Flight Scheduler

Proposal, Technical Project  
Plan #1  
to be presented on or before June 9th, 2018

Student: Scott Waldron  
In support of Bachelor of Computer Science  
Western Washington University

Client: Aran Clauson  
Email: Aran.Clauson@wwu.edu

Advisor: Dr. Aran Clauson  
Email: Aran.Clauson@wwu.edu

Table of Contents

[1. Introduction 2](#_Toc481350940)

[1.1 Problem Statement 2](#_Toc481350941)

[1.2 Background 2](#_Toc481350942)

[1.3 Needs Statement 2](#_Toc481350943)

[1.4 Objective 3](#_Toc481350944)

[2. Proposed Technical Approach 3](#_Toc481350945)

[2.1 Requirements 4](#_Toc481350946)

[2.2 Architecture Design 4](#_Toc481350947)

[2.3 Implementation Design 4](#_Toc481350948)

[2.4 Quality Assurance Plan 5](#_Toc481350949)

[3. Expected Project Results 5](#_Toc481350950)

[3.1 Measures of Success 5](#_Toc481350951)

[4. Schedule 6](#_Toc481350952)

# Introduction

Cadets and pilots seeking to schedule a flight through Civil Air Patrol (CAP) must communicate availability and request flights around specific times and dates. Cadets are paired with available pilots during both of their availabilities to complete a required sortie for the cadet. Broadcasting availability is currently restricted to individually logging available hours and having an administrator filter through the files to find potential matches. Flight Scheduler is a survey tool that collects data related to flight availability, and then stores and shares it with the CAP coordinator. Flight Scheduler I s intended to save time for both the coordinator and the cadets and pilots inputing their information. The survey is also designed to kickstart cadet and pilot communication with the coordinator. This application is designed to streamline the scheduling process allowing for quicker scheduling and more effective planning of sorties.

## Problem Statement

The problem of quickly and accurately scheduling a sortie affects both the cadet and the pilot by utilizing the time of both parties in an effective manner without too much spread between sortie dates. This could result in incorrect scheduling and/or mismatches between the cadet and pilot leading to an unsuccessfully planned sortie. The benefits of a streamlined survey and database collection service would not only result in more accurate and available data, but would also save time for the cadet, pilot and the coordinator planning the flights.

## Background

The existing system operates by requiring users to input their availability into the Web Mission Information Reporting System, or WMIRS. This can lead to inefficiency with planning as the availability data can be widespread through the site. Requiring cadets and pilots to enter credentials and navigate to the correct page can also lead to increased input times.

## Needs Statement

The most pertinent discrepancy in the existing system the the time required by the coordinator to collect the information to plan a successful sortie. The proposed flight scheduler will short-circuit this potential problem by providing the coordinator with a more organized and accurate set of availability data, allowing for more accurate schedules.

## Objective

A completed project will utilize an API to query the AthenaHealth database for relevant patient medical data, and insert the data into the appropriate TTYD survey question fields. The user will then have the option to review this data if it is outdated or otherwise erroneous. This will save the user from having to answer basic health questions by providing data from previous doctor visits, while ensuring the accuracy of supplied information.

# Proposed Technical Approach

AthenaHealth’s API program, titled *More Disruption Please*, is provided as a sandbox for small businesses to develop programs that interface with their medical data in an effort to expand availability of electronic health records. The interface uses a RESTful query structure that makes identifying and requesting patient data very straightforward. The documentation provides a list of example queries suitable for requesting the desired data for each survey question, which will then be stored in the survey via a PHP array prior to the user starting the survey. Once the user has started taking the survey, the answer to each question will be autofilled for them, only requiring validation of the data to continue.

Figure D-0 shows a use case defining how the API will fit between the TTYD survey and the AthenaHealth database.

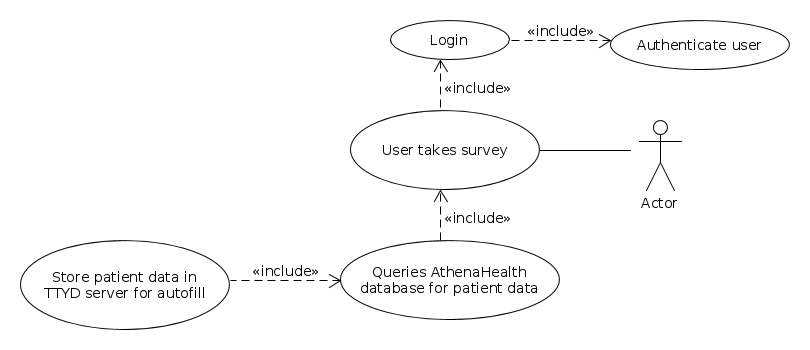
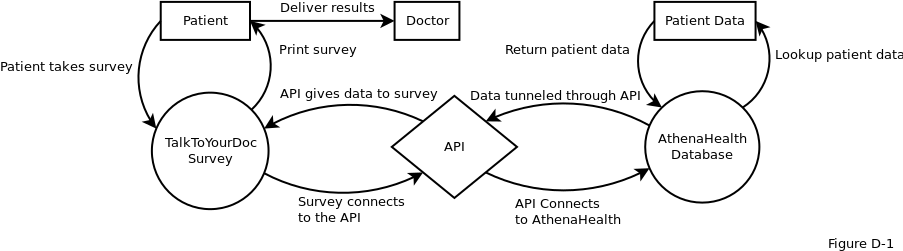


