Danfoss Enterprise Layered Process Auditing

Final Report

May 15-02

Zach Carlson

Jamie Countryman

Mitch Valenta

Table of Contents

Project Abstract

The current paper-based auditing process used by Danfoss lends itself to time-consuming or incomplete audits with too many possible points of failure and little to no accountability. Simplifying this process will allow audits to be completed quicker with more flexibility, creating a safer workplace and higher quality work output. This auditing system is a web application to be used by both PC and iPad devices, allowing for easy access on the shop floor.

Key Terms

Active Directory
A Microsoft domain service that provides user authentication, among other things.

Area
A section of the shop floor dedicated to part of the manufacturing process.

Audit
A series of questions whose answers represent the current state of an area, such as safety and cleanliness.

Auditor
An employee who answers questions in an audit, assessing the state of an area.

Entity Framework
An object-relational mapping framework created by Microsoft for use in applications developed using the .NET framework. Object-relational mapping systems map objects in a relational database to logical objects in the programming language (in this case, C#).

MVC
An acronym for Model-View-Controller, a popular design pattern for developing modular user interface-based software such as web applications. Specifically, Microsoft maintains an MVC framework for ASP.NET to assist in development using this pattern.

WCF
An acronym for Windows Communication Foundation. WCF provides a framework for creating a web service for fetching data, allowing separation of database and presentation layers.

Web Forms
A event-based framework for creating web pages in ASP.NET.

System Requirements

Functional

- The Danfoss Auditing system shall provide a process to complete and answer audit questions.
- The Danfoss Auditing system shall provide a process to create audits with editable questions, categories and hints.
- The Danfoss Auditing system shall provide a process to create audits from a standard template audit.
- The Danfoss Auditing system shall provide a process to assign auditors to audits/areas.
- The Danfoss Auditing system shall automatically schedule auditors with defined rotation.
- The Danfoss Auditing system shall automatically send email reminders of upcoming or overdue audits.
- The Danfoss Auditing system shall produce audit results and statistics and history.
- The Danfoss Auditing system shall automatically recur audits with assignment to auditors.
- The Danfoss Auditing system shall require users to log in using Active Directory credentials.
- The Danfoss Auditing system shall display a category for every audit question.
- The Danfoss Auditing system shall display a link to Standard work for Standard work audit questions.
- The Danfoss Auditing system shall save audit answers after each audit question.
- The Danfoss Auditing system shall provide a process to maintain roles for users.
- The Danfoss Auditing system shall display percentages of yes and no answered results for a completed audit.
- The Danfoss Auditing system shall include a statement that displays who last edited an audit.
- The Danfoss Auditing system shall display a list of pending and completed audits upon logging in.

Non-functional

- The Danfoss Auditing system must maintain a responsive design to display correctly in both tablet and desktop browsers.
- The Danfoss Auditing system must use bar graphs of audit results and score.
- The Danfoss Auditing system must have one audit question per web page while completing an audit.

