**Senior Design Final Report**

PDF Web Viewer / Filter

 ****

Version 1.0 - 03/11/21

Team Members:

Cristian Corrales

Chandel Buelna

 Phoebe Castanedo

 Brandon Gonzalez

 David Sanchez

 Joaquin Robles

 Sean Ybarra

 Roman Arias

Ares Ton-That

Faculty Advisor:

Keenan Knaur

Liaisons:

Ashwariya Chopra

**Table of Contents**

1. Introduction 3

1.1. Background 3

1.2. Design Principles 4

1.3. Design Benefits 4

1.4. Achievements 4

1. Related Technologies 4

2.1. Existing Solutions 5

2.2 Reused Products 5

1. System Architecture 5

3.1. Overview 5

3.2. Data Flow 6

3.3. Implementation 8

1. Conclusions 9

4.1. Results 9

4.2. Future 9

1. References 11

**1. Introduction:**

**1.1. Background:**

 QTC is a medical organization that uses advanced technology to provide state-of-the-art medical services to several government-owned agencies. QTC has an extensive network of licensed medical providers who are able to provide professional assistance to clients over the phone. QTC’s primary goal is to provide a quality service to their clients in a timely manner. In light of this, they suggested that our senior design team develop a new software tool to replace their current solution, helping save time and resources.

Currently, QTC views and saves their documents in a PDF format which are stored in a database. The current solution is a web application which allows these files to be viewed all at once as opposed to one at a time. This is accomplished by combining many smaller PDF files into one large PDF file behind the scenes, then opening that large file in the viewer. Although this slightly speeds up the process of viewing many files, this creates a storage issue as these larger PDFs are often many gigabytes in size. Due to these drawbacks to their current viewer, our team was to design our PDF viewer with a storage-efficient way to seamlessly scroll through files. It was also required to filter out different types of PDFs from within a database and store their locations in a quick, temporary, and easily accessible format that would allow them to be viewed in one continuous stream without the need for such large amounts of storage space. Additionally, they wanted a search feature constructed to allow users to search for words or phrases within the list of filtered PDF documents.

 The PDF Viewer is a web page containing the PDF viewer itself, a category search filter, and a PDF text search feature. Our team used QTC’s ultraviolet color theme as well as their design standards to create a user interface that is consistent with the appearance of their other software applications. At the top of the web page is the category search filter, which consists of a list of categories that can be used to limit the types of PDFs that will be displayed within the viewer. The filter provides users with the ability to search one or more categories in a single search. When a category search has been made, only PDFs related to any of the searched categories will be displayed in the viewer. If all categories are selected, every PDF within the database will be displayed.

Once the user has filtered by a category, the viewer will display the first PDF in a list of PDFs that are being temporarily stored in a local memory location, this allows users to switch between PDFs easily by using the scrollbar or the previous and next buttons. When a PDF is being displayed, a user can search for information within the PDF by performing a text search. The user can type any word or phrase into a text box at the bottom of the page and the viewer will try to locate the next instance of that word or phrase within the current PDF. Once it finds a match, the viewer will automatically display the section of the PDF where the word or phrase is located. Any matches that are found will automatically be highlighted so that the user can quickly find the information that they are looking for. If the keyword being searched for does not exist within the current document, the user will be notified that no matches were found. They can cancel the search, or proceed to click the find previous or find next buttons to search the previous or next PDF for a match.

**1.2. Design Principles:**

 The design for our web application aims to be extensible, maintainable, and user intuitive. The goal is for QTC developers to easily expand on the application and eventually launch our application for QTC employee use. QTC provided our group with coding standards and project set-up which we followed while creating the application. The standards included how our project should be structured, ensuring that QTC can more easily work with our application in the future. The user interface for our application should be user intuitive and should not appear cluttered. Since the aim for our application is to eventually launch and replace QTC’s existing viewer, the user interface should be simple for QTC employees to use and to adjust to.

**1.3. Design Benefits:**

By following the architecture provided in QTC’s coding standards and project set-up, we provide QTC developers the ability to easily expand on the application. QTC developers will be able to integrate their database or add future needed functions for the application and be within a project structure the company is accustomed to. This will allow for a faster deployment time should QTC choose to use the application to replace their current solution, which is not storage-efficient and much slower due to reasons expanded on in Section 2.1. The user interface is designed to be simple and easy to use. This allows a QTC employee to easily use the features provided in the interface such as filtering through the PDFs or searching for keywords within the PDFs. By keeping the interface simple, we hope it will allow QTC employees to easily adjust to the new PDF viewer.