Figure D-0

## Requirements

|  |  |
| --- | --- |
| **Requirement** | **Solution** |
| 1. TTYD survey provides accurate medical data. | Working in conjuction with AthenaHealth’s *More Disruption Please* program we will have access to relevant patient medical data that can be autofilled into the survey. AthenaHealth has a unique patientID that will be used to identify the patient, and will never be visible beyond the scope of their firewall. |
| 1. TTYD survey saves time for patient and doctor. | By autofilling survey questions with an API, our client estimates that patients will complete surveys up to 50% faster than those who have taken the survey without the autofill feature. |

Figure D-1 is a diagrammatic description of how the API will connect to the AthenaHealth database and return data to the TTYD survey.



## Architecture Design

The Autofill feature will act as a liaison between AthenaHealth’s SQL database and the PHP TTYD survey in an attempt to gather answers to the questions before the survey begins. Relevant data that is found in AthenaHealth’s EHR database is then stored in a PHP array on the TTYD webpage and subsequently inserted into a temporary MySQL database. Once all queries have been executed, the data for each question will be automatically inserted into the answer text field as soon as the user reaches a new question page.

## Implementation Design

The *More Disruption Please* program provides skeleton queries that can be used to request patient medical data for each survey question. Queries for all possible autofilled survey questions will be compiled into a batch, and will be executed the moment a user logs in to the TTYD survey. Once the data has been received from AthenaHealth, it will be populated into a temporary MySQL database on the TTYD server, and catalogued in a PHP array. When the user proceeds to a survey question, the array will be referenced to provide a possible pre-formulated answer. If an answer exists as a result of the API query, the answer will be inserted into the answer textbox, or a checkbox will be marked as appropriate to indicate the desired result. The user will then be given the option to change the supplied answer or they can simply continue on to the next question. The process will continue checking the PHP array for all subsequent questions, autofilling as appropriate, until the survey is completed.

## Quality Assurance Plan

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk #** | **Risk:** | **Risk Outcome:** | **Risk Remediation:** |
| **1** | The project fails to be HIPAA compliant. | If the deliverable does not meet HIPAA requirements, legally it cannot be allowed to perform its expected functions and may result in lawsuits. | A basic look through the HIPAA regulations for any obvious sections that we should have knowledge on should help mitigate the possibility that we will not be compliant with HIPAA regulations. |
| **2** | Product does not meet customer needs. | If the product does not meet customer needs, our product will become obsolete and development will have been wasted on an irrelevant product. | Continuous communication with the customer will allow both parties to be aware of any changes or concerns. The earlier changes in product requirements can be identified, the easier it will be to solve them. |
| **3** | AthenaHealth API or schema changes. | Changes in AthenaHealth’s API or database schema may break our final product as our programmed database calls are no longer appropriate to pull the required data. | If we ensure the software is easy to update, any changes to the API or database schema could easily be applied to accommodate future development requirements. |
| **4** | User medical data may be intercepted by a man-in-the-middle attack. | If transmission of personally identifiable medical information is intercepted, then TalkToYourDoc would be liable for violating HIPAA regulations. | The software will be placed inside the AthenaHealth firewall, which eliminates the transfer of personally identifiable information through the internet. |
| **5** | Autofill logic incorrectly answers survey questions. | Autofilling incorrect medical data into the TalkToYourDoc survey could lead to incorrect recommendations and/or failure to recommend hormone replacement therapy to a patient. | Rigorous testing will ensure that our program will perform as intended. As a failsafe, there will be visible warnings on all survey questions that request the patient to verify all autofilled data for correctness. |

# Expected Project Results

Upon completion of the project, our client can expect the following results:

1. The TTYD survey will successfully connect to the AthenaHealth database, and will automatically populate answers to all possible survey questions.
2. Time taken to complete the survey will be reduced by providing some answers for the user.
3. User and doctor satisfaction will rise due to increased survey efficiency.

## Measures of Success

As per our client, the project will be considered a success if the following criteria are met:

1. Survey provides accurate patient medical data for each survey question.
2. Data queried from the AthenaHealth database is relevant to the current user.
3. The process of autofilling survey results in time savings for overall survey completion.

# Schedule

|  |  |
| --- | --- |
| **Deliverable** | **Due Date** |
| API queries completed. | 5/7/2017 |
| PHP survey insertion completed. | 5/14/2017 |
| Autofilled data fully tested for accuracy. | 5/21/2017 |
| Client approval of autofill implementation. | 5/28/2017 |
| Handoff of project to AthenaHealth for testing behind the firewall. | 6/5/2017 |
| Bugs worked out and project finalized. | 12/8/2017 |
| Final project handed off to client. | 12/15/2017 |