Case Complexity Tool

System Administrator Manual

Version 1.0

Prepared by Adventurous Analytics, LLC

1 Purpose

This manual is intended for system administrators that will install the Case Complexity Tool (CCT) at their agency. For information about using CCT output, see the Case Complexity Tool User Manual.

CCT is written in Python and relies on installation of components supporting Python and Conda that enable data science applications to run in isolated environments.

2 Requirements

The Case Complexity Tool (CCT) has system requirements outlined in Table 1.

Table 1: System Requirements

Item	Description
Operating System	Windows Server 2016 or Windows 10 Pro/Enterprise
RAM	8GB minimum
Disk Space	Approximately 2 GB total space is need for installation The CCT application requires approximately 250MB of disk space. The Conda package and environment management tool uses most of the 2 GB space. Note: Once running, required disk space expands daily with output. It will be a function of how many days of retained input and output data you choose to keep. One day of input/output data is approximately 20MB for a moderate to large CBC handling under 2000 active cases, excluding FSS cases. This would utilize ~7 to 10 Gb per year of retained I/O files
Python / Conda	The application is written in Python and uses Conda virtual environments to manage the specific versions of libraries used.
Business Objects Queries	The application relies on data extracted from FSFN via pre-defined Business Objects reports/queries. Note: Scheduling Business Objects Queries is coordinated through Florida DCF, please use the information in the installation section of this manual to prepare queries and request them to be scheduled daily.
Internet access to GitHub	You will need access to GitHub to download the application. Currently Adventurous Analytics, LLC maintains the private code repository containing the CCT Tool
Google Account	A Google account is needed to obtain an API key for the geocoding service used to calculate distances used in the Case Complexity Tool.

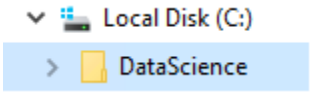
3 Installing CCT

Follow the directions in the sub sections below to install CCT.

3.1 Create a Data Science directory

For ease of installation and ongoing maintenance we recommend installing the application and Conda in a new data science directory on your server. You may install in any location. We will use C:\DataScience throughout the instructions, substitute your directory location as applicable. You should have a structure like Figure 1.

Figure 1: Create a Directory named "DataScience"



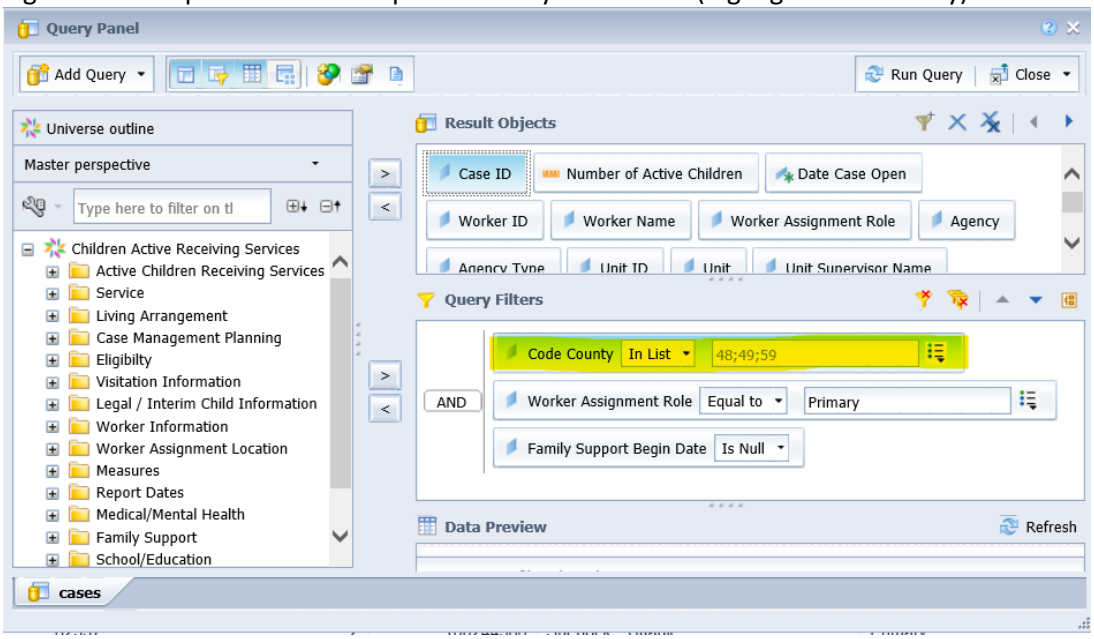
3.2 Setup Business Object Queries

The application uses data from FSFN, retrieved via pre-defined Business Objects queries. The application is intended to run daily and expects new data files are present when it kicks off. The Business Objects queries need to be scheduled with Florida DCF ahead of time to ensure data is available.

Note: The queries must be copied and altered to use your agency's county codes before scheduling.

The queries are setup to pull data for specific county codes associated with your CBC Lead Agency. Copy the set of queries to a folder you can use to schedule for you agency. Then update the county code(s) in each of the 9 reports prior to scheduling. The first query in each report's data provider contains the County Code filter that needs to be updated as seen in Figure 2.

Figure 2: Example of where to Update County Code Filter (highlighted filter only)



The set of template queries are located on Business Objects in Public Folders at this location:

Public Folders -> FTP Reports -> CCT

The Business Objects report files need to be saved at a location that is accessible by the application. By default, the application will expect the files in the "\EF-CCDT\data\input" folder. This can be modified in the configuration to another folder. CCT expects the report files to be in *.xlsx format.