**1.4. Achievements:**

 Our group has accomplished developing a PDF viewer that will hopefully assist QTC and their employees for years to come. Our application allows users to view PDFs in a continuous manner by retrieving them from a database and displaying them to the user. Users will also be able to filter through the multiple PDFs by selecting a category or many categories. Users will also be able to search keywords within the contents of all the PDFs in the viewer. All of this is done on a single easy to use web page ensuring that QTC employees will be able to swiftly adjust to their new viewer. The application has been developed in a way that QTC developers can easily add more features in the future if needed without having to adjust the main features of the application.

**2. Related Technologies:**

**2.1. Existing Solutions:**

There are many existing web PDF viewer applications. Adobe Acrobat is an excellent PDF viewer that allows its users to view, annotate and even collaborate on altering a PDF. Although it is an excellent PDF viewer it does not provide the necessary functions that QTC is looking for. The major function it lacks is the scrolling transition from one document to the next without the manual intervention of selecting each specific file to view. Other functions it does not provide include sorting the documents into categories, and the persistent text search across all documents which omits those without the desired keyword.

QTC currently also has a viewer being used within the company. Currently, QTC’s solution is combining their individual PDF files into larger files to create the effect of seamlessly scrolling through multiple files at a time. Creating these huge files and traversing them creates a problem in both time and resources. It can take a lengthy amount of time to find one specific document in a document consisting of hundreds. This process also takes a huge amount of storage space as the combined documents can reach a size of up to 2 gigabytes of essentially redundant data.

**2.2. Reused Products:**

 The application is developed using C#, JavaScript, ASP.NET, SQL Server Management Studio (SSMS), and Atalasoft’s DotImage SDK. C#, ASP.NET, and SSMS in specific are used for the project because they are designed by Microsoft to be integrated to work together among one another. This makes web development intuitive and flexible considering we will have the software written in C# communicating with the SQL server. Atalasoft’s DotImage SDK comes equipped with an embedded PDF viewer that has the function of displaying pages continuously in a single scrollable form and is the prominent form of existing software components we will reuse.

**3. System architecture:**

**3.1. Overview:**

The architecture for the QTC PDF Web Viewer can be broken down into 3 large overarching components. These are the Users, the Web Application itself, and the Database to which it is linked, with the second component being the most complex. Here we have provided a DFD Level 0 Diagram which illustrates how these large project components interact with each other.

* **The Users (QTC Employees):** This project was developed with the viewpoint of a QTC employee in mind, as it is intended to replace their current in-house solution for PDF viewing. As such, all requirements were drafted, finalized, and implemented for the purpose of satisfying what QTC employees would require from this application.
* **The PDF Web Application:** This complex component comprises several modules, which include the interface, the client side, and the server side. However, this large module as a whole is the combined implementation of each requirement. Note that the purposes and interactions of each component will be expanded on in the subsequent section of this document.
* **The Database:** Using a Microsoft SQL Server platform, the database holds all relevant information pertaining to the PDF files accessed by the Web Application. This includes file paths, file categories, folder names, and more.


**3.2. Data Flow:**

Herein the inter-module communication within the Web Application will be explained further with a description of each module’s purpose. Implementation will be detailed in the following section. For this purpose our DFD level 1 of the Web Application is provided below for reference. Interactions are labeled, as well as which module fulfills which requirement.



**3.2.1 Client Side** / The client side consists of JavaScript files that are responsible for initializing new document viewer and document thumbnailer objects as well as implementing the correct object configurations. Within these files, the Atalasoft SDK libraries are referenced and the default Atalasoft viewer is manipulated to satisfy our design requirements. These files have direct control of the viewer, thus they are in charge of any user-to-viewer interaction. The client side essentially satisfies the text-specific search requirement, and the continuous displaying of files requirement.

**3.2.2 Front End** / The Front End is the HTML files that serve up the UI containing the viewer and control bar for category and keyword searches. All logic is obscured behind this web page, which is the only page the user may access and interact with. It communicates directly with the Server side to retrieve PDF file data as needed.

**3.2.3 Server Side** / The Server Side is a set of controller classes that are implemented within the standards of MVC architecture. They are responsible for the satisfaction of the category filter requirement as they respond to user input retrieved from the category filter drop down within the Front End. The controller classes retrieve the desired categories that the user has selected and will call the corresponding stored procedures to fire up SQL queries within the database and retrieve the results to be stored in the view models.

**3.2.4 View Model** / This is a model class implemented within MVC architecture standards. These models are to store the data initially retrieved from the queries to the database, then further processed into strings that the Front end and Client side can access.

