apitext

*An **API** that has an appetite for **TE**I-**X**ML **T**ranscriptions*

Apitext: An API for TEI-XML Transcriptions

University of Washington Information School, INFO 490/491 Capstone Project

project team

Guiyan Bai
Design & User Experience

Chris Sumption
Project Manager, User Experience, Design & Development

Michael Andrea
Development & User Experience

introduction

Abstract

For over twenty years the Text Encoding Initiative (TEI) has managed and developed a set of encoding guidelines for the representation of humanities, social science, and linguistics -- to preserve and share -- texts in digital form. Using Extensible Markup Language (XML) as its backbone, TEI is the generally accepted encoding model for the digital humanities. Due to XML's extensible nature, it can often be difficult to share these files, and problematic to make them interoperable. Our Application Programming Interface (API) for TEI-XML documents addresses these challenges. It requires no prior programming experience to use, can be installed using standard File Transfer Protocols (FTP), and is able to return multiple interoperable views of a TEI-XML file using a Uniform Resource Identifier (URI) as its method of query.

introduction

Problem Statement

How do we build a RESTful Application Programming Interface (API) for TEI-XML documents that provides greater interoperability while at the same time is easy to use by an audience with backgrounds in the humanities, social sciences, and linguistics.

What is TEI?

- TEI stands for The Text Encoding Initiative
- It is a consortium which collectively develops and maintains a standard for the representation of texts in digital form.
- Its chief deliverable is a set of Guidelines which specify encoding methods for machine-readable texts, chiefly in the humanities, social sciences and linguistics
- XML is the language used for encoding

project documentation

proposal

Problem Statement, Background, Outcomes, Audience, Social Impact, Schedule

discovery

Literature Review, Audience Research, Comparator Analysis, Feature Requirements

design

Storyboards, Mood boards, Paper Prototype, User Testing, Findings, Design Mockup, API Requirements

development

Backlog
Usability Testing
Screenshots

media

Pitch Video
Project Poster
Documentation Website
Demonstration Website

future

Recommended Features

proposal

Problem Statement, Background, Goals, Audience, Social Impact, Schedule

project proposal

Background & Context

A typical TEI Digital Humanities project usually consists of the following components:

- A digitized collection of textual objects. The types of these objects could be diverse and might include things like letters, books, newspapers, and diaries.
- Digital transcriptions of these items in the form of XML files that use the TEI Schema. These encodings would contain tags that highlight items of interest to the respective project. For instance, one project may be interested in capturing information related to persons mentioned in their texts, while another focuses on places.
- An application that allows users to interact with the TEI-XML encoded documents. This interaction could take the form of the properly formatted display of the text, it also could be something as simple as listing person names contained within the text.

This project intends to help Digital Humanities projects with the third bullet point.

Background & Context (continued)

Chris Sumption has been a research intern with a TEI-XML Digital Humanities project for the last five quarters and has witnessed firsthand the challenges faced when building applications that display the contents of a project's TEI-XML archives. There are usually two scenarios that occur:

- The first, if it can afford to, a project brings in outside developers to custom build a viewer.
- The second, the project members look to other projects to see if they can adapt an open source solution to fit their own needs. At this point many projects hit a brick wall. The documentation and technologies presented to users who might want to apply these solutions are often nonexistent or worse, designed for computer scientists, not historians.

project proposal

Desired Outcomes

This project intends to deliver a working RESTful Application Programming Interface (API) for TEI-XML Encodings prototype by May 20th, 2016. This prototype will:

- Provide an interface for applications and humans that that will allow clients easy access to a DH project's archive of TEI-XML encoded documents. If applicable this interface will also enable the archived TEI data to be more interoperable with TEI data from other projects and domains.
- Be able to be implemented by individuals who have a minimum of technical expertise. This will be accomplished through the architecture of the API and the design of its documentation.

Intended Audience for this Prototype

Scholars, Archivists, Educators, Enthusiasts, and Programmers with varying backgrounds and levels of experience.

Social Impact

A Digital Humanities project preserves a record of the past while at the same time making that record available to the general public. Typically, these projects are staffed with individuals with backgrounds in the Humanities, not Computer Science. This lack of technical knowledge typically forces a project to bring in outside developers. If a project does not garner enough funding to cover these added costs, this record cannot be properly presented. Basically, if the project has money, the public gets access.

This project will not completely solve this barrier to access, but it will make it easier and more cost effective for these projects to publicize their materials.

project proposal

Resources, Experts & Tools

This project will be using the following technologies and tools:

- eXtensible Markup Language(XML), Text Encoding Initiative (TEI), Application Programming Interface (API), Hyper Text Markup Language (HTML), Representational State Transfer (REST), Hypertext Preprocessor (PHP), Javascript, GitHub, and Linux.

This project will be using the following resources:

- Web server space for hosting documentation relating to the project and its prototype.

This project will take advantage of the following experts:

- David Stearns Ph D (Information Systems Development)
- Sarah Ketchley Ph D (Digital Humanities, TEI, Egyptology).

project proposal

Schedule & Milestones

Initiation & Design Phase

01/13/2016	Project Proposal
02/03/2016	Discovery Brief
02/17/2016	Conceptual Design
03/09/2016	Design Blueprint

Development Phase

04/01/2016	Complete Sprint 1

Development Phase (continued)

04/08/2016	Complete Sprint 2
04/15/2016	Complete Sprint 3
04/22/2016	Complete Sprint 4
04/29/2016	Complete Sprint 5
05/06/2016	Complete Sprint 6
05/13/2016	Complete Sprint 7
05/20/2016	Prototype Delivery

discovery

Literature Review, Audience Research,
Comparator Analysis, Feature Requirements

major goals

In order to successfully solve the problem outlined in our problem statement, we identified three major goals:

Useability

Because this prototype and the documentation associated with it will be used by a diverse audience of users with varying levels of technical experience, it is important that it be useable by all audience members.

Interoperability

By the end of this development cycle our prototype must be able to make any TEI-XML file that it interacts with, more interoperable with other TEI-XML files.

Changeability

Our prototype must follow software architecture best practices. Future versions of the prototype must be compatible with past implementations.

literature review

Typically, digital humanitarians encode texts for three reasons: to preserve longevity, to help in the analyzing of a text, and to share that text with other audiences (Bauman, 2011). Using XML as its backbone, the Text Encoding Initiative (TEI) is the generally accepted encoding model for the Digital Humanities.

Schöch (2013) gives a very good overview of TEI, he explains:

Technically, TEI documents are usually considered semi-structured; usually, they follow a data model expressed in a schema, but such schemas allow for considerable flexibility. In addition to a very clean transcription of the text, digital editions using TEI can make a lot of information explicit: first of all, TEI files contain not just the full text, but also metadata associated with the text (in the teiHeader section); also, the data is structured and explicit: there is markup making the structure of the text explicit, identifying parts, chapters, headings, paragraphs, as well as page and line breaks, for example. Finally, many more types of information can be specified: for example person names in a novel or play, place names in a letters or documents, and many more things; and links to other parts of the documents and to external documents. Making all of these things explicit allows to visualize them in specific ways and to index, count and analyze them computationally.

literature review

We intend to address what Schöch touches on in the last sentence of his overview, how TEI documents can be shared or made interoperable with not just each other but other non TEI data. This literature review will: explain the challenges involved with TEI document interoperability; cover how current Digital Humanities usability research can influence how these challenges are approached; and review more recent attempts to further a TEI interoperability solution. I then intend to build upon this previous work by suggesting a new line of Design Research that holds potential for addressing the challenge of TEI interoperability.

A bone of contention within the Digital Humanities community is whether or not the TEI XML standard can allow its users to make their encoded texts interoperable. According to Bauman (2011), “An interoperable text is one that does not require any direct human intervention in order to prepare it to be used by a computer process other than the one(s) for which it was created.” The challenge with making TEI-encoded texts interoperable has to do with the “flexibility” that Schöch mentioned in his overview. There are viewpoints that express that interoperability is probably an impossible task due to the nature of the TEI XML tags used. If you set your tags up so they align with a particular application or process, more than likely you will start abusing the names of those tags. On the other hand, if you try to anticipate interoperability with processes not yet invented, you run the risk of decreased expressiveness in your tagging system (Bauman, 2011).

literature review

Other viewpoints are more optimistic and see this as a people problem by explaining that the goal of interoperability doesn’t necessarily have to be impossible, it just means that developers need to take into account that biases do get instilled into tagging choices because every person sees things differently when they look at a text (Schmidt, 2014). Voices within the library community emulate this “people” viewpoint. McDonough (2009) points out that, "if we are to deal with the issues of interoperability that continually manifest themselves in the realm of structural metadata standards for digital libraries, we need to cease viewing this purely as a technical problem, and acknowledge that it is the result of the interplay of technical and social factors.”

So, if it’s recognized that people are a major component to this challenge, it makes sense that Human Computer Interaction (HCI) and Design Research might be able to provide some answers. Recent usability research into Digital Humanities tool use seems to suggest that tools don’t need to be more sophisticated, they need to: be more transparent, applicable to a broader audience of traditional humanitarians, and favor ease of access over polished visualizations (Gibbs & Owens, 2012). Through prototyping, Daniel Carter explored different visualizations of TEI documents but more importantly he touched on the issue of interoperability by imagining the implications involved by the use of standoff markup when moving TEI annotations to external files (Carter, 2014).

literature review

Schmidt (2014) takes the idea of separation of markup a step further by explaining that:

> *To achieve interoperability—the ability to load a transcription into various programs without modification—it is thus obvious that the markup must first be removed from the text, because it is the markup that contains virtually all of the interpretation, and what little interpretation remains does not stop the text from being interoperable.*

He even goes so far as to suggest a structure for this separation by arguing that since TEI-encoded data can be organized into markup, annotation, metadata, and the plain text itself, those classes could be leveraged to divide the TEI document into a bundle of files containing: plain text or HTML files that witness the text, markup if plain text is used, files containing annotations, and files for metadata about the text (Schmidt, 2014).

That's where research into TEI interoperability is currently at. To summarize, the TEI-encoding standard is important to Digital Humanities research. A major drawback to the standard is that it's difficult to make its encoded documents interoperable without weakening it in some way. Furthermore, the interoperability challenge may not be a technical one but instead a people challenge. If that's the case, then best practice would suggest that a solution to this

literature review

challenge should be user focused, less sophisticated, but be able to support quick and easy visualization of TEI-encoded data originating from a broad audience of users. Research within the last couple of years shows that the separation of the different data classes within a TEI-encoded file may be a solution to the interoperability challenge.

Carter and Schmidt are on the right track by exploring the separation of data classes but I believe they are overlooking the people part of the problem. They are adding multiple files and complicating use of the TEI standard. What if instead of having users create multiple physical representations of the information contained within a TEI document, you instead keep the encoded file intact then use it as a data source from which an application interface could draw out those multiple class visualizations, which then could be presented to human or machine users. This type of technology already exists in what is known as Representational State Transfer (REST). A RESTful web service uses anything a web server can manage like XML files, whole database tables, specific database records, algorithms, etc... then allows those resources to be easily addressed, represented in different interchange formats, and in some cases manipulated (Stearns. 2015). Therefore, I propose design research that seeks to fill this “people” gap exposed in this literature by addressing the following research question: will the design, construction and implementation of a RESTful Application Programming Interface (API) for TEI-encodings contribute a solution to or at the very least produce additional avenues of exploration concerning TEI-encoding interoperability?

literature review (references)

Bauman, S. (2011). Interchange vs. interoperability. Balisage: The Markup Conference 2011. doi:10.4242/BalisageVol7.Bauman01

Carter, D. (2014). A design methodology for exploring and communicating system values and assumptions. Journal of the Text encoding initiative, (7). doi:10.4000/jtei.974

Gibbs, F., & Owens, T. (2012). Building better digital humanities tools: toward broader audiences and user-centered designs. Digital humanities quarterly, 6(2). Retrieved from http://www.digitalhumanities.org/dhq/vol/6/2/000136/000136.html

McDonough, J. (2009). XML, Interoperability and the social construction of Markup languages: The library example. Digital humanities quarterly, 3(3). Retrieved from http://www.digitalhumanities.org/dhq/vol/3/3/000064/000064.html

Schmidt, D. (2014). Towards an interoperable digital scholarly edition. Journal of the Text encoding initiative, (7). doi:10.4000/jtei.979

Schöch, C. (2013). Big? Smart? Clean? Messy? Data in the humanities. Journal of digital humanities, 2(3). Retrieved from http://journalofdigitalhumanities.org/2-3/big-smart-clean-messy-data-in-the-humanities/

Stearns, D. (2015, February 19). High-performance, scalable, and intuitive web services. Lecture presented at INFO 380 A: Information Systems Analysis And Design in University of Washington, Seattle.

audiences

During our audience definition workshop we came up with a general sketch of traits that we would expect to see in users of our RESTful Application Programming Interface (API) for TEI-XML documents.

- Scholars, archivists, educators, enthusiasts, and programmers.
- Have a background or interest in the Digital Humanities.
- May or may not have backgrounds in the humanities, social sciences, and linguistics.
- Ages 18 - 80
- Involved with a research project that:
  - works with textual documents that have been encoded using TEI-XML
  - wants to make available their texts available to a broader audience
  - wants to be able to output the data from their documents into other applications
  - has a web presence, and the technical expertise to be able to upload content to that presence

audiences

We then wanted to compare that sketch to actual TEI-XML users. We interviewed three TEI-XML users breaking up questions into four general categories:

User Background	Display Related	TEI-XML Related	Documentation
Questions relating to background and what drew the user towards TEI-XML research Questions relating to technologies worked with or exposed to	How the user publishes content to the web Questions relating to the display of their TEI-XML files	Basic assessment of the user's level of TEI-XML proficiency Things the user would like to be able to do with your TEI-XML files that you're not able to do right now?	Users success and failure stories working with API documentation

audiences

From those interviews we gained a sense of some of their experience, motivations and barriers real world users encounter while pursuing their TEI-XML research.

- They were all in their mid forties.
- Our group of users have a diverse range of technical experience, but common traits included: basic understanding of web content publishing practices and proficiency working with the File Transfer Protocol (FTP), XML experience, HTML experience, and experience working with a content management system like Wordpress.
- Our users all had a desire to make their TEI-XML research not only readily available by a broader audience, but also interoperable with other data sets (not necessarily TEI-XML data)
    - Consideration: It may be important to find at least one user who is not interested in interoperability.
- Our users have all had bad experiences working with TEI-XML documentation (among others), but when they had good things to say, they stated that relevant examples were what made those interfaces useable.

We then used those interviews to develop three personas for our project.

audiences

Persona One: Scholar

Background: Specialty is art history in the first millennium, has taught courses in Egyptian history and archeology, currently working on a project that digitized, transcribed, and encoded travel diaries from the turn of the twentieth century using TEI-XML.

Technologies Used: HTML, TEI-XML, XSLT, Wordpress, Mac, FTP, Omeka, Wikimedia, Oxygen

TEI-XML Related: Has never created or modified TEI-XML tags, When looking at other projects TEI-XML files: first looks if the project is tagging people names then looks at how the project tagged TEI-Header data.

Goal: Be able to dynamically connect the person names in the projects TEI-XML files to biographies about those people stored in the projects Omeka platform.

audiences

Persona Two: Semi Technically Proficient Enthusiast

Background: speculative fiction writer who is currently researching Annie Jump Cannon for use in a fiction story, wants to create a biographical work about Cannon. Has found writings connected to her, but are not publicly accessible. Wants to transcribe those records, encode them using TEI-XML and make them publicly accessible.

Technologies Used: HTML, XML, TEI-XML, Wordpress, Drupal, Windows, FTP

TEI-XML Related: Has never created or modified TEI-XML tags, and has never really looked at another projects TEI-XML files.

Goal: wants to create an online dynamic timeline of Annie Jump Cannon's writings that allows the viewer to see her writings in the context of data collected from other astronomers of the period.

Persona Two: Technically Proficient Enthusiast

Background: Works in the tech industry. In her spare time digitizes, transcribes, and encodes depression era diaries using TEI-XML standard. Discovered TEI while researching best practices for the preservation of archival records in digital formats.

Technologies Used: HTML, XML, TEI-XML, Javascript, PHP, Wordpress, Drupal, Windows, Terminal / Command Line, FTP, SSH, MySQL, Omeka, Wikimedia, Linux

TEI-XML Related: Uses TEI documentation extensively. Has never modified a tag name, but has created own attributes. When looking at TEI-XML files from other projects, is interested in how the project handles place metadata.

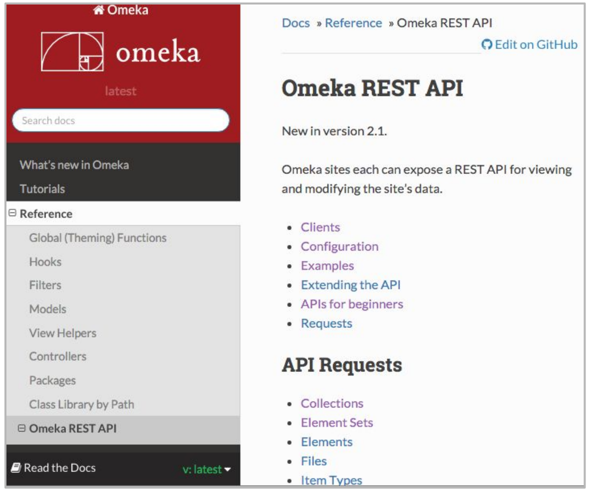
Goal: Wants to be able to easily display diary entries on a map, intermixed with entries from other historical documents from the same time period

comparator analysis

In order to not reinvent the wheel, our project group researched existing technologies that had similar functionality or goals. We were unable to find any existing tools that would provide the same functionality that our REST API prototype would deliver, however the analysis did uncover some interesting components from those tools that may be of use by our project. For instance:

- Usability of competing API documentation by a non technical audience seemed to be a common thread. Which means our project can't adapt their content style, but we might be able to look to their content structure for inspiration.
- The Omeka REST API Documentation educates the user by providing an explanation of what a REST API is. It also provides the user with examples of how to use their API. Should our API documentation take that a step further and provide a real world demonstration of functionality?
- Altova's XML Spy documentation showed people using their tool. As we looked at competitor API documentation, the one thing we didn't see was people. We should include images of people implementing our API.

comparator analysis


Omeka
omeka
latest
Search docs
What's new in Omeka
Tutorials
⊟ Reference
Global (Theming) Functions
Hooks
Filters
Models
View Helpers
Controllers
Packages
Class Library by Path
⊟ Omeka REST API
Read the Docs v: latest

Docs » Reference » Omeka REST API
Edit on GitHub

Omeka REST API

New in version 2.1.

Omeka sites each can expose a REST API for viewing and modifying the site's data.

- Clients
- Configuration
- Examples
- Extending the API
- APIs for beginners
- Requests

API Requests

- Collections
- Element Sets
- Elements
- Files
- Item Types

Omeka
omeka
latest

APIs for beginners

What's a REST API?

Put simply, an API is a formal line of communication between

The Omeka REST API Documentation

Interesting Features:

- Provides the user with an explanation of what a REST API is
  - Covers basic topics geared towards beginning users
  - Also speaks to developers
- Provides user with examples of how to use the API

Challenges:

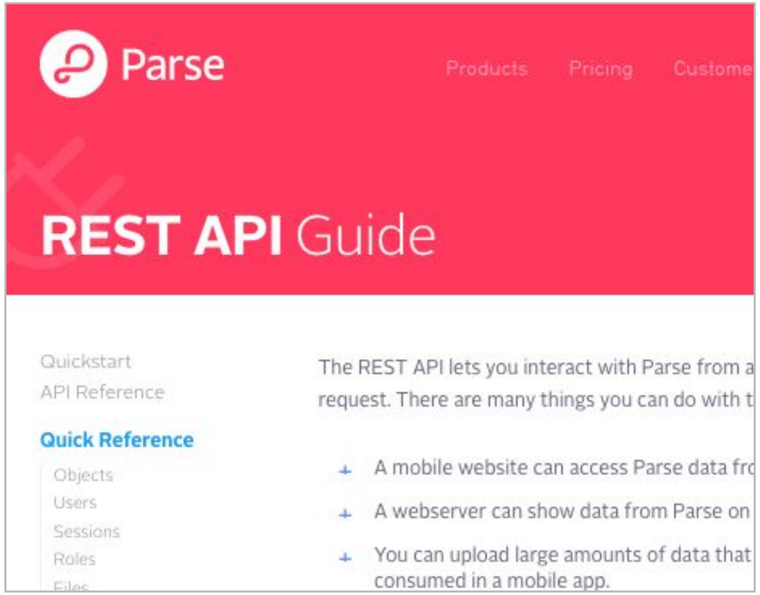
- Explanations are too short; beginners would have trouble grasping the concepts.

Further Exploration:

- The Omeka code is open source, it might be beneficial to take a closer look at how the developers approached the development of their API.

http://omeka.readthedocs.org/en/latest/Reference/api/index.html

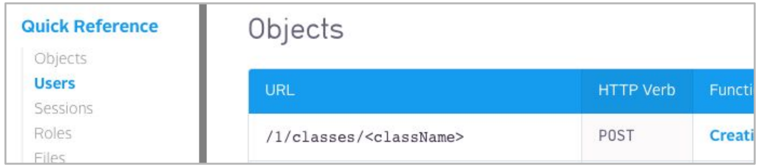
comparator analysis



REST API Guide

The REST API lets you interact with Parse from a
request. There are many things you can do with t

- A mobile website can access Parse data fro
- A webserver can show data from Parse on
- You can upload large amounts of data that consumed in a mobile app.



Objects

URL	HTTP Verb	Functi
/1/classes/<className>	POST	Creati

The Parse REST API Documentation

Interesting Features:

- Is more organized than other API documentations

Challenges:

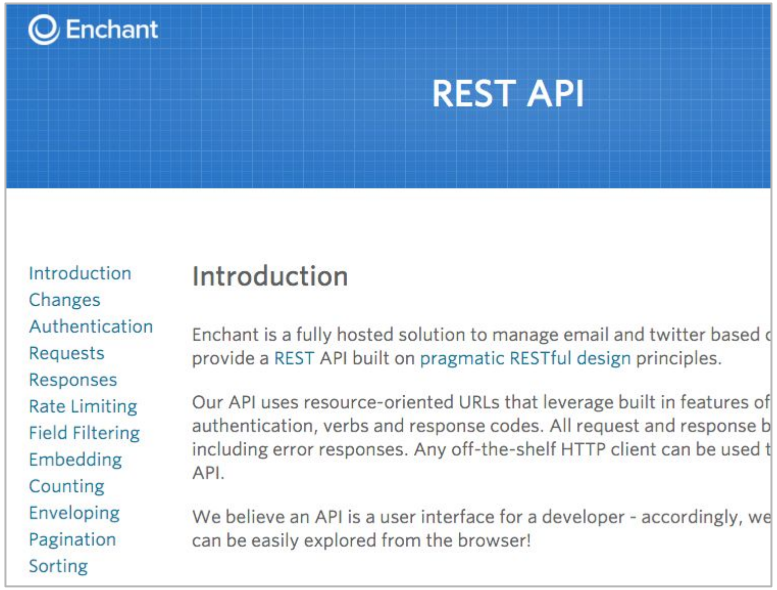
- Target audience of the documentation is developers
- Difficult for a beginner user to understand

Further Exploration:

- None at this time

https://parse.com/docs/rest/guide#quick-reference-objects

comparator analysis



Enchant

REST API

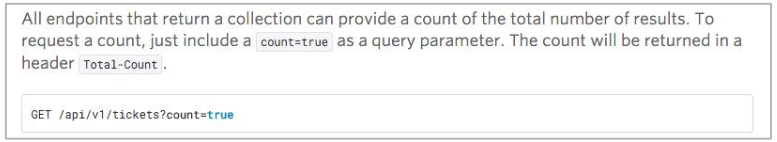
Introduction
Changes
Authentication
Requests
Responses
Rate Limiting
Field Filtering
Embedding
Counting
Enveloping
Pagination
Sorting

Introduction

Enchant is a fully hosted solution to manage email and twitter based c
provide a REST API built on pragmatic RESTful design principles.

Our API uses resource-oriented URLs that leverage built in features of
authentication, verbs and response codes. All request and response b
including error responses. Any off-the-shelf HTTP client can be used t
API.

We believe an API is a user interface for a developer - accordingly, we
can be easily explored from the browser!