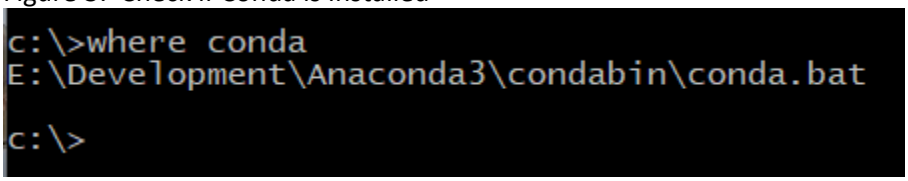
3.3 Install Conda

CCT uses Conda to manage Python environments keeping versions of Python and supporting libraries consistent. If you are running existing Python applications on your server, you may have Conda already installed.

3.3.1 Check if Conda is Already Installed

To check if you have Conda installed, open a command prompt, and use the command in Figure 3, "where conda".

Figure 3: Check if Conda is Installed



If a version of Conda is already installed you will see a response like Figure 3. Take note of the path up to "\Anaconda3" or "\Miniconda3" as you will need these in later steps. You may skip the remaining steps in section 3.3.

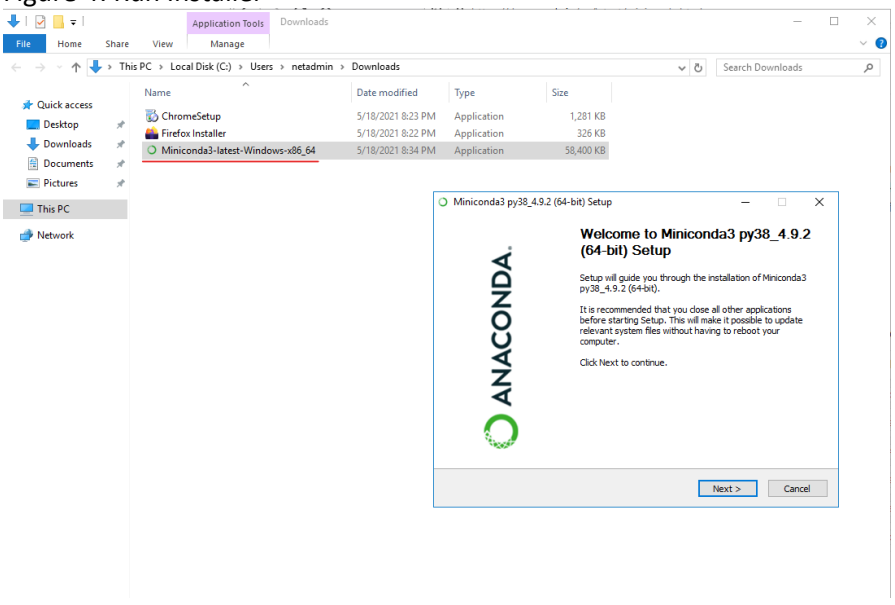
If Conda is not installed, please install the lightweight version, Miniconda found at the following URL .

Miniconda: https://docs.conda.io/en/latest/miniconda.html

A Miniconda installation will progress as seen in the following steps of section 3.3.

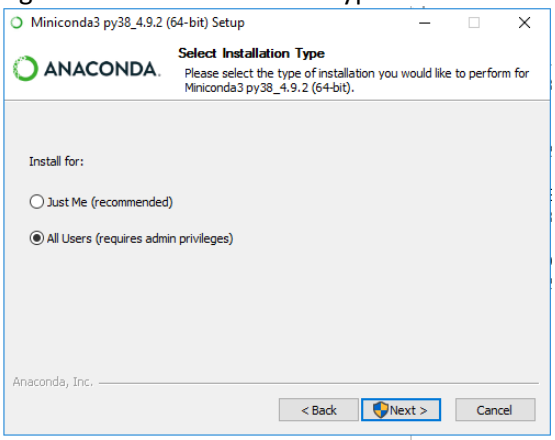
3.3.2 Download and Launch installer

Figure 4: Run Installer



3.3.3 Choose All Users installation type

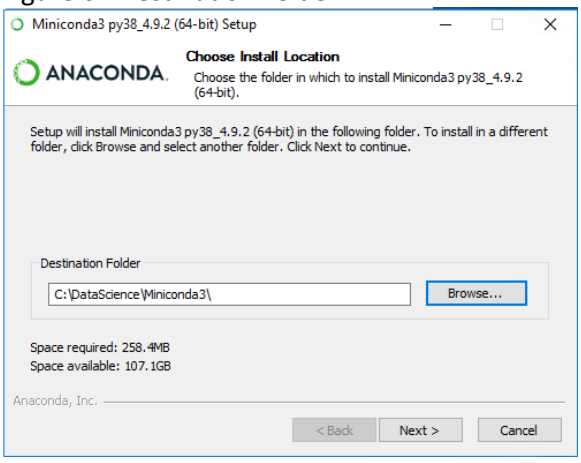
Figure 5: Select Installation Type



3.3.4 Choose a destination folder

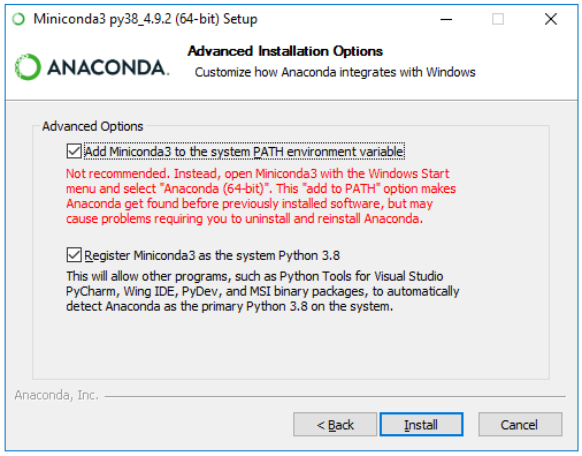
Place Conda in the \DataScience directory we created earlier as seen in Figure 6.

