# ISS Secure File Repository Installation

The Secure File Repository (SFR) application can be implemented as a distributed service, requiring one or more web application components called “MVC nodes”, and a data tier running MySQL database management. Components can reside on a single server or separate devices with network connectivity permitting inbound TCP connections from the MVC node to the data tier, typically port 3306. The web application utilizes much of the Flask framework, including the web server. Protection measures such as an HTTP proxy are beyond the scope of this README and recommended for internet-facing production.

## Software installation

The GitHub repository contains two main directories, server\_build and app. Server\_build contains configuration files needed for the initial implementation of the solution, a standalone python CLI application to enable user registration functions, an SQL script for creating the application database and a sample dataset which can be restored to the database once created. The “app” directory is the top-level collection of files needed to run an MVC node. These contents can be copied to any location readable by the user account created to the application.

## MVC infrastructure preparation

The Flask server operating system is presumed to be Linux with the following core packages installed and all package dependencies automatically included:

python3, python3-pip

1.  Create a non-privileged user on the MVC node

2.  Download the source code from the [following repository:] (https://github.com/tbuildssd2022) , to a staging location.

3.  Recursively copy the contents of the SFR app directory to the non-privileged user’s home directory E.G.

`cp -R ./app /opt/appuser`

4.  Change user context to the non-privileged user, E.G.,

`sudo su – appuser`

5.  While in the appuser context, install the required python packages using the file requirements.txt to install all required libraries with the following command:

`pip3 install --user -r requirements.txt`

6.  Add flask to the appuser’s path with the following command:

`export PATH=$PATH:/opt/appuser/.local/bin`

7.  Edit the sample file  app/sfrapp/env-dev, modify the database connection settings and save the file as .env

## Data tier infrastructure preparation

Any operating system or container supporting MySql version 8 is a suitable data tier.

1.  Modify the username, password and IP restrictions in the file srever\_build/sfrdev-v3.sql to suit the local environment.

2.  Create the application database using the modified sfrdev-v3.sql file with the following command:

`mysql -u root -p sfrdev < /path/to/sfrdev-v3.sql`

3.  Restore the sample database content using the following command:

`mysql -u root -p sfrdev < /path/to/sdrdev\_dump.sql`

4.  Alternatively create application users with mksfruserdev.py and generate groups using sfrgroups-sample.sql.

## Starting the SFR application

1. Change directory to the application home folder E.G. /opt/appuser/app

2. Export two environment variables as shown below (use development if verbose console output needed):

`export FLASK\_APP=sfrapp`

`export FLASK\_ENV=production`

3. Start the flask server, specifying the IP address and TCP port as shown below:

`flask run -h 10.100.200.6. -p 8080`

## Starting real-time log monitoring

Server\_build/appmon.py is a standalone program for detecting SFR application log events of interest, as defined by application developers.  Utilizing the Python generator pattern, read access to the Flask application logfile, (currently /var/tmp/sfrdev.log) and write access to its own log directory is required.

1.  Configure file locations and permissions

2.  Modify syslog client destination

3.  Run in background shell such as screen

## Key Python modules

The application has two custom modules, auth.py and main.py as well as common Flask modules like models.py, \_\_init\_py etc.  Flask views related to authentication are contained within auth.py, remaining control functions reside in main.py. The tbutility.py module is used extensively for repeatable functions, with extensive comments regarding use.