

# Quick Start



## Installation

You must contact [Aspire Support](#) in order to install the StageR Staging Repository.

## On this page

- [Run StageR](#)
- [Download and Configure](#)
- [Storage](#)
- [Environment Configurations](#)
  - [All configuration example](#)
  - [Environment specific configuration \(secure.js\)](#)
  - [Start StageR with a specific environment configuration](#)

## Run StageR

StageR runs on top of [NodeJS](#) and uses a No-SQL database for storage. It uses [MongoDB](#) by default.

## Download and Configure

1. Download and install [NodeJS](#).
2. Download and install [MongoDB](#).
3. Download StageR latest source [here](#) (or download latest [binary](#) and skip step 4 and 5)
4. Configure npm to connect to Search and Content Analytics' artifactory repository for local dependencies:

- a. Open cmd as Administrator

```
npm install npmrc -g
```

```
npmrc -c local
```

```
npm config set registry https://repository.searchtechnologies.com/artifactory/api/npm/npm-virtual
```

- b. Get an Authentication token (using *curl* or *Postman*)

```
curl -u<USERNAME>:<PASSWORD> https://repository.searchtechnologies.com/artifactory/api/npm/auth
```

- c. Append a response to: C:\Users\<USERNAME>\.npmrcs\local



It seems that some versions of software will create C:\Users\<USERNAME>\.npmrcs\default when you perform step a) above. If you have a *default* file (rather than a *local* file in .npmrcs), append the authentication token to that. You should have a single file that has content similar to the following:

```
registry=http://repository.searchtechnologies.com/artifactory/api/npm/npm-virtual
_auth = 124567890abcdefghijklmnopqrstuvwxyz1234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ
always-auth = true
email = my.email@my.company.com
```

d. Save the file.

5. Install dependencies.

- Unzip the stager-1.0.0.zip file (downloaded in the previous step)
- Change to this directory: `npm install --production`

```
npm install --production
```

6. Start MongoDB.

```
mongod
```

7. Launch StageR.

```
node server.js
```

## Storage

---

StageR data is stored in a No-SQL database.

- [MongoDB](#) is the default storage option

You can develop custom plugins for other No-SQL databases.

StageR will create the database automatically, including all necessary collections (*contents*, *transactions*, *settings*, *foreign keys*, and *reprocesses*) that are used by each **Storage Unit**.

## Environment Configurations

---

StageR can hold the configuration for multiple environments. The `config/env` folder holds the different configuration files for all environments. An environment configuration will inherit the configuration present in the `config/env/all.js` file and will overwrite any configuration field by redefining it on the environments file. New configuration fields can be added to the environment file.

### All configuration example

```
'use strict'

var path = require('path')
module.exports = {
  app: {
    title: 'StageR',
    description: 'Staging Repository',
    keywords: 'Storage Unit, NoSQL, Replication, Content Processing, Staging'
  },
  port: process.env.PORT || 3000,
  templateEngine: 'swig',
  dbConn: {
    type: 'mongo',
    hbase: {
      server: 'centos-hadoop3',
      port: 9090
    },
  },
  mongo: {
    namespace: 'stager-',
    server: 'localhost:27017',
    connectionOptions: {
      /* Uncomment to add secure configuration
      server: {
        ssl: true,
```

```

        sslValidate: true,
        sslCA: fs.readFileSync('./config/sslcerts/mongo/cacert.pem'),
        sslKey: fs.readFileSync('./config/sslcerts/mongo/testuser_key.pem'),
        sslCert: fs.readFileSync('./config/sslcerts/mongo/testuser_cert.crt'),
        sslPass: '123456'
    },
    user: 'CN=testuser-aws,OU=CS,O=SearchTech,L=Herndon,ST=VA,C=US',
    auth:{
        authMechanism: 'MONGODB-X509'
    }*/
},
sharding: false
}
},
keyManager: {
    type: 'filebased',
    filebased: {
        keysNumber: 5,
        masterKeyLocation: 'config/MasterKey.txt',
        dbName: 'DEK'
    },
    basic: {
        masterKey: 'MTIzNDU2Nzg5MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMzI='
    }
},
reqBodySize: '1gb',
reprocessQueue: {
    batchSize: 20,
    querySize: 40,
    timeout: 5000
},
replication: {
    compression: true
},
logging: {
    express: {
        request: [
            // 'body', //only add if it's safe to display data
            'connection.remoteAddress',
            'authorizedCN'
        ],
        response: [
            // 'body' //only add if it's safe to display data
        ]
    },
    transports: {
        console: {
            level: 'error',
            enable: true
        },
        file: {
            level: 'info',
            enable: false
        }
    }
},
response: {
    showStack: true
}
},
workers: {
    restapi: 1,
    reprocess: 1,
    replication: 1
},
fileHandler: {
    type: 'local',
    local: {
        destination: path.resolve(process.cwd(), 'uploads')
    }
}
},

```

```
tempFolder: path.resolve(process.cwd(), 'tmp'),
zoneId: 'APP_ZONE'
}
```

## Environment specific configuration (secure.js)

In this example, the port and clusterMode parameter will be overridden. authList, logging, keyLocation, certLocation, caLocation, and passphrase will be added to the applications configuration.

```
'use strict'

module.exports = {
  port: 44300,
  logging: {
    express: {
      request: [
        // 'body', //only add if it's safe to display data
        'connection.remoteAddress',
        'authorizedCN'
      ],
      response: [
        // 'body' //only add if it's safe to display data
      ]
    },
  },
  transports: {
    console: {
      level: 'info',
      enable: true
    },
    file: {
      level: 'info',
      enable: true
    }
  },
  response: {
    showStack: false
  },
  clusterMode: false,
  secure: {
    adminUser: 'TestUser',
    keyLocation: 'config/sslcerts/server/server.key',
    certLocation: 'config/sslcerts/server/server.crt',
    caLocation: 'config/sslcerts/ca/ca.crt',
    passphrase: '123456',
    requestCert: true,
    rejectUnauthorized: false
  }
}
```

## Start StageR with a specific environment configuration

1. Set the environment configuration name to the NODE\_ENV environment variable.
2. If the variable is not set, the application will start with the **development** configuration by default.