All endpoints that return a collection can provide a count of the total number of results. To request a count, just include a `count=true` as a query parameter. The count will be returned in a header `Total-Count`.

```
GET /api/v1/tickets?count=true
```

The Enchant REST API Documentation

Interesting Features:

- Does a very good job of explaining how to use the API (to developers)
- Focuses on pragmatic RESTful design

Challenges:

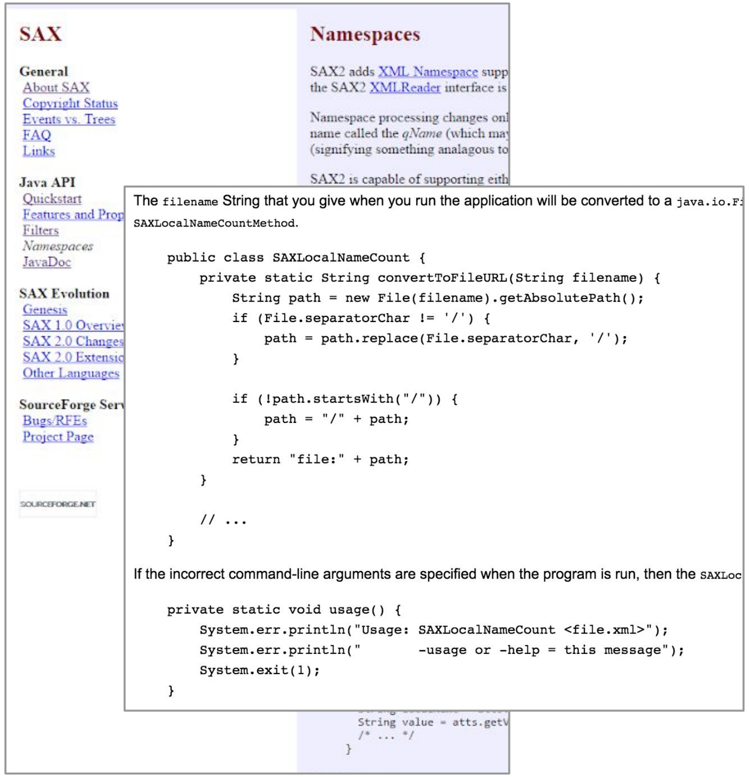
- Target audience of the documentation is developers

Further Exploration:

- None at this time

http://dev.enchant.com/api/v1

comparator analysis



SAX

General
About SAX
Copyright Status
Events vs. Trees
FAQ
Links

Java API
Quickstart
Features and Prop
Filters
Namespaces
JavaDoc

SAX Evolution
Genesis
SAX 1.0 Overvie
SAX 2.0 Changes
SAX 2.0 Extensio
Other Languages

SourceForge Serv
Bugs/RFEs
Project Page

SOURCEFORGE.NET

Namespaces

SAX2 adds XML Namespace supp
the SAX2 XMLReader interface is

Namespace processing changes onl
name called the *qName* (which may
(signifying something analagous to

SAX2 is capable of supporting eith

The `filename` String that you give when you run the application will be converted to a `java.io.Fi`
`SAXLocalNameCountMethod`.

```
    public class SAXLocalNameCount {
        private static String convertToFileURL(String filename) {
            String path = new File(filename).getAbsolutePath();
            if (File.separatorChar != '/') {
                path = path.replace(File.separatorChar, '/');
            }

            if (!path.startsWith("/")) {
                path = "/" + path;
            }
            return "file:" + path;
        }

        // ...
    }
```

If the incorrect command-line arguments are specified when the program is run, then the `SAXLoc`

```
    private static void usage() {
        System.err.println("Usage: SAXLocalNameCount <file.xml>");
        System.err.println("       -usage or -help = this message");
        System.exit(1);
    }
```

```
        String value = atts.getV
        /* ... */
    }
```

SAX: Simple API for XML

Interesting Features:

- An API developed by Oracle to deal with XML conversion and management

Challenges:

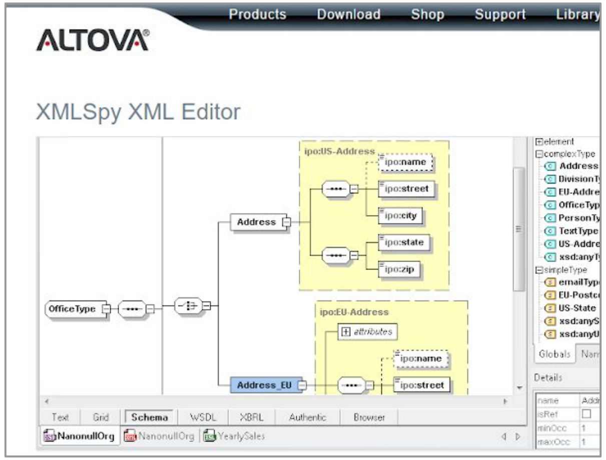
- documentation is hard to find and not clear whether or not you should use Oracle's or the SAX-made documentation.
- Oracle API documentation has hard code in it, which will make it difficult for non-code savvy users to implement their API

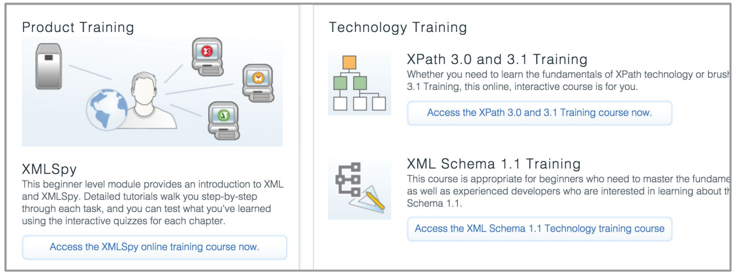
Further Exploration:

- None at this time

https://docs.oracle.com/javase/tutorial/jaxp/sax/

comparator analysis





Product Training

XMLSpy

This beginner level module provides an introduction to XML and XMLSpy. Detailed tutorials walk you step-by-step through each task, and you can test what you've learned using the interactive quizzes for each chapter.

Access the XMLSpy online training course now.

Technology Training

XPath 3.0 and 3.1 Training

Whether you need to learn the fundamentals of XPath technology or brush 3.1 Training, this online, interactive course is for you.

Access the XPath 3.0 and 3.1 Training course now.

XML Schema 1.1 Training

This course is appropriate for beginners who need to master the fundame as well as experienced developers who are interested in learning about th Schema 1.1.

Access the XML Schema 1.1 Technology training course

Altova XML Spy

Interesting Features:

- Visual/graphical interface to manage XML files
- Allows comparisons between different XML documents
- Documentation is very user friendly, directed towards less technology-savvy people (uses visual examples)

Challenges:

- Not open source
- Is a tool, not an API

Further Exploration:

- Their use of imagery in their documentation

http://www.altova.com/xmlspy.html

required features

Using the information collected from our literature review, audience research, and comparator analysis, we defined the following list of required features for our prototype.

Feature	Description	Map to Goal
Understandable Documentation	In order to remove barriers from useage, adequate documentation must be provided and should meet the following requirements: ● It must be easy to find ● It must be easy to understand (it cannot be written for a programmer)	Usability

required features

Feature	Description	Map to Goal
Easy Implementation	In order to accommodate varying levels of technical proficiency, the prototype must: ● Be able to be installed via FTP ● Be able to be configured using commonly available text editing software ● Be able to use technologies that would typically be found on a LAMP server	Usability
Understandable Interface	Because the interface for a RESTful API is text based (via a URL), it is important that the syntax used by the interface make structural sense while at the same time be human readable.	Usability
Open Source	In order to make TEI-XML files more interoperable, users need to adopt technologies that assist in that effort. Charging for these tools would act as a barrier to interoperability.	Interoperability, Usability

required features

Feature	Description	Map to Goal
Provides a Text View	Upon request, the prototype should deliver a view of the TEI-XML file's text only content that exists between the opening and closing "text" tags. This view would be the most interoperable view because most if not all applications can use straight text.	Interoperability
Provides a HTML Markup View	Upon request, the prototype should deliver a view of the TEI-XML file's text and markup content that exists between the opening and closing "text" tags. This view would not be as interoperable as text but HTML is well supported by technologies such as web browsers..	Interoperability

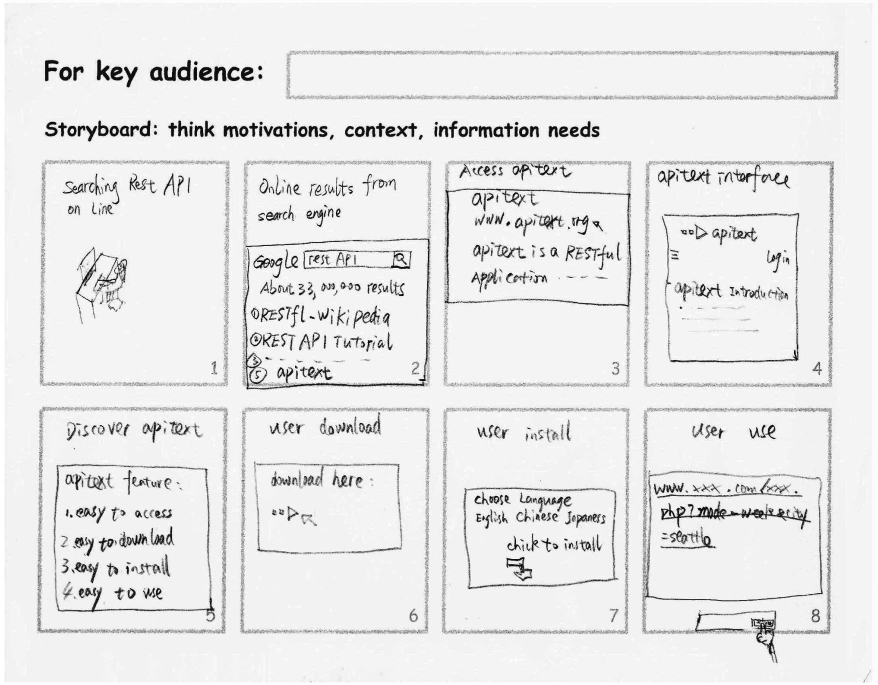
required features

Feature	Description	Map to Goal
Provides an Annotation View	Upon request, the prototype should deliver a view of the any of the “annotation” tags that exist within the opening and closing “text” tags of the TEI-XML file.	Interoperability
Provides a Document Metadata View	Upon request, the prototype should deliver a view of the any of the “document metadata” tags that exist within the opening and closing “teiHeader”” tags of the TEI-XML file.	Interoperability
Must Have Version Control	In order to allow for the prototype to be updatable, it will have the ability to support versioning and backwards compatibility.	Changeability

design

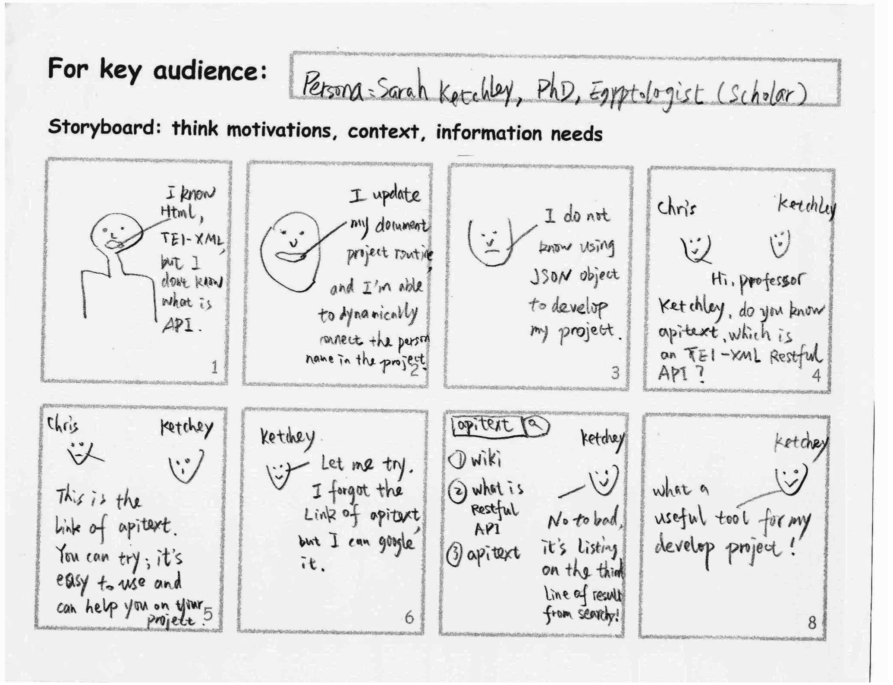
Storyboards, Mood Boards, Paper Prototype, User Testing, Findings, Design Mockup, API Requirements, Introduction Video

sketches / storyboards



For key audience:

Storyboard: think motivations, context, information needs

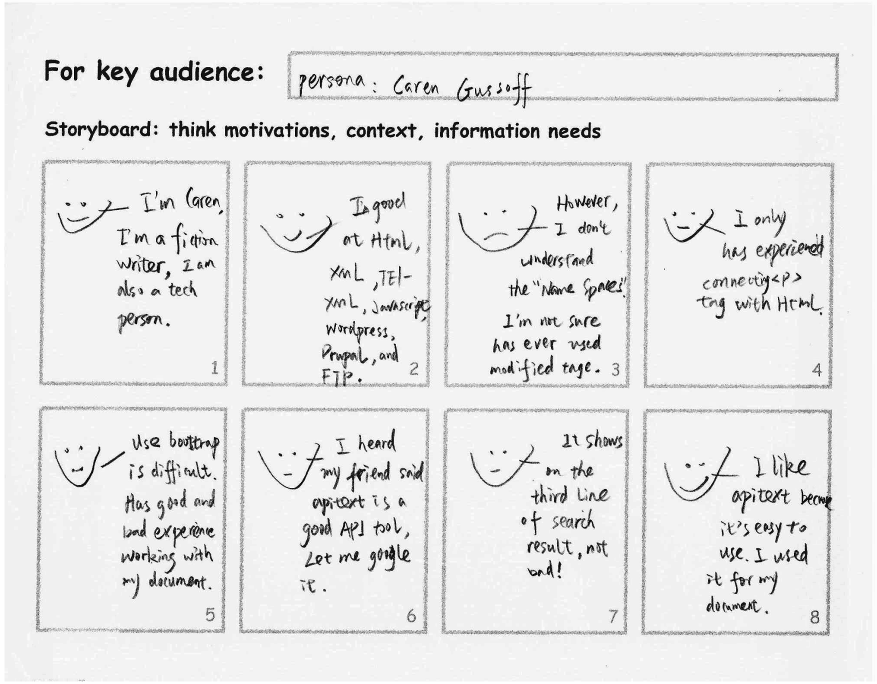


For key audience: Persona: Sarah Ketchley, PhD, Egyptologist (Scholar)

Storyboard: think motivations, context, information needs

I know Html, TEI-XML, but I dont know what is API.
1
I update my document project routine, and I'm able to dynamically connect the person name in the project.
2
I do not know using JSON object to develop my project.
3
Chris
Ketchley
Hi, professor Ketchley, do you know apitext, which is an TEI-XML Restful API?
4
Chris
Ketchey
This is the link of apitext. You can try; it's easy to use and can help you on your projects.
5
Ketchey
Let me try. I forgot the Link of apitext, but I can google it.
6
apitext
① wiki
② what is Restful API
③ apitext
Ketchey
No to bad, it's listing on the third line of result from search!
7
Ketchey
what a useful tool for my develop project!
8

sketches / storyboards



For key audience: persona: Caren Gussoff

Storyboard: think motivations, context, information needs

I'm Caren, I'm a fiction writer, I am also a tech person. 1

I's good at Html, XmL, TEI-XmL, Javascript, Wordpress, Drupal, and FTP. 2

However, I don't understand the "Name Spaces". I'm not sure has ever used modified tage. 3

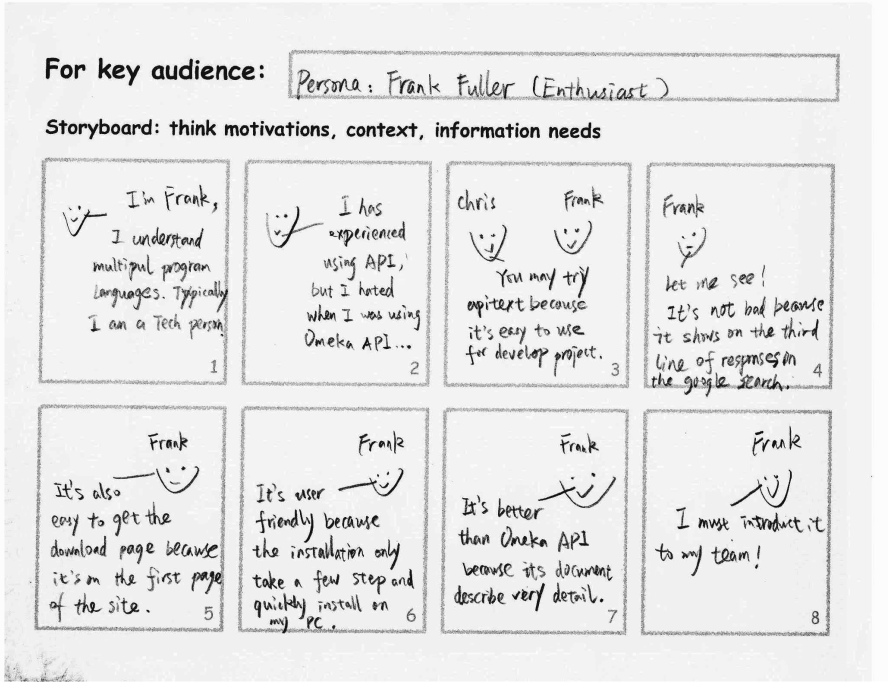
I only has experienced connecting <p> tag with Html. 4

Use bouttrap is difficult. Has good and bad experience working with my document. 5

I heard my friend said apitext is a good API tool, Let me google it. 6

It shows on the third line of search result, not bad! 7

I like apitext becaue it's easy to use. I used it for my document. 8



For key audience: Persona: Frank Fuller (Enthusiast)

Storyboard: think motivations, context, information needs

I'm Frank, I understand multipul program Languages. Typically I an a Tech person 1

I has experienced using API, but I hated when I was using Omeka API... 2

Chris Frank You may try apitext because it's easy to use for develop project. 3

Frank let me see! It's not bad because it shows on the third line of responses on the google search. 4

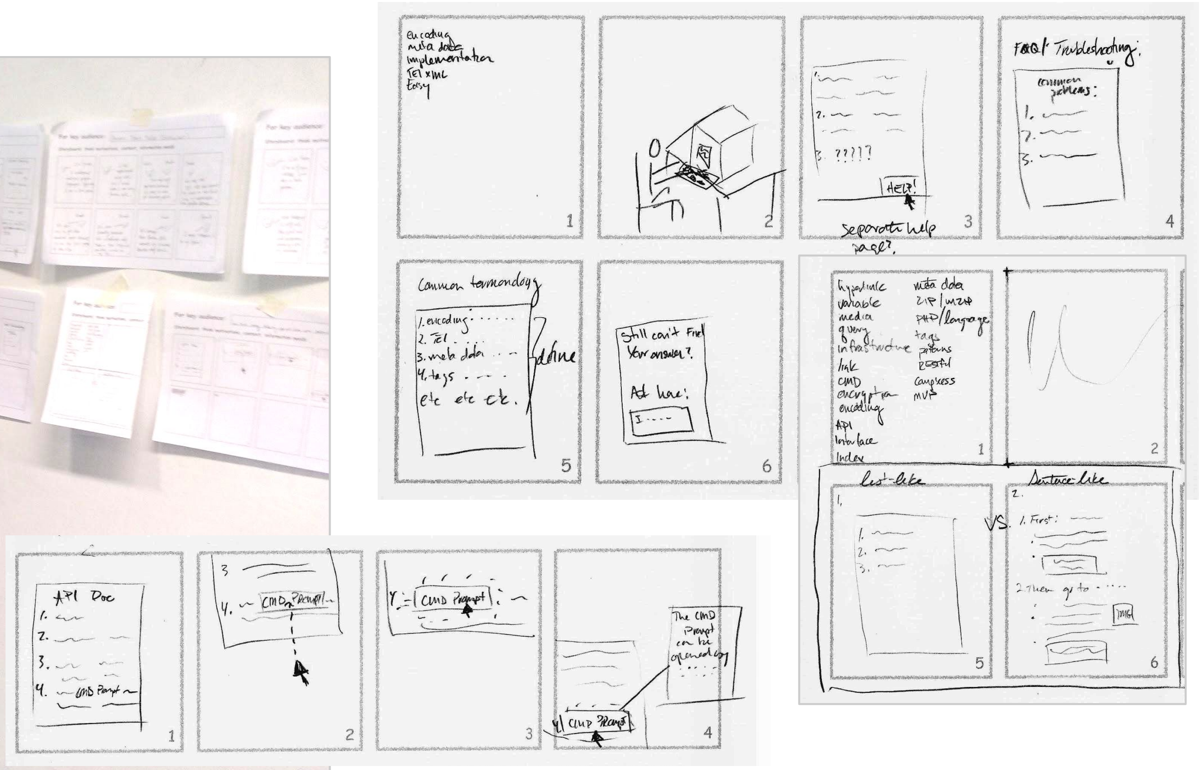
Frank It's also easy to get the download page because it's on the first page of the site. 5

Frank It's user friendly because the installation only take a few step and quickly install on my PC. 6

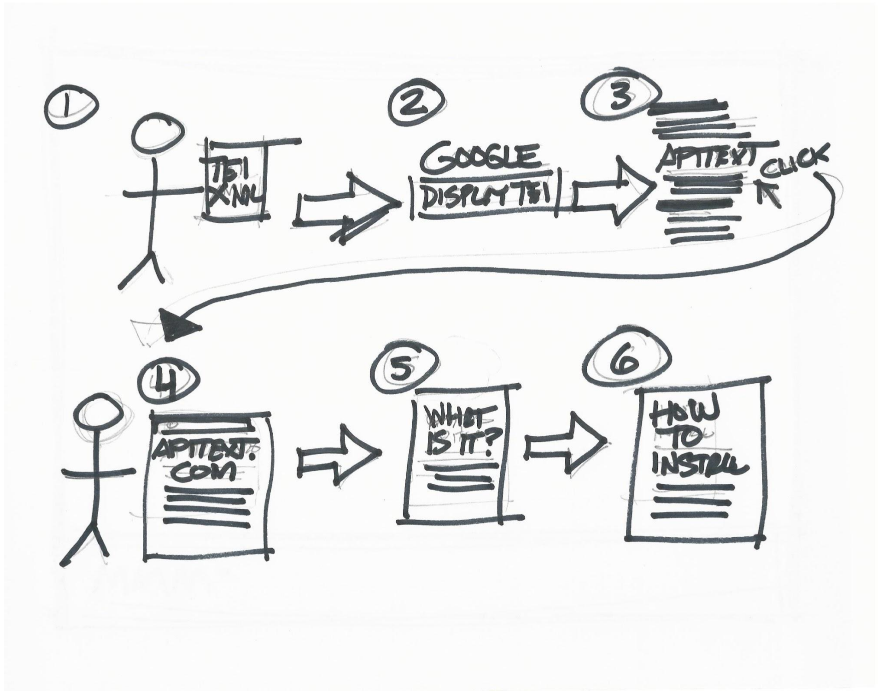
Frank It's better than Omeka API because its document describe very detail. 7

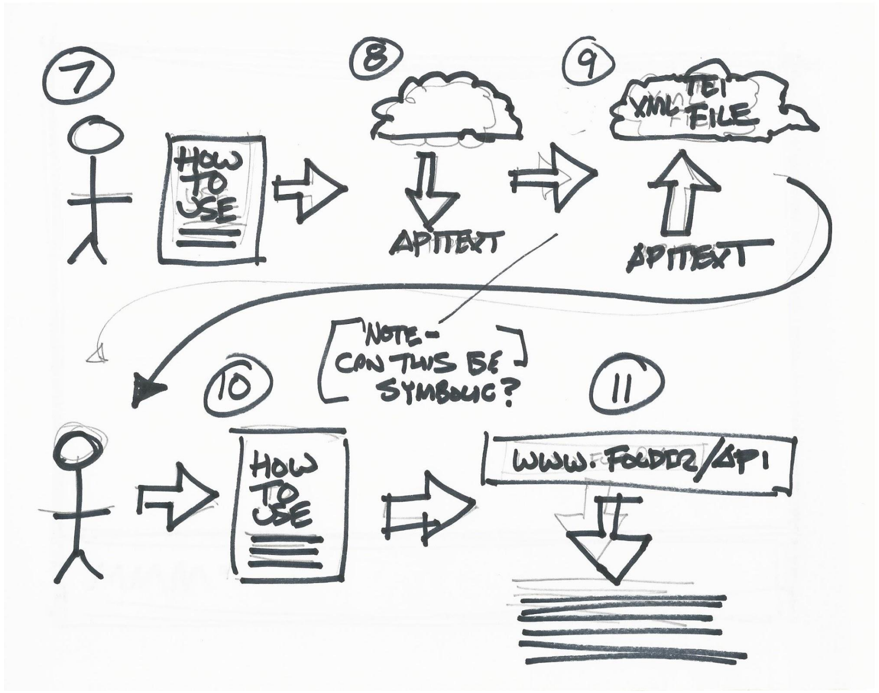
Frank I must introduct it to my team! 8

sketches / storyboards

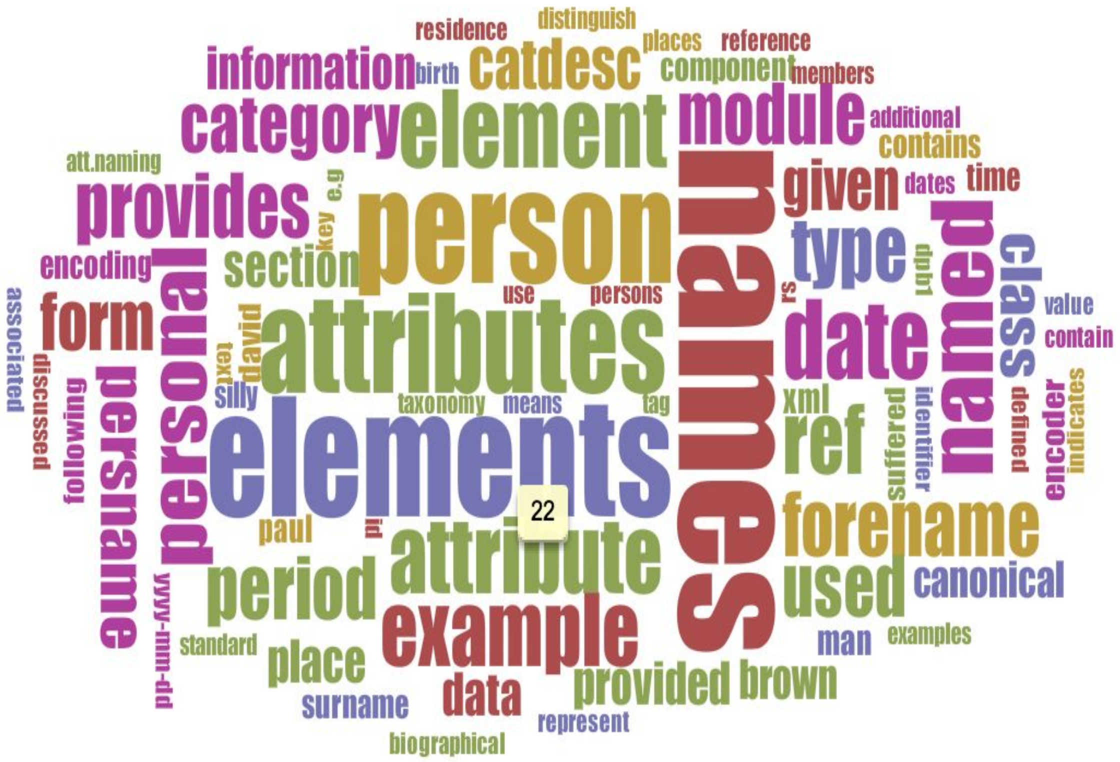


sketches / storyboards





mood board (for the api)



Since the interface for our API is unformatted text, we decided that a textual analysis of the content of the Text Encoding Initiative (TEI) P5 guidelines could serve as a kind of “textual” mood board that informs the syntax used for API calls and responses.

After analyzing different segments of the TEI documentation, we found the introductory parts to be the most user friendly, especially when considering our target audience.

mood board (for the documentation)

apitext

⬛ #000000

🟦 # 1B3B61

🟦 #2AB2EA

🟩 # 7EB335

🟨 # FDC809

🟧 # E1963D

⬜ # F3F3F3

HEADER

THIS IS A HEADER.

ARIAL BLACK

SUB HEADER

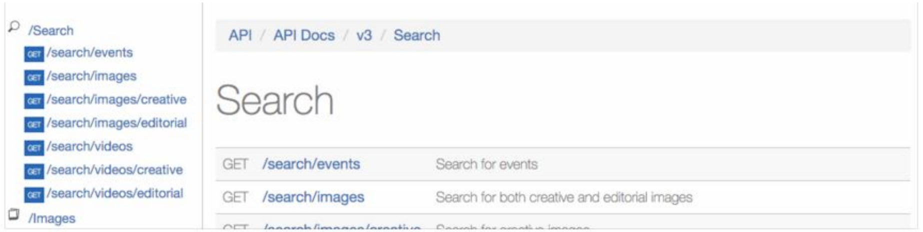
THIS IS A SUB HEADER.