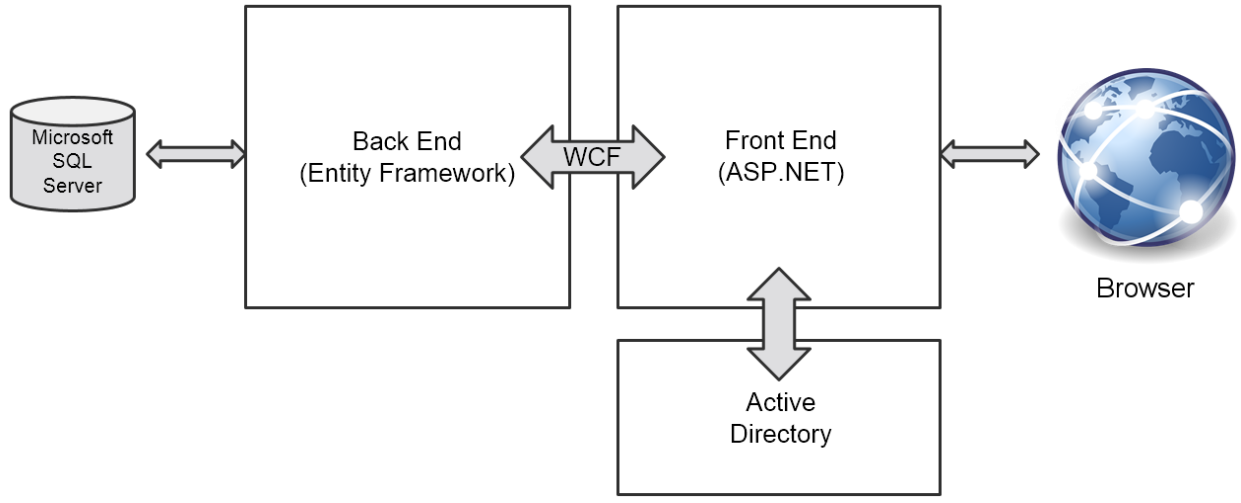
- The Danfoss Auditing system must have yes or no answer and a comment section for every audit question.
- The Danfoss Auditing system must have a tooltip hint for every audit question that provides helpful information on where to find the answer.
- The Danfoss Auditing system must provide navigation buttons for each audit question.
- The Danfoss Auditing system must track individual audit questions.
- The Danfoss Auditing system must change the color of the yes answer button to green when selected.
- The Danfoss Auditing system must change the color of the no answer button to red when selected.
- The Danfoss Auditing system must distinguish a separate page for administrative operations.
- The Danfoss Auditing system must distinguish a separate page to display results for a completed audit.
- The Danfoss Auditing system must be able to display text in different languages.

Design

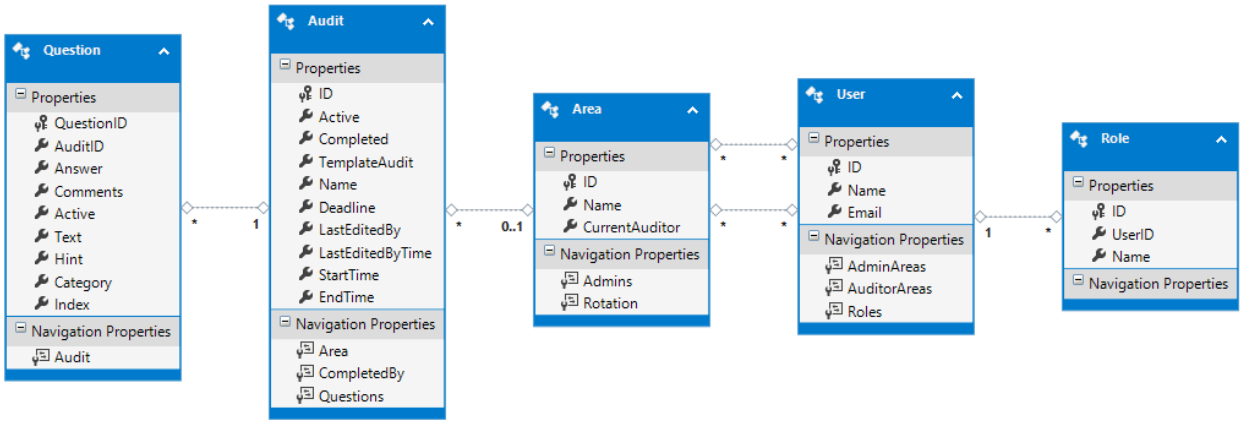
Specifications

The auditing system must work for both desktop and mobile browsers. This allows for mobile completion on the shop floor as well as administration from the office. The system should be created using an ASP.NET front end with an Entity Framework back end interfacing with a Microsoft SQL Server database.