Figure 6: Destination Folder



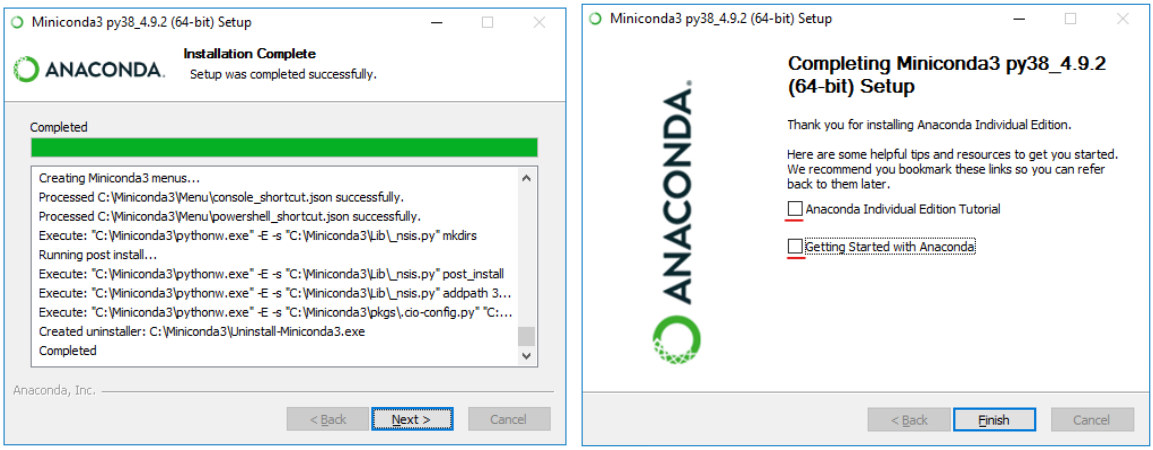
3.3.5 Register Conda Path and Python

Figure 7: Register Conda



3.3.6 Installation Complete

Figure 8: Complete Installation



3.4 Install Case Complexity Tool (CCT)

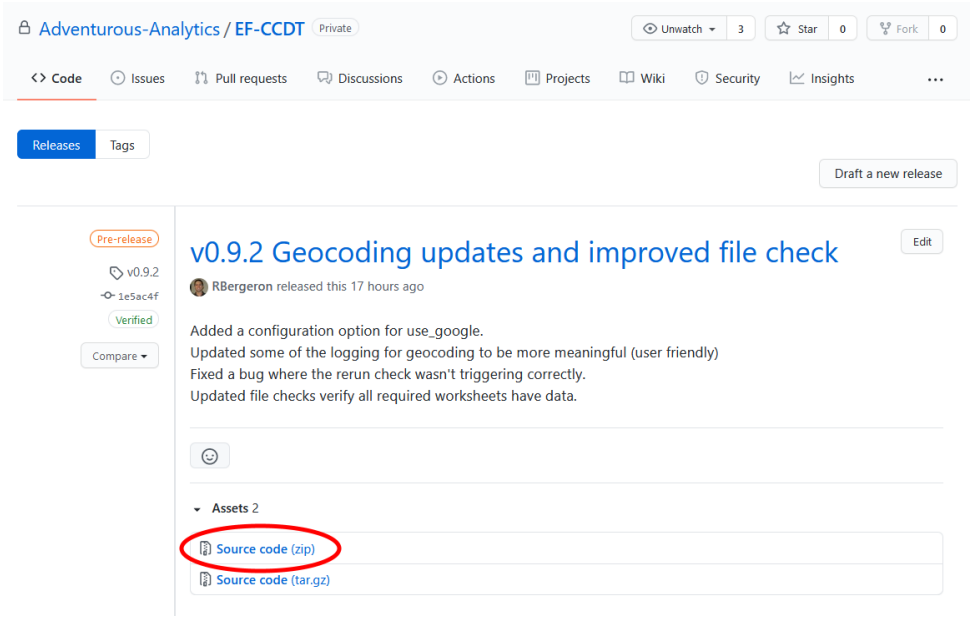
3.4.1 Download CCT Zip File

CCT is available from the GitHub repository as a ZIP file. Download the most recent release's ZIP file from this location.

https://github.com/Adventurous-Analytics/EF-CCDT/releases

On this web page, click the small arrow next to the "Assets" label and download "Source code (zip)" file as seen in the Figure 9.

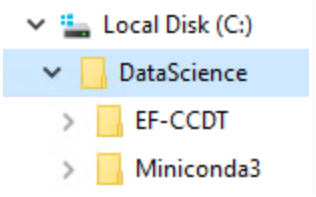
Figure 9: Downloading the CCT Zip file



3.4.2 Unzip the file to \DataScience directory

Unzip the zip file to the \DataScience directory we created earlier. The unzipped folder will contain the release number in the name, such as "EF-CCDT-v0.9.2". Rename the folder to "EF-CCDT". You should end up with a folder structure like the one seen in Figure 10.