ARIAL BLACK

ALTERNATE HEADER

THIS IS AN ALTERNATE HEADER.

ARIAL BLACK



/Search
GET /search/events
GET /search/images
GET /search/images/creative
GET /search/images/editorial
GET /search/videos
GET /search/videos/creative
GET /search/videos/editorial
/Images

API / API Docs / v3 / Search

Search

GET	/search/events	Search for events
GET	/search/images	Search for both creative and editorial images

Arial Black

Aa Bb Cc Dd Ee Ff Gg

Hh Ii Jj Kk Ll Mm Nn

Oo Pp Qq Rr Ss Tt Uu

Arial Bold

Aa Bb Cc Dd Ee Ff Gg Hh

Ii Jj Kk Ll Mm Nn Oo Pp

Qq Rr Ss Tt Uu Vv Ww Xx

Yy Zz

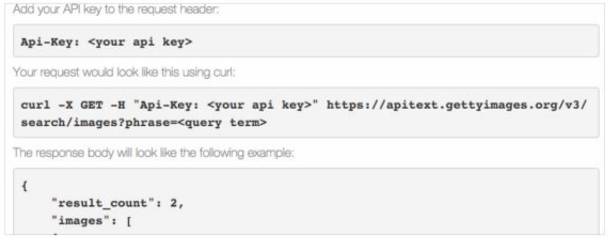
Times New Roman

Aa Bb Cc Dd Ee Ff Gg Hh Ii

Jj Kk Ll Mm Nn Oo Pp Qq Rr

Ss Tt Uu Vv Ww Xx Yy Zz

0123456789



Add your API key to the request header:

```
Api-Key: <your api key>
```

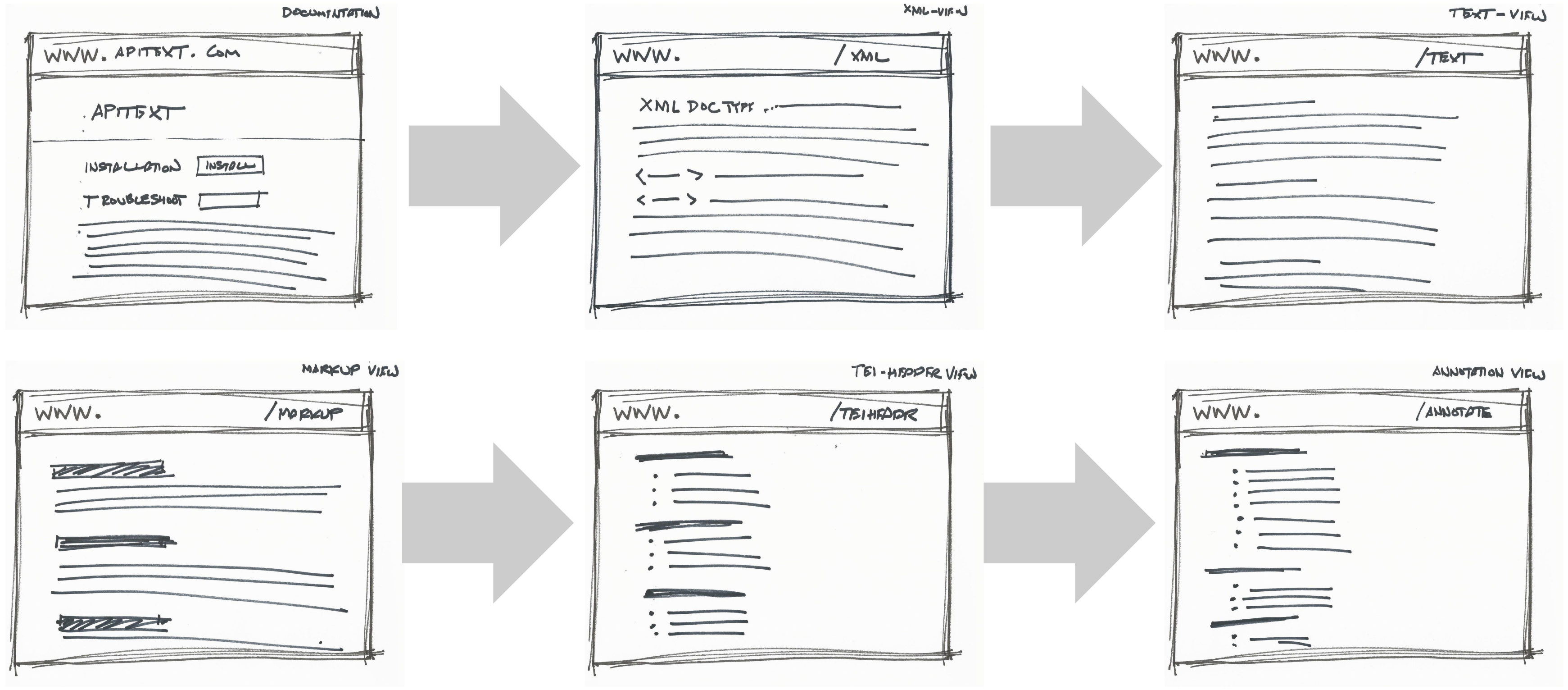
Your request would look like this using curl:

```
curl -X GET -H "Api-Key: <your api key>" https://apitext.gettyimages.org/v3/
search/images?phrase=<query term>
```

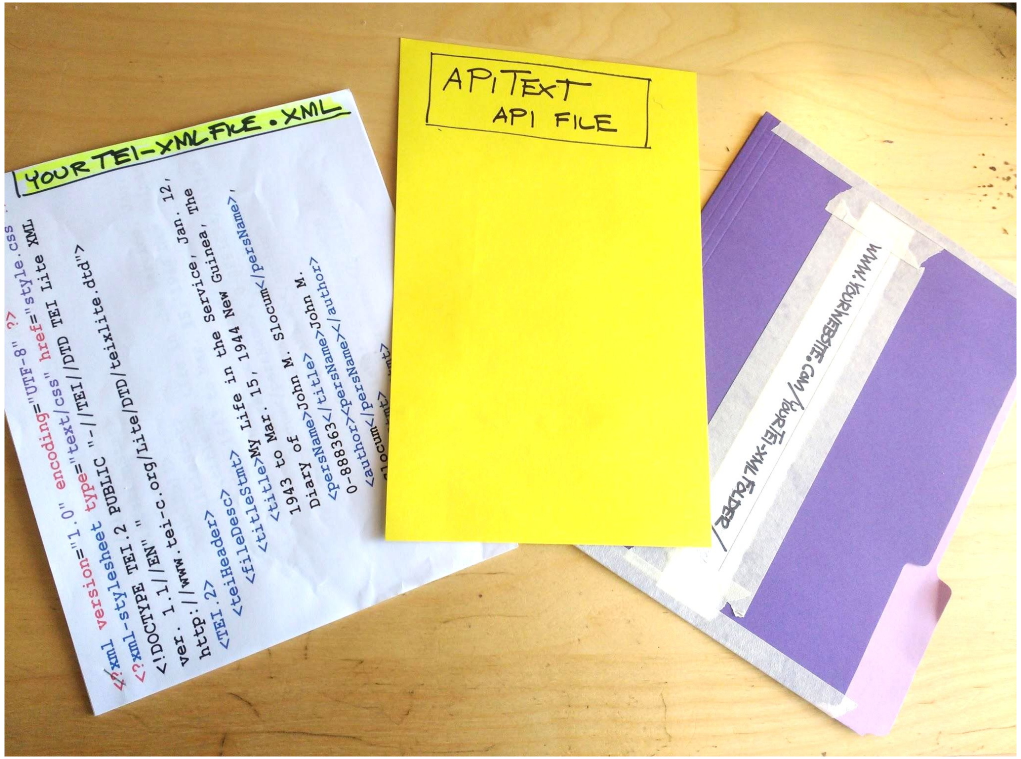
The response body will look like the following example:

```
{
    "result_count": 2,
    "images": [
```

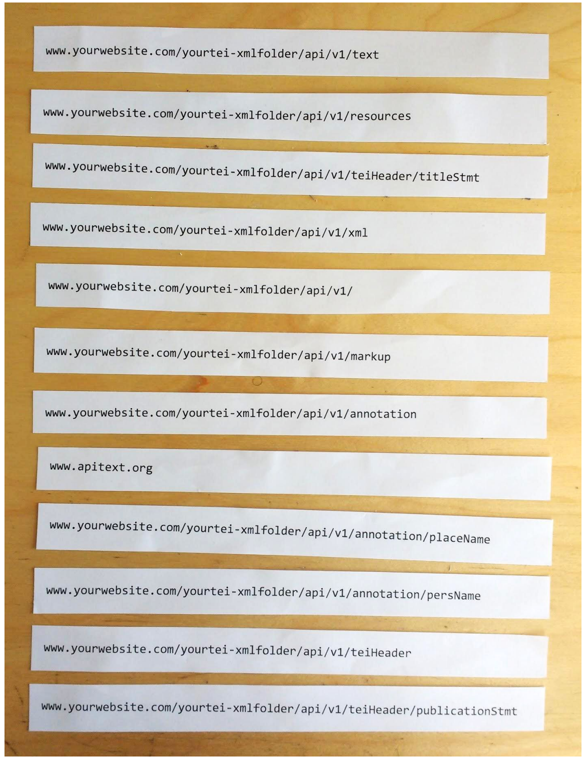
paper prototype (rough concept)



paper prototype



```
<?xml version="1.0" encoding="UTF-8" ?>
<?xml-stylesheet type="text/css" href="style.css
<!DOCTYPE TEI.2 PUBLIC "-//TEI//DTD TEI Lite XML
ver. 1.1//EN" "
http://www.tei-c.org/Lite/DTD/teixlite.dtd">
<TEI.2>
<teiHeader>
<fileDesc>
<titleStmt>
<title>My Life in the Service, Jan. 12,
1943 to Mar. 15, 1944 New Guinea, The
Diary of John M. Slocum</persName>,
<persName>John M.
0-888363</title>
<author><persName></author>
<author></persName>
</locum
```



www.yourwebsite.com/yourtei-xmlfolder/api/v1/text

www.yourwebsite.com/yourtei-xmlfolder/api/v1/resources

www.yourwebsite.com/yourtei-xmlfolder/api/v1/teiHeader/titleStmt

www.yourwebsite.com/yourtei-xmlfolder/api/v1/xml

www.yourwebsite.com/yourtei-xmlfolder/api/v1/

www.yourwebsite.com/yourtei-xmlfolder/api/v1/markup

www.yourwebsite.com/yourtei-xmlfolder/api/v1/annotation

www.apitext.org

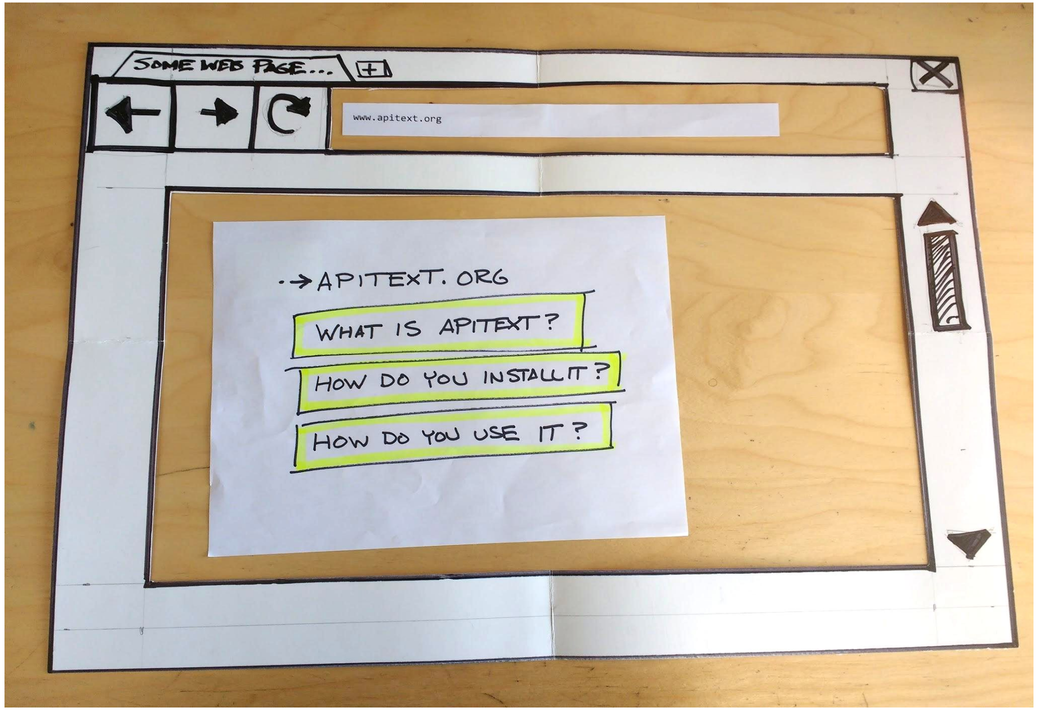
www.yourwebsite.com/yourtei-xmlfolder/api/v1/annotation/placeName

www.yourwebsite.com/yourtei-xmlfolder/api/v1/annotation/persName

www.yourwebsite.com/yourtei-xmlfolder/api/v1/teiHeader

www.yourwebsite.com/yourtei-xmlfolder/api/v1/teiHeader/publicationStmt

paper prototype

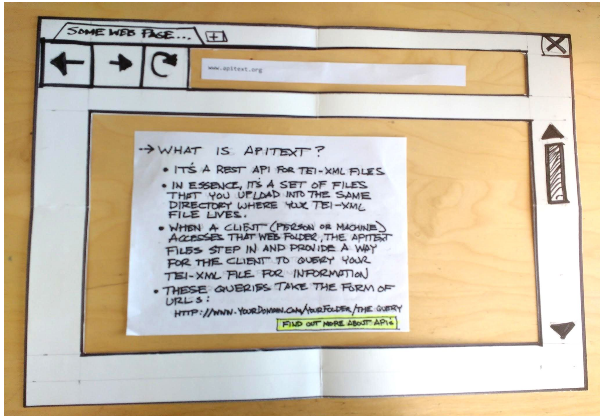


→ APITEXT. ORG

WHAT IS APITEXT?

HOW DO YOU INSTALL IT?

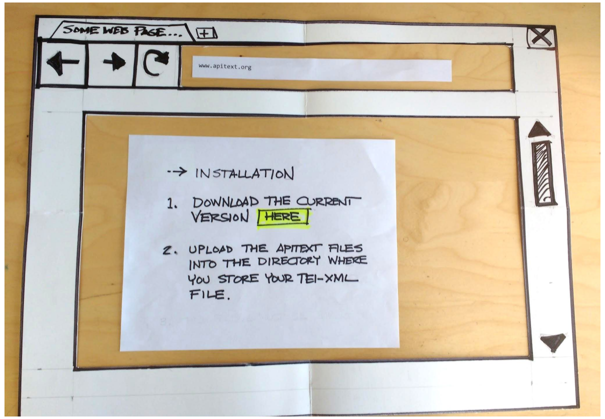
HOW DO YOU USE IT?



→ WHAT IS APITEXT?

- IT'S A REST API FOR TEI-XML FILES
- IN ESSENCE, IT'S A SET OF FILES THAT YOU UPLOAD INTO THE SAME DIRECTORY WHERE YOUR TEI-XML FILE LIVES.
- WHEN A CLIENT (PERSON OR MACHINE) ACCESSES THAT WEB FOLDER, THE APITEXT FILES STEP IN AND PROVIDE A WAY FOR THE CLIENT TO QUERY YOUR TEI-XML FILE FOR INFORMATION
- THESE QUERIES TAKE THE FORM OF URLS:
  HTTP://WWW.YOURDOMAIN.COM/YOURFOLDER/THE QUERY

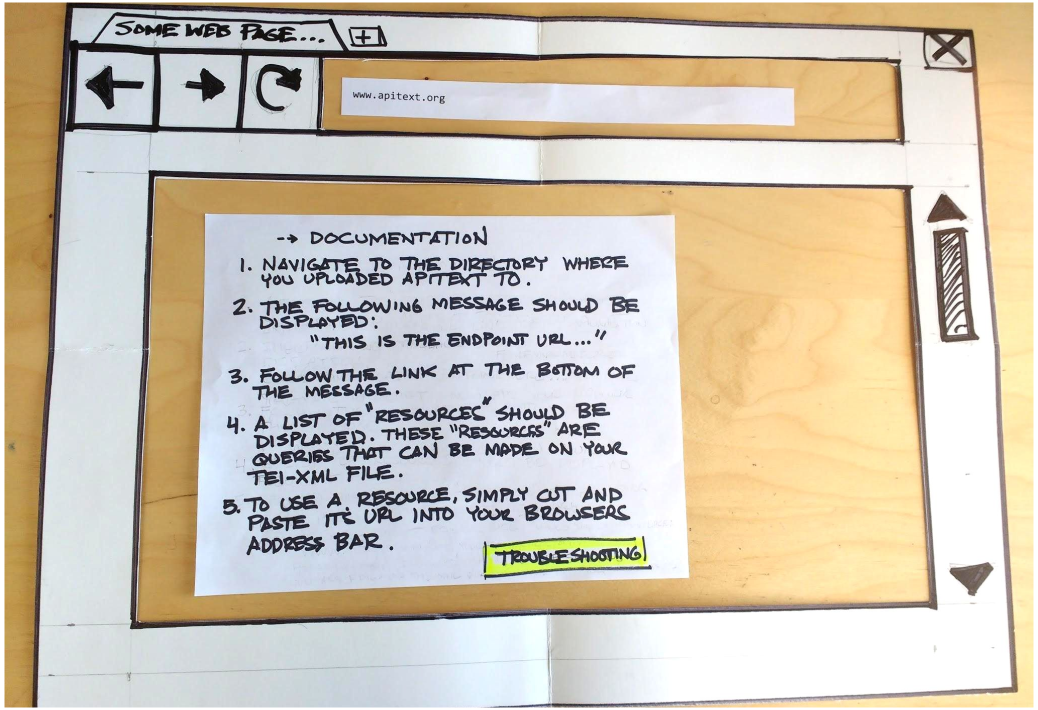
FIND OUT MORE ABOUT APIs



→ INSTALLATION

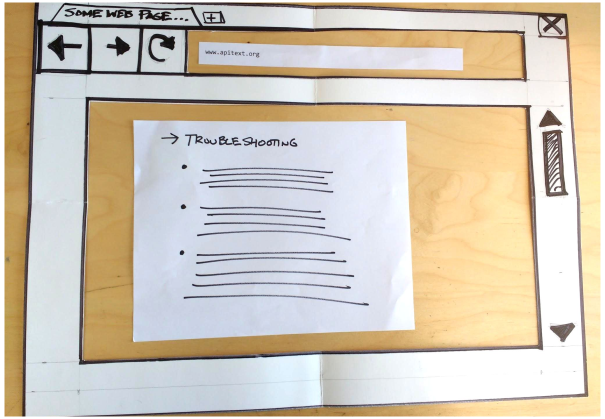
1. DOWNLOAD THE CURRENT VERSION HERE
2. UPLOAD THE APITEXT FILES INTO THE DIRECTORY WHERE YOU STORE YOUR TEI-XML FILE.

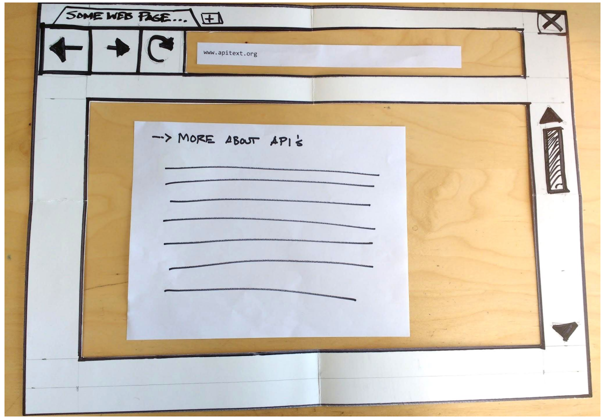
paper prototype



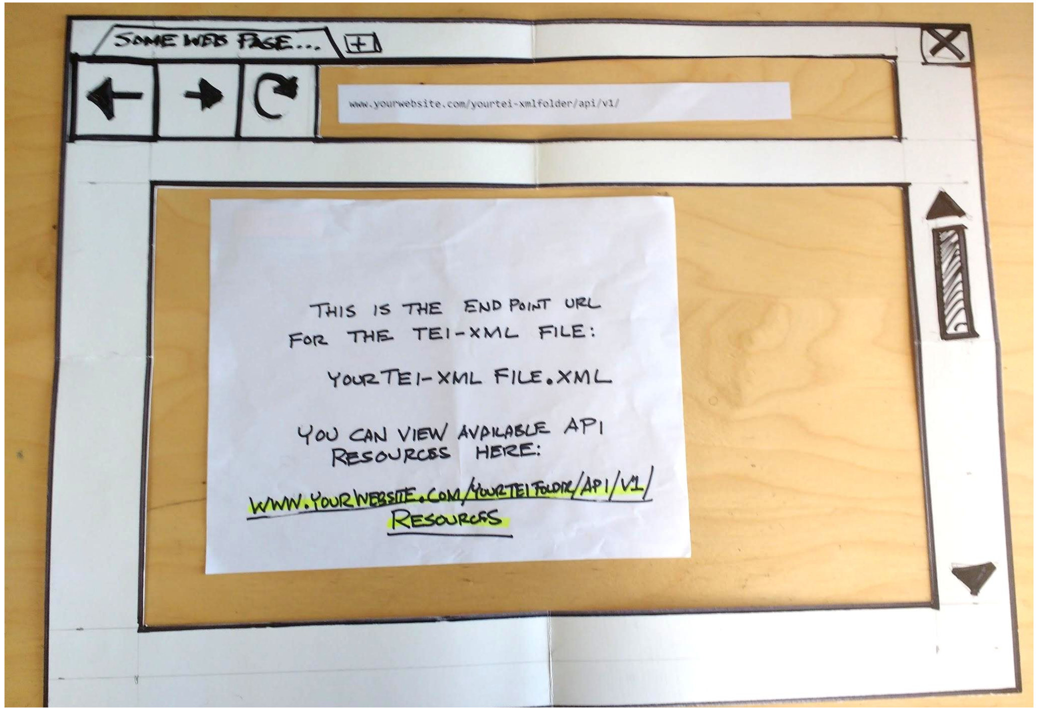
→ DOCUMENTATION

1. NAVIGATE TO THE DIRECTORY WHERE YOU UPLOADED APITEXT TO.
2. THE FOLLOWING MESSAGE SHOULD BE DISPLAYED:
   "THIS IS THE ENDPOINT URL..."
3. FOLLOW THE LINK AT THE BOTTOM OF THE MESSAGE.
4. A LIST OF "RESOURCES" SHOULD BE DISPLAYED. THESE "RESOURCES" ARE QUERIES THAT CAN BE MADE ON YOUR TEI-XML FILE.
5. TO USE A RESOURCE, SIMPLY CUT AND PASTE IT'S URL INTO YOUR BROWSERS ADDRESS BAR.





paper prototype

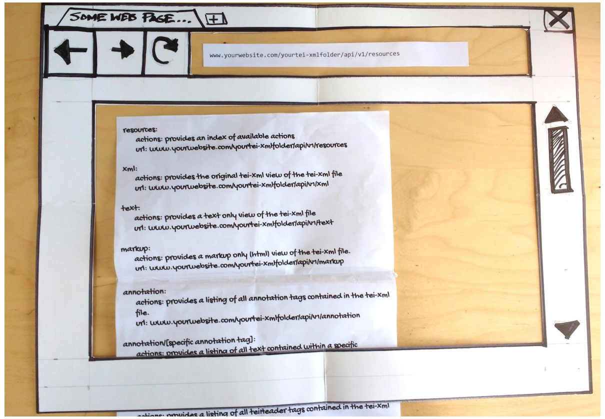


THIS IS THE END POINT URL
FOR THE TEI-XML FILE:

YOURTEI-XML FILE.XML

YOU CAN VIEW AVAILABLE API
RESOURCES HERE:

WWW.YOURWEBSITE.COM/YOURTEIFOLDER/API/V1/
RESOURCES



resources:
  actions: provides an index of available actions
  url: www.yourwebsite.com/yourtei-xmlfolder/api/v1/resources

xml:
  actions: provides the original tei-xml view of the tei-xml file
  url: www.yourwebsite.com/yourtei-xmlfolder/api/v1/xml

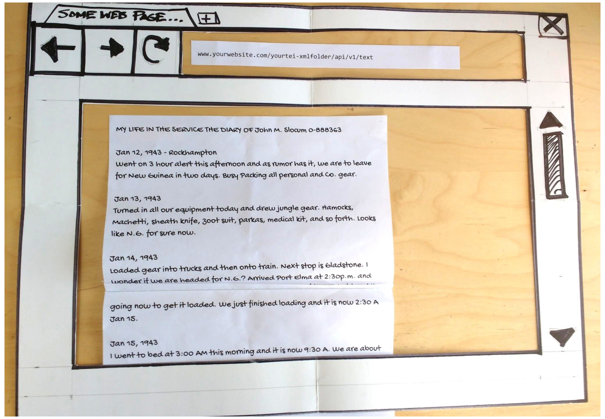
text:
  actions: provides a text only view of the tei-xml file
  url: www.yourwebsite.com/yourtei-xmlfolder/api/v1/text

markup:
  actions: provides a markup only (html) view of the tei-xml file.
  url: www.yourwebsite.com/yourtei-xmlfolder/api/v1/markup

annotation:
  actions: provides a listing of all annotation tags contained in the tei-xml file.
  url: www.yourwebsite.com/yourtei-xmlfolder/api/v1/annotation

annotation/[specific annotation tag]:
  actions: provides a listing of all text contained within a specific

actions: provides a listing of all teiHeader tags contained in the tei-xml



MY LIFE IN THE SERVICE THE DIARY OF John M. Slocum 0-888363

Jan 12, 1943 - Rockhampton
Went on 3 hour alert this afternoon and as rumor has it, we are to leave for New Guinea in two days. Busy Packing all personal and Co. gear.