Block Diagram



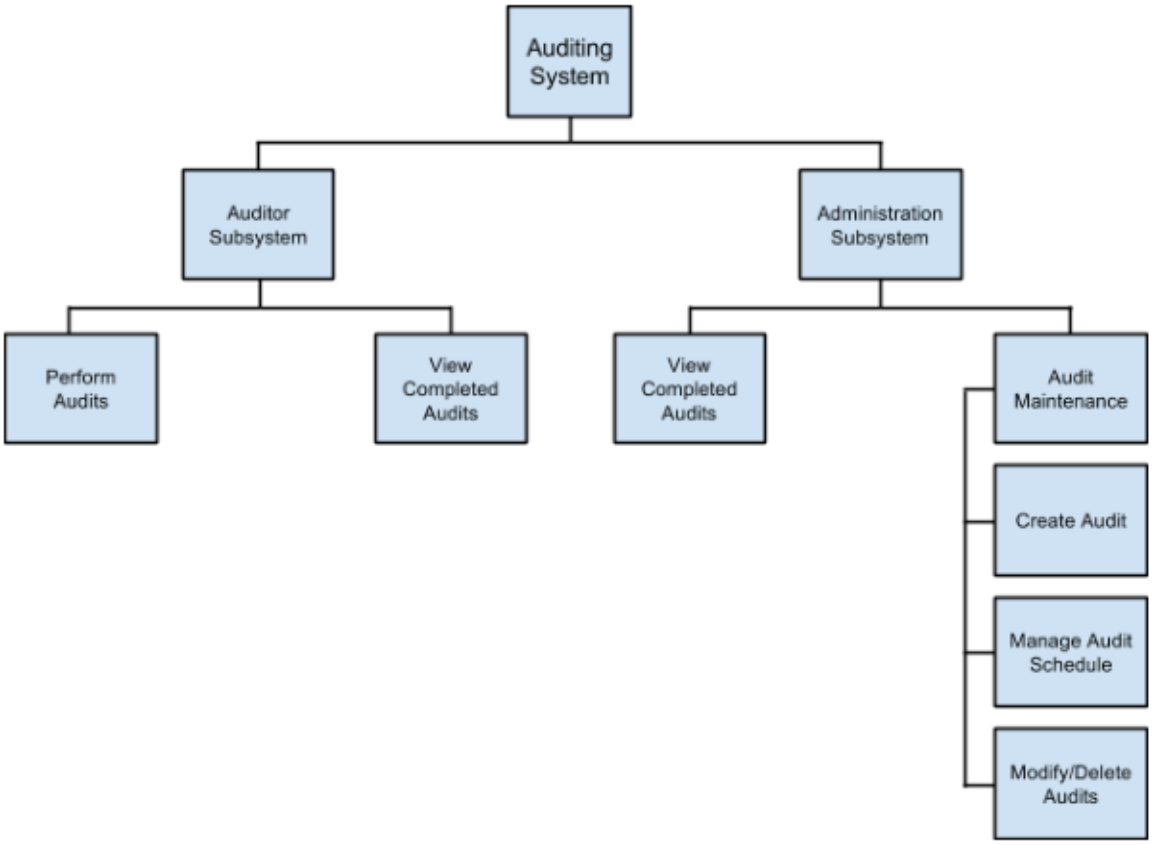
Database Schema



The database contains five tables. Starting on the left, the Question table contains questions along with their answers and comments. Questions are assigned to audits, represented by the Audit table. This table stores the name, deadline, active/completed states, and tracks the times at which the audit is started, finished, or edited, along with the user who performed those actions. The Area table maintains links to Audits and Users, defining a rotation of auditors for

each area and which users are defined as administrators in that area.The User table stores information about each user, such as their username and email address, and is linked to the Roles table, defining system administrators.

Functional Decomposition



Use Cases

Use Case 1

Auditor is required to do their scheduled audit.

Preconditions

User is logged in

Main Success Scenario

1. Auditor opens the application
2. Application displays the audits that the auditor needs to do
3. Auditor selects the audit they need to do
4. Application displays audit questions one at a time
5. Auditor fills out entire audit

Postconditions

A new audit entry is added into the application

Exceptions

1. Server is unable to be reached

Use Case 2

Alternate auditor has to do an audit.

Preconditions

User is logged in

Main Success Scenario

1. Administrator clicks option to view audit schedule
2. Administrator selects audit location
3. Application displays audit schedule and allows for addition and removal of auditors into schedule
4. Administrator adds auditor to schedule
5. Application updates schedule with new auditor in schedule

Postconditions

Auditor is added into audit schedule

Exceptions

1. Server is unable to be reached

Use Case 3

A user wishes to see audit results.