Figure 10: Example Folder Structure after Unzipping CCT and Renaming Folder to EF-CCDT

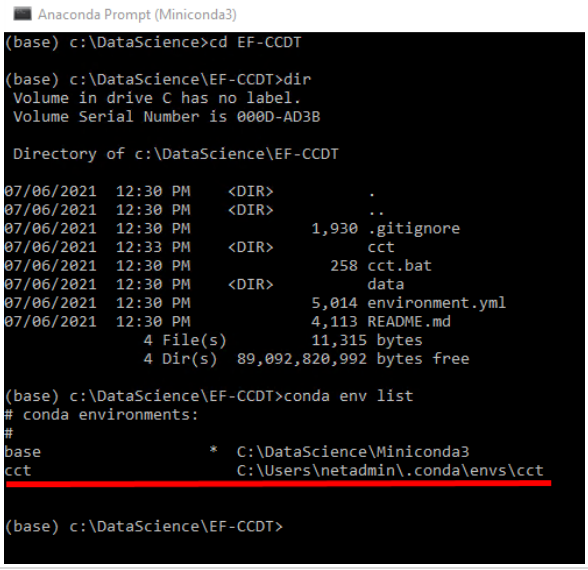


3.5 Setup the Conda Environment

Once Conda and CCT are installed, we need to setup the Conda environment. Take the following steps and see Figure 11 as an example.

1. From Windows Start button, open a Conda command prompt (this was installed with Conda installation in previous steps) as seen in Figure 11.
2. Navigate to the directory where you installed CCT (default is C:\DataScience\EF-CCDT).
3. Locate the file, "environment.yml".
4. Run the command: `conda env create -f environment.yml`
5. Verify the environment with: `conda env list`
    a. The list of environments should include cct and look similar to Figure 11.

Figure 11: Example Command window with Conda environments listed



3.6 Setup a Windows Scheduled Job

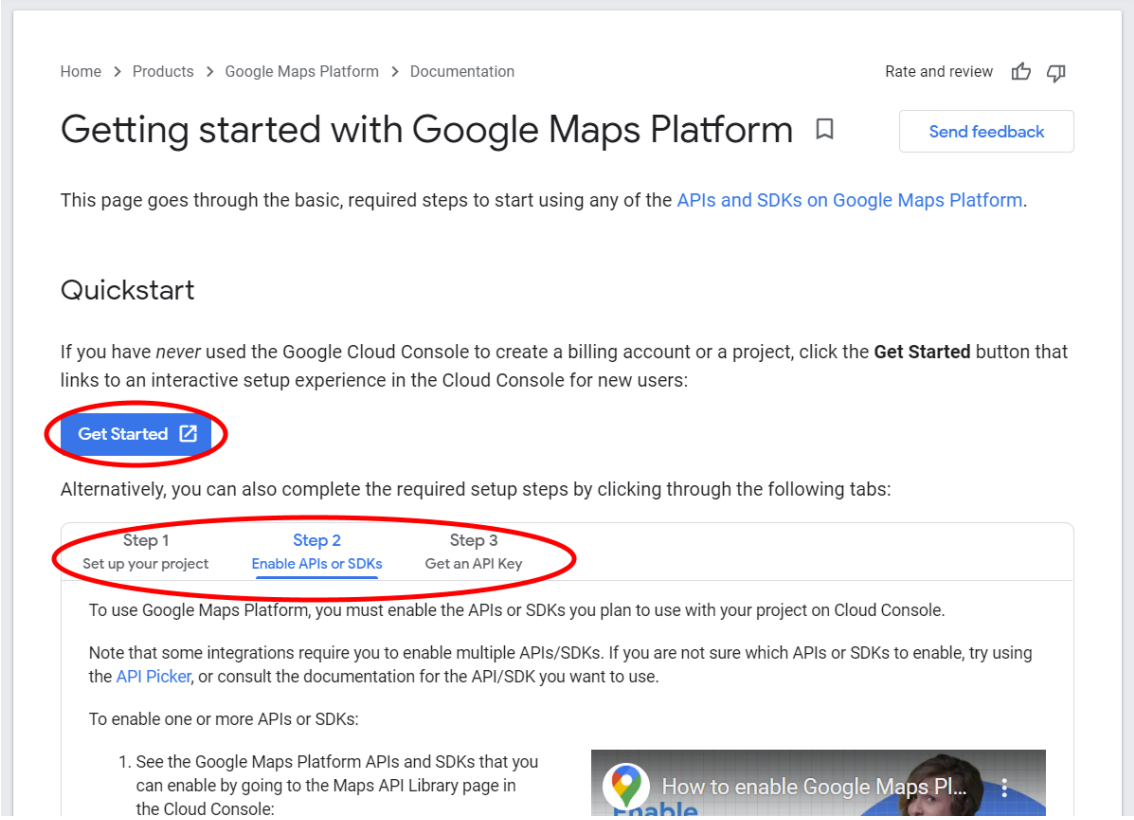
CCT is intended to run on a daily schedule. A template batch file named cct.bat is included in the CCT directory to assist with setup. If you have used the default directory structure for installation, the file can be used without modification. If you installed either Conda or the application somewhere other than the default "C:\DataScience" directory, the file needs to be edited in a text editor like Notepad or Notepad++ to use your directory paths.

3.7 Setup your Google Account

3.7.1 Create Google Account

A Google account is necessary to use the Google geocoding service that provides information needed to calculate distances. The Google account needs to be associated with the business (agency). It needs to be setup to have a billing account and project. You can also set limits to prevent accidental or unauthorized requests from creating unexpected costs. At the time of writing this manual, the Google cost structure for the geocoding API will usually not generate a monthly cost. To setup your account and project with Google, please walking through the "Getting started with Google Maps Platform" process available from Google at https://developers.google.com/maps/gmp-get-started. During this process you will setup a billing account and create project and enable APIs. You can use either the "Get Started" option or the step-by-step instructions on Google's site as seen in Figure 12.