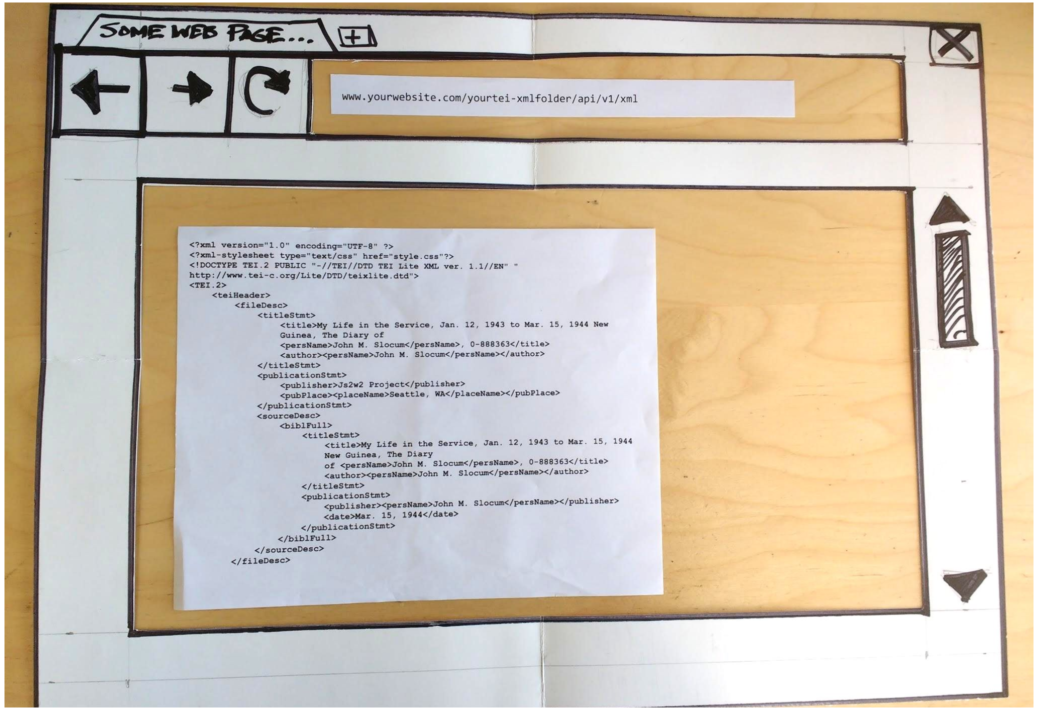
Jan 13, 1943
Turned in all our equipment today and drew jungle gear. Hamocks, Machetti, sheath knife, 3oot suit, parkas, medical kit, and so forth. Looks like N.G. for sure now.

Jan 14, 1943
Loaded gear into trucks and then onto train. Next stop is Gladstone. I wonder if we are headed for N.G.? Arrived Port Elma at 2:30p.m. and

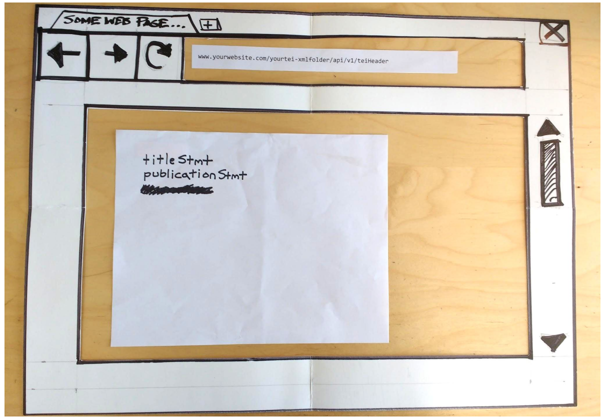
going now to get it loaded. We just finished loading and it is now 2:30 A Jan 15.

Jan 15, 1943
I went to bed at 3:00 AM this morning and it is now 9:30 A. We are about

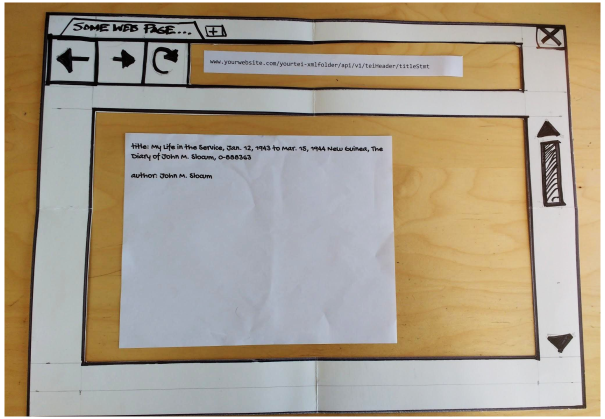
paper prototype



```
<?xml version="1.0" encoding="UTF-8" ?>
<?xml-stylesheet type="text/css" href="style.css"?>
<!DOCTYPE TEI.2 PUBLIC "-//TEI//DTD TEI Lite XML ver. 1.1//EN" "
http://www.tei-c.org/Lite/DTD/teixlite.dtd">
<TEI.2>
    <teiHeader>
        <fileDesc>
            <titleStmt>
                <title>My Life in the Service, Jan. 12, 1943 to Mar. 15, 1944 New
                Guinea, The Diary of
                <persName>John M. Slocum</persName>, 0-888363</title>
                <author><persName>John M. Slocum</persName></author>
            </titleStmt>
            <publicationStmt>
                <publisher>Js2w2 Project</publisher>
                <pubPlace><placeName>Seattle, WA</placeName></pubPlace>
            </publicationStmt>
            <sourceDesc>
                <biblFull>
                    <titleStmt>
                        <title>My Life in the Service, Jan. 12, 1943 to Mar. 15, 1944
                        New Guinea, The Diary
                        of <persName>John M. Slocum</persName>, 0-888363</title>
                        <author><persName>John M. Slocum</persName></author>
                    </titleStmt>
                    <publicationStmt>
                        <publisher><persName>John M. Slocum</persName></publisher>
                        <date>Mar. 15, 1944</date>
                    </publicationStmt>
                </biblFull>
            </sourceDesc>
        </fileDesc>
```



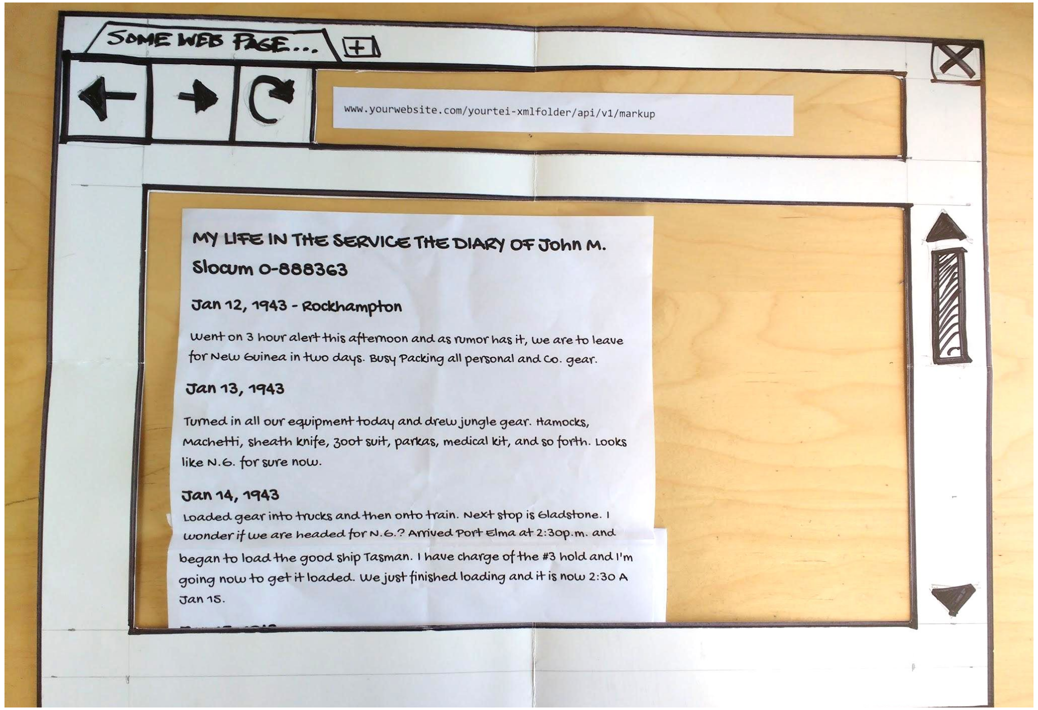
titleStmt
publicationStmt
~~[illegible]~~


SOME WEB PAGE...
www.yourwebsite.com/yourtei-xmlfolder/api/v1/teiHeader/titleStmt

title: My Life in the Service, Jan. 12, 1943 to Mar. 15, 1944 New Guinea, The Diary of John M. Slocum, 0-888363

author: John M. Slocum

paper prototype



www.yourwebsite.com/yourtei-xmlfolder/api/v1/markup

MY LIFE IN THE SERVICE THE DIARY OF John M. Slocum 0-888363

Jan 12, 1943 - Rockhampton

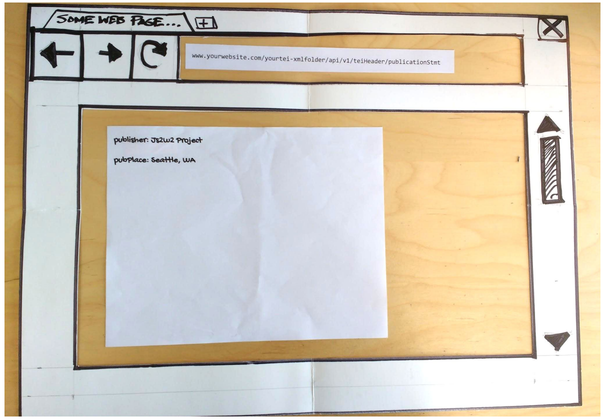
Went on 3 hour alert this afternoon and as rumor has it, we are to leave for New Guinea in two days. Busy Packing all personal and Co. gear.

Jan 13, 1943

Turned in all our equipment today and drew jungle gear. Hamocks, Machetti, sheath knife, 300t suit, parkas, medical kit, and so forth. Looks like N.G. for sure now.

Jan 14, 1943

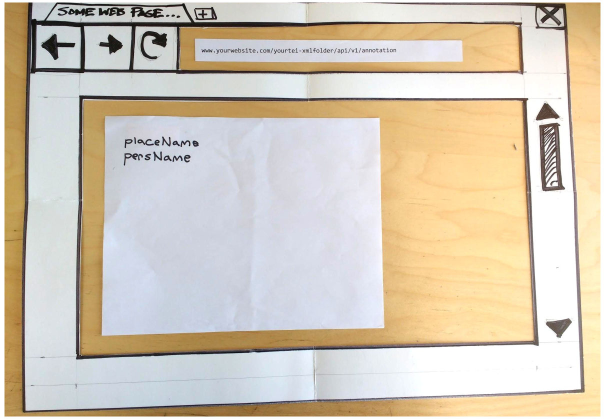
Loaded gear into trucks and then onto train. Next stop is Gladstone. I wonder if we are headed for N.G.? Arrived Port Elma at 2:30p.m. and began to load the good ship Tasman. I have charge of the #3 hold and I'm going now to get it loaded. We just finished loading and it is now 2:30 A Jan 15.



www.yourwebsite.com/yourtei-xmlfolder/api/v1/teiHeader/publicationStmt

publisher: JS2W2 Project

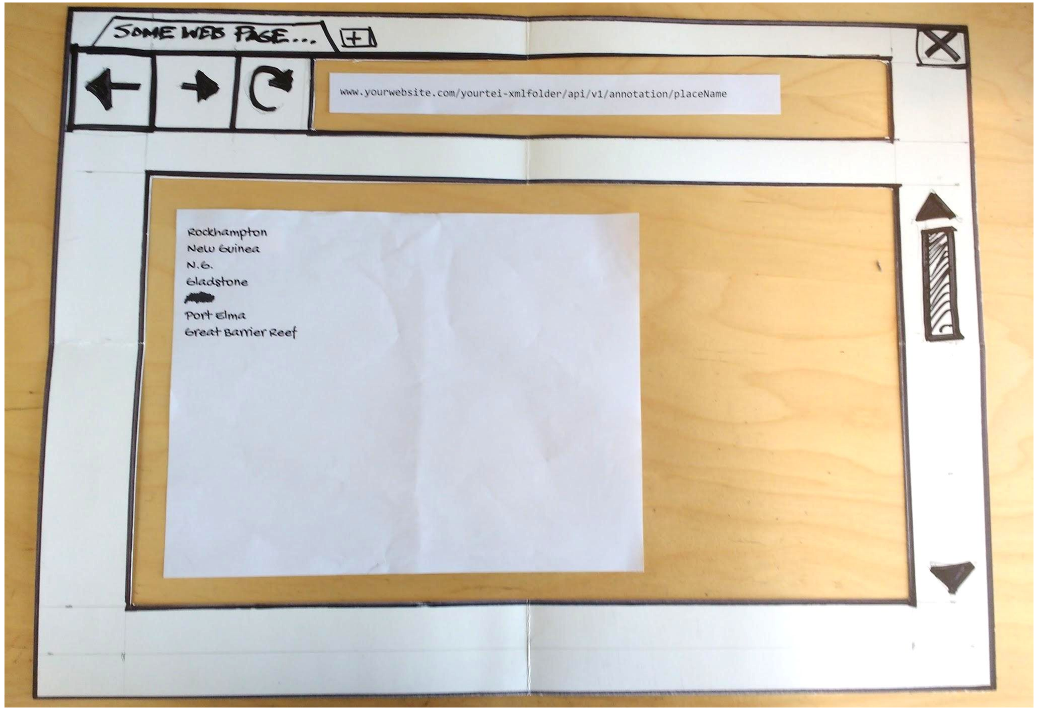
pubPlace: Seattle, WA



www.yourwebsite.com/yourtei-xmlfolder/api/v1/annotation

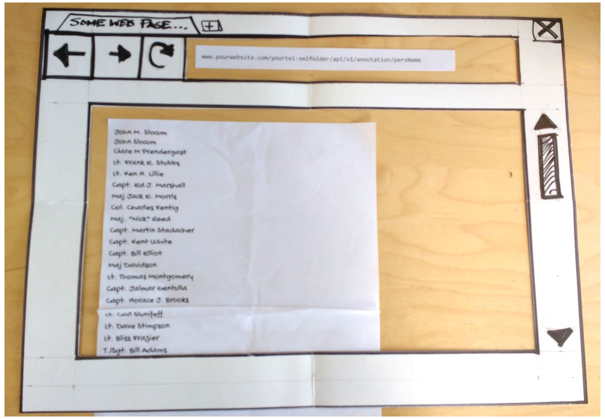
placeName
persName

paper prototype



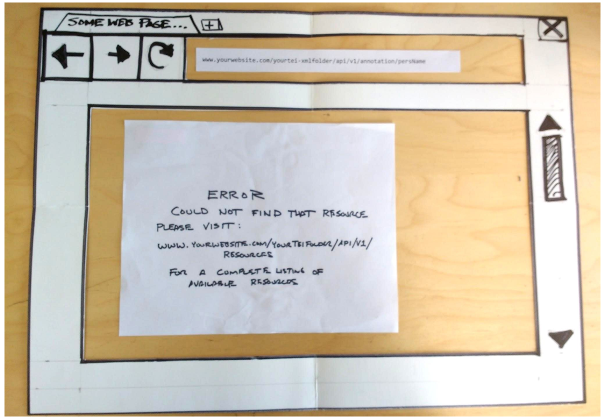
www.yourwebsite.com/yourtei-xmlfolder/api/v1/annotation/placeName

Rockhampton
New Guinea
N.G.
Gladstone
[redacted]
Port Elma
Great Barrier Reef



www.yourwebsite.com/yourtei-xmlfolder/api/v1/annotation/persName

John M. Slocum
John Slocum
Clare M Prendergast
Lt. Frank E. Stubbs
Lt. Ken H. Lillie
Capt. Ed J. Marshall
Maj Jack E. Morris
Col. Charles Fentig
Maj. "Nick" Reed
Capt. Martin Stadacher
Capt. Kent White
Capt. Bill Elliot
Maj Davidson
Lt. Thomas Montgomery
Capt. Jalmar Gentulla
Capt. Horace J. Brooks
Lt. Carl Slunifeff
Lt. Dave Stimpson
Lt. Bliss Frazier
T./Sgt. Bill Adams



www.yourwebsite.com/yourtei-xmlfolder/api/v1/annotation/persName

ERROR
COULD NOT FIND THAT RESOURCE
PLEASE VISIT:
WWW.YOURWEBSITE.COM/YOURTEIFOLDER/API/V1/RESOURCES
FOR A COMPLETE LISTING OF AVAILABLE RESOURCES

user testing - script

Introduction:

Hi, My name is [name]...

We may be taking pictures of you interacting with our prototype. Is that okay? [permission form]

[Confirm they are a researcher who is familiar with the Text Encoding Initiative (TEI)]

Our project, apitext is building a RESTful Application Programming Interface (API) for TEI-XML documents provides greater interoperability for these documents easy to use by an audience with backgrounds in the humanities, social sciences, and linguistics.

We are at the preliminary stages of our design process. We have developed what is called a low fidelity or paper prototype. It's basically a rough representation of how we think our prototype will be used.

You are going to help us by testing this prototype. You will be given two or three tasks to complete. While you are working on those tasks, it would great if you could talk aloud your actions and your thought process as you work through these tasks.

Introduce the prototype web interface…does this browser interface seem familiar?

For instance If you wanted to type a url to a website, where would you type it?

user testing - script

Demonstrate to user:

clicking, links, typing, cutting and pasting, back button, etc…

Remember to talk aloud your actions

[Explain the scenario] You're a TEI researcher - You have your TEI-XML file that you have created - You have a folder where your file lives.

You have heard about apitext and want to try it out with your own TEI-XML file.

You are at the Apitext website.

Tasks to complete:

Using the apitext website, install the apitext API

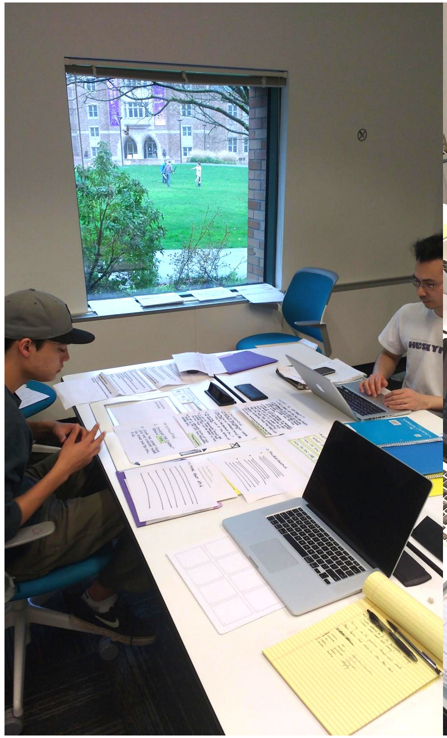
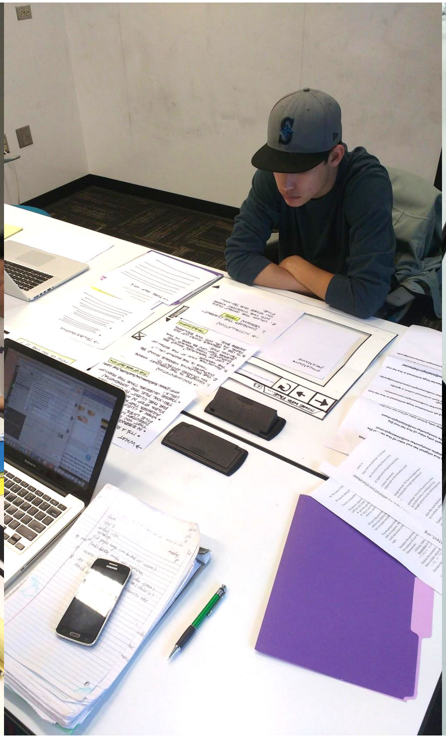
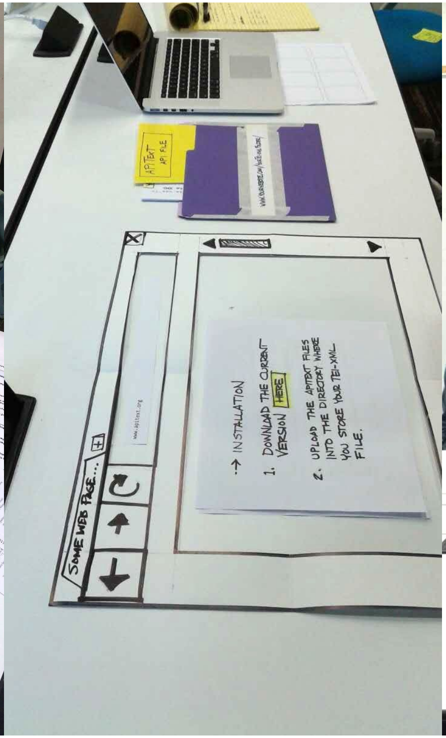
Once installed, spend some time experimenting with the interface

Who is the the publisher of this TEI-XML file?

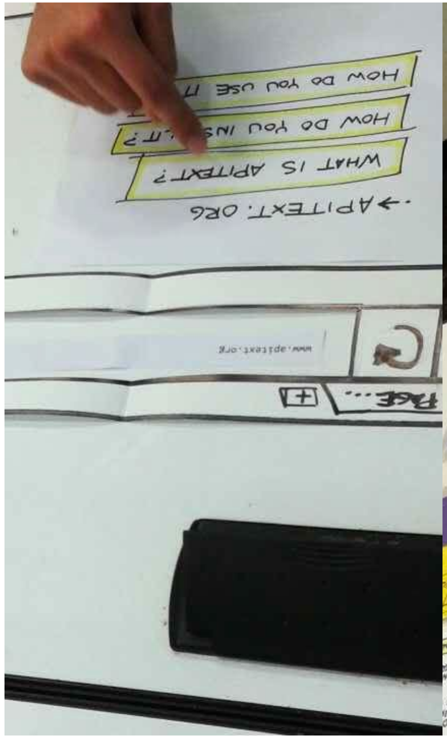
Is the Great Barrier Reef mentioned in your TEI-XML file?

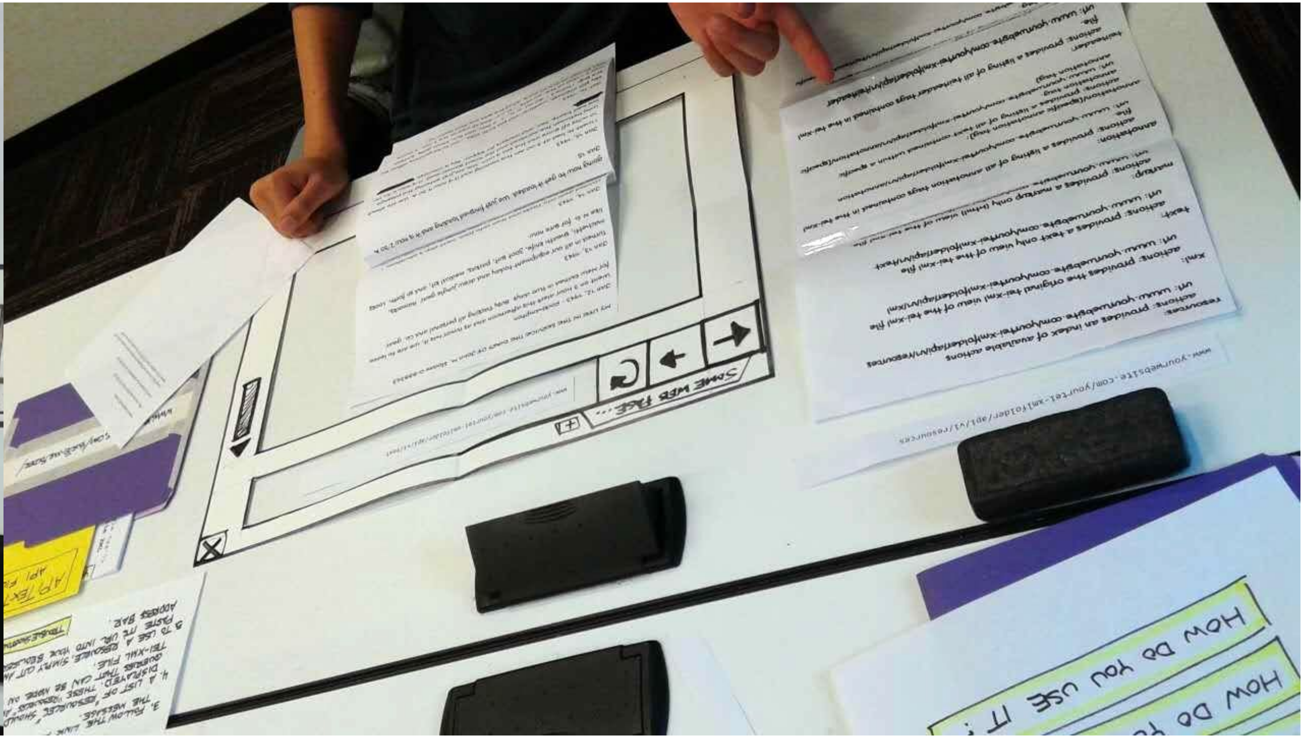
I would like to run your text through my text analysis tool, do you have a text only version of the contents of your TEI-XML file. If so, could you give me a link?

user testing - sessions

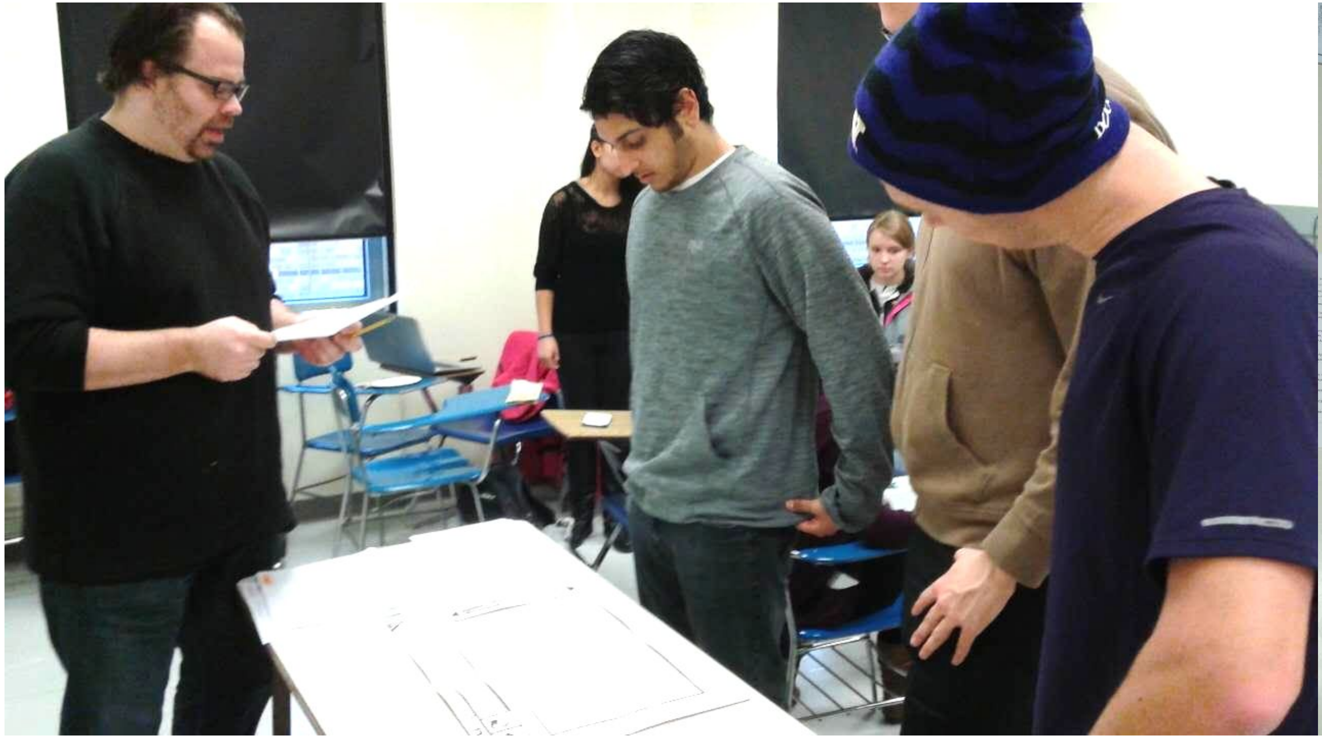



APITEXT
API FILE
SOME WEB PAGE...
→ INSTALLATION
1. DOWNLOAD THE CURRENT VERSION HERE
2. UPLOAD THE APITEXT FILES INTO THE DIRECTORY WHERE YOU STORE YOUR TEI-XML FILE.