Preconditions

User is logged in

Main Success Scenario

1. User selects location of audit results
2. Application retrieves results of audits for specified location
3. Application view displays historical chart of audits and their results

Postconditions

Audit results are displayed to the user

Exceptions

1. Server is unable to be reached
2. No audits have been made at the location

Use Case 4

A user wishes to change the audit form

Preconditions

User is logged in

Main Success Scenario

1. Administrator clicks the option to edit audit forms
2. Administrator selects audit location to be edited
3. Application displays the audit questions for the selected location
4. Administrator alters/adds/deletes questions as desired

Postconditions

Audit form is changed

Exceptions

1. Server is unable to be reached

UI Layout

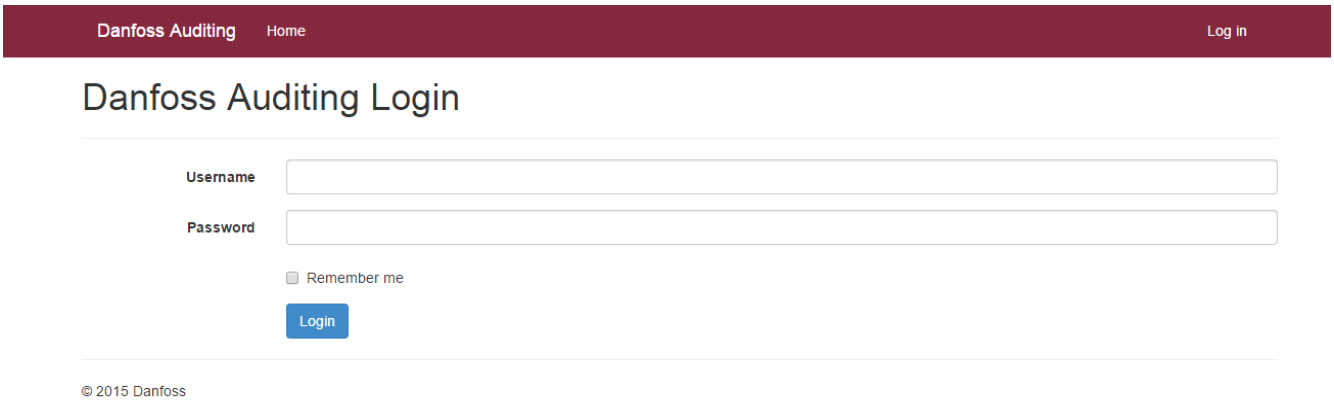


Figure 1 - Login with Active Directory

Figure 1 (shown above) displays what a user will see when logging in to the auditing system. Login credentials are authenticated with Active Directory. After logging in, the user is presented with a dashboard shown in Figure 2.

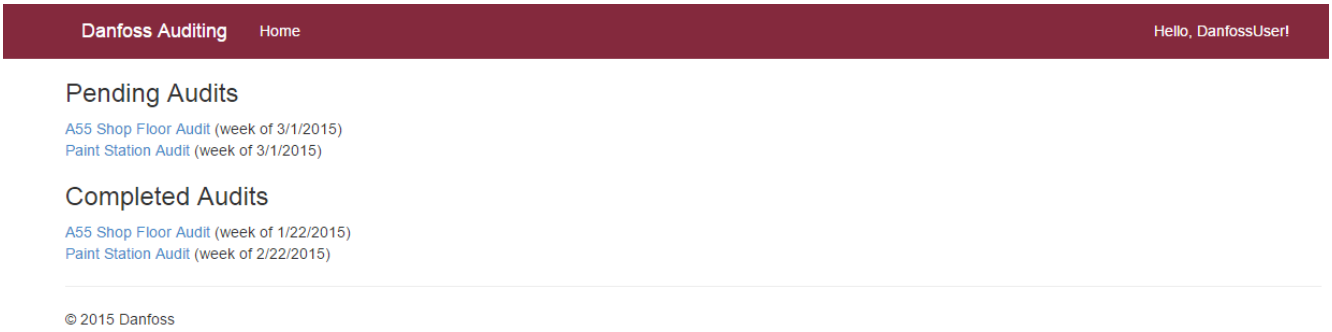


Figure 2 - User dashboard

The dashboard (Figure 2) is what a user sees after logging into the auditing system. Pending Audits are audits that need to be completed. Clicking on the title of the audit takes the user to the first question in the audit, describe in Figure 3. Completed audits has a list of links to the results of previously completed audits.