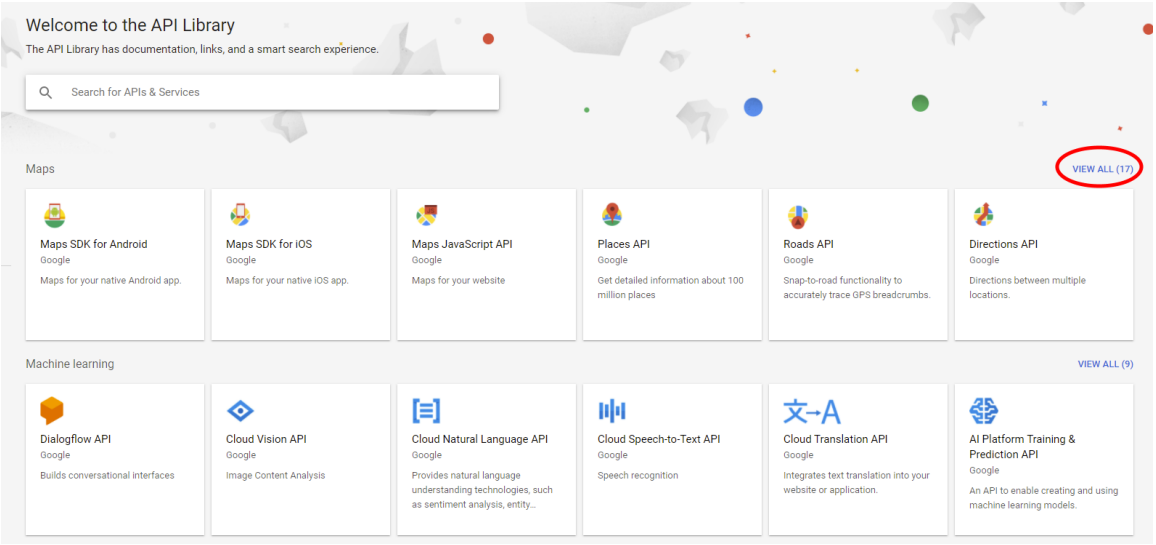
Figure 12: Google's Getting started with Google Maps Platform



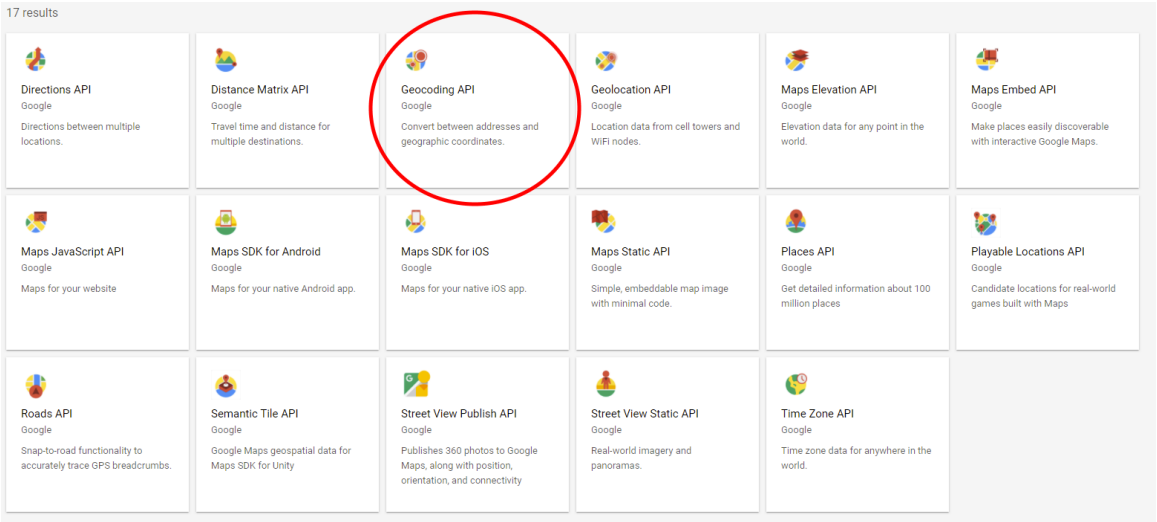
3.7.2 Enable Geocoding API

Once you have your account and project setup, you want to enable the Geocoding API on the project. No other API is required. Examples of the Google API selection are in Figure 13. You may need to select "View All" option on the right of the Maps group to see the Geocoding API option as seen in Figure 13. Be sure to take note of the API key you create in the last steps of account setup. You'll need it in the next setup step.

