Point-In-Time Survey Documentation and Guide

Setup / Running

1. Install MongoDB
2. Install python (pip is a command line tool that will come with)
3. Run `pip install flask pymongo pygal`
4. Download the git repo from https://github.com/code-for-tb/PITCensus
5. Start MongoDB. By default, it runs on port 27017. See the Configuration section to change this.
6. Navigate to backend.py and run '`python backend.py`'

Technical Details

1. Components
    a. We're using Flask to host the web server.
    b. We're using surveyjs to create the surveys.
    c. We're using PyMongo and MongoDB to store the data from the surveys in the database.
    d. We're using PyGal to create the charts for the data.

2. Where do I look if I want to…?
    a. Edit the survey - `src/static/js/questions.js`. See the section "Updating and Expanding Questions"
    b. Change the behavior of the backend (what we do with traffic, routing) - `src/backend.py`
    c. Change the way the database works - `src/database.py`
    d. Change the way the site looks - `src/templates/index.html`

Troubleshooting

Problem	**Suggestion**
The admin page does not load	Make sure MongoDB is running
Exception: Cannot read main config file pitconfig.ini	Make sure the pitconfig.ini file is where the error indicates. Edit config.py to change the location or the name of the file.
The map on the webpage is not working	Try different browsers and incognito or private modes. If location is blocked on the browser, the map will not work. re-adding the Google API key

Configuration

Located within the repository is a **pitconfig.ini**. This is the central configuration of the system. Edit this file to change the survey and admin password, and update the current PIT year. The configuration file is split into sections.

The admin page will display the current contents of the pitconfig.ini file, and direct readers to edit it in case of changing things.

[database]
Controls database configuration

Option	Meaning
host	The address of the host running MongoDB
port	The port the on host that the application will read/write to the MongoDB connection
pityear	The selected year to read/write records from. Internally is is done by accessing the database as an array, similar to “database[pityear]”

[web]
Controls configuration settings about the web server itself.

Option	Meaning
port	The address the web service will run on, including the index page, admin page, and API
debug	If True, Flask will run in debug mode, which will (among other things) restart the application automatically if any python files are changed
homeUsername	The username required to access the main page containing the survey
homePassword	The password required to access the main page containing the survey
adminUsername	The username required to access the admin page
adminPassword	The password required to access the admin page

Updating and Expanding Questions

1. If you want to change / update / add / remove questions, go to `src/stratic/js/questions.js`. This is in the format accepted by surveyjs (http://surveyjs.org/examples/knockout/), specifically the knockout type. “`triggers`” cause things to happen, and “`questions`” are the format for the different questions. Most of the triggers cause questions to be shown or hidden based on answers to previous questions.

2. If you navigate to the survey editor on the website (http://surveyjs.org/builder/) and copy all of the code in `src/static/js/questions.js` except for the leading ‘var surveydata = ’ (so starting with the ‘{‘ and ending with a ‘}’), and then click the ‘Survey Designer’ tab, you will see the working version of the survey. You can use this tool to make sure that you have valid questions / triggers. You can either edit the json directly or use the interactive survey editor to add and remove questions here. Once you are done, copy all the JSON (the text in the JSON editor) and put it back in questions.json (after the ‘var surveydata = ‘). Now those questions should be updated.

3. If you add new questions, answers, etc, that information will automatically be stored in the database for any new surveys

Updating and Expanding The Website

The code for the website is located in `src/templates`. The HTML files are rendered using Flask and its template engine to fill in data on the page. The layout.html file is the base for all other pages, which includes the background, Bootstrap JavaScipt, and other required HTML. `index.html` is the page which the survey is displayed on, and `admin.html` contains the admin page.

Retrieval of Completed Surveys

To retrieve completed surveys, call the **/getLastSurveys** REST API call, as detailed in the REST API section.

Making Use of Data

The keys in the database should be somewhat self-explanatory. They are pulled directly from the questions.js file, so if you need to know what question goes with what answer you can always refer to questions.js.

REST API

The API of the server is the main way to import and export data from the system.

End point	**Method**	**Description**
/completedSurvey	POST	Adds a new survey to the database. The data does not have to be in any particular format, except in JSON. The new survey will be added to the PIT year as detailed in the configuration file. Be careful! No checks are made to validate the incoming data.
/getSurveyCount	GET	Return the number of surveys added so far for the configured PIT year
/getLastSurveys/<amount>	GET	Return the last <amount> surveys recorded for the configured PIT year as JSON. Use this call to export survey data to any other system. An example call would be “http://localhost:5000/getLastSurveys/10”