

- `search-app-project.zip`

  Use this file to share your project with other search interface developers.

- `search-app.war`

  You can upload this file to Fusion to deploy it. See the Deployment Guide.

Upload the .war file

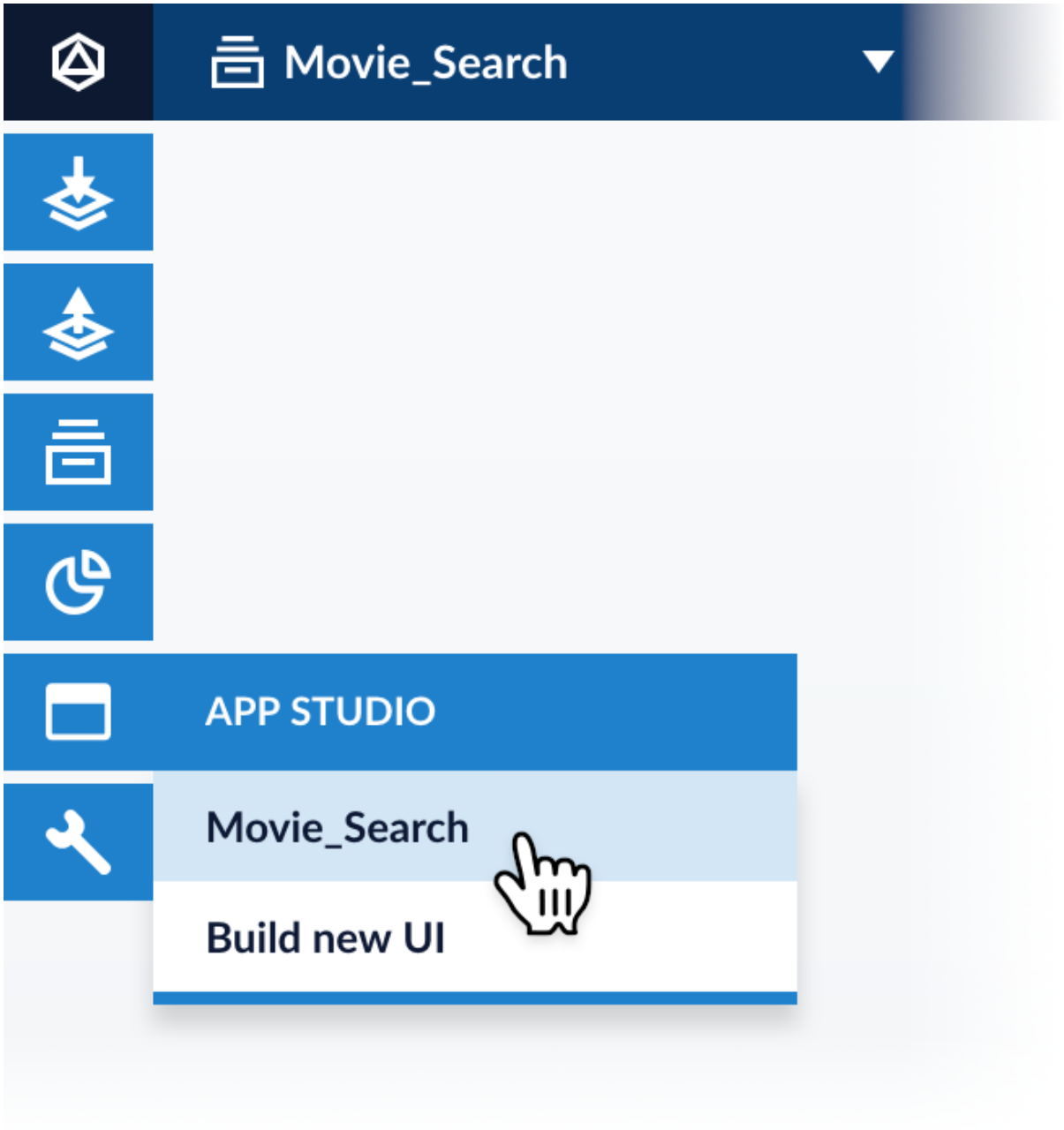
You can upload the `.war` file through the Fusion UI or the REST API.