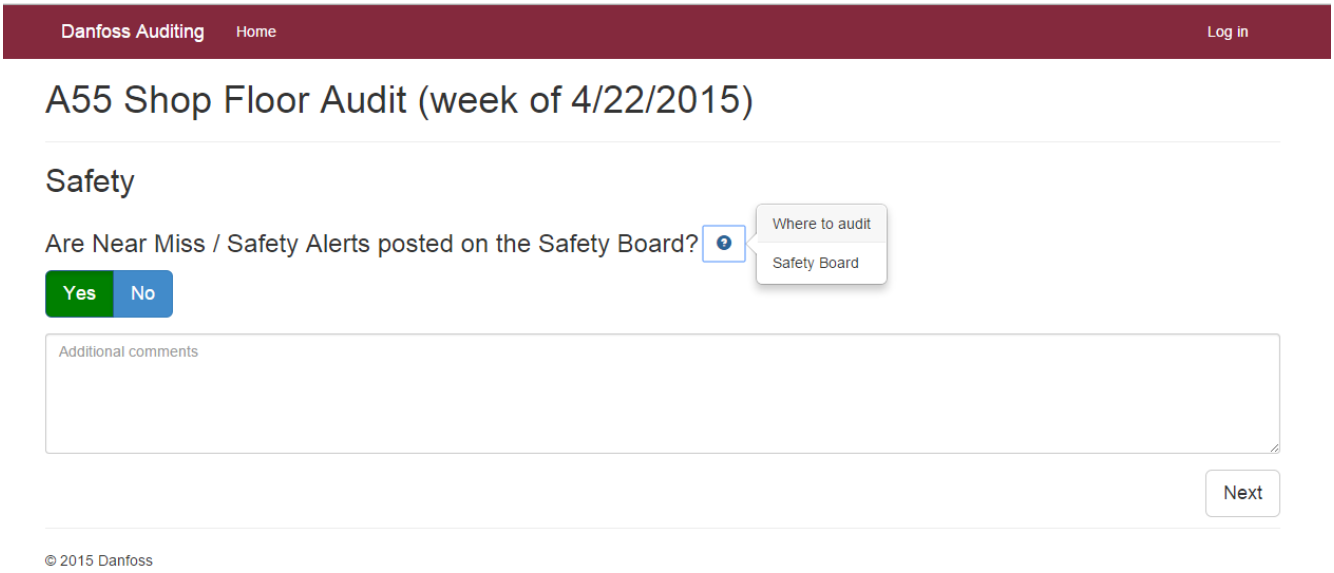


Figure 3 - Completing an audit

The audit completion pages, shown in Figure 3, display one question at a time, with a “hint” button the auditor can click on to trigger a tooltip with information about where to find the answer. Each page has simple yes/no toggle buttons, a field for comments, and “Next” and “Back” buttons for navigation between questions.

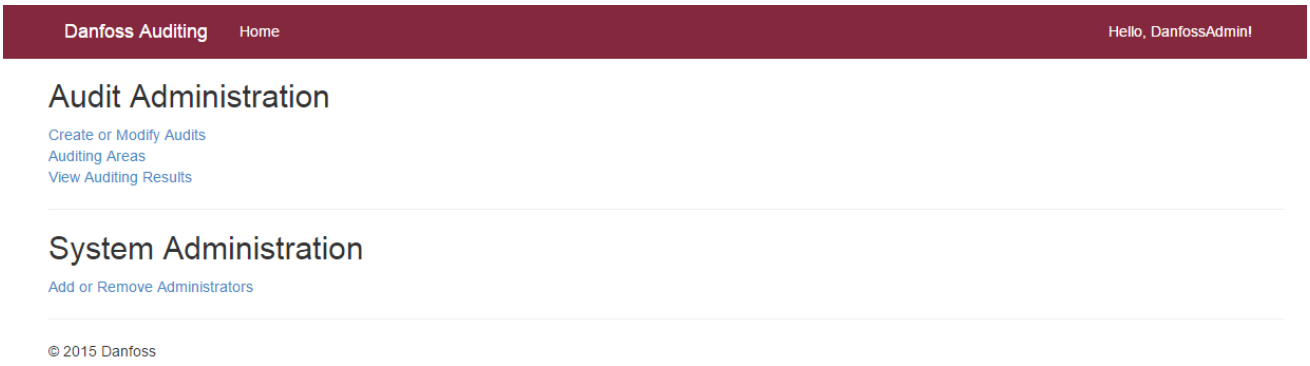


Figure 4 - Administration dashboard

The administration dashboard (Figure 4), available only to users with appropriate permissions, links to the other administration pages, such as those to modify audits and schedules and view auditing results.

