**Administration Guide for the Ecosystems Goods & Services (EGS) Web Application**

The EGS web app consists of 3 parts – a python-based source data processor to read CSV and MS Word docx files and create data files in JSON format, a python-based data server and a presentation front-end. The user interacts with the presentation front-end in a web-browser, and the data displayed comes from the data server in JSON format.

The majority of the text displayed in the front-end is generated in the data server, but there are some text strings hard-coded in JavaScript in the front-end.

**Part 1. Source Data Processor**

The source data processor part consists of 2 python scripts that read 'source' files in CSV and MS Word format and create JSON files. This part is not directly used by the web app, and should not (does not need to) be put in a folder accessible to the web server.

Directory structure of the source data processor

/cygdrive/c/Data\_and\_Tools/epa\_egs\_backend/working

├── scripts/

├──-- mk\_ecosystem\_goods\_and\_services\_json.py

├──-- mk\_egs\_literature\_json.py

├──-- lib/

│ ├── epa\_egs\_backend.config

├── data/

├──-- ecosystem\_goods\_and\_services.json

├──-- egs\_literature.json

├──-- benefit\_category\_csv/

│  ├── biodiversity.csv

│ ├── clean\_air.csv

│ ├── clean\_water.csv

│ ├── climate\_stabilization.csv

│ ├── food\_fuel\_and\_materials.csv

│ ├── hazard\_mitigation.csv

│ ├── recreation.csv

├──- benefit\_category\_word\_docs/

│ ├── BiodiversityConservation\_Agroecosystems.docx

│ ├── BiodiversityConservation\_Atmosphere.docx

│ ├── BiodiversityConservation\_BarrenRockSand.docx

│ ├── … 101 additional files

Notes:

1. There are two python scripts that each read a directory of source files and output a single JSON file. The output JSON files can then be used to overwrite the files in the data server.
2. The scripts are tested and work using python 2.7.
3. The python script to process 'literature' MS Word docx files requires installing python library 'docx' (pip install docx).
4. There is one configuration file 'epa\_egs\_backend.config' in the lib/ directory that contains the directory paths and file names for all parts of the system. It also contains settings that control how much debugging information is printed out by the python scripts.

Note: the following screen grab shows the commands used to run the python scripts and create the JSON files.

Microsoft Windows [Version 6.1.7601]

Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Data\_and\_Tools>cd epa\_egs\_backend

C:\Data\_and\_Tools\epa\_egs\_backend>cd working\scripts

C:\Data\_and\_Tools\epa\_egs\_backend\working\scripts>ls

boneyard lib mk\_ecosystem\_goods\_and\_services\_json.py mk\_egs\_literature\_json.py

C:\Data\_and\_Tools\epa\_egs\_backend\working\scripts>python mk\_ecosystem\_goods\_and\_services\_json.py

[2018-08-16 08:12:07 MDT] Reading 7 CSV (text) files from directory:

C:\Data\_and\_Tools\epa\_egs\_backend\working\data\benefit\_category\_csv

[2018-08-16 08:12:07 MDT] Writing JSON (text) file:

C:\Data\_and\_Tools\epa\_egs\_backend\working\data\ecosystem\_goods\_and\_services.json

[2018-08-16 08:12:07 MDT] Reading: biodiversity.csv

[2018-08-16 08:12:07 MDT] Reading: clean\_air.csv

[2018-08-16 08:12:07 MDT] Reading: clean\_water.csv

[2018-08-16 08:12:07 MDT] Reading: climate\_stabilization.csv

[2018-08-16 08:12:08 MDT] Reading: food\_fuel\_and\_materials.csv

[2018-08-16 08:12:08 MDT] Reading: hazard\_mitigation.csv

[2018-08-16 08:12:08 MDT] Reading: recreation.csv

[2018-08-16 08:12:08 MDT]

Wrote JSON (text) file. File Size 674 KB:

C:\Data\_and\_Tools\epa\_egs\_backend\working\data\ecosystem\_goods\_and\_services.json

C:\Data\_and\_Tools\epa\_egs\_backend\working\scripts>python mk\_egs\_literature\_json.py

Traceback (most recent call last):

File "mk\_egs\_literature\_json.py", line 265, in <module>

raise Exception("Python 2.7 is required. Encoding not completely working in python > 2.7")

Exception: Python 2.7 is required. Encoding not completely working in python > 2.7

C:\Data\_and\_Tools\epa\_egs\_backend\working\scripts> python2 mk\_egs\_literature\_json.py

[2018-08-16 08:13:03 MDT] Reading 104 MS Word Documents from directory:

C:\Data\_and\_Tools\epa\_egs\_backend\working\data\benefit\_category\_word\_docs

[2018-08-16 08:13:03 MDT] Writing JSON (text) file:

