Criterion B: Analysis

Proposed solution:

I discussed with my client the possibility of a browser extension (WebExtension) to add functionality to veterinary electronic medical record (EMR) web interfaces.

Requirement specification

IT system requirements

Hardware

- Home computer:
  OS: GNU/Linux (Manjaro 17.1-rc1 Hakoila)
  CPU: AMD A4-5300B APU with Radeon HD Graphics @ 2x 3.4GHz
  RAM: 8 GB
- Home network connection (measured using the Speakeasy Speed Test):
  Download speed: 9.8 Mbps
  Upload speed: 2.8 Mbps

Services

- Cloud storage:
  Google Drive: 15 GB of free storage (source: Google)
  Github: no fixed limit; files must be <100 MB, recommended <1 GB per repo (source: Github)
- Extension hosting and signing: AMO

Software

- Firefox 57.0.2
- Vim 8.0.1297
- Git 2.15.1
- Webpack for bundling JavaScript sources
- Babel.js, so I can use modern JavaScript features
- Mozilla web-ext for building and testing the extension
- Python 3 for running the mock server
- Assorted GNU utilities

System interaction

- The extension will be compatible with Firefox and Chrome, as my client uses both.
- The browser extension will be hosted on AMO and probably also the Chrome Web Store (I currently do not have a Chrome Web Store developer account and Google charges $5 to create one)
- As an alternative to simply installing the extension from AMO or the Chrome Web Store, extension bundles will be distributed along with source code and can be temporarily loaded by both browsers
- Clicking on the extension icon will reveal an interface that provides some of the simple tools specified by my client.

- This interface will also have a utility for searching PDF documents attached to the current record. Searching will be case-insensitive.
- The interface will have calculators, which are automatically filled with information on the page, but can also be used on any other site.
- The interface will have a button to open up an editor with a feature to auto-fill information from templates.
- The distribution will come with a mock site so that some of these tools and their integration with the EMR interface can be used without access to the actual site. The mock server will be bundled into a standalone executable using pyinstaller, which can be run without installing Python. The mocked site will be viewable by navigating to the printed url in the browser of choice.

Input/output requirements

Input requirements

- My client will provide me with templates for the editor. I will ask my client to create sample templates to get a clear idea of the required capabilities, specify a format for the templates that is similar, but more formal than the samples which fulfils these needs, and request further templates in this format.
- For utilities such as energy calculators, I will ask my client for formulas and possibly some test cases too.
- The extension will obtain patient records and data from the EMR site. I will create a mock site based on the behavior of the records page of the EMR site to test my extension. The mock site will mostly just serve static web pages and record files downloaded from a representative sample medical record on the real site, although it may also mock more complex queries on the site as well when desired for the extension implementation.

Output requirements

- Information about each patient should be concisely summarized in a tabular format.
- The document search should output urls of matching attachments
- The editor should provide suggestions based on keywords which will auto-fill information based on the templates provided
- The mock server should display output that closely mimics that of the EMR site.

Processing

- Medical record files of various formats will be parsed and data will be extracted for various purposes.
- Text will be extracted from the PDF documents and matched to a search pattern.
- Formulas for calculating things like energy requirements will be applied to extracted data.

Security

- Access to patient records must not be exposed. I will not press my client for login information to the EMR site, preferring to have my client sign in to the site for me when needed. For most of my testing, I will be creating a mock site based on information and data gathered from the site through an initial technical investigation. After this investigation, limited use of the site should be required until the project nears completion. These

measures will prevent privileged access to the EMR site, and thus all of my client's records, from being exposed to anyone -- even myself.

- Patient confidentiality is important. I will only view records of my pets, hospital pets, and sample records used by the hospital for training.
- Straying even farther from purely security concerns and into ethics, I will make an effort to ensure that the extension behaves correctly and does not silently make assumptions about abnormal data. In cases of ambiguous data, the extension should fail rather than silently use possibly incorrect data.

Specific performance criteria

Comprehensive unit-tests will be created for the extension, likely including automated in-browser tests with the Selenium web driver. These tests should pass and indicate reasonably high code coverage while we're at it. I'll list highlights from these tests here later on.

The user interface for all added functionality should be intuitive and consistent.

Justification of chosen solution

My client wants software to ease the process of working with EMRs. Of all considered methods to assist my client, a browser extension that adds functionality to the EMR site is the most obvious and useful. My client wants a variety of tools for common tasks encountered when working with EMRs. Some of these could be implemented as standalone programs, but this may be more difficult for my client to install and would separate the tools from my client's work. With a browser extension, the tools can be integrated into the page itself. Additionally, tools that require data from the EMR site would require login information to be input into the program and perform the login process itself. With a browser extension, the login process is the same for both normal access for the site as well as the added tools. I also have some prior familiarity with creating extensions for both Firefox and Chrome (albeit a while ago) and am experienced enough with programming methodology to feel comfortable working on this project as opposed to looking for other ways I can help my client.