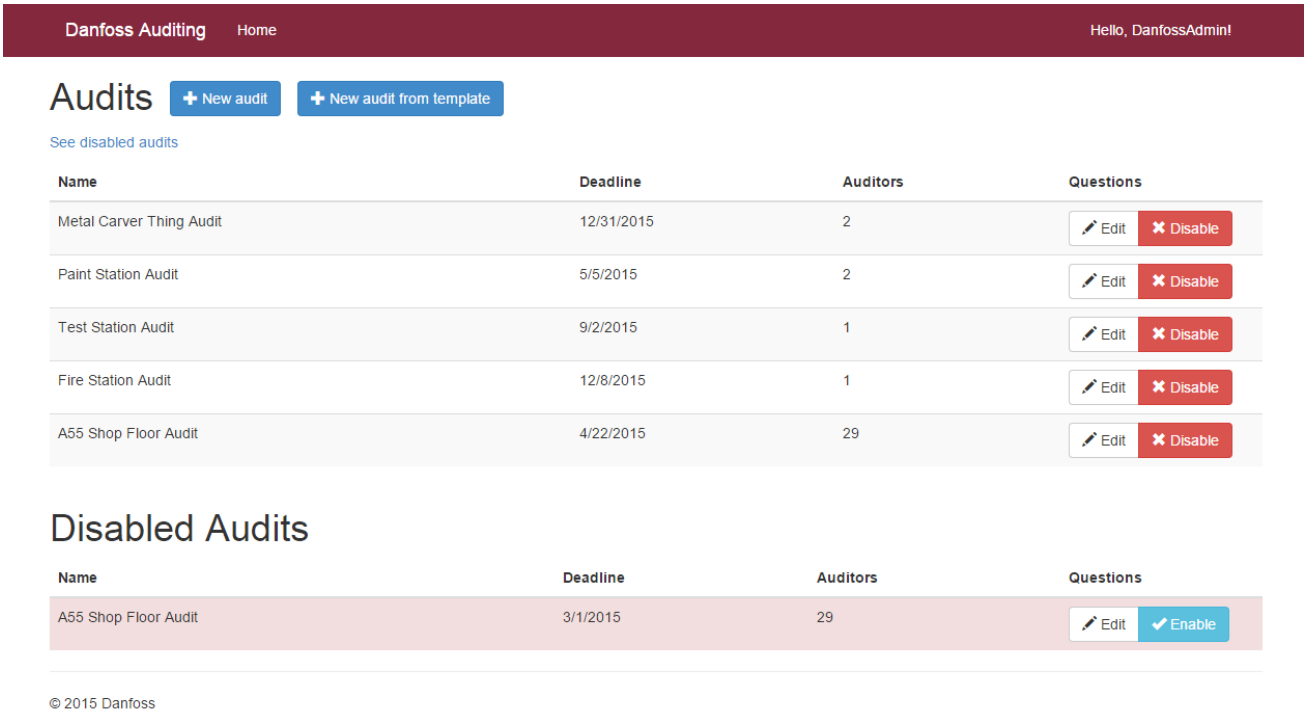


Figure 5 - Audit administration

The audit administration page, shown in Figure 5, is where administrators can create, modify, and disable audits.