Figure 13: Google Maps – Geocoding API Selection



Then the Geocoding API will be visible



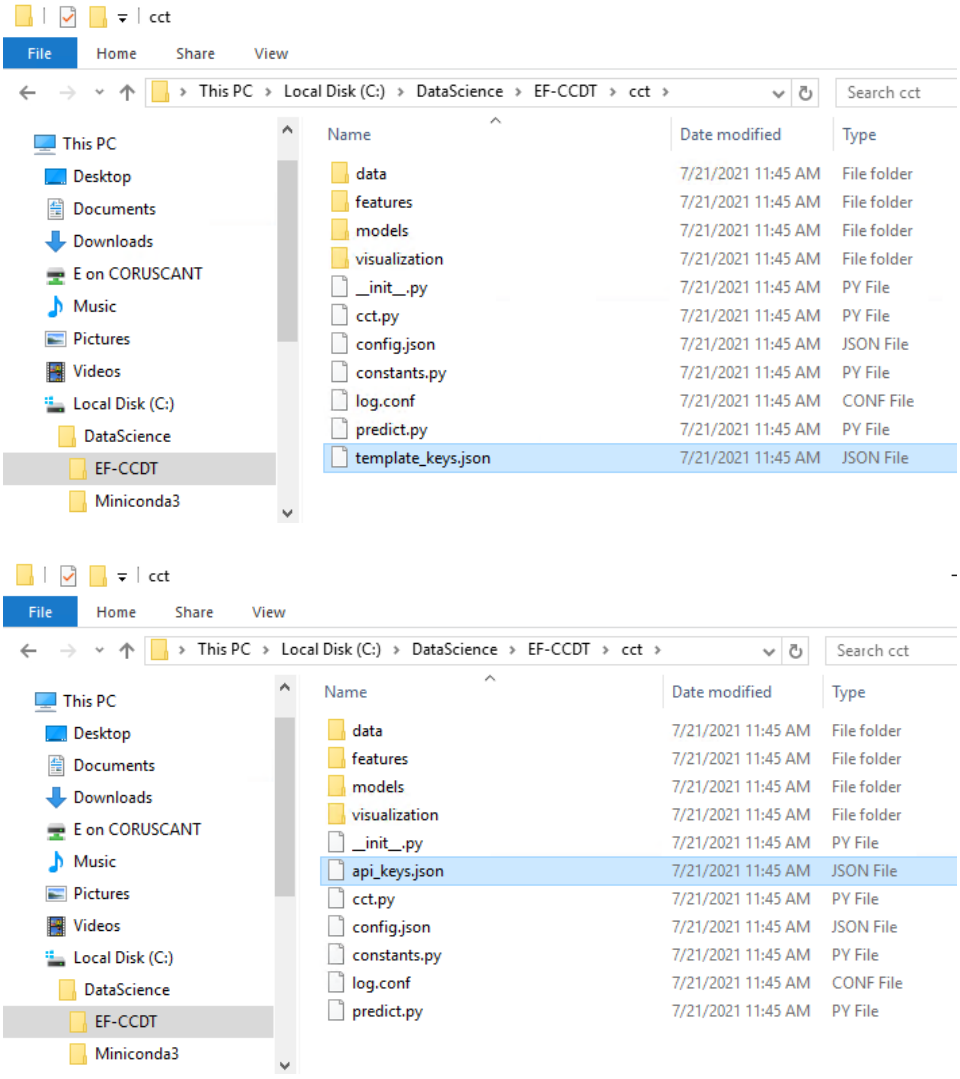
3.7.3 Save API Key

Once you have setup your Google account and have your API key, you'll need to modify the template file to add your key and rename the file from "template_keys.json" to "api_keys.json" as seen in Figure 14.

3.7.3.1 Locate and rename the template file

A template key file is included in the project that needs to be renamed.

Figure 14: Rename template_keys.json file to api_keys.json

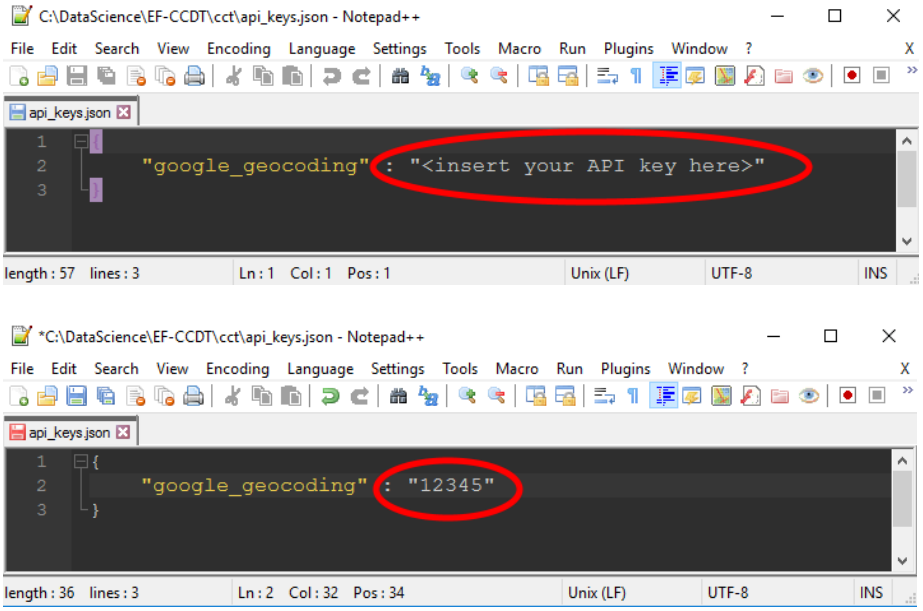


3.7.3.2 Add Your API key

Edit the "api_keys.json" file with a text editor such as notepad or notepad++ to add your API key similar to what's seen in Figure 15. Replace the text inside the quotes "<insert your API key here>", leaving the

quotes. For example, if your API key is 12345, then the finished edit would look like the second screen snip in Figure 15.

Figure 15: Add Your API key to api_keys.json



4 Configure CCT

CCT has configurable options that can be edited for your environment. Configuration settings are in the \EF-CCDT\cct\config.json file. Configurable options are described in Table 2. The application will run with the default configuration as long as application is installed in the default locations and the Business Objects reports data are placed in the default input folder location (\EF-CCDT\data\input). **Note: Settings in the config.json file other than the ones listed in Table 2 should not be changed.**

Table 2: Configurable Settings

<table>
<tr><th>Config Option</th><th>Description</th></tr>
<tr><td>fullpaths: use_fullpaths</td><td>Default is false.

By default, the application sets its paths based on the location of installation. If you want to override where it looks for data files or places output, set to true and update ALL the entries for your system.