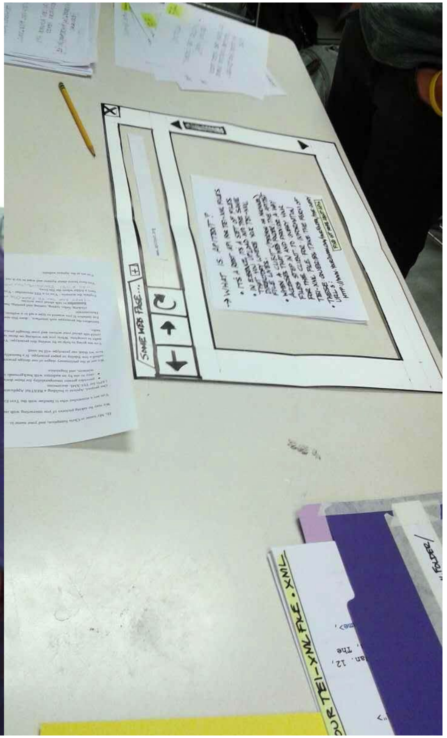
user testing - sessions





user testing - sessions





summary of findings

Our design not only includes a RESTful API for TEI-XML documents, it includes the documentation for how to use it. In our design process we made the conscious decision to treat both the documentation and the API as one prototype. Much of our testing to date has focused on how to get the user from successfully reading about the API to actually using it. Our most recent round of user testing showed that our design fundamentally supports that goal, however we uncovered the following list of improvements that should be included in the next iteration of the design.

It’s important to give the users the ability to use the interface early on in the API documentation process.

On multiple occasions users stated that after using the paper prototype they had a better understanding of what the purpose of the API was, this was even after reading the documentation. This feedback can be used in two ways:

1. Our API documentation should contain an interactive demonstration version of the interface.
2. This demo should be presented early on in the documentation, maybe even the first content the user processes after the apitext logo. This should aid in better user comprehension of our content.

summary of findings

Syntax used in the API and its accompanying documentation cannot be targeted exclusively towards users experienced with TEI.

We geared the paper prototype towards users experienced with TEI. When we tested users who were very technically savvy, but unfamiliar with TEI, they had difficulties completing the provided tasks. This raises the question, what if a TEI researcher brings on a programmer who is unfamiliar with TEI terminology? We think the answer to that question is, element syntax can still be TEI heavy, but element descriptions should be useable by a broader audience.

Multiple users did not want to follow a step by step process when using the paper prototype documentation.

On two occasions, users interacting with the prototype documentation did not follow the order of steps provided. These users wanted to explore how to use the API before installing it. This observation supports the previous idea of providing users a demonstration version of the API as part of our documentation.

summary of findings

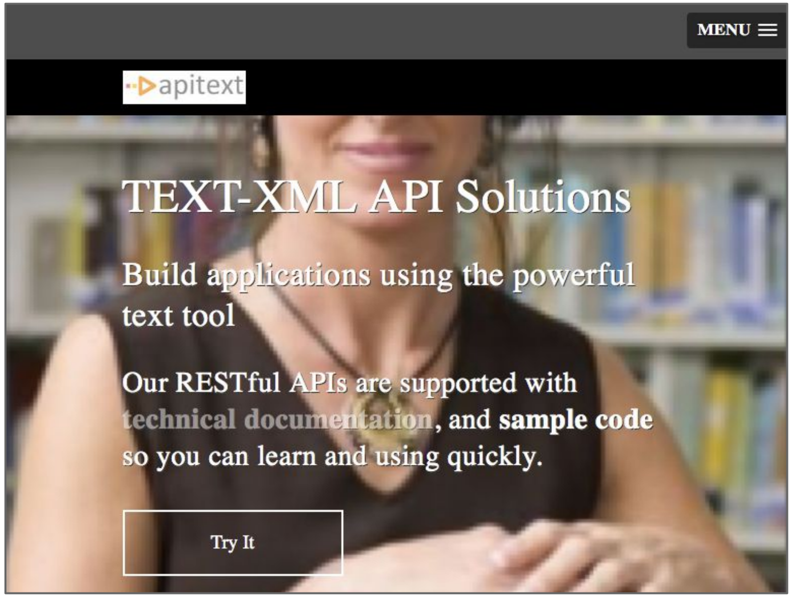
Cutting and pasting support for our documentation.

The initial version of our paper prototype had users writing in Uniform Resource Identifier (URI) endpoints onto a small whiteboard surface in order to simulate typing those requests into the search bar of a browser. When tested with actual users, this was a disaster. Version two of the paper prototype supported symbolic cutting and pasting of the necessary URI's. This was obviously a failure of the initial design of the paper prototype itself, but it does illustrate a potential barrier to the use of our documentation. We need to support and include examples of URI requests that can be cut and pasted into the user's browser.

The transition from the API documentation to actually using the API is important.

On each user test, there was confusion when the user crossed the threshold from reading the documentation to actually using the API. As previously observed, having a demonstration of the API within the documentation may assist with that transition, but what happens when the user actually installs our API, will that confusion still exist? In future iterations of this design we will need to address this transition issue.

design mockup (for the documentation)


MENU

apitext

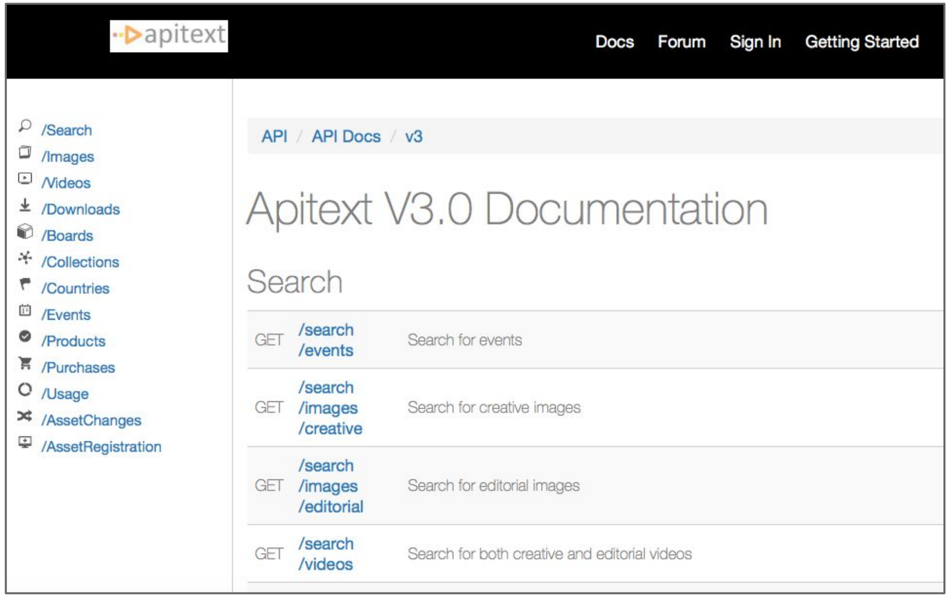
TEXT-XML API Solutions

Build applications using the powerful text tool

Our RESTful APIs are supported with **technical documentation**, and **sample code** so you can learn and using quickly.

Try It

Home Page


apitext

Docs Forum Sign In Getting Started

- /Search
- /Images
- /Videos
- /Downloads
- /Boards
- /Collections
- /Countries
- /Events
- /Products
- /Purchases
- /Usage
- /AssetChanges
- /AssetRegistration

API / API Docs / v3

Apitext V3.0 Documentation

Search

GET	/search /events	Search for events
GET	/search /images /creative	Search for creative images
GET	/search /images /editorial	Search for editorial images
GET	/search /videos	Search for both creative and editorial videos

Representative Page

api design requirements

Scope of the Product

The API will deal with properly formatted TEI-XML documents. Apitext will not be a parser for TEI-XML documents that do not follow typical XML guidelines. By utilizing our API's GET endpoints, the user will have access to four general views: text, markup, XML, annotation, and teiHeader.

User Characteristics

The API is meant to be used by academics, historians, humanists, etc. We want the API and its documentation to be user friendly, and understandable by people with a minimal exposure to this type of technology. It needs to be easy to understand, implement, and use.

General Constraints

The API will not validate TEI files. Our product is also not a TEI-XML document viewer. Instead, we are providing the tools necessary for software platforms, such as viewers to view helpful elements of TEI-XML documents.

api design requirements

System Requirements

The API will be written in PHP because because our audience typically has previous experience working within a LAMP stack environment. We are assuming that users who will be using our product will have some knowledge of TEI-XML files, will has access to a web server, and have the ability to upload files to that server. Our audience may also have prior experience working with PHP content management systems. Additionally, we must assume and depend on the fact that the TEI-XML files that are being used with our product are in fact valid TEI-XML files.

API Functionality

- The API will expose multiple URI endpoints that will only support HTTP GET requests.
- There will be four general categories of HTTP responses: text, markup, XML, annotation, and teiHeader.
- Annotation and teiHeader responses will support the ability to return tag specific information (see the following endpoint listing for more details).

The following is a detailed listing of the API's endpoints and their expected responses.

api design requirements

`www.[website]/[tei-xmlfolder]`

- **Endpoint Name:** Base Folder
- **Response**: Redirects the client to the API (...`/api/v1/`)
- **Format:** Not Applicable

`www.[website]/[tei-xmlfolder]/api/v1/`

- **Endpoint Name:** Welcome
- **Response**: Returns a welcome message and a link to the Resources endpoint.
- **Format:** HTML, UTF-8

`www.[website]/[tei-xmlfolder]/api/v1/resources`

- **Endpoint Name:** Resources
- **Response**: Returns a listing of available URI endpoints and their response descriptions.
- **Format:** JSON, UTF-8

`www.[website]/[tei-xmlfolder]/api/v1/xml`

- **Endpoint Name:** XML
- **Response**: Returns the TEI-XML view of the TEI-XML file.
- **Format:** XML, UTF-8

api design requirements

`www.[website]/[tei-xmlfolder]/api/v1/text`

- **Endpoint Name:** Text
- **Response**: Returns a text only view of the tei-xml file
- **Format:** Text, UTF-8

`www.[website]/[tei-xmlfolder]/api/v1/markup`

- **Endpoint Name:** Markup
- **Response**: Returns a markup only (html) view of the tei-xml file.
- **Format:** HTML, UTF-8

`www.[website]/[tei-xmlfolder]/api/v1/annotation`

- **Endpoint Name:** Annotation
- **Response**: Returns a listing of all annotation tags contained in the tei-xml.
- **Format:** JSON, UTF-8

`www.[website]/[tei-xmlfolder]/api/v1/annotation/[specific tag]`

- **Endpoint Name:** Annotation/[tag]
- **Response**: Returns a listing of all textual elements that are “tagged” with the specific TEI-XML tag.
- **Format:** JSON, UTF-8

api design requirements

`www.[website]/[tei-xmlfolder]/api/v1/teiHeader`

- **Endpoint Name:** teiHeader
- **Response**: Returns a listing of all the first level teiHeader tags contained in the tei-xml file.
- **Format:** JSON, UTF-8

`www.[website]/[tei-xmlfolder]/api/v1/teiHeader/[specific tag]`

- **Endpoint Name:** teiHeader
- **Response**: Returns a listing of the content that is “tagged” with the specific TEI-XML teiheader tag.
- **Format:** JSON, UTF-8

development

Backlog, Usability Testing, Screenshots

backlog

The apitext API will process all URI endpoints for a researchers TEI-XML file folder.

- As a TEI-XML Researcher I can use a web browser to navigate to the directory where I store my TEI-XML files so that I can get confirmation that the apitext API is handling any HTTP requests for resources located in that directory.
- As a TEI-XML Researcher I can use a web browser to navigate to the directory where I store my TEI-XML files and add a unsupported endpoint to the URI endpoint so that I can view a standard error message which includes a link to the resource's URI endpoint.
- As a TEI-XML Researcher I can use a web browser to navigate to the directory where I store my TEI-XML files and add “/api/v1/” to the end of the URI so that I can view a welcome message for the API.
- As a TEI-XML Researcher I can use a web browser to navigate to the directory where I store my TEI-XML files and add “/api/v1/resources” to the end of the URI so that I can view a listing of available URI endpoints and their respective human readable resource descriptions in JSON format.

backlog

The apitext API will process all URI endpoints for a researchers TEI-XML file folder (continued).

- As a TEI-XML Researcher I can use a web browser to navigate to the directory where I store my TEI-XML files and add “/api/v1/xml” to the end of the URI so that I can view the XML of the TEI-XML file located within that directory.
- As a TEI-XML Researcher I can use a web browser to navigate to the directory where I store my TEI-XML files and add “/api/v1/text" so that I can view a text representation of the contents of the TEI-XML file located within that directory.
- As a TEI-XML Researcher I can use a web browser to navigate to the directory where I store my TEI-XML files and add “/api/v1/markup" so that I can view a HTML representation of the contents of the TEI-XML file located within that directory.
- As a TEI-XML Researcher I can use a web browser to navigate to the directory where I store my TEI-XML files and add “/api/v1/annotation" so that I can view a JSON formatted listing of all "annotation" tags contained within the TEI-XML file located within that directory.

backlog

The apitext API will process all URI endpoints for a researchers TEI-XML file folder (continued).

- As a TEI-XML Researcher I can use a web browser to navigate to the directory where I store my TEI-XML files and add “/api/v1/teiHeader" so that I can view a JSON formatted listing of all "teiHeader" tags contained within the TEI-XML file located within that directory.
- As a TEI-XML Researcher I can use a web browser to navigate to the directory where I store my TEI-XML files and add “/api/v1/teiHeader/[specific tag]" so that I can view a JSON formatted listing of all textual elements that are “tagged” with a specific "teiHeader" TEI-XML tag contained within the TEI-XML file located within that directory.
- As a TEI-XML Researcher I can use a web browser to navigate to the directory where I store my TEI-XML files and add “/api/v1/annotation/[specific tag]" so that I can view a JSON formatted listing of all textual elements that are “tagged” with a specific "annotation" TEI-XML tag contained within the TEI-XML file located within that directory.

backlog

The apitext API documentation can be easily used by users with varying levels of technical proficiency.

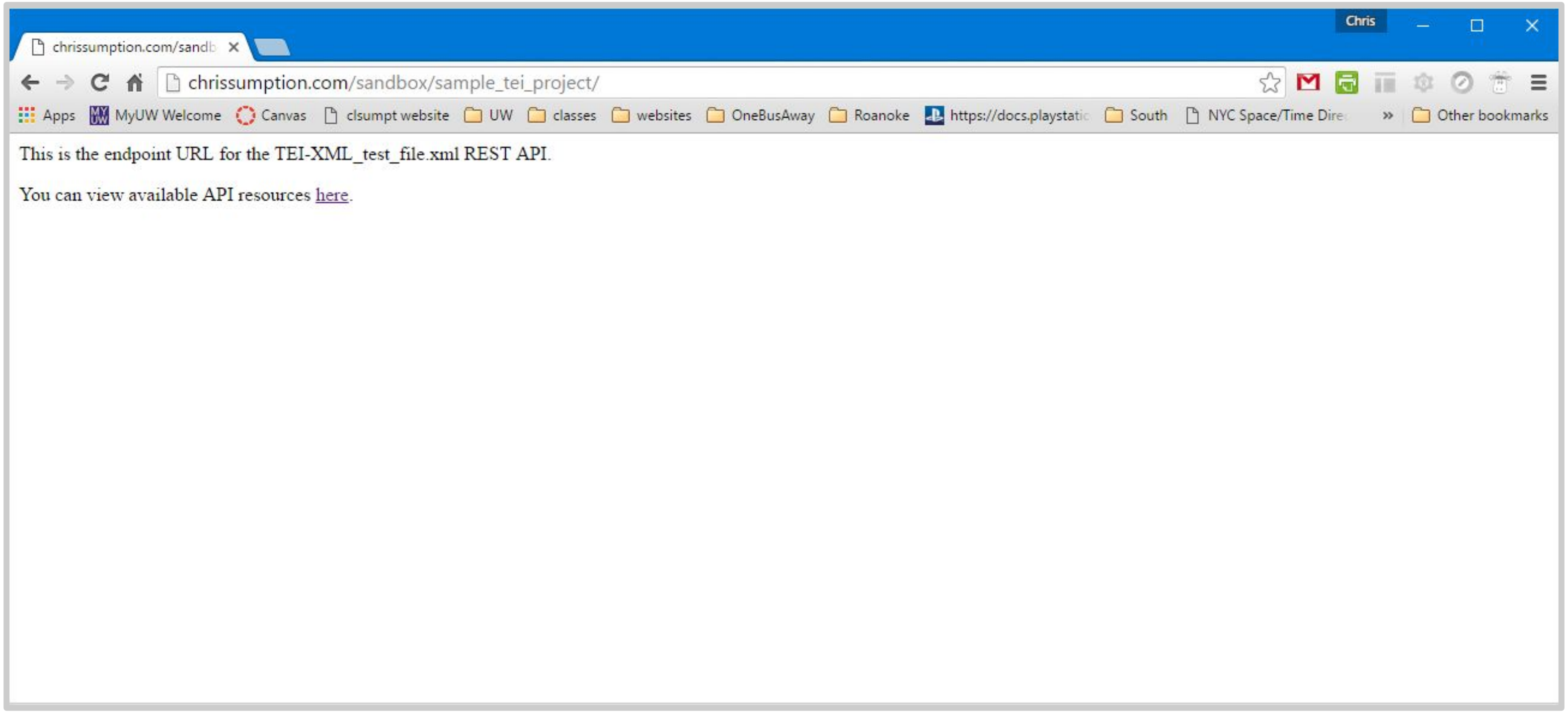
- As a TEI-XML Researcher I can use the apitext documentation so that I can learn more about what the apitext API does.
- As a TEI-XML Researcher I can use the apitext documentation so that I can learn more about what a RESTful API is.
- As a TEI-XML Researcher I can use the apitext documentation so that I can find out how to install the apitext api.
- As a TEI-XML Researcher I can use the apitext documentation so that I can find out how to view a listing of available URI endpoints and their respective human readable resource descriptions.
- As a TEI-XML Researcher I can use the apitext documentation so that I can find out how to view the XML of the TEI-XML file located within my TEI-XML directory.

backlog

The apitext API documentation can be easily used by users with varying levels of technical proficiency (continued).

- As a TEI-XML Researcher I can use the apitext documentation so that I can find out how to view the text representation of the TEI-XML file located within my TEI-XML directory.
- As a TEI-XML Researcher I can use the apitext documentation so that I can find out how to view the HTML representation of the TEI-XML file located within my TEI-XML directory.
- As a TEI-XML Researcher I can use the apitext documentation so that I can find out how to view all annotation tags within the TEI-XML file located within my TEI-XML directory.
- As a TEI-XML Researcher I can use the apitext documentation so that I can find out how to view all teiHeader tags within the TEI-XML file located within my TEI-XML directory.

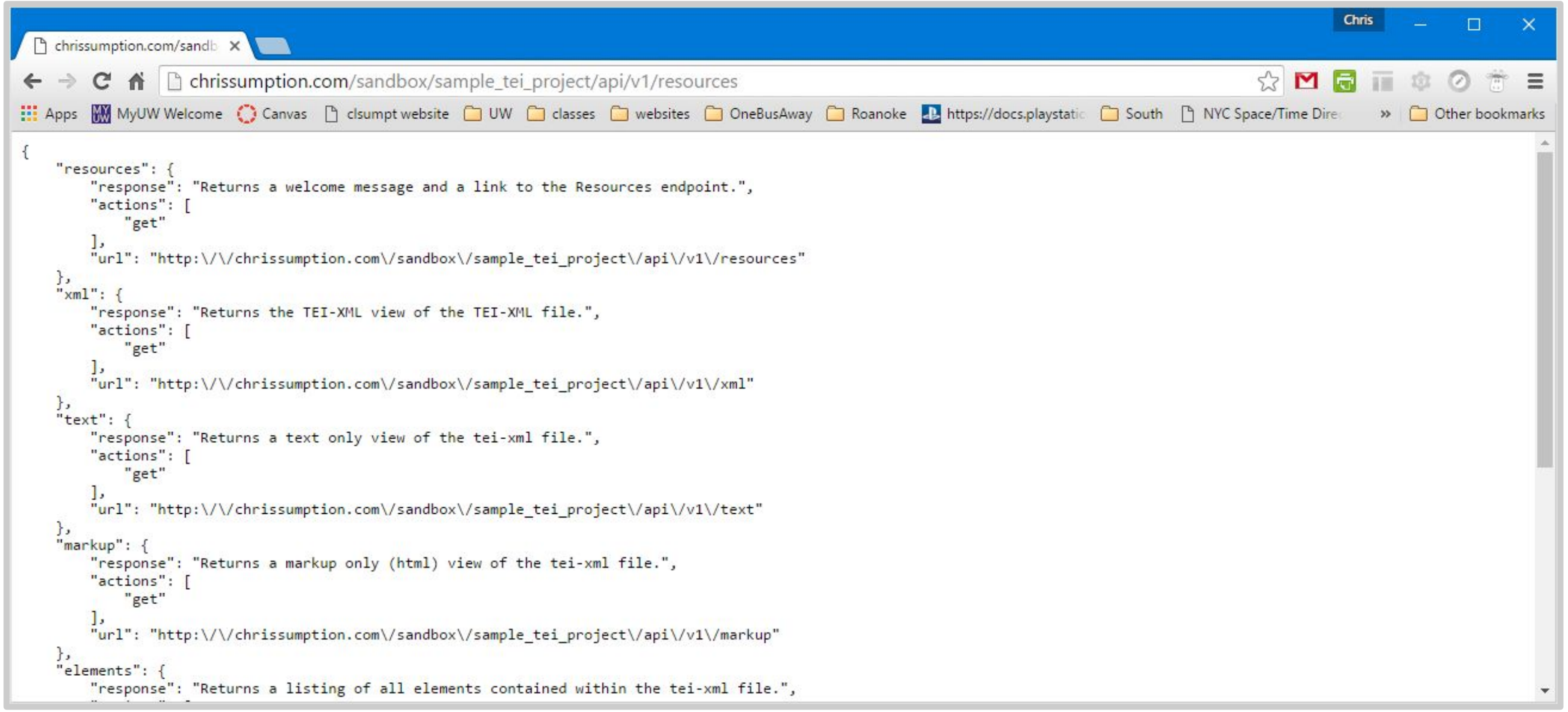
screenshots - api



This is the endpoint URL for the TEI-XML_test_file.xml REST API.

