MC EDT Web API Project

About DoctorCare

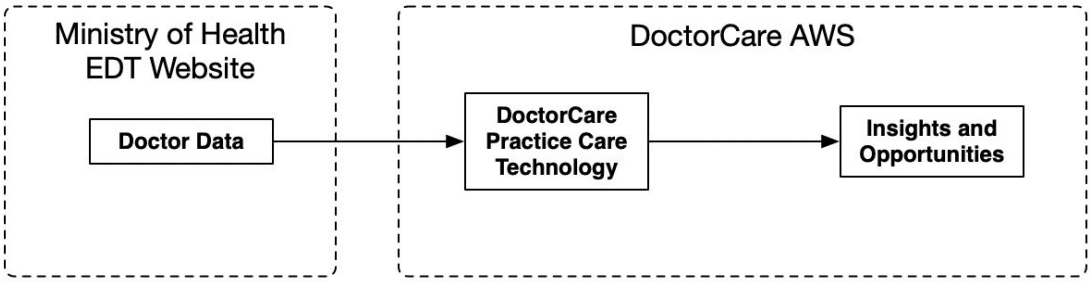
DoctorCare is a group of clinicians and business professionals dedicated to improving healthcare in Ontario. We are the only provider focused exclusively on helping Primary Care Physicians enhance patient care, run better businesses – and improve their own quality of life.

Our services focus on enhancing chronic disease management, billing and technology, and clinic workflow, helping 1,200+ family doctors improve the lives of more than 1.5 million patients. Ontarians get tremendous benefits from our healthcare system. But inefficiencies, inconsistencies and lost opportunities can limit the quality of care they receive. This is particularly evident in primary care practices, where doctors are caring for complex patient populations and simply don't have the time or resources to focus on maximizing efficiency.

The founders of DoctorCare, Paulo Gomes and Adam Hutton, both with extensive experience providing systems for the healthcare industry, realized that the solution was simple. Doctors needed expert support to help them run their practices – support they simply didn't have the resources for in-house. So in 2010, they founded a service model with a clear approach: patients count on doctors – so doctors need someone to count on too. That's DoctorCare: a solution provider to help doctors cut wait times, improve the delivery of preventive care, better manage chronic disease programs – and help manage OHIP fee schedules and add billing efficiencies.

Project Motivation

DoctorCare's Practice Care Product is an analytic product that provides primary care practitioners with insights about opportunities to improve their medical practice and business. DoctorCare has developed technology product (Practice Care) that translates a set of data into actionable insights. In simple terms:



Currently, the Doctor Data input to the product is acquired by downloading files from the ministry of health's website using a web scraping technology. The data is then stored in the Practice Care platform (hosted on Amazon Web Services - AWS) at which point it is analyzed with outputs stored in the same Practice Care platform.

DoctorCare would like to replace the web scraping technique for acquiring the Doctor Data and instead, acquire the data from the Ministry of Health's proper web service API - MC EDT Web service. This migration from web scraping to api is required as:

- the web scraping is prone to errors
- The web scraping is not efficient
- The web scraping is not officially supported by the Ministry of Health for the purposes of automation.

Project Goals

1. Develop python library capable of connecting to ministry of health's MC EDT API and acquiring the Doctor Data from the MC EDT Web Service.
2. Pass Ministry of Health certification. The Ministry of Health conducts certification for all applications that would like to consume from their MC EDT API. Passing this certification is a required goal.

In Scope

1. Develop python library capable of passing ministry of healths certification tests.
2. Develop supporting tooling (as required) in order to pass the ministry of health's certification tests.
3. Project management of entire development and testing process.
4. Support any and all testing as required by the ministry of health.

Out of Scope

1. Integration of final python library into Practice Care product.
2. Conformance testing registration (DoctorCare to register and manage ministry of health registration needs).

Useful Documentation

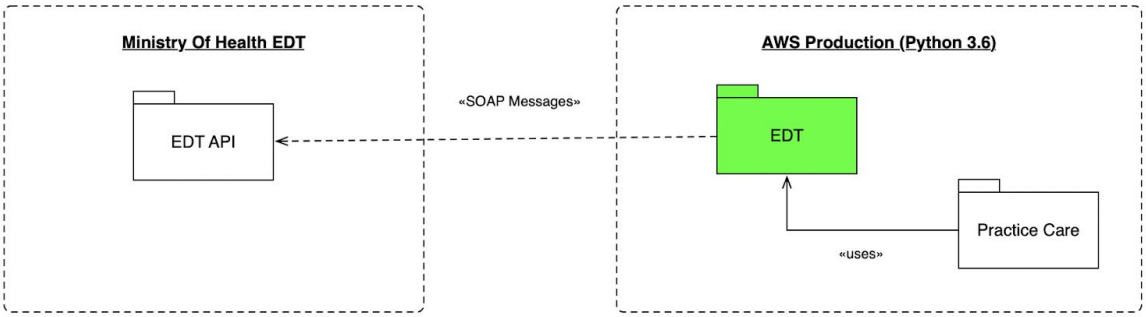
See “Documentation” folder for various documents on the MC EDT service.