<table>
<tr><td>Input</td><td>Path where app expects to find Business Objects input files</td></tr>
<tr><td>Train</td><td>Not currently used</td></tr>
<tr><td>Reset</td><td>Path to default geocode and poverty level files</td></tr>
<tr><td>Processed</td><td>Path to consumed input files</td></tr>
<tr><td>Transformed</td><td>Path to transformed input files</td></tr>
<tr><td>Output</td><td>Path to output files</td></tr>
<tr><td>Logs</td><td>Path to log files</td></tr>
</table>
</td></tr>
<tr><td>logging: level</td><td>Sets the level of detail captured in logging messages from the application. The default is “info”

<table>
<tr><td>Debug</td><td>Most verbose level meant for troubleshooting</td></tr>
<tr><td>Info</td><td>Default, informational and more sever messages are logged.</td></tr>
<tr><td>Warning</td><td>Less verbose, captures important messages about issues the application encountered but did not cause the application to stop. Suggest only using after a period of consistent runs without issue.</td></tr>
<tr><td>Error</td><td>Less verbose, only captures messages that cause problems with application execution</td></tr>
<tr><td>Critical</td><td>least verbose, captures only the most severe messages that crash the application</td></tr>
</table>
</td></tr>
<tr><td>logging: log_to_console</td><td>true/false (case sensitive) – show log messages in console as program runs</td></tr>
<tr><td>logging: log_to_file</td><td>true/false (case sensitive) – record log messages to file</td></tr>
</table>

Config Option	Description
urls: poverty_level	Web address of the site where the application gathers published poverty level data. Rarely changes. If this needs to be changed, a code change is most likely needed.
urls: zip_codes	Web address of the Google geocoding API interface. Rarely changes. If this needs to change, there are likely other code changes needed.
geo_data: use_google	true/false (case sensitive) – Whether or not to use Google API for geocoding zip codes. Alternative is to manually update the CCT_zip_code_geo_data.xlsx file in the /reset folder.
geo_data: fl_zip_codes:filename	Default filename of the zip code geodata file required for distance calculation in the application.
geo_data: max_age	The age in days of how old a geodata point can be before the application will attempt to refresh the latitude and longitude coordinates. Default value is 30.
geo_data: key_filename	The file name of the .json file that contains your Google API key. This file needs to be located in the /cct folder.
geo_data: zip_prefixes	List of prefixes for your state. This rarely changes. It is used to limit zip code geodata look ups to your state preventing calculation of long distances to out of state case participants.
output: filename	The template for the output file name. the file name may be changed except that it must contain the "{batch_id}" string in the name. The application expects to pre-pend the batch_id to the output file name and subsequently uses the existence of an output file with a batch_id from the current day as a trigger to know when not to re-run.
output: sheets	This represents the Excel worksheets in the output file. The keys within this configuration hierarchy should not be changed. You may alter the values within the following keys as desired. "sheet_name" – change the name of the Excel worksheet "column_order" – changes the order of columns in the worksheet "sort_columns" – changes which columns to sort on in order or priority "sort_ascending" – must be one true/false value for each column listed in sort_columns. True = sort in ascending order, False = descending order
output: columns	This represents the columns within the output file Excel worksheets. The keys within this configuration hierarchy should not be changed. You may alter the values within the following keys as desired. "header" – change the name of the column's header "width" – changes the column width

5 Output

The Case Complexity Tool generates a combination of Excel and text output files. Descriptions of the output files are in Table 3. Most of the output files are prepended with a batch ID composed of a timestamp in the format YYYY_MM_DD_HHMISS. It designates which batch run of the process generated the file. All files generated from the same batch run receive the same batch ID. The batch ID is set internally by CCT when it starts.

Table 3: Output Files (default locations C:\DataScience\EF-CCDT…)

File Name	Output location	Description
<Batch-ID>_CCT_log.txt	...\data\logs	A text file capturing the status of the batch run and any errors encountered during the run. The main purpose is to help system administrators and developers monitor the process and troubleshoot problems. The amount of information logged is set by the logging: level configuration noted in section 4.
CCT_case_scores.xlsx	...\data\output	Main output Excel file containing case scores and associated summary data. Intended for end user distribution to case managers and supervisors. This file is overwritten each time CCT is run.
<Batch-ID>_CCT_case_scores.xlsx	...\data\output	Copy of the main output file prefixed with the batch-id. Intended to keep a historical record of the case scores generated.
<Batch-ID>_CCT_caregiver_data.xlsx <Batch-ID>_CCT_case_data.xlsx <Batch-ID>_CCT_case_history.xlsx <Batch-ID>_CCT_case_visits.xlsx <Batch-ID>_CCT_case_zip_codes.xlsx <Batch-ID>_CCT_cases.xlsx <Batch-ID>_CCT_csec.xlsx <Batch-ID>_CCT_parent_data.xlsx <Batch-ID>_CCT_poverty_level.xlsx <Batch-ID>_CCT_zip_code_geo_data.xlsx	...\data\processed	Copies of the input files associated with a batch run. These files are useful for trouble shooting and may be loaded to a data warehouse for tracking the long-term trends in case factors and scores
<Batch-ID>_CCT_all_factors.xlsx	...\data\transformed	The set of input data transformed into model factors used to generate the case score for a given batch ID. This file is useful for trouble shooting and may be loaded to a data warehouse for tracking the long-term trends in case factors and scores

6 Maintenance Tasks

The CCT Application has a few periodic maintenance requirements for ongoing operation outlined in the subsections below.