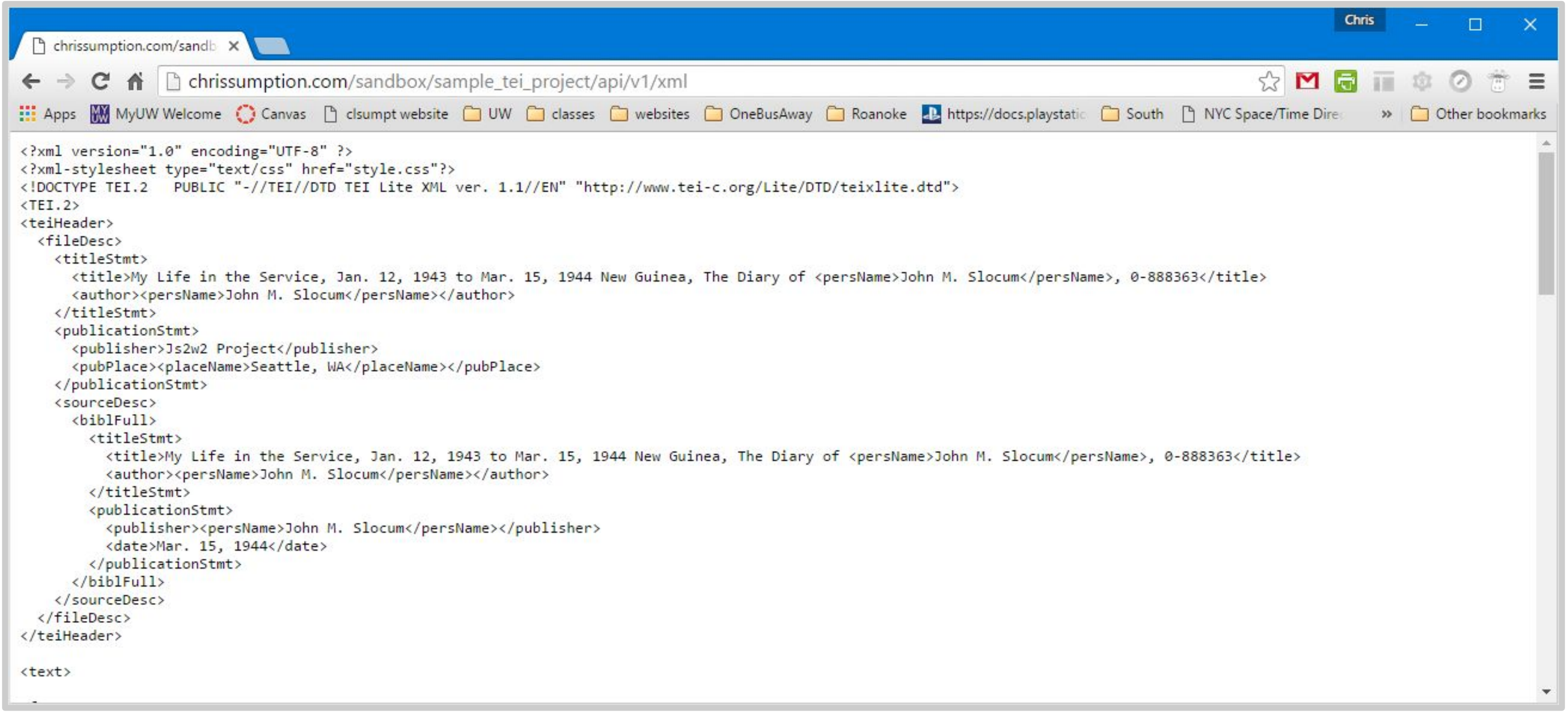
You can view available API resources here.

screenshots - api



```
{
    "resources": {
        "response": "Returns a welcome message and a link to the Resources endpoint.",
        "actions": [
            "get"
        ],
        "url": "http:\/\/chrissumption.com\/sandbox\/sample_tei_project\/api\/v1\/resources"
    },
    "xml": {
        "response": "Returns the TEI-XML view of the TEI-XML file.",
        "actions": [
            "get"
        ],
        "url": "http:\/\/chrissumption.com\/sandbox\/sample_tei_project\/api\/v1\/xml"
    },
    "text": {
        "response": "Returns a text only view of the tei-xml file.",
        "actions": [
            "get"
        ],
        "url": "http:\/\/chrissumption.com\/sandbox\/sample_tei_project\/api\/v1\/text"
    },
    "markup": {
        "response": "Returns a markup only (html) view of the tei-xml file.",
        "actions": [
            "get"
        ],
        "url": "http:\/\/chrissumption.com\/sandbox\/sample_tei_project\/api\/v1\/markup"
    },
    "elements": {
        "response": "Returns a listing of all elements contained within the tei-xml file.",
```

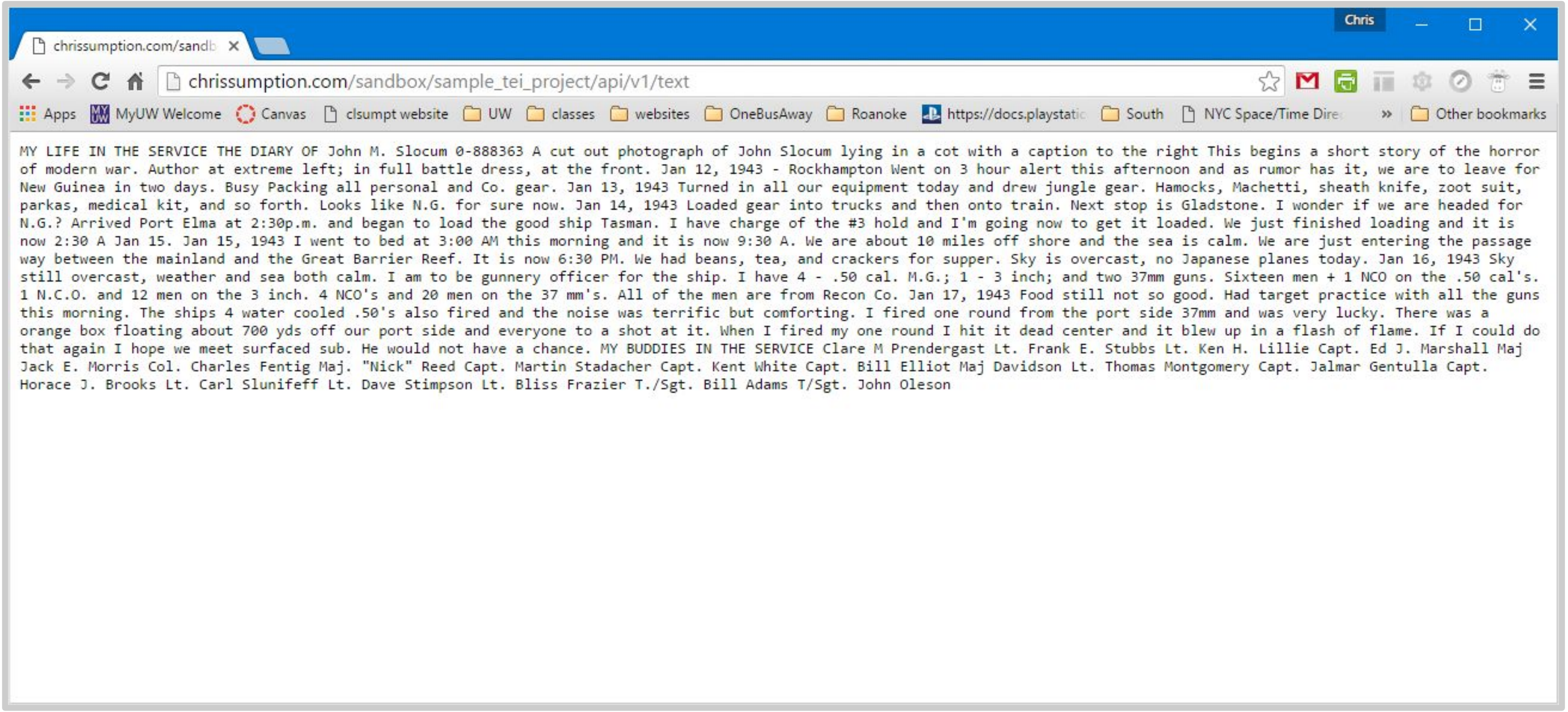
screenshots - api


chrissumption.com/sandb
chrissumption.com/sandbox/sample_tei_project/api/v1/xml
Apps MyUW Welcome Canvas clsumpt website UW classes websites OneBusAway Roanoke https://docs.playstatic South NYC Space/Time Dire Other bookmarks
Chris

```
<?xml version="1.0" encoding="UTF-8" ?>
<?xml-stylesheet type="text/css" href="style.css"?>
<!DOCTYPE TEI.2   PUBLIC "-//TEI//DTD TEI Lite XML ver. 1.1//EN" "http://www.tei-c.org/Lite/DTD/teixlite.dtd">
<TEI.2>
<teiHeader>
  <fileDesc>
    <titleStmt>
      <title>My Life in the Service, Jan. 12, 1943 to Mar. 15, 1944 New Guinea, The Diary of <persName>John M. Slocum</persName>, 0-888363</title>
      <author><persName>John M. Slocum</persName></author>
    </titleStmt>
    <publicationStmt>
      <publisher>Js2w2 Project</publisher>
      <pubPlace><placeName>Seattle, WA</placeName></pubPlace>
    </publicationStmt>
    <sourceDesc>
      <biblFull>
        <titleStmt>
          <title>My Life in the Service, Jan. 12, 1943 to Mar. 15, 1944 New Guinea, The Diary of <persName>John M. Slocum</persName>, 0-888363</title>
          <author><persName>John M. Slocum</persName></author>
        </titleStmt>
        <publicationStmt>
          <publisher><persName>John M. Slocum</persName></publisher>
          <date>Mar. 15, 1944</date>
        </publicationStmt>
      </biblFull>
    </sourceDesc>
  </fileDesc>
</teiHeader>

<text>
```

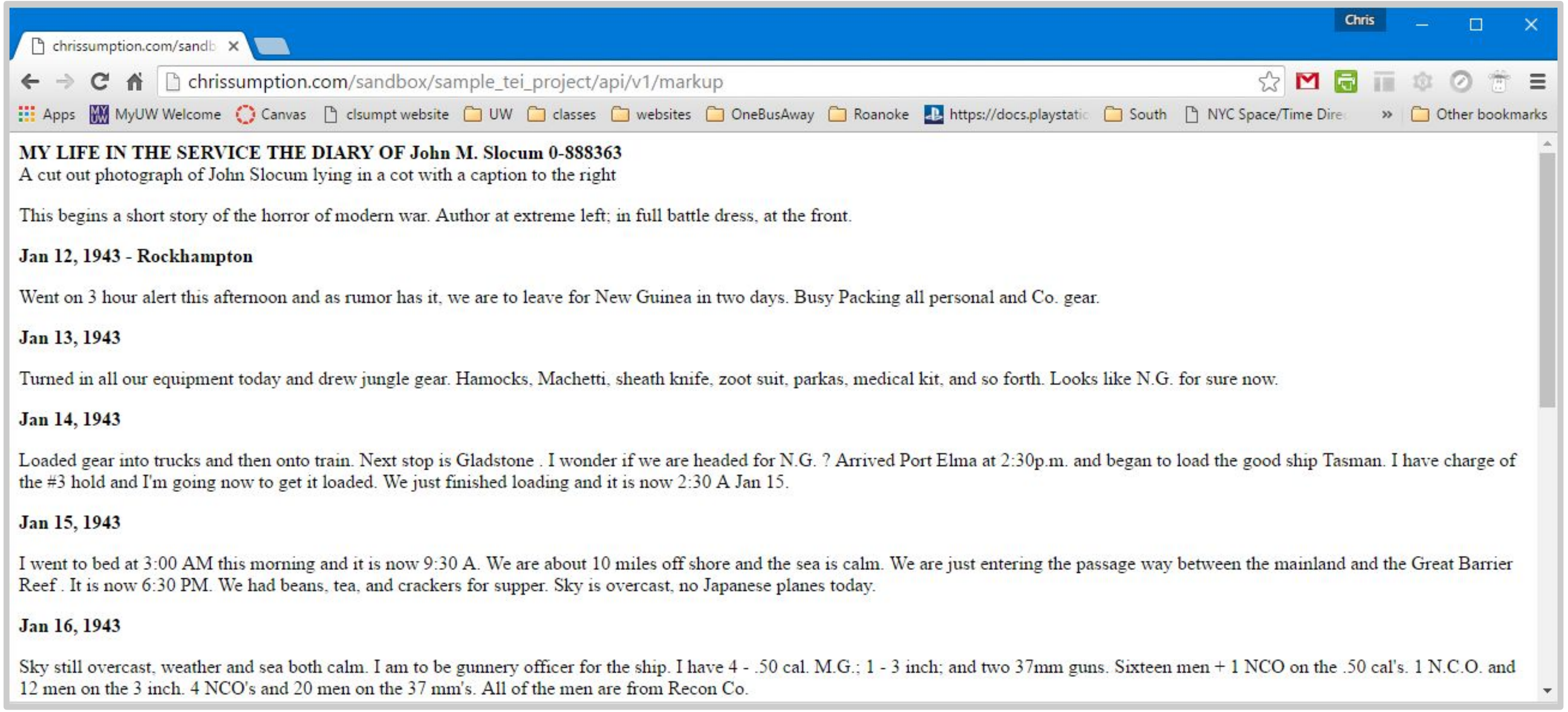
screenshots - api



chrissumption.com/sandbox/sample_tei_project/api/v1/text

```
MY LIFE IN THE SERVICE THE DIARY OF John M. Slocum 0-888363 A cut out photograph of John Slocum lying in a cot with a caption to the right This begins a short story of the horror
of modern war. Author at extreme left; in full battle dress, at the front. Jan 12, 1943 - Rockhampton Went on 3 hour alert this afternoon and as rumor has it, we are to leave for
New Guinea in two days. Busy Packing all personal and Co. gear. Jan 13, 1943 Turned in all our equipment today and drew jungle gear. Hamocks, Machetti, sheath knife, zoot suit,
parkas, medical kit, and so forth. Looks like N.G. for sure now. Jan 14, 1943 Loaded gear into trucks and then onto train. Next stop is Gladstone. I wonder if we are headed for
N.G.? Arrived Port Elma at 2:30p.m. and began to load the good ship Tasman. I have charge of the #3 hold and I'm going now to get it loaded. We just finished loading and it is
now 2:30 A Jan 15. Jan 15, 1943 I went to bed at 3:00 AM this morning and it is now 9:30 A. We are about 10 miles off shore and the sea is calm. We are just entering the passage
way between the mainland and the Great Barrier Reef. It is now 6:30 PM. We had beans, tea, and crackers for supper. Sky is overcast, no Japanese planes today. Jan 16, 1943 Sky
still overcast, weather and sea both calm. I am to be gunnery officer for the ship. I have 4 - .50 cal. M.G.; 1 - 3 inch; and two 37mm guns. Sixteen men + 1 NCO on the .50 cal's.
1 N.C.O. and 12 men on the 3 inch. 4 NCO's and 20 men on the 37 mm's. All of the men are from Recon Co. Jan 17, 1943 Food still not so good. Had target practice with all the guns
this morning. The ships 4 water cooled .50's also fired and the noise was terrific but comforting. I fired one round from the port side 37mm and was very lucky. There was a
orange box floating about 700 yds off our port side and everyone to a shot at it. When I fired my one round I hit it dead center and it blew up in a flash of flame. If I could do
that again I hope we meet surfaced sub. He would not have a chance. MY BUDDIES IN THE SERVICE Clare M Prendergast Lt. Frank E. Stubbs Lt. Ken H. Lillie Capt. Ed J. Marshall Maj
Jack E. Morris Col. Charles Fentig Maj. "Nick" Reed Capt. Martin Stadacher Capt. Kent White Capt. Bill Elliot Maj Davidson Lt. Thomas Montgomery Capt. Jalmar Gentulla Capt.
Horace J. Brooks Lt. Carl Slunifeff Lt. Dave Stimpson Lt. Bliss Frazier T./Sgt. Bill Adams T/Sgt. John Oleson
```

screenshots - api


chrissumption.com/sandb
chrissumption.com/sandbox/sample_tei_project/api/v1/markup
Apps MyUW Welcome Canvas clsumpt website UW classes websites OneBusAway Roanoke https://docs.playstatic South NYC Space/Time Dire Other bookmarks

MY LIFE IN THE SERVICE THE DIARY OF John M. Slocum 0-888363
A cut out photograph of John Slocum lying in a cot with a caption to the right

This begins a short story of the horror of modern war. Author at extreme left; in full battle dress, at the front.

Jan 12, 1943 - Rockhampton

Went on 3 hour alert this afternoon and as rumor has it, we are to leave for New Guinea in two days. Busy Packing all personal and Co. gear.

Jan 13, 1943

Turned in all our equipment today and drew jungle gear. Hamocks, Machetti, sheath knife, zoot suit, parkas, medical kit, and so forth. Looks like N.G. for sure now.

Jan 14, 1943

Loaded gear into trucks and then onto train. Next stop is Gladstone . I wonder if we are headed for N.G. ? Arrived Port Elma at 2:30p.m. and began to load the good ship Tasman. I have charge of the #3 hold and I'm going now to get it loaded. We just finished loading and it is now 2:30 A Jan 15.

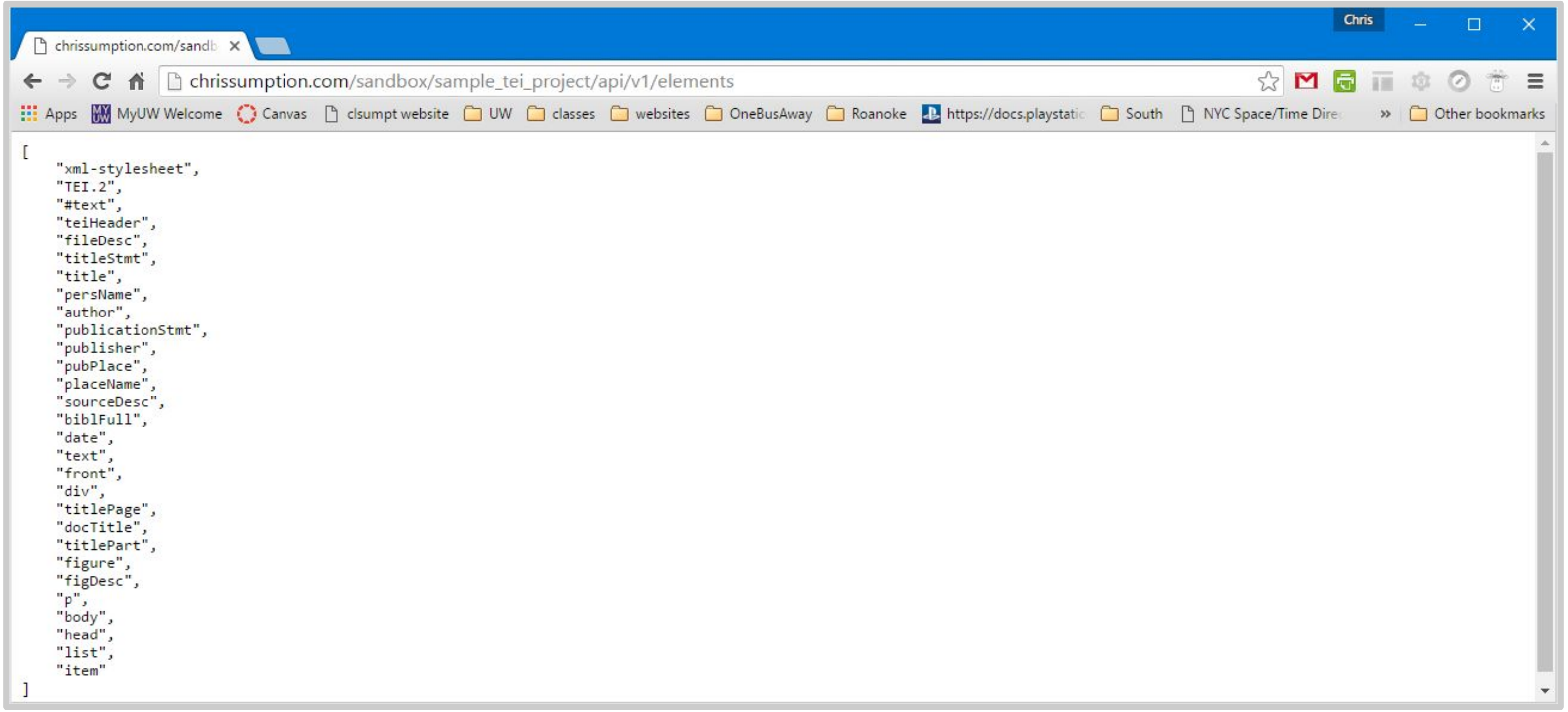
Jan 15, 1943

I went to bed at 3:00 AM this morning and it is now 9:30 A. We are about 10 miles off shore and the sea is calm. We are just entering the passage way between the mainland and the Great Barrier Reef . It is now 6:30 PM. We had beans, tea, and crackers for supper. Sky is overcast, no Japanese planes today.

Jan 16, 1943

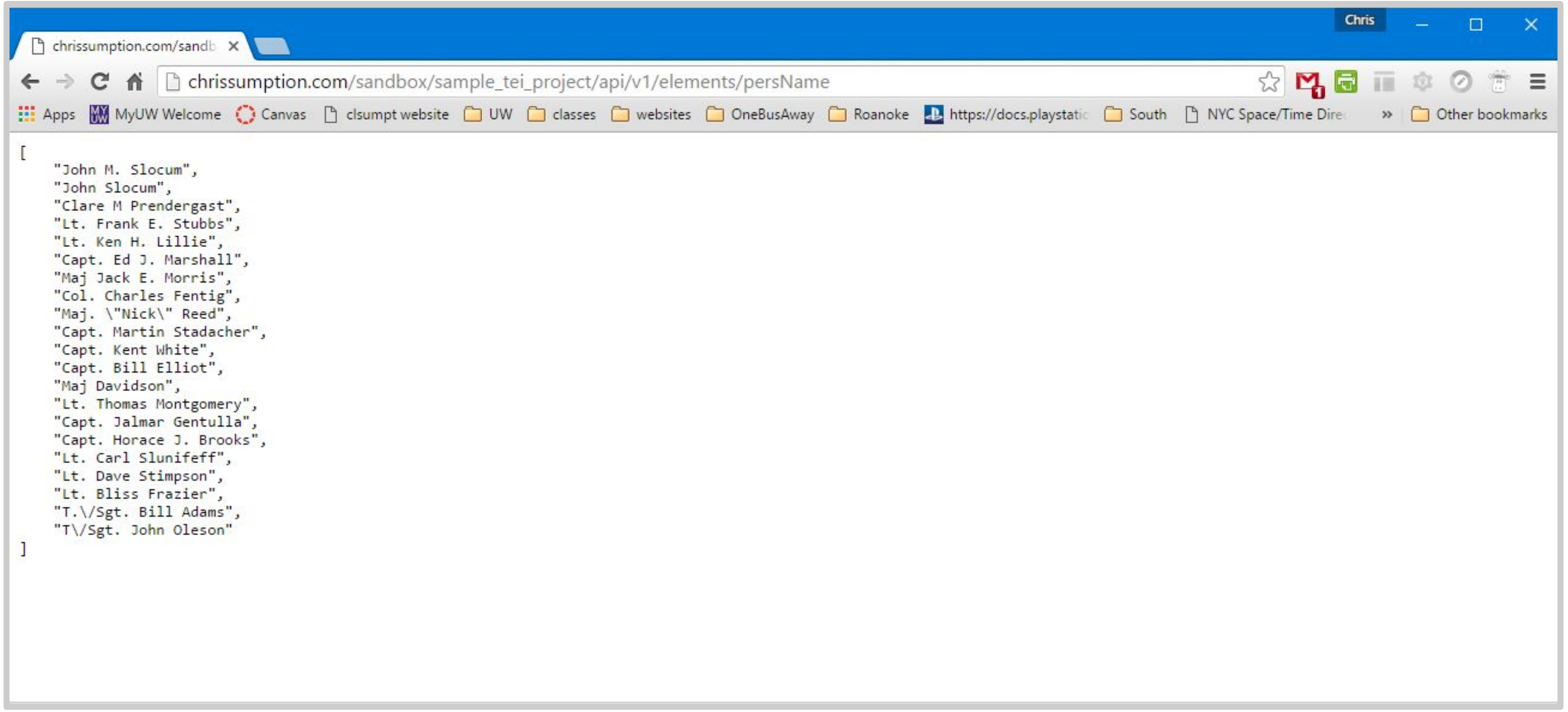
Sky still overcast, weather and sea both calm. I am to be gunnery officer for the ship. I have 4 - .50 cal. M.G.; 1 - 3 inch; and two 37mm guns. Sixteen men + 1 NCO on the .50 cal's. 1 N.C.O. and 12 men on the 3 inch. 4 NCO's and 20 men on the 37 mm's. All of the men are from Recon Co.

screenshots - api



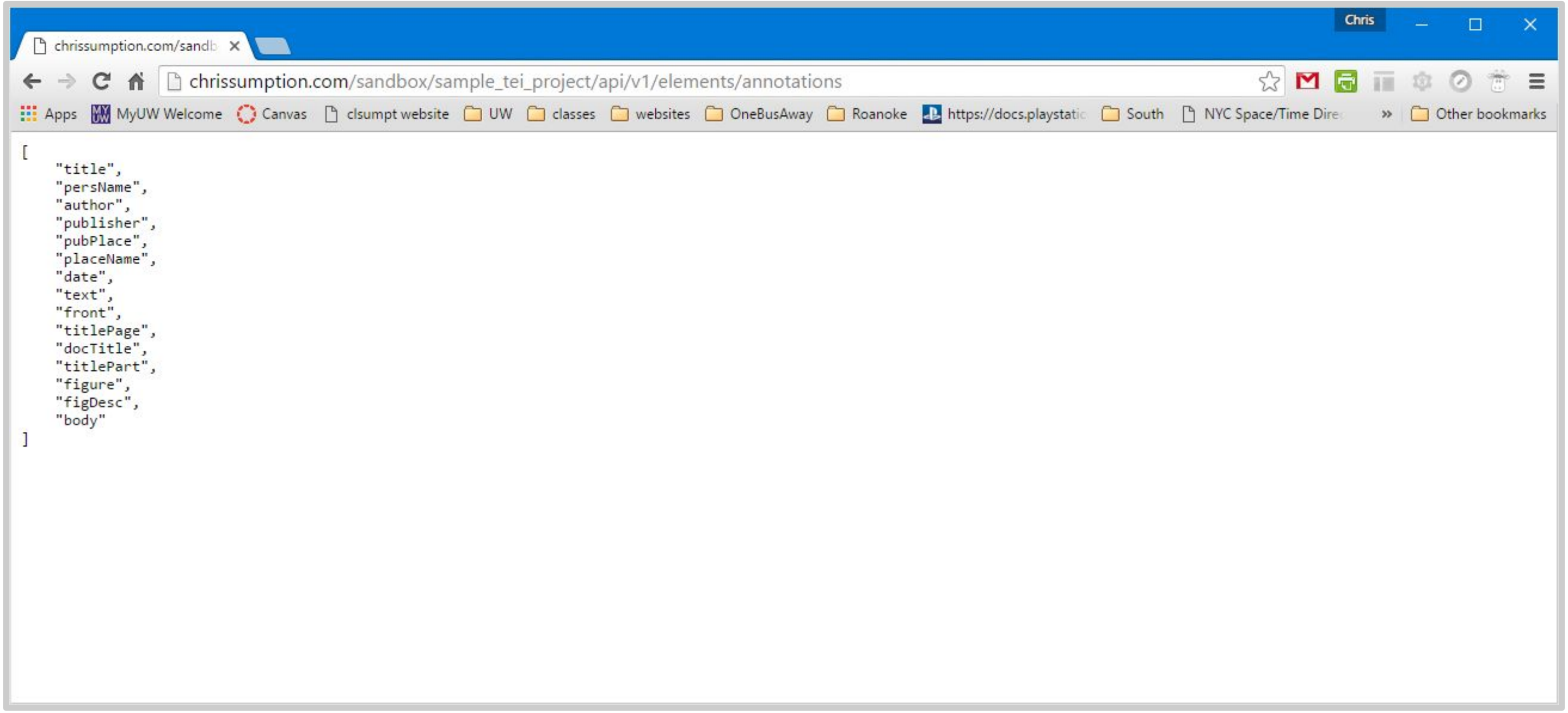
```
[
    "xml-stylesheet",
    "TEI.2",
    "#text",
    "teiHeader",
    "fileDesc",
    "titleStmt",
    "title",
    "persName",
    "author",
    "publicationStmt",
    "publisher",
    "pubPlace",
    "placeName",
    "sourceDesc",
    "biblFull",
    "date",
    "text",
    "front",
    "div",
    "titlePage",
    "docTitle",
    "titlePart",
    "figure",
    "figDesc",
    "p",
    "body",
    "head",
    "list",
    "item"
]
```

screenshots - api



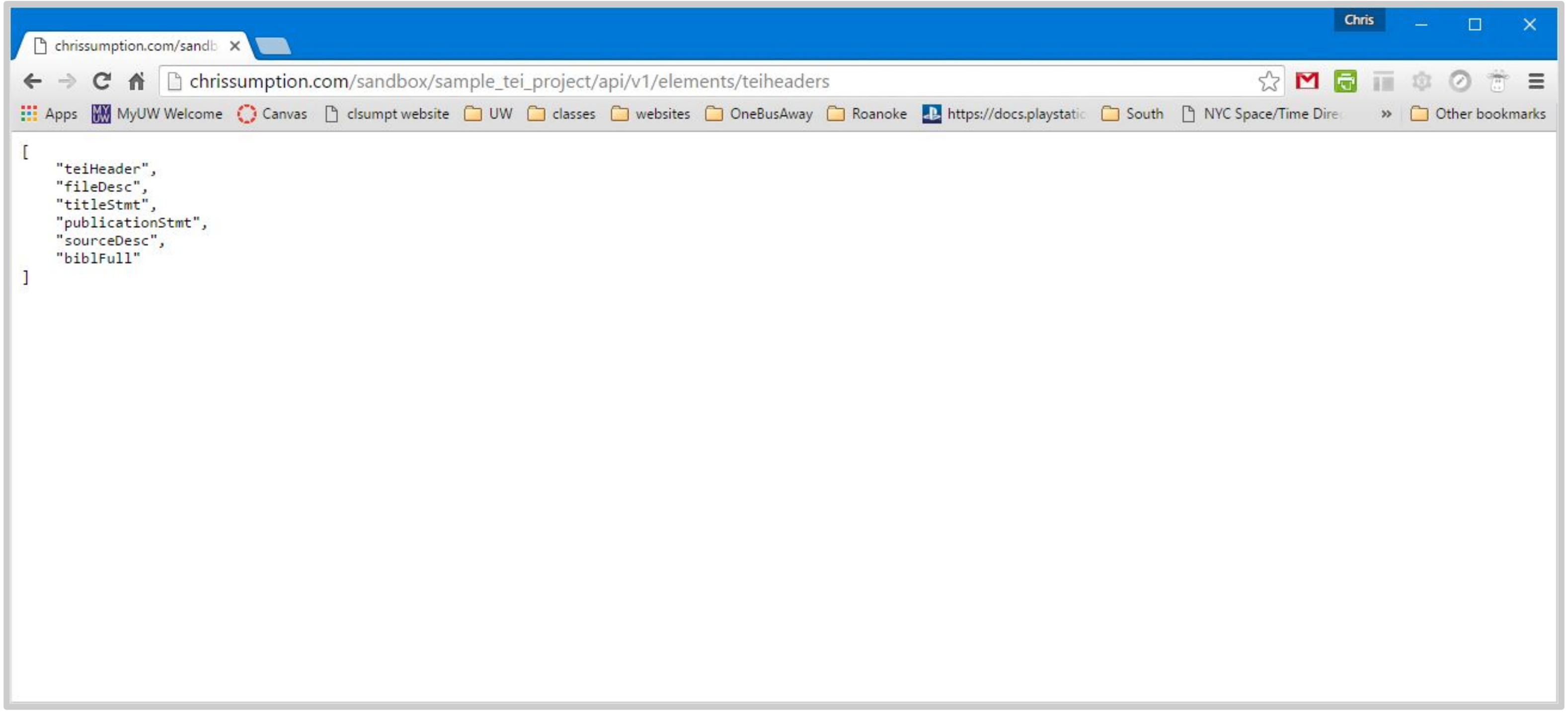
```
[
    "John M. Slocum",
    "John Slocum",
    "Clare M Prendergast",
    "Lt. Frank E. Stubbs",
    "Lt. Ken H. Lillie",
    "Capt. Ed J. Marshall",
    "Maj Jack E. Morris",
    "Col. Charles Fentig",
    "Maj. \"Nick\" Reed",
    "Capt. Martin Stadacher",
    "Capt. Kent White",
    "Capt. Bill Elliot",
    "Maj Davidson",
    "Lt. Thomas Montgomery",
    "Capt. Jalmar Gentulla",
    "Capt. Horace J. Brooks",
    "Lt. Carl Slunifeff",
    "Lt. Dave Stimpson",
    "Lt. Bliss Frazier",
    "T.\/Sgt. Bill Adams",
    "T\/Sgt. John Oleson"
]
```

screenshots - api



```
[
    "title",
    "persName",
    "author",
    "publisher",
    "pubPlace",
    "placeName",
    "date",
    "text",
    "front",
    "titlePage",
    "docTitle",
    "titlePart",
    "figure",
    "figDesc",
    "body"
]
```

screenshots - api



```
[
    "teiHeader",
    "fileDesc",
    "titleStmt",
    "publicationStmt",
    "sourceDesc",
    "biblFull"
]
```

usability testing - methodology

Moderated In-Person Usability Test Plan

Scope: We will be testing how well the naming conventions of the apitext API matched the expectations of it’s users.

