**CSSP Desktop**

CSSP Desktop objectives: **Speed**, **Available Offline**, **Security**, **Continuity**, **Bilingual**, **Cost** and **Empowering Users**.

CSSP Desktop

Microsoft Azure Storage

wmon01dtchlebl

CSSP database

and CSSPWebTools

Create and store files to Azure

ECCC Firewall

CSSP Web APIs Local

CSSP Client (Angular) Web

User

Computer

Starts

Need file

Store in memory

Manage local files

(more below)

File requested

Request file

View info on web page

Manage local files

Provide file

yes

no

Once a day for now

Is file localized

Check Azure for changed file

once

Provide file

Download from Azure

Provide file

yes

no

Localize file

Send Changes

(more left)

Send Changes (coming soon)

Send Changes

Send Local CSSP Database

Microsoft Azure Storage

Download local database stored from Azure



Internet required

No Internet required (once files are localized)

file changed

Sync with

wmon01dtchlebl

CSSP database

by designated ECCC staff

**Speed**

Speed is probably the most important aspect of this new CSSP Desktop application. Compared to the existing CSSP Web Tools currently being used inside ECCC, it’s probably more than 100 times faster while reducing internet traffic considerably. Its design and the use of Microsoft Azure (cloud) as a file storage and provider has allowed us to improve the performance substantially from its previous version.

**Available Offline**

One of the crucial requirements is to be able to use the application in the field where no internet connection is readily available. In order to do this the user can pre-download all required files from Azure (cloud) with a single click. These files are automatically stored on the user local hard drive for future use. The users are then able to bring their laptop in the field and use it as if they were connected to the internet once all the required files are on the user hard drive

**Security**

Having some sensitive data (presumably) within our database requires some level of access control for different users and no access at all to the others. All information stored in Azure is only available to allowed users and is password protected. In order to be able to use CSSP Desktop, an Administrator of the system need to provide access to the user (done once). All users can only use CSSP Desktop with their unique email and password. In no time can outside user access Azure stored information without the CSSP Desktop application working in the background.

**Continuity**

In no time should the existing CSSP Web Tools be out of service while this new version (CSSP Desktop) is being created and implemented. The fact that the CSSP Web Tools is being used by many ECCC staff across the country and on a regular basis is a critical application for ECCC delivery of the Canadian Shellfish Sanitation Program (CSSP) with its departmental partners (DFO and CFIA).

**Bilingual**

CSSP Desktop is, like its predecessor, fully bilingual. The user has the ability to work in the language of their choice. Changing from one language to another is done with one click and the application does not have to be restarted of reloaded.

**Cost**

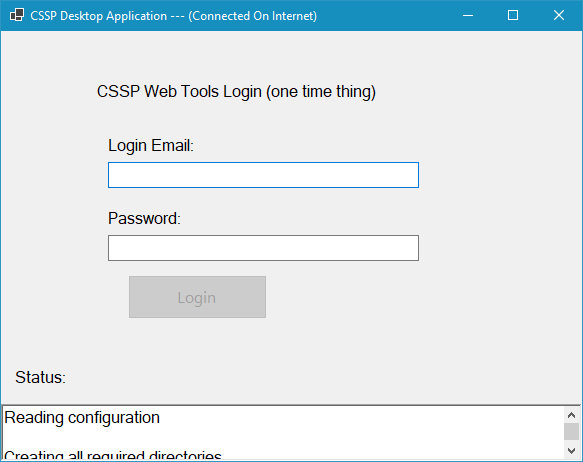
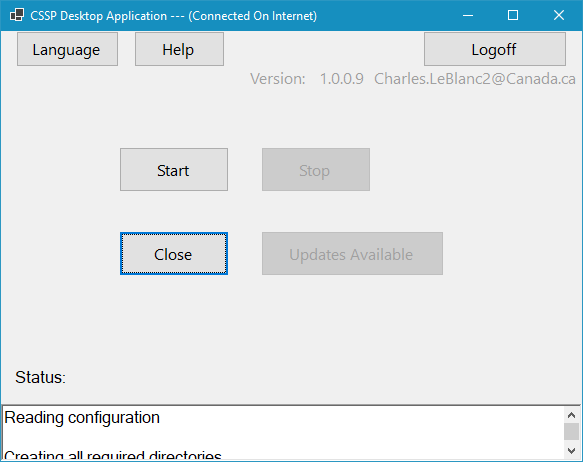
The cloud (Azure or others), if used properly can be a huge saving in cost and time. Currently for the CSSP Desktop, the amount of information stored in the cloud is ~ 200 GB and its current cost per month is less than $50. This is currently being covered by the $200 monthly credit received because ECCC own copies of Visual Studio.

**Empowering Users**

User should have access to all their data and have all the tools to manage it. They should also feel empowered while using CSSP Desktop. In no time should they have to rely on an IT or database person to accomplish their daily tasks. At the end of the full implementation of CSSP Desktop, the user should see the application as the go to tools for numerous components of their yearly tasks. Like being able to produce reports or partial reports with a single click. Setup and run modelling scenarios. Add and manage sanitary surveys related information. Quickly calculate the statistics required to proposed closures to DFO. Etc.