Uploading with the Fusion UI

You can upload your project (as a `.war` file) in the App Studio configuration panel.

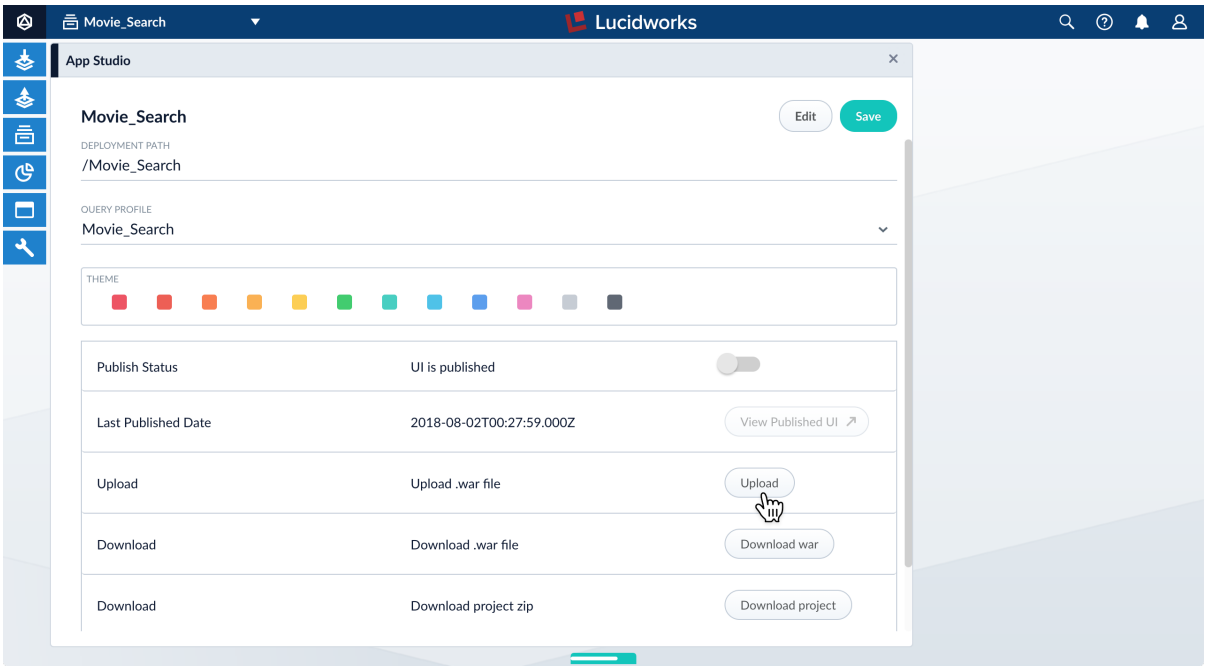
Note	If you are uploading the file into a Fusion app other than the one in which it was created, the new Fusion app must include a query profile and data fields whose names are identical to the ones already configured in the search interface.

1. From the App Studio menu, select your project:



   The project configuration panel appears.

2. Click **Upload**.



Fusion prompts you to confirm that you want to upload a project that will delete the existing one.

3. Click **Yes, continue**.
4. Navigate to your `.war` file and select it.

   Once the upload is complete, you can edit or publish the uploaded project.

Uploading with the REST API

Use the `webapps/{id}/war` endpoint of the Webapps API to upload a `.war` file, as in this example:

```
curl -u admin:password123 -X PUT http://localhost:8764/api/webapps/Movie_Search/war -F 'file=@movies.war'
```

Running a dedicated node for a search interface

When deploying Fusion nodes dedicated to serving one or more search interfaces, only the following Fusion services need to run:

- `agent`
- `api`
- `log-shipper`
- `webapps`

When starting a dedicated node, you can minimize the number of running services by starting it like this:

```
fusion/4.1.x/bin/agent start
fusion/4.1.x/bin/api start
fusion/4.1.x/bin/log-shipping start
fusion/4.1.x/bin/webapps start
```

Standalone Deployments

There are two ways to deploy a standalone search interface:

- Self-contained application

  This deployment type has an embedded Tomcat server and is ready to deploy in any production environment.