6.1 Update Business Objects Query Dates

Due to limitations in Business Objects queries, certain date filters, which limit how far back the query looks into history, must be updated. In an ideal situation these dates could be set dynamically based on the date when the query is run, however, this is a limitation of the Business Objects tool.

These dates should be updated at least once per year in January. If the amount of data being returned becomes too large, causing queries to time out or becoming a strain on your available disk space, the queries may be updated more frequently. Table 4 contains the Business objects reports and queries that need periodic updates.

Table 4: Business Objects Query Maintenance

Report	Query	Date Filter Value
case_history	IH_history	Set to 1 year prior to today's date
case_history	IH__leagal_history	Set to 1 year prior to today's date
case_history	OH_history	Set to 1 year prior to today's date
case_visits	case_visits	Set to 90 days prior to today's date
child_data	juvenile_1	Set to 3 year prior to today's date
parent_data	protective_capacity	Set to 3 year prior to today's date
parent_data	safety_summary	Set to 3 year prior to today's date
parent_data	eligibility	Set to 3 year prior to today's date

6.2 Determine File Retention Policy

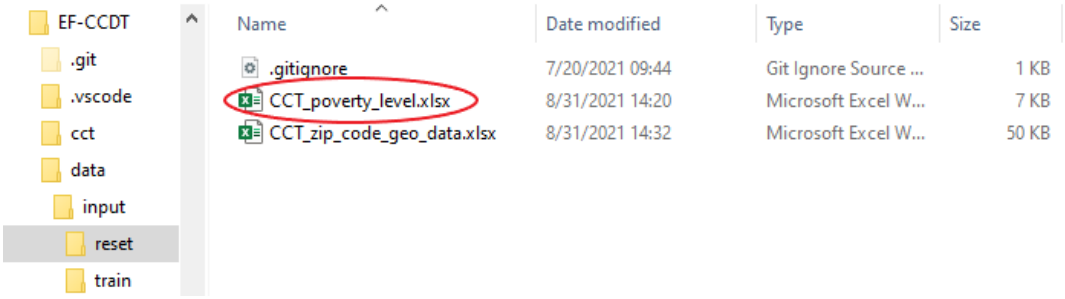
The input and output files retained with each run of CCT are not large but can add up over time. It is your discretion as System Administrator in coordination with your end user audience as to how long to retain files. With caseloads at a moderate to large size CBC, you can expect disk space growth at approximately 7-10 GB per year, which may allow you to retain a year or more of data if that is desirable for your team of users.

6.3 Federal Poverty Level Data

The Department of Health and Human Services (HHS) publishes the national poverty level data once per year. CCT uses poverty level data as an input. The information is stored in an Excel file located in the

/reset directory (directory may be different if your CCT is configured to use something other than the default directory in config.json) as seen in Figure 16. .

Figure 16: Poverty Level data file location



It is recommended to update the file once per year, in January or as soon as the new year's data is published. The HHS web site typically provides information on in the format shown in Figure 17 at https://aspe.hhs.gov/topics/poverty-economic-mobility/poverty-guidelines

Figure 17: HHS Poverty Level data published online

HHS Poverty Guidelines for 2021

The 2021 poverty guidelines are in effect as of January 13, 2021
Federal Register Notice, February 1, 2021 - Full text.

2021 POVERTY GUIDELINES FOR THE 48 CONTIGUOUS STATES AND THE DISTRICT OF COLUMBIA	
Persons in family/household	**Poverty guideline**
1	$12,880
2	$17,420
3	$21,960
4	$26,500
5	$31,040
6	$35,580
7	$40,120
8	$44,660
For families/households with more than 8 persons, add $4,540 for each additional person.	

Our data file expands the baseline data up to a household of 50 members, use the HHS published information to update the associated yearly (100_percent) and monthly income numbers associated with household size as seen in the example in Figure 18.

Figure 18: Poverty Level Excel File Example

house_hold_size	100_percent	monthly
1	12880	1073.33
2	17420	1451.67
3	21960	1830
4	26500	2208.33
5	31040	2586.67
46	217180	18098.33
47	221720	18476.67
48	226260	18855
49	230800	19233.33
50	235340	19611.67

6.4 Updating to New Releases

Occasionally a new version of CCT may be published to GitHub. Sometimes releases are required to address a critical issue, other times it may be for improvements or enhancements. The steps below outline how to update to a newer version of CCT. Note: all directory paths below refer to the default installation locations, please modify them as needed if your configuration used an alternate location.

6.4.1 Rename the existing install directory (do not delete)

You will already have a folder named C:\DataScience\EF-CCDT. Rename it by appending "_old" to the end of the directory name resulting in C:\DataScience\EF-CCDT_old

6.4.2 Download the latest *.zip file from GitHub

Download the most recent release's ZIP file from this location (see section 3.4.1 of this document for figures, this is identical to initial installation step):

https://github.com/Adventurous-Analytics/EF-CCDT/releases

6.4.3 Extract the zip file to your C:/DataScience directory

Extract the zip file to your C:\DataScience directory. This will result in a folder named like C:\DataScience\EF-CCDT-v1.0.0

6.4.4 Rename the extracted folder

The extracted folder will have the version number appended to the end. Rename it by removing the version number which will leave you with a folder named C:\DataScience\EF-CCDT.

6.4.5 Copy batch and configuration files

During the first installation, you have changed settings and/or paths in the cct.bat file, api_keys.json, and config.json file. Copy these files from the EF-CCDT_old directory to the same subfolders under the new EF-CCDT folder.

6.4.6 Delete EF-CCDT_old directory after new version completes successfully

We recommend waiting until at least one scheduled run of the new version completes, before deleting the EF-CCDT_old directory.