**3.2.5 Atalasoft Libraries** / The Atalasoft libraries are the SDK we are using. They define all the namespaces, classes, and objects used in the HTML and JS files. They also provide an abundance of event triggers for the document viewer and thumbnail viewer so we can check to see when changes have occurred and implement functions for those events. The library also has predefined variables that we can access for conditional statements.

**3.3. Implementation:**

The project was split into 3 distinct components: the UI/UX, the backend, and the database/data management, to allow for a modular implementation process.

**3.3.1 UI/UX**

The UI of this project was designed to comply with QTC’s UI standards. It consists mainly of the Details.CSHTML file (3.2.2), which is styled with Bootstrap. Its purpose is to present two major functionalities to the user, those being the Document Viewer, and keyword search and category filter functionality. The functions it contains are bound to triggers provided by Atalasoft that enable it to display a seamless scrolling view of the PDF files.

**3.3.2 Backend Logic**

This is a large module which contains the entirety of both the client side and the server side. Starting with the client side (3.2.1), variable and function initialization is handled by the viewer-hadler-script.JS (viewer, thumbnail, and trigger function initialization) and common-config.JS (global variable initialization). While the program is running, page-change.JS is in charge of loading PDFs into the viewer, changing the PDFs when appropriate (when the user scrolls or clicks “Next”) and is responsible for the functionality of the thumbnails. The search-function.JS file is in charge of the text-specific search functionality. As for the server side (3.2.3), the PdfViewerController.CS component of this module communicates continuously with Front end to receive user input regarding the category filter. It translates this input into the correct database queries, storing the results in the View Model (3.2.4) within this module, then serving it back to the Front end for further processing.

**3.3.3 Database/Data Management**

Our backend logic was designed to interface with a database running on Microsoft SQL server framework. This database consists of several tables, containing file metadata, file locations, document-keyword association, and document-category association as well as containing several stored procedures. The PDFViewerViewModel will call these stored procedures to retrieve the relevant file paths, enabling their display in IndexView. This database is fully obscured from their user and cannot be modified or directly accessed in any way from within the QTC PDF Viewer Web App.

**4. Conclusions:**

**4.1. Results:**

Our team has developed a PDF viewer that will greatly expedite the process of finding information within PDFs by using a category search filter and a text search box. The search filter initially limits the types of PDFs that will be displayed within the viewer. The category filter drastically speeds up the access time of PDFs and slightly reduces computer resource consumption. The search box allows users to search the current PDF for any word or phrase that has been typed into the box. Once a match has been found, the user will automatically be taken to the section of the document containing the match. If no match is contained within the current document, the user can quickly search the previous or next document for a match by using the find previous and find next buttons.

Our user interface design team made the web-application user-friendly by organizing the different components of the application into different sections. The category search is at the top, with the viewer in the middle of the page and the search box at the bottom. Additionally, the design team used QTC’s ultraviolet color scheme to give the application a design that bears great similarity to other QTC web applications. Furthermore, the team provided several navigation buttons at the bottom of the page to make it easier to transition between PDFs.

**4.2. Future:**

 The PDF Viewer is an application that is somewhat similar to other PDF viewers like Adobe Acrobat or QTC’s PDF Viewer. In fact, the viewer module and much of its functionality uses Atalasoft’s library which QTC currently uses. However, the team also created several features that QTC’s current viewer didn’t have, such as the category search filter and navigation buttons. Because our PDF Viewer was designed for private use by QTC, only QTC will have access to it. Thus, only QTC’s software teams will be able to add new features or improve current features.

 A few new features that could be implemented are editing and printing PDFs. Editing PDFs is a useful tool provided by some companies like Adobe. Unfortunately, our PDF viewer does not provide this tool as all of the displayed PDFs are read-only. Printing PDFs is another useful feature that can be developed in future versions of the application. As of right now, printing would only print the web page as well as whatever page of the PDF the user is on, rather than printing the current PDF.

 Improvements on some of the currently implemented features would be to add more ways to filter through the documents. Such as filtering them by folders that the files may be inside of. Another would be to display which category or categories each document falls under. Currently, there is no way to tell when the list of documents has a change in category.

**5. References:**

Atalasoft DotImage documentation: <https://www.atalasoft.com/docs/dotimage/docs-concept/AtalasoftDotImageDevelopersGuide_EN.pdf>

SQL Server documentation: <https://docs.microsoft.com/en-us/sql/sql-server/?view=sql-server-ver15>