Milestones

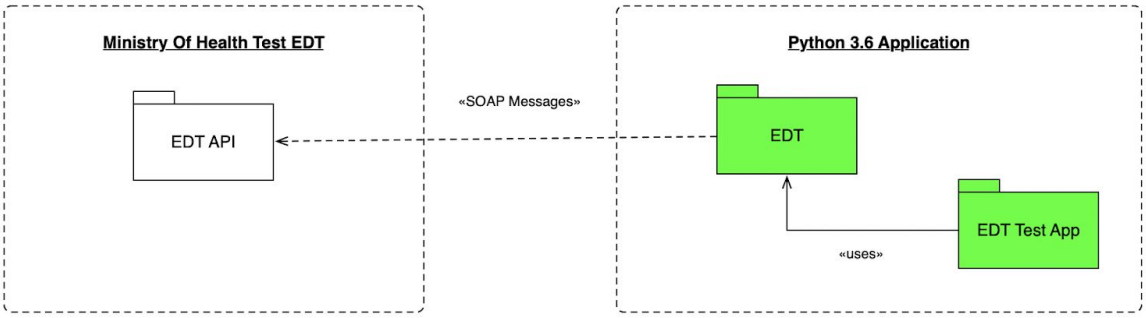
1. DoctorCare officially registered with Ministry of Health's conformance testing program.
2. Python library API design complete.
3. Python library passes internal conformance testing.
4. Python library passes Ministry of Health conformance testing.

System Architecture

Production Configuration



Test Configuration



Components

1. **EDT API**. The API provided by the Ministry of Health.
2. **EDT**. The python library capable of connecting to the EDT API.
3. **Practice Care application**. The application developed by DoctorCare.
4. **EDT Test App**. A test harness application that is sufficiently capable such that it can pass the ministry of health's conformance tests.

Note that components in green are to developed as part of the scope of this project. Also note that the production environment is the final target deployment configuration but that the test environment is to be developed for the purposes of internal testing as well as for conformance test passing.

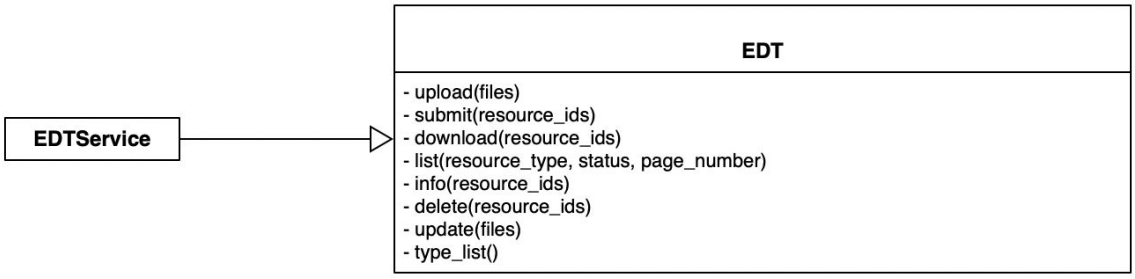
EDT API

Relevant documentation for the ministry of health's EDT API can be found online.

- Technical Specification for Medical Claims Electronic Transfer (MCEDT) v 4.4
- FAQ for EDT service
- EBS Generic Security Specification

EDT

The edt package is a python library that supports python 3.6. The minimum api for the package will look like the following (and mirrors the api in the technical specification):



Details of the EDT interface are as follows

Method Name	**Inputs**	**Outputs**
upload	• Files. An array of dictionaries where each dictionary has the following	Dictionary. • **audit_id**. String.

	keys: ○ **content**. string. The contents of the file. ○ **description**. String. ○ **resource_type.** String.	● **description**. String. ● **resource_id.** Integer. ● **code**. string
submit	● **resource_ids**. Array of integers.	Dictionary. ● **audit_id**. String. ● **description**. String. ● **resource_id**. Integer. ● **code**. string
download	● **resource_ids**. Array of integers	Dictionary. ● **audit_id**. String ● **content**. String ● **resource_id**. Integer ● **description**. String ● **code**. String. ● **message**. string
list	● **resource_type**. String. ● **status**. String. Allowable values “UPLOADED”, “SUBMITTED”, “WIP”, “DOWNLOADABLE”, “APPROVED”, “DENIED”. ● **page_number**. integer	● **audit_id**. String ● **create_timestamp**. Datetime. ● **description**. String ● **modify_timestamp**. Datetime ● **resource_id**. Integer ● **code**. String. ● **message**. String. ● **status**. String. Allowable values “UPLOADED”, “SUBMITTED”, “WIP”, “DOWNLOADABLE”, “APPROVED”, “DENIED” ● **result_size**. integer
info	**resource_ids**. Array of integers	Array of dictionary values ● **audit_id**. String ● **create_timestamp**. Datetime ● **description**. String ● **resource_type**. String. ● **modify_timestamp**. Datetime.

		• **resource_id**. Integer • **code**. String. • **message**. String. • **status**. String. Allowable values “UPLOADED”, “SUBMITTED”, “WIP”, “DOWNLOADABLE”, “APPROVED”, “DENIED”
delete	**resource_id**. Array of integers	Dictionary: • **audit_id**. String • **create_timestamp**. Datetime • **description**. String • **resource_type**. String. • **modify_timestamp**. Datetime. • **resource_id**. Integer • **code**. String. • **message**. String. • **status**. String. Allowable values “UPLOADED”, “SUBMITTED”, “WIP”, “DOWNLOADABLE”, “APPROVED”, “DENIED”
update	• **content**. String. • **resource_id**. integer	Dictionary: • **audit_id**. String • **description**. String • **resource_id**. Integer • **code**. String. • **message**. String. • **status**. String. Allowable values “UPLOADED”, “SUBMITTED”, “WIP”, “DOWNLOADABLE”, “APPROVED”, “DENIED
get_type_list	N/A	Dictionary: • **audit_id**. String

		● **group_required**. Bool ● **description**. String. ● **description_french**. String ● **resource_type**. String. ● **billing_numbers**. Array of dictionaries where each dictionary has the following: ○ **billing_number**. String. ○ **group_number**. String ● **code**. String. ● **message**. string.