**Explaining the moving parts of CSSP Desktop**

The CSSP Desktop is really 3 applications in one. We have the conductor (CSSP Desktop) responsible for orchestrating the other 2 (CSSP Web APIs Local and CSSP Client). It’s also the application managing all the updates which will be coming in the future. This is the application that is being installed from the web site <https://csspdesktopinstall.azurewebsites.net/> . It is a Windows application with basic visuals to help the user navigate through required login and future updates. It looks like:

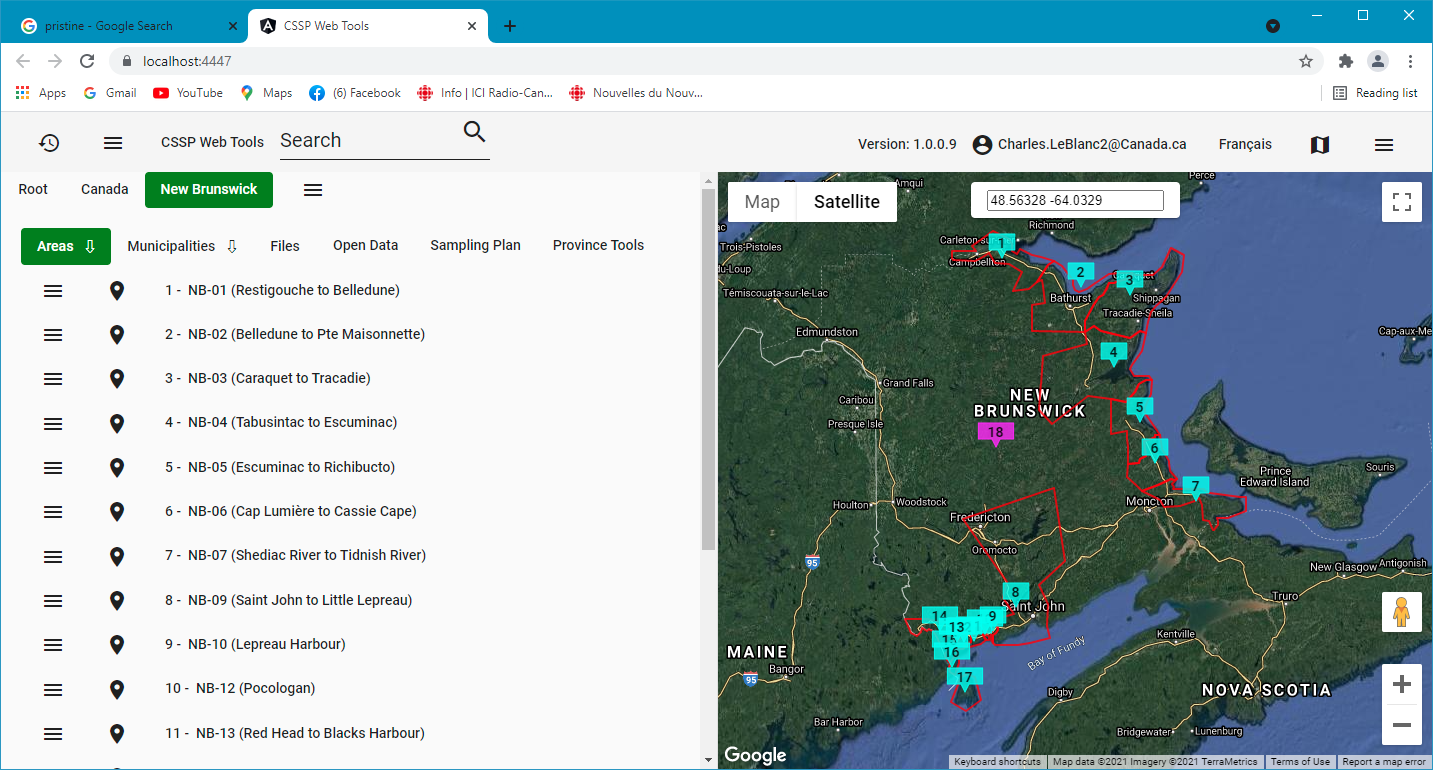
Once you have logged in you will see another screen to help with updates and start the other applications. Once you have click on the Start button, CSSP Desktop automatically starts CSSP Web APIs Local and CSSP Client. The first is an actual web server running on the user computer in the background. Its main purpose is to wait and provide the user with information/files when needed. The second (CSSP Client) is just a web page that is provided by the background web server (CSSP Web APIs Local). It’s the user visual.

**CSSP Web APIs Local**

This background web Application Programing Interface (API) which runs on the user computer is responsible for receiving commands from the user via the web (CSSP Client) and converting all this into something the user can use. So, it will receive a request and do everything necessary to reply back something to the user. Sometime it will be sending a file (json, picture, report, excel, etc.). But before that, it will verify if the file is already downloaded from Azure in order to limit the traffic to and from Azure (cloud). It is also responsible for managing and knowing which file is already localized, which files need to be redownloaded because Azure has a newer copy. It is also responsible for sending error message when needed. There is visual for this application. However, if you want to find out if it’s running, you can open the task manager (Ctrl+Shift+ESC) and look for CSSPWebAPIsLocal within the list of running applications. You will find it under the running application CSSPDesktop.