C:\Data\_and\_Tools\epa\_egs\_backend\working\data\egs\_literature.json

[2018-08-16 08:13:03 MDT] Reading: BiodiversityConservation\_Agroecosystems.docx

[2018-08-16 08:13:03 MDT] Reading: BiodiversityConservation\_Atmosphere.docx

[2018-08-16 08:13:03 MDT] Reading: BiodiversityConservation\_BarrenRockSand.docx

[2018-08-16 08:13:03 MDT] Reading: BiodiversityConservation\_CreatedGreenspace.docx

[2018-08-16 08:13:03 MDT] Reading: BiodiversityConservation\_EstuariesNearCoastalMarine.docx

….. continues

[2018-08-16 08:13:06 MDT]

Wrote JSON (text) file. File Size 402 KB:

C:\Data\_and\_Tools\epa\_egs\_backend\working\data\egs\_literature.json

C:\Data\_and\_Tools\epa\_egs\_backend\working\scripts>

Note: if the source CSV and(or) MS Word data files are updated and the scripts re-run, the new output JSON files need to be moved manually to the data server 'data' subdirectory.

**Part 2. Data Server**

The data server is a REST service written in python that reads data from either a JSON file or a python 'pickle' file and outputs JSON.

Directory structure of data server

/cygdrive/c/inetpub/wwwroot/epa\_egs-cgi

├── data

│ ├── ecosystem\_goods\_and\_services.json

│ ├── ecosystem\_goods\_and\_services.p

│ ├── egs\_custom\_text.config

│ ├── egs\_literature.json

│ ├── egs\_literature.p

├── index.py

├── lib

│ ├── BenefitTree.py

│ ├── epa\_egs.config

├── web.config

Notes:

1. The data server is a python script 'index.py'. It can be used with python 2.7 or 3.5+. To install and use in IIS7 it requires creating a script processor for python.
2. In the data folder there are 2 files with the '.p' extension. These are python 'pickle' files used to optimize performance. They are generated automatically if they are not present when the data server runs. The '.p' files are updated automatically when the index.py script is run if the '.p' file is older than the '.json' file
3. The IIS server (or other web server daemon on linux) needs permission to write in the 'data' directory to enable the automatic creation of the '.p' files. But if it doesn't have write permission, the index.py script will NOT report a problem – it will just use the '.json' file.
4. All text output from the data server comes from the contents of the data directory.
5. The order of all the information in the JSON output from the data server is defined in the 'egs\_custom\_text.config' configuration file in the [ORDERED\_LISTS] section.
6. The data server can be tested on the command line by setting/commenting/uncommenting one or more arguments in the [TESTING] section of the epa\_egs.config file and then running in a windows command window. These settings are ignored when the script is run from the web-server.

The following screen grab shows the commands used run the python script index.py when all the command line settings in the 'TESTING' section of the config file are commented out:

Microsoft Windows [Version 6.1.7601]

Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Data\_and\_Tools> cd C:\inetpub\wwwroot\epa\_egs-cgi\

C:\inetpub\wwwroot\epa\_egs-cgi>python index.py

{

"name": "Seven Broad Benefit Categories",

"children": [

{

"code": "Biodiversity",

"name": "Biodiversity Conservation",

"extended\_text": "Biodiversity is the variety of all forms of life and it is essential to the existence and proper functioning of all ecosystems. Biodiversity supports habitats for all species by providing many unique environments in which species can exist; these include ecosystems of all types and sizes, rare ecosystems, and corridors between habitats. Many scientists believe biodiversity, as it represents all forms of life on earth, provides or supports the core benefits that humans derive from their environment. Biodiversity is fundamental for the provision of ecosystem services, which we depend on for food, air, and water security, and multiple other natural benefits.",

"primary\_key": "benefit\_category"

