**Steps**

1. Create conda environment for new app

conda create -n msiapp python=3

1. Activate environment and add Conda Forge channel (if not in channel list yet)

source activate msiapp

conda config --add channels conda-forge

1. Install required packages

conda install flask

conda install flask-sqlalchemy

1. Create directory structure (see repo)
2. Create a config file with the information necessary to create your database connection. **Do not commit this file.** You never want to commit files with keys or passwords. The file should look like this:
3. SECRET\_KEY = 'development\_key'
4. SQLALCHEMY\_DATABASE\_URI = 'sqlite:////tmp/tracks.db'

SQLALCHEMY\_TRACK\_MODIFICATIONS = True

The SECRET\_KEY should be a unique, arbitrary string specific to the application that should be random/difficult to guess - "development\_key" is just an example.

When you create your RDS database, you will change these parameters.

1. msiapp/\_\_init\_\_.py includes the line of code:

application.config.from\_envvar('MSIA\_SETTINGS', silent=True)

which tells the application to look at the environmental variable MSIA\_SETTINGS for the path to your config file. This means you need to set this environmental variable yourself by going to command line and entering:

export APP\_SETTINGS="path/to/where/your/config/file/is.config

1. Define data model (see msiapp/models.py) - this can be a placeholder and change as the students better define their project's data needs
2. Create local table with data model and insert some data (see create\_db.py)
3. Write html template for main page that displays (all or some of the )data from the table in some way.

Flask uses [jinja2](http://jinja.pocoo.org/docs/2.10/templates/) syntax for templates. In it's most basic application, the template is a normal html page but with

{{ datafield }}

in the locations where data from the table is wanted. See msiapp/templates/index.html for a very basic example

1. Create a view (see application.py) that presents data from the table through html rendered from the msiapp/templates/index.html template.

A view separates the way in which a user interacts with information and the the way that it is manipulated by the app. The "user" in this case can be an actual user interacting through a web page, or it could be a remote application interacting with it through web services.

1. Get up application running

If you haven't executed create\_db() and created your database, from the command line:

python create\_db.py

Now run your application:

python application.py

You should be able to go to the IP address that it responds with and see your web app

1. (Optional for now) Add some basic formatting via a CSS file (see msiapp/static/basic.css)