**C868 – Software Capstone Project Summary**

**Task 2 – Section A**



|  |  |
| --- | --- |
| **Capstone Proposal Project Name:** | FS InfoCat – File System Information Cataloging Tool |
| **Student Name:** | Leonard T. Erwine |

Table of Contents

[**Business Problem** 2](#_Toc83605517)

[**The Customer** 2](#_Toc83605518)

[**Business Case** 3](#_Toc83605519)

[**Fulfillment** 4](#_Toc83605520)

[**Existing Gaps** 4](#_Toc83605521)

[**SDLC Methodology** 5](#_Toc83605522)

[**Requirements** **Validation** 5](#_Toc83605523)

[**Planning and Design** 6](#_Toc83605524)

[**Deliverables** 6](#_Toc83605525)

[**Deliverables for Pastoral Staff and Stakeholders** 6](#_Toc83605526)

[**Technical Deliverables** 7](#_Toc83605527)

[**End-User Deliverables** 7](#_Toc83605528)

[**Implementation** 7](#_Toc83605529)

[**Software Implementation and Concurrent Testing** 7](#_Toc83605530)

[**Validation and Verification** 8](#_Toc83605531)

[**Environments and Costs** 8](#_Toc83605532)

[**Programming Environment** 8](#_Toc83605533)

[**Environment Costs** 9](#_Toc83605534)

[**Human Resource Requirements** 9](#_Toc83605535)

[**Project Timeline** 9](#_Toc83605536)

[References 11](#_Toc83605537)

# **Business Problem**

**The Customer**

Centreville Baptist Church (CBC Staff, 2021) is active with an extensive community support ministry whose staffing predominantly comprises part-time and volunteer positions. In addition to the humanitarian services provided to the community, the audio/visual team embraces the newest technologies, making full use of the church’s IT assets. The services provided by volunteer personnel tend to be transient but still very valuable to the success of the overall goals of the local religious community. As technology plays an increasingly vital to even the most menial of tasks, the support staff is continually working to ensure that IT resources are properly maintained and available to all participants. Inevitably due to the part-time nature of the work schedules, there is not always enough time to keep up with everything. In addition, as systems degrade due to inattention, IT staff frequently finds themselves reacting to incidents rather than proactively preventing them. The management of storage devices and their content is one area that often presents problems. Many times, the A/V team would like to re-use material that someone had previously created. However, they cannot always locate the files because the original creators are not available for consultation. Additionally, the employment of new technology increases the demand for storage space, leading to the depletion of free space on shared drives.

The pastoral staff worked with the IT staff to formulate policy and governance on the usage of storage devices but found it cumbersome to verify or enforce due to the transiency of most participants. They determined that instituting an IT training regimen would not be beneficial in most instances. If volunteers were required to complete training sessions before utilizing IT equipment, it would detract from the time spent on their humanitarian work, which is the church’s primary goal.

## **Business Case**

FS InfoCat will scan files distributed across multiple storage locations, devices, and media, creating a catalog of characteristics about each discovered file. The information in the database can then be searched and analyzed to look for specific patterns or values which may expose unwanted files. More specifically, this software will assist with the mitigation of shared network drive overuse by identifying possible file duplication or files with a size that has exceeded a specific threshold. Not only does it compare by file size, but it also calculates cryptographic hash codes of files that share the same length. Thus, files with the same size and cryptographic hash are highly likely to be exact copies of one another. Additionally, the software offers a way to view file metadata according to their file type, allowing users to identify redundancies even if the file sizes differ. Finally, the user can exempt files and subdirectories when redundancy is permitted or notate them for further action.

A pastoral staff member initially objected to the prospect of using the software on local workstation hard drives, citing the risk that confidential information could be compromised if metadata gets published to the shared database. This risk was mitigated by separating the data gathering process from the publication process. The application will initially store information to a local database within the user’s windows profile. It will only get published to the centralized database when the user explicitly opts to do so. More specifically, while the software application can share information with a centralized database, it will not depend upon it and function independently. Additionally, users can configure the software to skip specific files and folders, and they can specify multiple configurations that only scan particular folders.