- Self-executing application

  This is a `.jar` package.

Self-Contained Deployment

This deployment type has an embedded Tomcat server and is ready to deploy in any production environment.

Compiling a self-contained application

To compile the application, run the `app-studio` script with the `dist` target, like this:

Unix:

```
./app-studio dist
```

Windows:

```
app-studio.bat dist
```

You can find the output in the `dist` directory:

```
search-app-project.zip
search-app-standalone/
    README.md
    app/
    build/
    config/
    keystore.jks
    lib
    search-app-start.bat
    search-app-stop.bat
    search-app-stop.ps1
    search-app.sh
```

The `search-app-project.zip` is a snapshot of your project at the time you compiled the application. The `search-app-standalone/` directory is the self-contained application.

Self-Executing Deployment

In this deployment type, your search interface runs within the Java runtime environment (JRE).

Compiling a .jar file

A downloaded project comes with an `app-studio` script that creates a `.jar` file.

1. In a shell window, switch to your project directory:

   ```
   cd /path/to/Project_Directory
   ```

2. Run the following command:

   ```
   ./app-studio package
   ```

The script creates the following files in the `dist` directory of your project:

- `search-app-4.1.x-y.y.y.jar`

  This is your self-executing application file.

- `search-app-project.zip`

  Use this file to share your project with other search interface developers.

- `search-app.war`

  You can upload this file to Fusion to deploy it as a Fusion Web app.

Launching a self-executing deployment

```
java -jar search-app-4.1.x-y.y.y.jar
```

SSL Configuration

App Studio can be served over HTTPS using SSL encryption. You can use the default keystore for development and testing, or use your own keystore for production. Then, invoke the App Studio startup script using the SSL parameters.

SSL keystore

We include a keystore file with a default self-signed key for development and testing.

For proper security in a production environment, import your own keystore into the `keystore.jks` file, or copy it to a new file. If you copy it to a new file, use the `-Dtwigkit.keystore.file` (described below) to specify its location.

SSL parameters

To enable SSL, you specify the following parameters on the command line when invoking the startup script:

Parameter	Description	Default
`-Dtwigkit.ssl=true`	Enable SSL.	`false`
`-Dtwigkit.https.port`	Set the port.	`8765`
`-Dtwigkit.keystore.file`	The keystore path/filename, relative to the `app-studio` directory.	`keystore.jks`
`-Dtwigkit.keystore.password`	The keystore password.	`p4ssw0rd`
`-Dtwigkit.keystore.alias`	The name of the key in the keystore to be used.	`default-key`

Encrypting Sensitive Values

In a downloaded project, the `bin/twigcrypt/twigcrypt.sh` utility is available to encrypt sensitive string values, such a passwords, at the command line. It uses a two-way encryption mechanism so anywhere in the code this is used the value can be decrypted.

To encrypt a value, run the following (note the single quotes around `yourSensitiveValue`):

```
./twigcrypt.sh yourSecretSeed 'yourSensitiveValue'
```

This outputs an encrypted string. You must copy the whole string and paste into your configuration file.

For example, in your `src/main/resources/conf/platforms/fusion/fusion.conf` file, add:

```
username:jbloggs
password:Enc(ABC123==)
```

Then you must also configure the seed in the application's security configuration, in `src/main/resources/conf/security/security.conf`:

```
password: yourSecretSeed
```

Wherever this configuration parameter is used, it will be decrypted back to plain text at the time it's used.

Note	Lucidworks recommends using a randomly-generated **alphanumeric** seed (special characters can cause problems).