Purpose: Will the function and hierarchy of the API’s URI endpoints make enough sense to users without the need to consult additional documentation.

Schedule & Location: There were five user test sessions that occurred over a four week period that coincided with our four development sprints.

- Wednesday April 13, 2016 - Communications Building - Quiet Classroom - Expert Users
- Wednesday April 20, 2016 - Research Commons - Moderately Noisy Open Space - Expert Users
- Wednesday April 27, 2016 - Eastlake Starbucks - Very Noisy Public Space - Expert Users
- Wednesday May 4, 2016 - Communications Building - Quiet Classroom - Expert Users
- Friday, May 6, 2016 - Anderson Hall - Very Noisy Open Space - Non Expert Users

usability testing - methodology

Moderated In-Person Usability Test Plan (continued)

Sessions: Each usability testing session lasted between 15 - 20 minutes. The moderator would spend the first 5 minutes of the session friendly conversing with the user. This served as a warm up for the upcoming task completions. The moderator then spent 2 minutes explaining the testing scenario. The user would then be observed for approximately 10 to 13 minutes as they completed tasks.

Equipment: The users completed tasks with an Acer Chromebook, with touchscreen capability and a resolution of 1536 x 864. There were no audio or video recordings made, all documentation was captured with paper and pencil.

Participants: There were four expert users (1 male and 3 female), each with varying levels of previous experience with the TEI-XML standard. They were previous contacts of Chris Sumption and were asked by him to participate in the testing. There was one non expert user (male) with no previous TEI-XML experience that was recruited at random during a INFO 491 Capstone class event.

usability testing - methodology

Moderated In-Person Usability Test Plan (continued)

Scenarios: Each user test session consisted of 4 tasks. Each task was connected to a URI endpoint. Users were provided with the URL to a TEI-XML file and were then asked to retrieve specific information from that file using the URI endpoints exposed by the API.

Metrics: Only qualitative data was captured during each testing scenario. Users were asked to communicate their thought processes as they navigated between the different URI endpoints. They were also encouraged to report instances where they experienced any confusion.

Roles: Chris Sumption acted as moderator and observer during the 4 expert user testing sessions. For the non expert user session, Chris Sumption acted as the moderator and Guiyan Bai acted as an observer.

usability testing - script

Expert User Testing Script

<u>Introduction (5 minutes):</u>
Hi, My name is [name]. [Small talk with the user, ask questions, if conversation permits attempt to illicit thought processes -- attempt to get user comfortable sharing with you]

<u>Test Scenario Introduction (2 minutes):</u>
Our project, apitext is building a prototype RESTful Application Programming Interface (API) for TEI-XML documents that provides greater interoperability for these documents and easy to use by an audience with backgrounds in the humanities, social sciences, and linguistics. An important component of our prototype are the names of the resources that our API exposes and how they are organized hierarchically. You are going to help us by testing those names and how they are organized. You will be given four tasks to complete that are related to information retrieval from a TEI-XML file. You will be given a URL to our API that you will use to complete those tasks. While you are working, it would great if you could talk aloud your actions and your thought process as you work through these tasks. It will especially be helpful if you alert us to interactions with the API that are confusing to you.

usability testing - script

Expert User Testing Script (continued)

Testing Scenarios - Wednesday April 13, 2016 - (13 minutes):

1. You would like to perform a text analysis of the provided TEI-XML file. Attempt to use the Apitext API to assist with this task.
2. You would like to prepare the textual content of the provided TEI-XML file for presentation on your projects website. Attempt to use the Apitext API to assist with this task.
3. You would like to display a listing of the types bibliographical information connected with this TEI-XML file on your project's web site. Attempt to use the Apitext API to assist with this task.
4. You would like to display a listing of the types of researcher annotation tags connected with this TEI-XML file on your project's web site. Attempt to use the Apitext API to assist with this task.

usability testing - script

Expert User Testing Script (continued)

Testing Scenarios - Wednesday April 20, 2016 - (13 minutes):

1. You would like to perform a text analysis of the provided TEI-XML file. Attempt to use the Apitext API to assist with this task.
2. You would like to prepare the textual content of the provided TEI-XML file for presentation on your projects website. Attempt to use the Apitext API to assist with this task.
3. You would like to display a listing of the types bibliographical information connected with this TEI-XML file on your project's web site. Attempt to use the Apitext API to assist with this task.
4. You would like to display a listing of the types of researcher annotation tags connected with this TEI-XML file on your project's web site. Attempt to use the Apitext API to assist with this task.

usability testing - script

Expert User Testing Script (continued)

Testing Scenarios - Wednesday April 27, 2016 - (13 minutes):

1. You would like to perform a text analysis of the provided TEI-XML file. Attempt to use the Apitext API to assist with this task.
2. You would like to prepare the textual content of the provided TEI-XML file for presentation on your projects website. Attempt to use the Apitext API to assist with this task.
3. You would like to display a listing of the types of researcher annotation tags connected with this TEI-XML file on your project's web site. Attempt to use the Apitext API to assist with this task.
4. You would like to display a listing of all people mentioned in the provided TEI-XML file on your project's web site. Attempt to use the Apitext API to assist with this task.

usability testing - script

Expert User Testing Script (continued)

Testing Scenarios - Wednesday May 4, 2016 - (13 minutes):

1. You would like to perform a text analysis of the provided TEI-XML file. Attempt to use the Apitext API to assist with this task.
2. You would like to prepare the textual content of the provided TEI-XML file for presentation on your projects website. Attempt to use the Apitext API to assist with this task.
3. You would like to display a listing of the types of researcher annotation tags connected with this TEI-XML file on your project's web site. Attempt to use the Apitext API to assist with this task.
4. You would like to display a listing of all people mentioned in the provided TEI-XML file on your project's web site. Attempt to use the Apitext API to assist with this task.

usability testing - script

Non Expert User Testing Script

Introduction (5 minutes):
Hi, My name is [name]. [Small talk with the user, ask questions, if conversation permits attempt to illicit thought processes -- attempt to get user comfortable sharing with you]

Test Scenario Introduction (2 minutes):
Our project, apitext is building a prototype RESTful Application Programming Interface (API) for TEI-XML documents that provides greater interoperability for these documents and easy to use by an audience with backgrounds in the humanities, social sciences, and linguistics. An important component of our prototype are the names of the resources that our API exposes and how they are organized hierarchically. You are going to help us by testing those names and how they are organized. You will be given four tasks to complete that are related to information retrieval from a TEI-XML file. You will be given a URL to our API that you will use to complete those tasks. While you are working, it would great if you could talk aloud your actions and your thought process as you work through these tasks. It will especially be helpful if you alert us to interactions with the API that are confusing to you.

usability testing - script

Non Expert User Testing Script (continued)

Testing Scenarios - Friday, May 6, 2016 - (13 minutes):

You are a developer that supports a tool for processing data from books (think Amazon, Google Books, etc…). Your company has just acquired a new library of books converted to XML files using something called “TEI.” Your boss says the library has an API, but no documentation for it. He hands you a URL and asks you to see if the API can be of any use to us:

1. The tool you support has text analysis functionality. Can the API provide raw text for this tool or will we need to pre process the text with one of our tools?
2. The tool you support has geographical mapping capabilities. Can the API provide a listing of geographical places that can then be ported into your tool?

usability testing - key findings

Wednesday April 13, 2016

Communications Building - Quiet Classroom - Expert Users

<u>Key Findings:</u>

- User seemed to have difficulty understanding the hierarchy of the names listed in the resource page. They systematically went down the list trying out each endpoint. They experienced confusion while navigating to the “base folder” and “welcome” endpoints.

<u>Recommended Improvements:</u>

- Remove the “base folder” and “welcome” endpoints from the “resources” endpoint list.
    - Rationale: The user has already navigated to the resources page, these endpoints occur before it. It interrupts the hierarchy of the navigation.

usability testing - key findings

Wednesday April 20, 2016

Research Commons - Moderately Noisy Open Space - Expert Users

Key Findings:

- User had difficulty distinguishing the difference between the “text” and “markup” endpoints when attempting to complete tasks 1 and 2.
  - The user felt that both the “text” and the “ markup endpoints were basically the same since they provided formatting.
  - User suggested that it might be better to have two separate “text” endpoints. One that only provided just the “text” and one that is “formatted text”.

Recommended Improvements:

- Change the output of the “text” endpoint so that it only returns unformatted textual content.
  - Rationale: When you see the word “text” you should only be getting text. That seems more true to REST. At this time providing a second “formatted text” option is deemed out of scope due to time constraints.

usability testing - key findings

Wednesday April 27, 2016

Eastlake Starbucks - Very Noisy Public Space - Expert Users

Key Findings:

- The user experienced confusion when trying to complete tasks 3 and 4. When asked to talk more about the confusion, the didn’t understand what the word “tag” referred to in the resources listing. When explained what the reference meant, the user explained that “element” would be a better word for that endpoint.

Recommended Improvements:

- Change the word “tag” to “element” in the resource page descriptions.
  - Rationale: TEI documentation uses the word “element” or “element tag”.

usability testing - key findings

Wednesday May 4, 2016

Communications Building - Quiet Classroom - Expert Users

<u>Key Findings:</u>

- The user experienced confusion when trying to complete tasks 3 and 4. When asked to talk more about the confusion, the user didn't understand the hierarchy of the "annotation" and "teiheader" elements. They wanted to see a listing of all elements.

<u>Recommended Improvements:</u>

- Change the hierarchy of the endpoints from "annotation" or "teiheader" to:
    - "element" - a listing of all elements that are found within the document
    - Then "element/annotation" or "element/teiheader".
    - Rationale: This provides a more understandable hierarchy for the endpoints. It also gives the user more flexibility in the type of views that can be returned.

usability testing - key findings

Friday, May 6, 2016

Anderson Hall - Very Noisy Open Space - Non Expert Users

Key Findings:

- User experienced frustration with the escape characters in the URL’s. The user wanted to be able to just cut and paste them into the browser's search bar.
- The user experienced confusion when trying to retrieve a listing of a specific element.
- The user experienced confusion when reading the “url” descriptions of the endpoints.

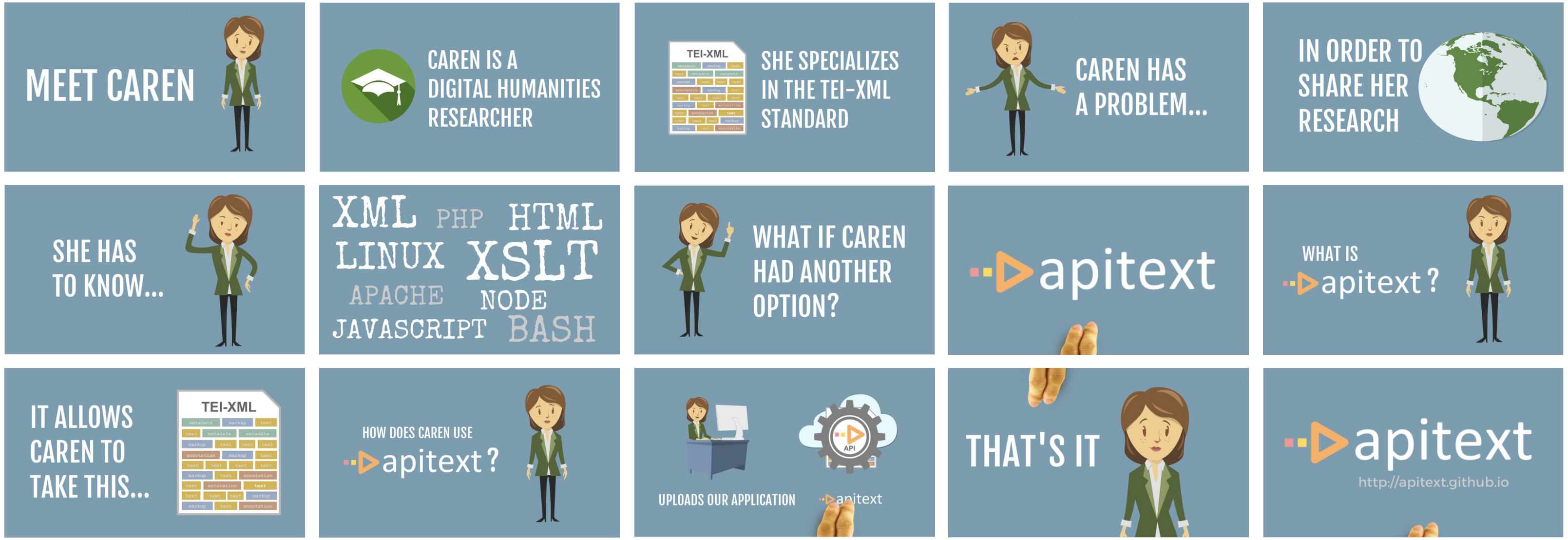
Recommended Improvements:

- Add a non escaped URL entry for each endpoint “url” description.
- Add a more verbose explanation for each endpoints “response” description.
- Create or better Highlight the ability to return “index” information from a “elements” enpoint call.

media

Pitch Video, Project Poster, Documentation Website, Demonstration Website

pitch video


MEET CAREN
CAREN IS A DIGITAL HUMANITIES RESEARCHER
TEI-XML
SHE SPECIALIZES IN THE TEI-XML STANDARD
CAREN HAS A PROBLEM...
IN ORDER TO SHARE HER RESEARCH
SHE HAS TO KNOW...
XML PHP HTML LINUX XSLT APACHE NODE JAVASCRIPT BASH
WHAT IF CAREN HAD ANOTHER OPTION?
apitext
WHAT IS apitext ?
IT ALLOWS CAREN TO TAKE THIS...
TEI-XML
HOW DOES CAREN USE apitext ?
UPLOADS OUR APPLICATION
API
apitext
THAT'S IT
apitext
http://apitext.github.io

https://youtu.be/lIKqgCywfe8

project poster - design

apitext

A RESTful Application Programming Interface (API) for TEI-XML documents

Problem Statement

How do we build a RESTful Application Programming Interface (API) for TEI-XML documents that provides greater interoperability while at the same time is easy to use by an audience with backgrounds in the humanities, social sciences, and linguistics.

What is TEI?

- TEI stands for The Text Encoding Initiative
- It is a consortium which collectively develops and maintains a standard for the representation of texts in digital form.
- Its chief deliverable is a set of Guidelines which specify encoding methods for machine-readable texts, chiefly in the humanities, social sciences and linguistics.
- XML is the language used for encoding.

Desired Outcomes

- Provide an interface for applications and humans that that will allow clients easy access to a DH project's archive of TEI-XML encoded documents. If applicable this interface will also enable the archived TEI data to be more interoperable with TEI data from other projects and domains.
- Be able to be implemented by individuals who have a minimum of technical expertise. This will be accomplished through the architecture of the API and the design of its documentation.

Information School

apitext

An API that has an appetite for TEI-XML Transcriptions

Problem

How we did it

Background

Solution

Social impact

Tools:

Mentor:

Capstone Project Members: Coyote Bai, Michael Andrea, Chris Sumption

apitext

An API that has an appetite for TEI-XML Transcriptions

Problem

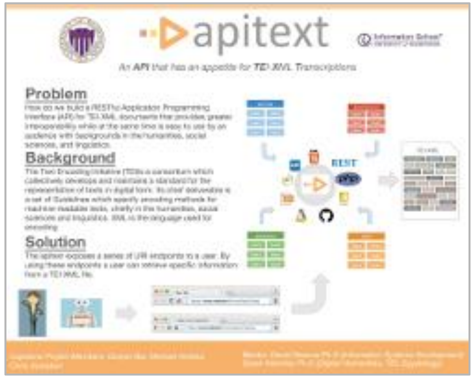
How we did it

Background

Solution

Social impact

Tools:



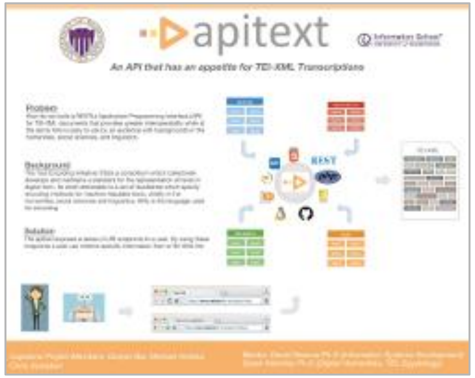
apitext

An API that has an appetite for TEI-XML Transcriptions

Problem

Background

Solution



apitext

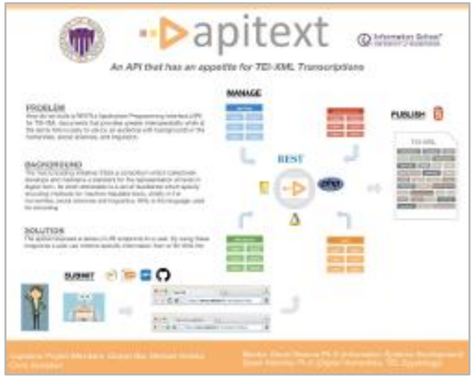
An API that has an appetite for TEI-XML Transcriptions

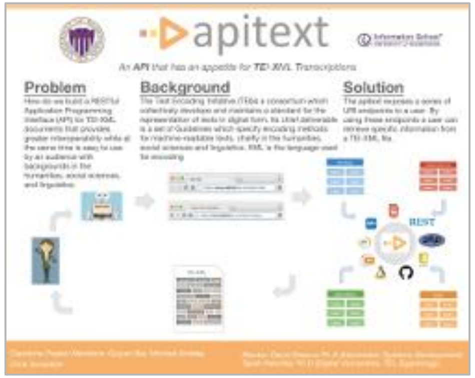
Problem

Background

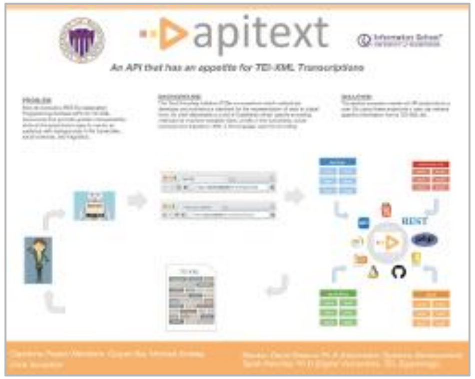
Solution

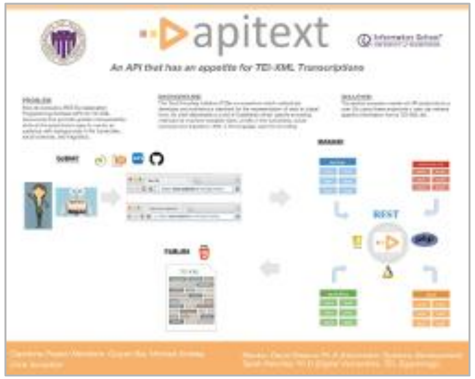
project poster - design

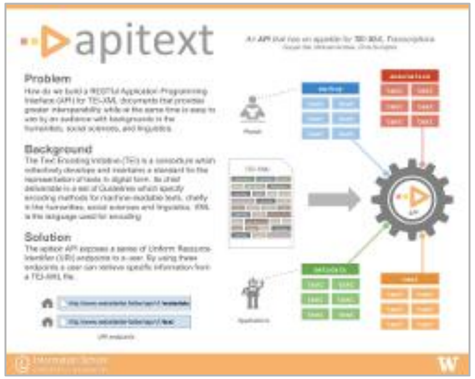


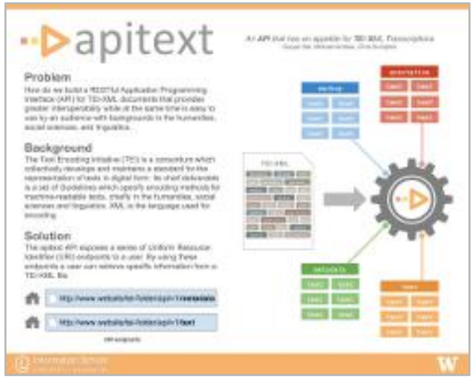


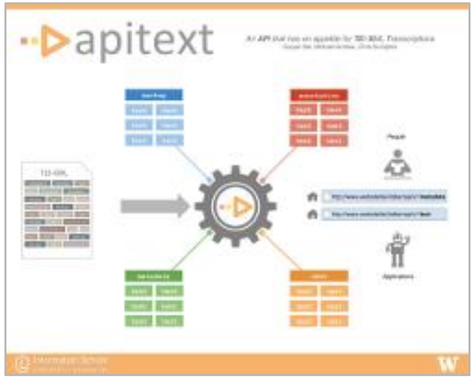
project poster - design

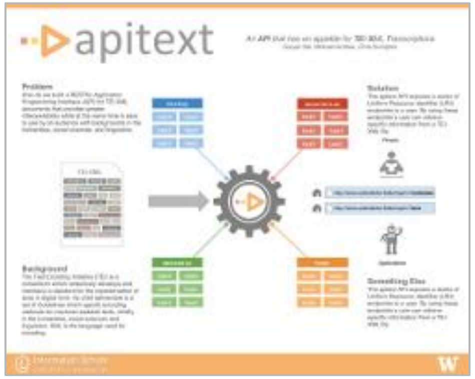




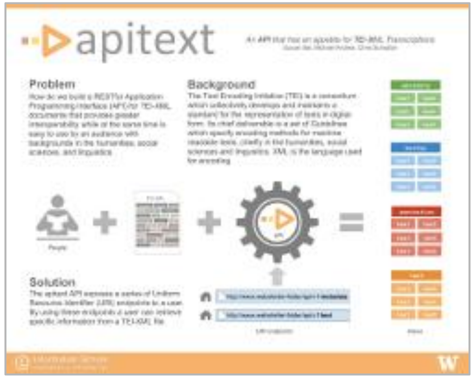
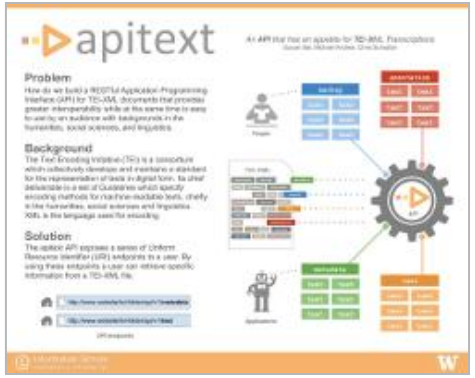
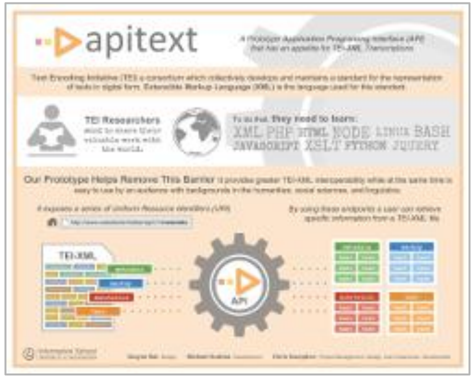
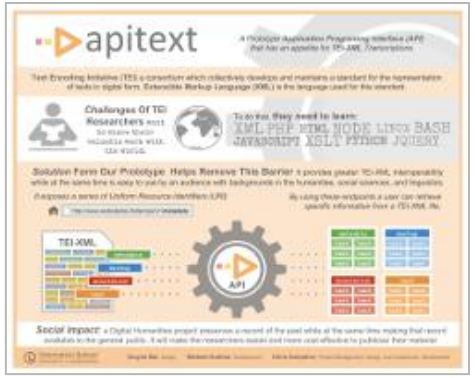
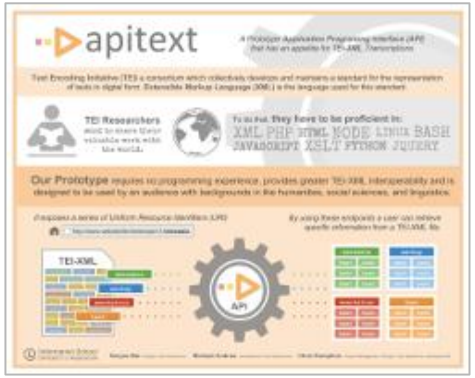
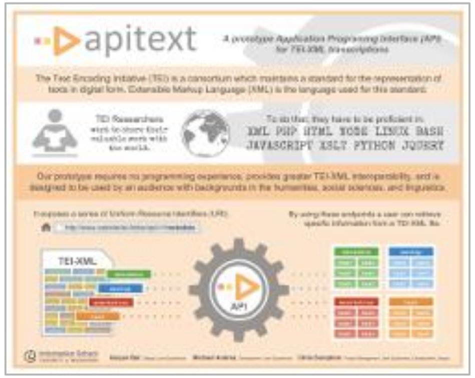