## **Fulfillment**

FS InfoCat will be a windows desktop application built on the Windows Presentation Foundation (De George & Coulter, 2021) framework. It will leverage the Windows Property System (Schofield, Batchelor, Sharkey, Coulter, & Satran, 2018) to extract metadata from files, such as author names, audio titles, and video frame rates. Since not all devices support the Windows Property System, the application allows people to provide relevant information for each file manually

# **Existing Gaps**

Shared network drives used by the staff and volunteers alike lack sufficient structure. The cost of file storage steadily increased, but the value to the business was not proportionate due to the wasted space. There were many instances where a staff member had to re-create something simply because nobody knew where the person who did the original work saved all their files. They also found themselves periodically running out of storage space, causing work stoppage or lost data. Manually cleaning up the common storage areas may be as costly as the price of increasing storage capacity. Due to the uncertainty of the actual content within the shared drives, it is impossible to assess the cost in person-hours, should they decide to go through each file to determine what can be consolidated or discarded. It is also conceivable that the pace of unmanaged data contributions could outpace the mitigation efforts. Even if they did reduce the storage footprint, reducing the cost of file storage services will not solve the problem of recalling specific stored files when needed.

The original intent was to produce an application in the first iteration that could synchronize with a centralized database. However, in the interest of utilizing the software sooner, the first release will not include this capability and will be deferred to the next release. However, in this state, it is still viable for gathering and analyzing information locally.

# **SDLC Methodology**

I will design this software using a modified Waterfall development approach. The main reason for using this method is that there will most likely only be a single developer. Another factor contributing to this decision was that none of the stakeholders think they will consistently have enough free time to participate in Scrum meetings. Additionally, the capabilities required in the first release would require much more time than I could fit within a typical Agile Sprint.

## **Requirements** **Validation**

The development process will begin with validating the requirements gathered in the initial meeting with the customer. In doing so, I will explicitly state all assumptions to be sure that I understand customer expectations. Then, to capitalize on the advantages of the Agile disciplines, I will organize the use cases and requirements in a GitHub Project (About projects (beta), 2021). From there, I will define the structure for the data models. Then, using several detailed diagrams, I will depict the design of the user interface to include how each part of the user interface corresponds with database entities.

## **Planning and Design**

Next, I will define the software validation process using three stages: Automated testing, scripted developer testing, and user acceptance testing. Automated unit tests are created synchronously with the development process to validate each component as I implement them and expose potential bugs. The purpose of the scripted developer testing is to ensure that it is fit for use. I will script the tests to ensure all features within the graphical user interface produce the intended result. Finally, the church’s system administrator and some audio/visual team stakeholders will participate in the user acceptance testing. The user acceptance testing will ensure that the software meets its intended purpose as understood by the customer. I will submit the entity model, user interface design, and testing plans to the customer for final scrutiny before I begin the implementation phase

# **Deliverables**

## **Deliverables for Pastoral Staff and Stakeholders**

Before initiating the implementation phase, I will prepare a requirements document that enumerates the use cases, expectations, and parameters set forth during the requirements gathering phase. Additionally, I included the entity models, user interface diagrams, and testing plans created during the design process. Finally, each test will need to indicate a way to measure performance or a metric of success.

## **Technical Deliverables**

Upon completing the implementation phase, I will finalize technical instructions on deploying, configuring, and monitoring the software. The system administrator will validate these technical instructions during the first part of the user acceptance testing phase. Then, at the start of the user acceptance testing, I will deliver the technical documentation and the software binaries to the system administrator.

## **End-User Deliverables**

I will create documentation that will be accessible to the end-user on the features and intended usage of the software. The user documentation can be a mixture of in-application documentation and tooltips, as well as online documentation. I will also produce a documentation file that contains any information presented in online documentation as a fallback when online resources are unavailable.

# **Implementation**

I will use Visual Studio 2019 (Lee, et al., 2021) as the development platform because it is better suited to using Windows operating system APIs than other platforms. This way, by using an OS-native platform, I will decrease the likelihood of unforeseen complications. Additionally, I will use .NET Framework 5.0 (Dykstra, et al., 2020) because that was the latest framework which I could use that still was able to use the Windows Property System from managed code.

## **Software Implementation and Concurrent Testing**

During the implementation phase, I will use test-driven development (What is Test Driven Development (TDD)?, 2021 ) techniques for the components that are critical to reliability or security. Additionally, code segments that are difficult to monitor or validate using event tracing or other debugging means will be validated using unit tests. For the test-driven components, I will write the tests before implementing the code. After the code has passed all unit tests, I will configure the unit tests to automatically run with every build, to ensure additional changes produce no adverse effects. Otherwise, I will carry out the scripted developer testing after implementing each target user interface element. I will also re-execute scripted developer tests at the culmination of the implementation phase

# **Validation and Verification**

The final testing phase will have two parts. The first part will be conducted with the system administrator to ensure that it can be installed and executed in the target environment. Upon the second part of the testing phase, we will install the software on the workstations of user who are designated to be testers and will evaluate the software. Testers will have a predetermined series of actions to invoke within the application and verify the expected result.