},

{

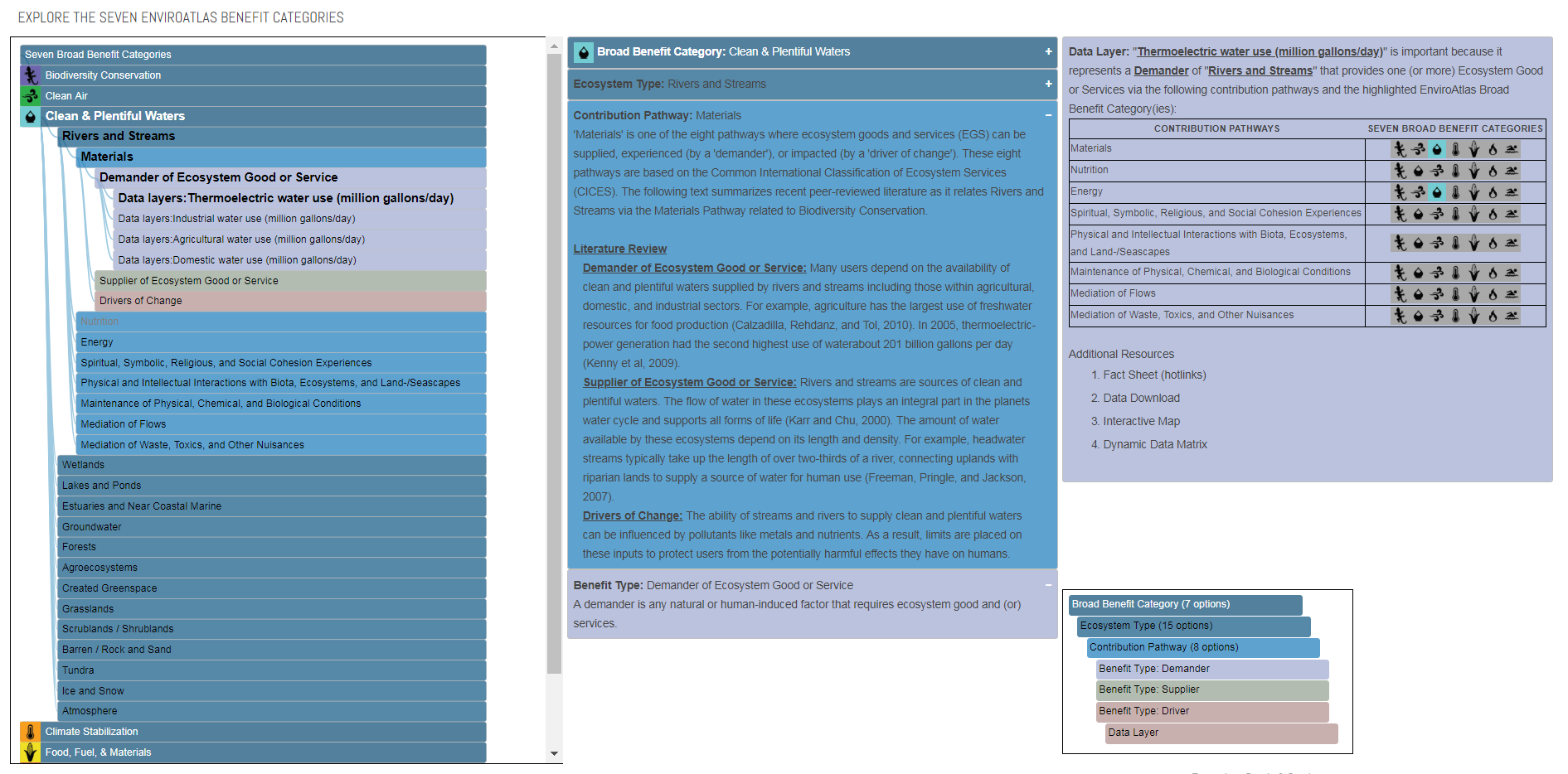
"code": "Clean Air",

"name": "Clean Air",

… continues

**Part 3. Presentation Front-End**

The presentation front-end is the web application that users interact with via a web browser. Users click on 'nodes' in the left-side panel, and drill-down to explore the relationships between the 7 EnviroAtlas Benefit Categories, Ecosystem Types, Contribution Pathways, Benefit Types, and Data Layers. Users can expand or collapse the information shown in the center panel by clicking anywhere in any of the sections.



Directory structure of presentation front-end

/cygdrive/c/inetpub/wwwroot/epa\_egs

├── css

│ ├── style.css

├── images

│ ├── air.png

│ ├── air\_bw.png

│ ├── bio.png

│ ├── bio\_bw.png

│ ├── clim.png

│ ├── clim\_bw.png

│ ├── comm.png

│ ├── food.png

│ ├── food\_bw.png

│ ├── haz.png

│ ├── haz\_bw.png

│ ├── nat.png

│ ├── rec.png

│ ├── rec\_bw.png

│ ├── water.png

│ ├── water\_bw.png

├── index.html

├── index.js

├── Web.config

Notes:

1. The web app is a 'pure javascript' app that uses the D3 JavaScript library for visualizing data with HTML, SVG, and CSS (<https://d3js.org/>)
2. The data layout used in the web app is called Collapsible Indented Tree. The code used started with an example from Mike Bostock (https://bl.ocks.org/mbostock/1093025)
3. All the data displayed comes from either the data server or is hard-coded in the index.js file. As much as possible, all text displayed are either in configuration variables at the top of the index.js file, or in the functions that populate the center and right panels.
4. The colors for each data element are defined in variables hard-coded in the index.js JavaScript file. There should not be any (important) colors defined in the single 'style.css' CSS file.
5. The names for the image files which are displayed for the Seven Broad Benefit Categories are hard-coded in the index.js JavaScript file in the JavaScript array 'benefit\_categories'. For each 'image' in this data structure, there are two '.png' images in the file system – 1 with and 1 without a '\_bw' (black and white) suffix.
6. The order of all (almost all – see next Note) data shown in the presentation front-end is inherited from the JSON data output from the data server.
7. The order of the icons shown in the 'Data Layer' section of the right-panel comes either from the JSON from the data server, or – in the case where a CONTRIBUTION PATHWAY doesn't have any available SEVEN BROAD BENEFIT CATEGORIES – the hard-coded JavaScript array 'benefit\_categories'. If the order is changed in the data server, it needs to be changed in this array.