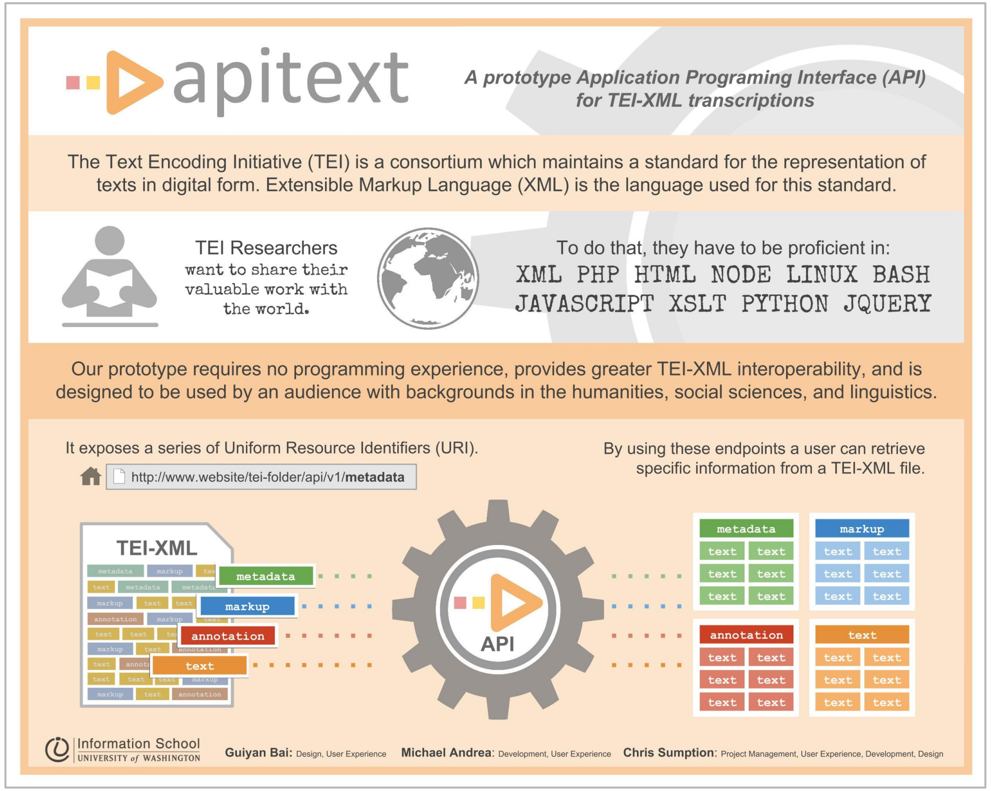




project poster - design

project poster - final



apitext

A prototype Application Programing Interface (API) for TEI-XML transcriptions

The Text Encoding Initiative (TEI) is a consortium which maintains a standard for the representation of texts in digital form. Extensible Markup Language (XML) is the language used for this standard.

TEI Researchers want to share their valuable work with the world.

To do that, they have to be proficient in:
XML PHP HTML NODE LINUX BASH
JAVASCRIPT XSLT PYTHON JQUERY

Our prototype requires no programming experience, provides greater TEI-XML interoperability, and is designed to be used by an audience with backgrounds in the humanities, social sciences, and linguistics.

It exposes a series of Uniform Resource Identifiers (URI).

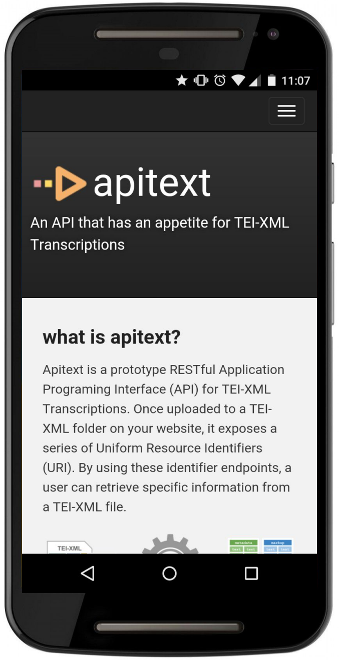
http://www.website/tei-folder/api/v1/**metadata**

By using these endpoints a user can retrieve specific information from a TEI-XML file.

Information School UNIVERSITY *of* WASHINGTON

Guiyan Bai: Design, User Experience **Michael Andrea:** Development, User Experience **Chris Sumption:** Project Management, User Experience, Development, Design

documentation website http://apitext.github.io

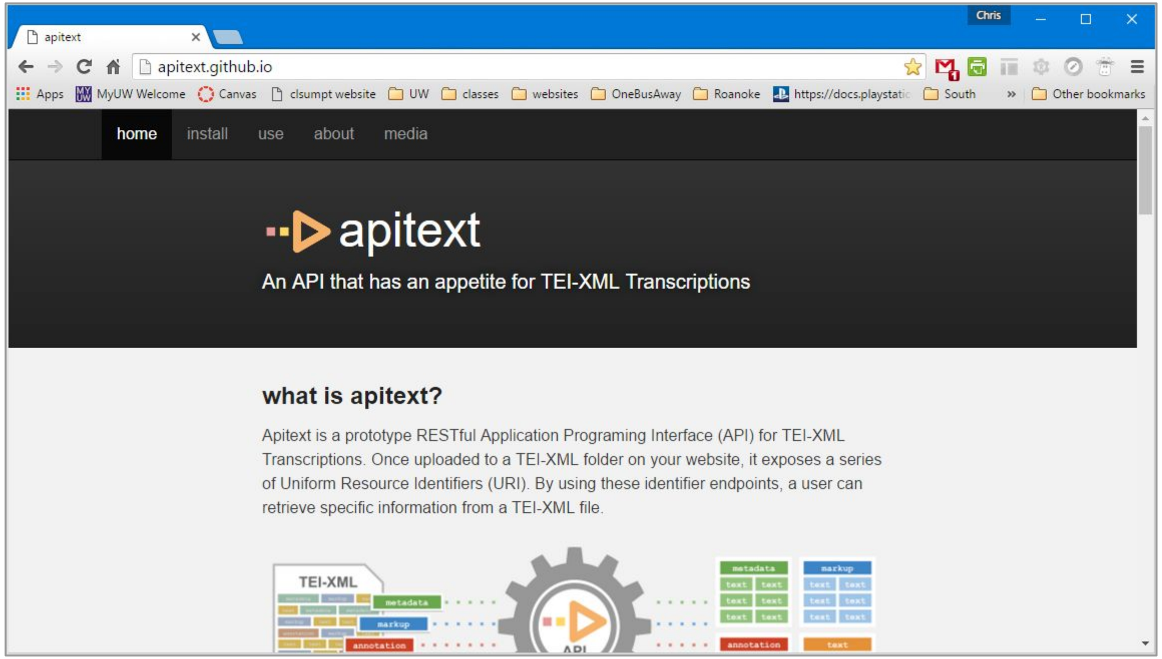


apitext

An API that has an appetite for TEI-XML Transcriptions

what is apitext?

Apitext is a prototype RESTful Application Programing Interface (API) for TEI-XML Transcriptions. Once uploaded to a TEI-XML folder on your website, it exposes a series of Uniform Resource Identifiers (URI). By using these identifier endpoints, a user can retrieve specific information from a TEI-XML file.



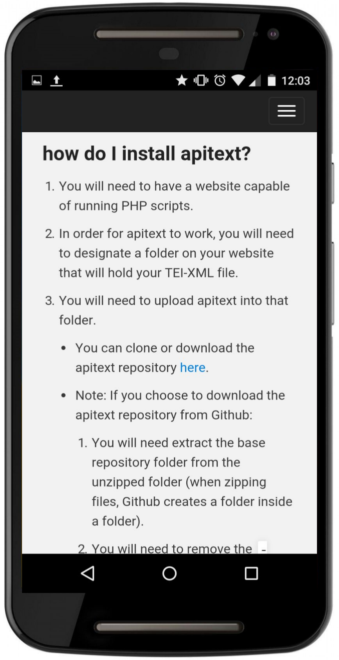
apitext

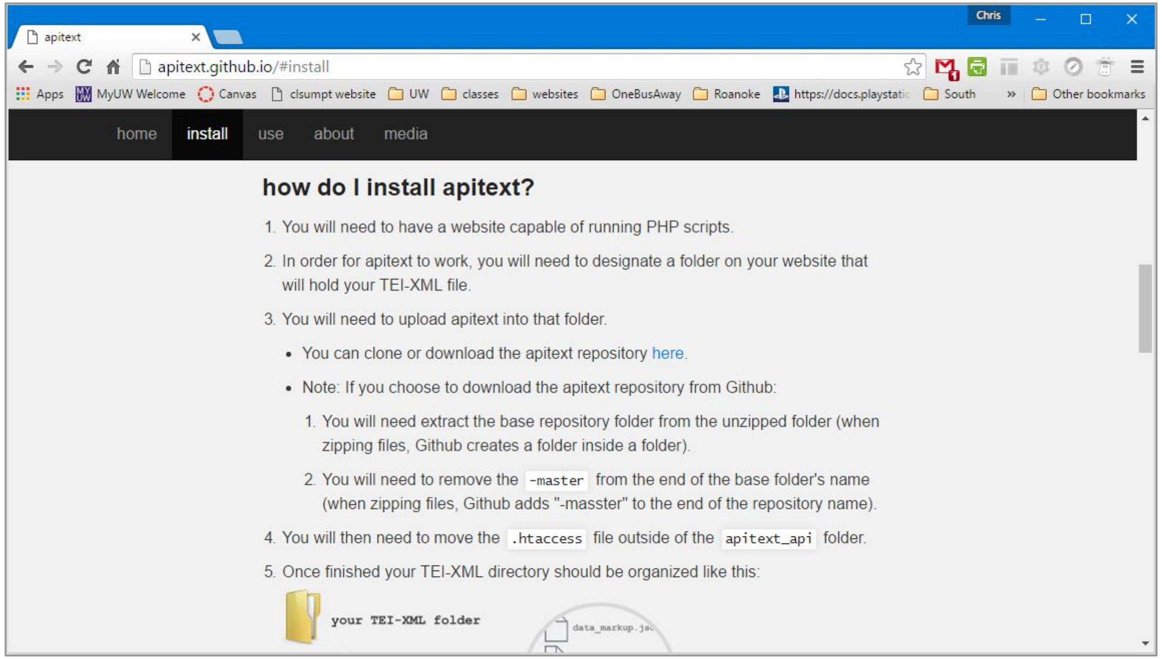
An API that has an appetite for TEI-XML Transcriptions

what is apitext?

Apitext is a prototype RESTful Application Programing Interface (API) for TEI-XML Transcriptions. Once uploaded to a TEI-XML folder on your website, it exposes a series of Uniform Resource Identifiers (URI). By using these identifier endpoints, a user can retrieve specific information from a TEI-XML file.

documentation website http://apitext.github.io/#install

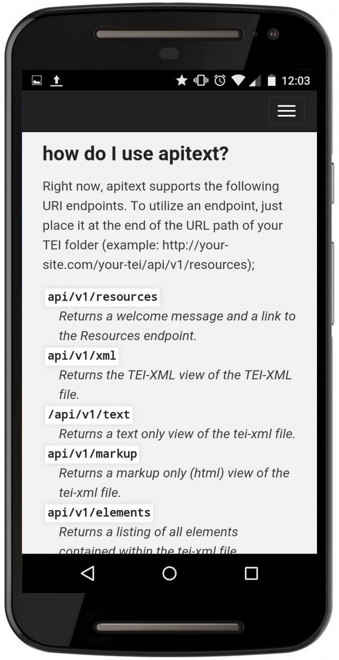




how do I install apitext?

1. You will need to have a website capable of running PHP scripts.
2. In order for apitext to work, you will need to designate a folder on your website that will hold your TEI-XML file.
3. You will need to upload apitext into that folder.
   - You can clone or download the apitext repository here.
   - Note: If you choose to download the apitext repository from Github:
     1. You will need extract the base repository folder from the unzipped folder (when zipping files, Github creates a folder inside a folder).
     2. You will need to remove the `-master` from the end of the base folder's name (when zipping files, Github adds "-masster" to the end of the repository name).
4. You will then need to move the `.htaccess` file outside of the `apitext_api` folder.
5. Once finished your TEI-XML directory should be organized like this:

documentation website http://apitext.github.io/#use



how do I use apitext?

Right now, apitext supports the following URI endpoints. To utilize an endpoint, just place it at the end of the URL path of your TEI folder (example: http://your-site.com/your-tei/api/v1/resources);

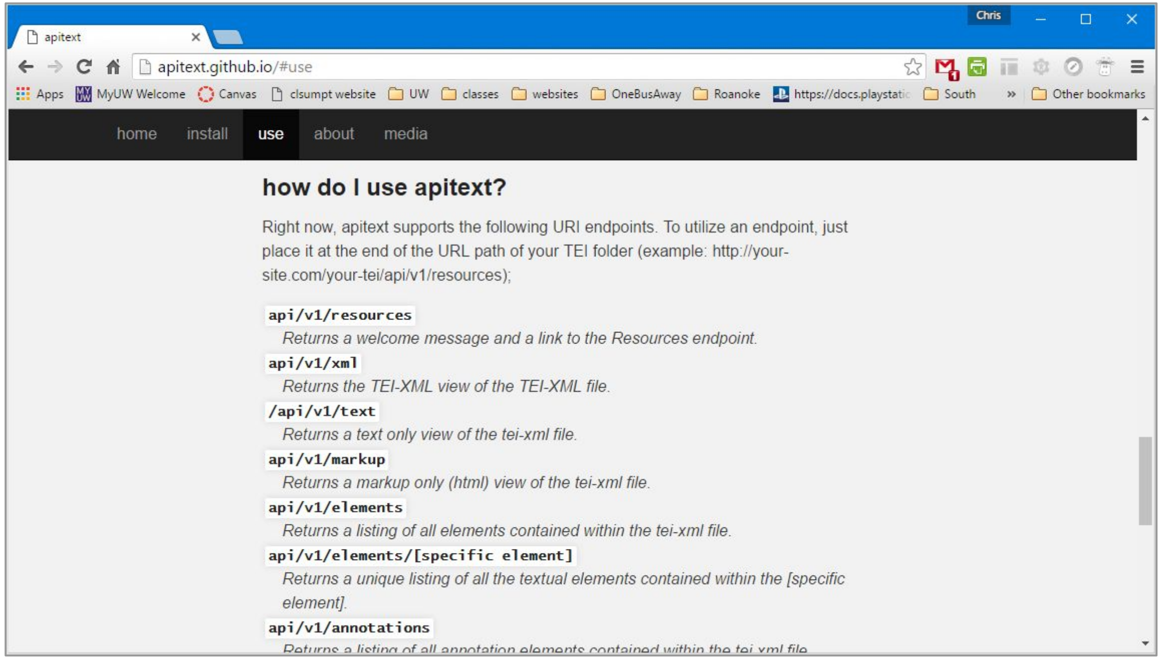
`api/v1/resources`
Returns a welcome message and a link to the Resources endpoint.

`api/v1/xml`
Returns the TEI-XML view of the TEI-XML file.

`/api/v1/text`
Returns a text only view of the tei-xml file.

`api/v1/markup`
Returns a markup only (html) view of the tei-xml file.

`api/v1/elements`
Returns a listing of all elements contained within the tei-xml file



how do I use apitext?

Right now, apitext supports the following URI endpoints. To utilize an endpoint, just place it at the end of the URL path of your TEI folder (example: http://your-site.com/your-tei/api/v1/resources);

`api/v1/resources`
Returns a welcome message and a link to the Resources endpoint.

`api/v1/xml`
Returns the TEI-XML view of the TEI-XML file.

`/api/v1/text`
Returns a text only view of the tei-xml file.

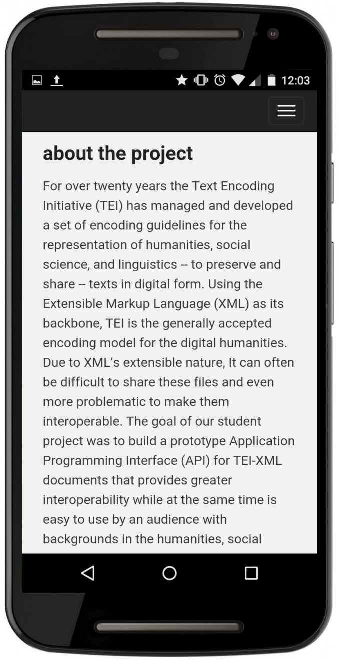
`api/v1/markup`
Returns a markup only (html) view of the tei-xml file.

`api/v1/elements`
Returns a listing of all elements contained within the tei-xml file.

`api/v1/elements/[specific element]`
Returns a unique listing of all the textual elements contained within the [specific element].

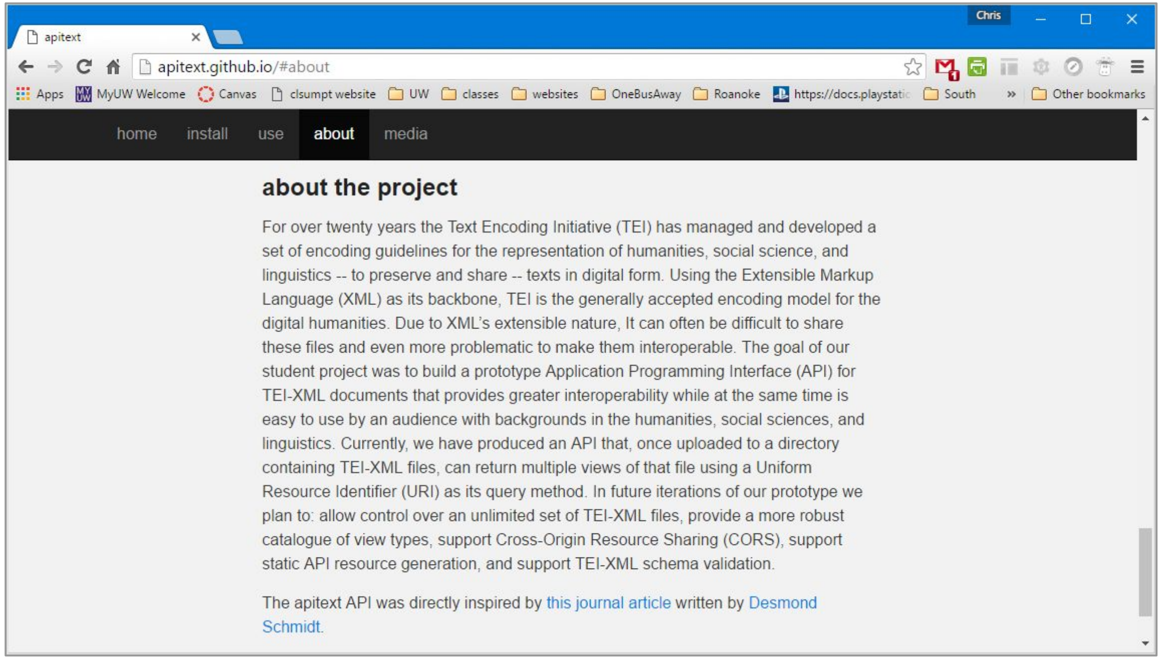
`api/v1/annotations`
Returns a listing of all annotation elements contained within the tei-xml file

documentation website http://apitext.github.io/#about



about the project

For over twenty years the Text Encoding Initiative (TEI) has managed and developed a set of encoding guidelines for the representation of humanities, social science, and linguistics -- to preserve and share -- texts in digital form. Using the Extensible Markup Language (XML) as its backbone, TEI is the generally accepted encoding model for the digital humanities. Due to XML's extensible nature, It can often be difficult to share these files and even more problematic to make them interoperable. The goal of our student project was to build a prototype Application Programming Interface (API) for TEI-XML documents that provides greater interoperability while at the same time is easy to use by an audience with backgrounds in the humanities, social

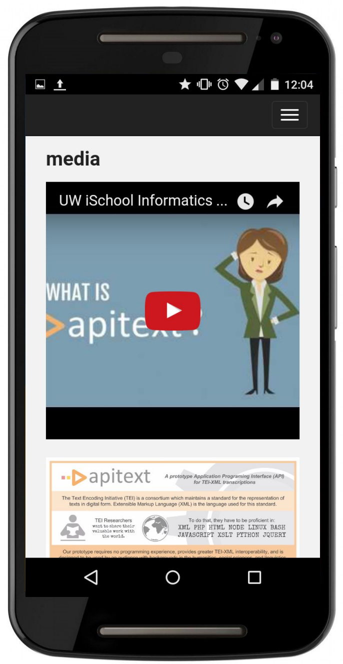


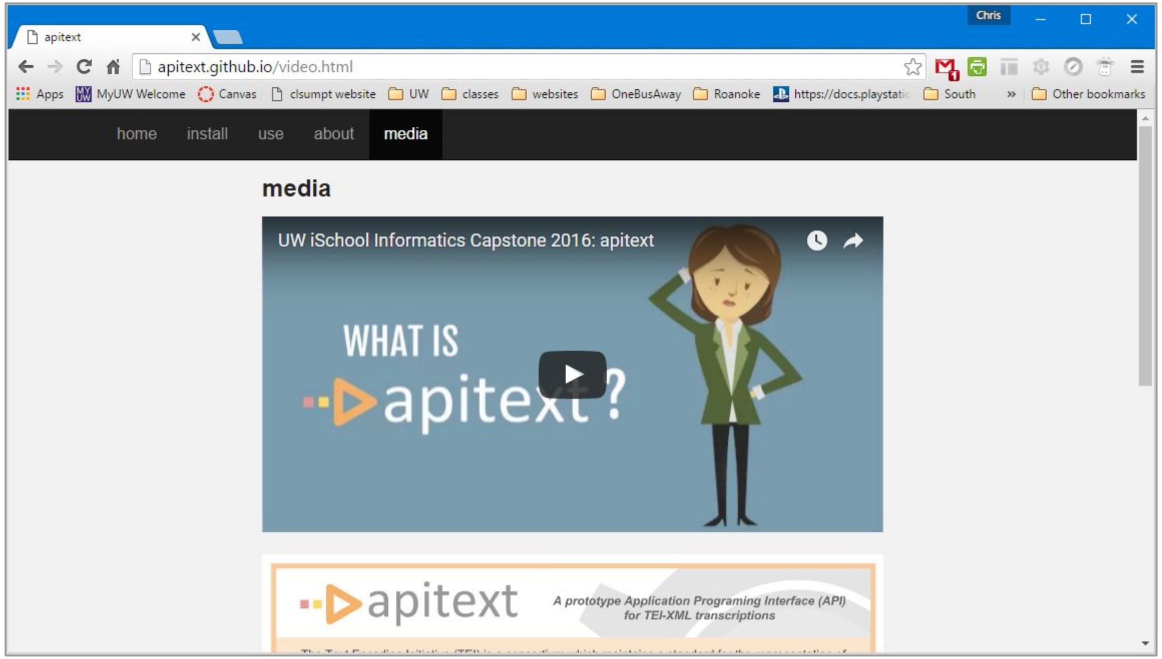
about the project

For over twenty years the Text Encoding Initiative (TEI) has managed and developed a set of encoding guidelines for the representation of humanities, social science, and linguistics -- to preserve and share -- texts in digital form. Using the Extensible Markup Language (XML) as its backbone, TEI is the generally accepted encoding model for the digital humanities. Due to XML's extensible nature, It can often be difficult to share these files and even more problematic to make them interoperable. The goal of our student project was to build a prototype Application Programming Interface (API) for TEI-XML documents that provides greater interoperability while at the same time is easy to use by an audience with backgrounds in the humanities, social sciences, and linguistics. Currently, we have produced an API that, once uploaded to a directory containing TEI-XML files, can return multiple views of that file using a Uniform Resource Identifier (URI) as its query method. In future iterations of our prototype we plan to: allow control over an unlimited set of TEI-XML files, provide a more robust catalogue of view types, support Cross-Origin Resource Sharing (CORS), support static API resource generation, and support TEI-XML schema validation.

The apitext API was directly inspired by this journal article written by Desmond Schmidt.

documentation website http://apitext.github.io/video.html


12:04
media
UW iSchool Informatics ...
WHAT IS
apitext?
apitext
A prototype Application Programing Interface (API) for TEI-XML transcriptions



future

Recommended Features

recommended features

Delivering a working prototype into the hands of both users and developers has opened the doors to new avenues of research and development. Future iterations of this API intend to:

- Provide a more robust catalogue of view types
    - The return of both formatted and unformatted “text” views.
    - When returning a listing of the contents of a “specific element”, the API should be able to:
        - Return both unique and multiple occurrence views.
        - Return unformated, formatted, HTML, and JSON views.
    - The return of element attribute metadata views.
- Allow control over an unlimited set of TEI-XML files
    - Currently, this version of the prototype is capable of controlling only one TEI-XML file.
    - Future versions will be able to control whole libraries of TEI-XML files.
- Support TEI-XML schema validation
    - This version of the prototype does not support schema validation; it assumes the provided XML document is well formed XML.
    - Future versions will be able to return information about whether or not the document is valid XML and if it is valid TEI.

recommended features (continued)

- Support static API resource generation
  - Currently, the API dynamically queries the TEI-XML file each time a request is made.
  - If a researcher has published their TEI-XML file, more than likely they will not be modifying that document on a regular basis.
  - Future versions should acknowledge this behavior and will:
    - Create static JSON files that will hold the data needed for requested views.
    - Be able to check for new versions of TEI-XML files and be able to react appropriately.
- Support Cross-Origin Resource Sharing (CORS)
  - Currently, clients of the API are required to be located on the same server as the API.
  - Future versions should include functionality that allows users from other domains access to the API.
    - Client domain registration support.
    - Documentation that would assist users but also educate users on the pros and cons of implementing this functionality.