# **Environments and Costs**

## **Programming Environment**

All development work will be carried out on my personal laptop, and no additional hardware will be required. I will use Visual Studio 2019 (Lee, et al., 2021) as the integrated development environment because it is better suited to using Windows operating system APIs than other platforms. This way, by using an OS-native IDE, I will decrease the likelihood of unforeseen complications. Additionally, I will use .NET Framework 5.0 (Dykstra, et al., 2020) because that was the latest framework which I could use that still was able to use the Windows Property System from managed code. SQLite will be used as the database engine. I will need access to a PaaS cloud service so I can create remote file shares for application testing.

## **Environment Costs**

Licenses for Visual Studio is based upon a subscription at 467 per year (Microsoft Visual Studio 2019 Professional - License - 1 User, 2021). This will be required for initial development. The Community Edition of Visual Studio is free and will be sufficient for post-deployment support. File storages services with 1 TB capacity through Azure is estimated to be $108.31 per month. This will only be needed until development is complete.

## **Human Resource Requirements**

It is estimated that it will take cost about $37,026,000 for the services of a single developer to complete implementation within 18 weeks, whereby the current median annual software developer salary of $107,000 averages out to $2057 per week. The cost associated with the System Administrator’s time is approximately $1100 for periodic attention being given to deployment. The cost of five testers is about $8170 for 5 days of testing. This brings the estimated cost of human resource requirements to 37,035,270.

# **Project Timeline**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Phase | Milestone/Task | Deliverable | Description | Dates |
| Requirements Gathering | Task 1 | Requirements | Meeting with customer and procedure review | 6/1/2018 – 6/8/2018 |
| Design |  | Low fidelity wireframe  High fidelity mockup | Design abstract data entities. Create low-fidelity mockups followed by high fidelity renderings. | 6/8/2018 – 7/1/2018 |
| Data Layer |  | Data layer implementation. | Demonstrably functional entity models and database | 7/1/2021-8/1/2021 |
| UI Layer | Task 2 | Beta version of app | Produce working application. | 8/1/2021-9/1/2021 |
| Deployment Validation |  | Complete deployment package | System administrator validates deployment instructions | 9/1/2021-9/8/2021 |
| User Acceptance Testing |  | Test Results | Test user evaluate software. | 9/8/2021-9/15/2021 |
| Release and Deployment |  | Release version of software | Release version of Application installer. | 9/15/2021 |

# References

*About projects (beta)*. (2021). Retrieved from GitHub: https://docs.github.com/en/issues/trying-out-the-new-projects-experience/about-projects

CBC Staff. (2021). *Home*. Retrieved from Centreville Baptist Church: https://centrevillebaptist.org/

De George, A., & Coulter, D. (2021, Jan 14). *Desktop Guide (WPF .NET)*. Retrieved from Microsoft Docs: https://docs.microsoft.com/en-us/dotnet/desktop/wpf/overview/?view=netdesktop-5.0

Dykstra, T., Warren, G., Jawahar, T., Wagner, B., Sharkey, K., Victor, Y., . . . Lander, R. (2020, Nov 16). *Introduction to .NET*. Retrieved from Microsoft Docs: https://docs.microsoft.com/en-us/dotnet/core/introduction

Lee, T. G., Petersen, T., Anand, M., Martens, J., Hogenson, G., Parente, J., . . . Robertson, C. (2021, Sept 14). *Welcome to the Visual Studio IDE*. Retrieved from MicrosoftDocs: https://docs.microsoft.com/en-us/visualstudio/get-started/visual-studio-ide?view=vs-2019

*Microsoft Visual Studio 2019 Professional - License - 1 User*. (2021). Retrieved from Newegg: https://www.newegg.com/microsoft-visual-studio-2019-professional-license-1-user/p/N82E16832350246?nm\_mc=KNC-MSNSearch&cm\_mmc=KNC-MSNSearch-\_-Software+-+Software+Application+License-\_-Microsoft-\_-32350246&source=region&msclkid=683d24e2001010e4d25d45b363070

Schofield, M., Batchelor, D., Sharkey, K., Coulter, D., & Satran, M. (2018, May 31). *Windows Property System*. Retrieved from Microsoft Docs: https://docs.microsoft.com/en-us/windows/win32/properties/windows-properties-system

*What is Test Driven Development (TDD)?* (2021 ). Retrieved from Agile Alliance: https://www.agilealliance.org/glossary/tdd