**CSSP Client**

This is really the web page showing all the data, information and files. This web frontend is the visual for the user. It’s where the user can see all the data and uses inputs fields and buttons to send commands to CSSP Web APIs Local. In order to accelerate the whole system even more, this web page is designed with the capability to use browser memory to reduce the number of reload the web page needs to do. So, it will only reload the information/file if not already in memory. You will feel like home, because the visual is very similar to the visual of CSSP Web Tools (see image below).



At this time the CSSP Web Tools continues to be the official data/information manager, it is also the location where the official CSSP database resides (inside the firewall of ECCC). In the future this database will be hosted by Azure (cloud). The CSSP Web Tools is also responsible for the creation of all the .gz files (compressed .json files) which all together (~ 10 000) replicates the content of CSSP database. At this time and for continuity with the CSSP Web Tools, the .gz files are created every day at midnight and only the changed files are sent to Azure (cloud). In the future, this will only be done when the CSSP database changes.

The CSSP Client, which is actually a web page, is the visual via web. All information sent to the web page is done with request to CSSP Web APIs Local application which works in tandem with the CSSP Client. All files are stored locally (localized) before being sent to the web page. CSSP Web APIs Local take cares of managing all the Azure files and local files in order to only download required files once. It has a mechanism to check Azure cloud to verify if the file changed since the last download. If it did change then it is also responsible to localize the file, which is downloading the file and storing it on the user computer for later use. CSSP Client also manage memory in the browser to accelerate the reading and parsing of the .json files.

Although, at this time, the user will not have the capability to change the information using CSSP Desktop, future version will allow this. Taking advantage of changing everything local by using an empty replica of the server CSSP database. All changes will reside locally until the user decides to send all the changes at once to Azure. Some other application will read the database file just sent and synchronize it with CSSP database while being validated by designated ECCC staff. This would keep the master database pristine.

One of the major differences between CSSP Web Tools and CSSP Desktop is the way the information is being transferred and parsed. For the CSSP Web Tools, all information is digested on the server before being sent to the user. That is, html pages are created by reading the database and generating an html file that is then being sent to the client. This is done every time the user browses to another page. Some of these pages can be large like the Analysis pages which can go up to 1 MB. The downside to this way of doing thing is also that the database is heavily solicited and has to work very hard all the time as more and more people need access the site and browse the information. This way of doing things was the standard way for many years, and web developers were just asking for bigger and faster computers/servers to satisfy the ever-growing need for large amount of information.

Lately, however, with the help of multinational companies like Google, Facebook and Microsoft, better ways and tools are now available to reduce web traffic considerably. Google is pushing something call “Angular” which is a JavaScript based framework which can be downloaded to the browser once at the beginning of the browsing session. It’s responsible for downloading and parsing .json files into html documents for which the browser can then show to users. Just this part already reduced the amount of information being transferred from the servers to the user computer/browser. To make this part even more streamlined, we do not request the information from the database directly but via .json files which are created only once and stored on Azure cloud. This also greatly limits the stress put on the ECCC servers and databases. So rather than creating these files every time a user request it, they are already created and all that needs to be done is to provide the file. These files could be stored anywhere. For now, and because of speed and low cost, we opted to stored them in the cloud (Azure). Furthermore, to satisfy the need for users to be able to work on their data without internet, and also reduce the number of times a particular file needs to be downloaded, local storage on the user computer hard drive is being used to temporary store Azure files that are needed or will be needed while working in the field.

Next steps, CSSP Desktop will eventually replace the CSSP Web Tools, CSSP Pollution Source Input Tool and the CSSP Water Quality Input Tool currently being used by ECCC staff, contractors and provincial partners.

**Note**: A Commercial of The Shelf (COTS) product is currently in its final stage of being purchased. Its full implementation will take a few years. We, SWCP, should not accept anything less than what is being proposed here.

CSSP Desktop File structure

**Note**: These directory structure and files could change in the future

**C:\CSSPDesktop\**

All necessary files associated with the CSSP Desktop resides under this directory

**C:\CSSPDesktop\csspfiles\**

Contains localized files (reports, images, spreadsheets, modelling related, etc.)

**C:\CSSPDesktop\csspjson\**

Contains localized .gz files (CSSP database related information converted into json files)

**C:\CSSPDesktop\csspjsonlocal\**

Contains user changes in the form of .gz files

**C:\CSSPDesktop\cssplocaldatabases\**

**CSSPDBLocal.db** is an SQLite database storing user changes

**CSSPDBManage.db** is an SQLite database for managing Azure and local files

**C:\CSSPDesktop\csspotherfiles\**

Contains some files required for web visuals when working offline

**C:\CSSPDesktop\cssptempfiles\**

Contains a copy of the files created from web page graphics and tables downloads

**C:\CSSPDesktop\csspwebapislocal\**

Store the .dll and .exe needed to run CSSP Web APIs Local and CSSP Client