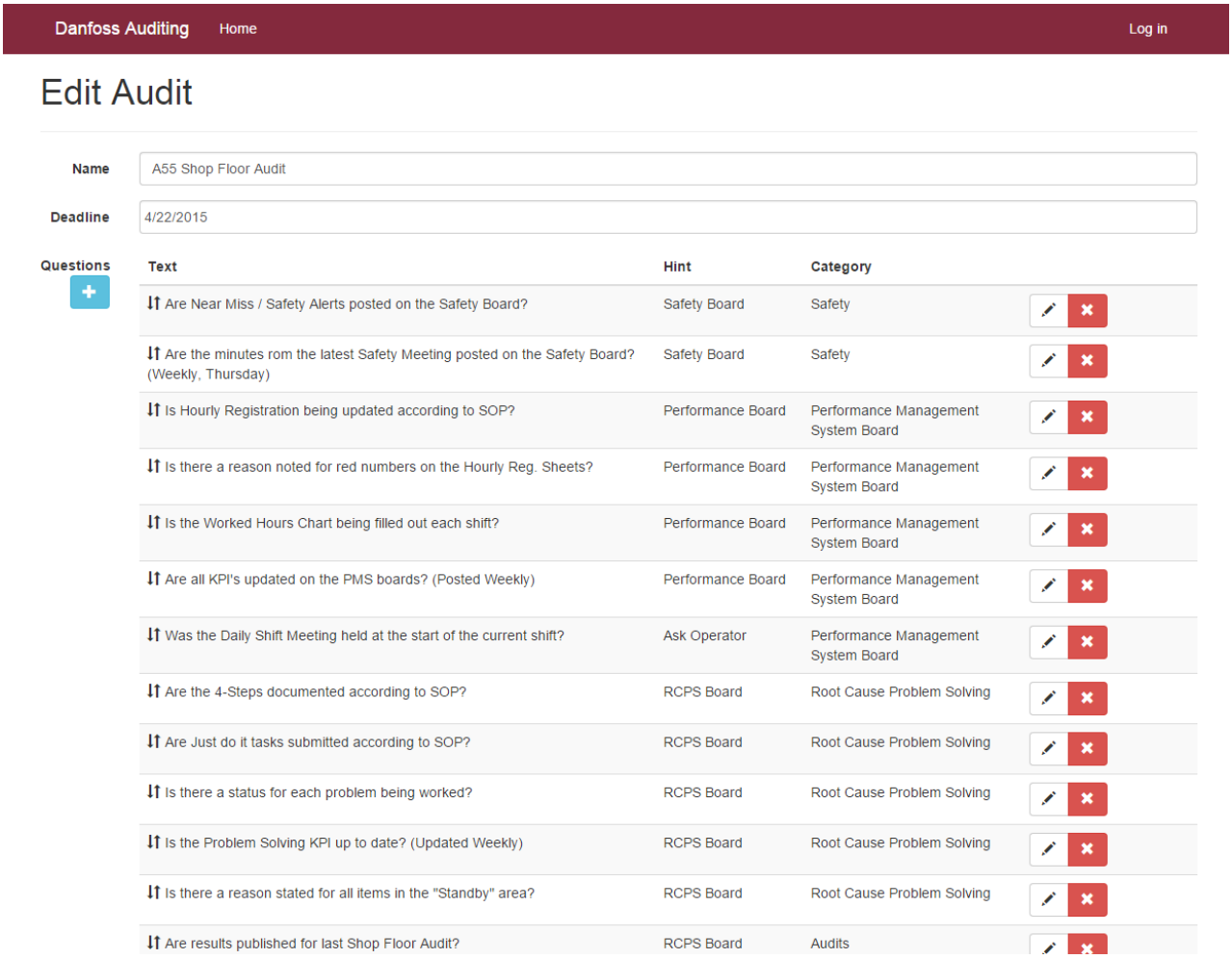


Figure 6 - Editing an audit

When editing an audit, as shown in Figure 6, administrators can add new questions or reorder, modify, or delete existing questions.

Standards

Danfoss Standards

Danfoss imposed a few standards that were required for this project. This included coding style, comment formatting, color choices, and logo placement. Danfoss's coding style is enforced by an application called StyleCop, making maintenance and documentation easier. Color choices and logo placement are rules imposed by Danfoss corporate to keep a consistent look and feel.

WCF and Web Forms Standards

Danfoss uses WCF to centralize business logic into one location in the system. It essentially creates an API to be used across the entire system instead of splitting logic and repeating code. This standard imposed on this project required restructuring calls and logic, but makes the system easier to maintain. Developing this system using ASP .NET Web Forms standards required conforming our system to this structure. Most applications created internally by Danfoss are developed using Web Forms, so using this structure is more familiar with developers, thus costing less to maintain.

Software Development and Evolution Standards

This project followed a waterfall process initially, with designing and documentation phases within the first semester. During this timeframe, meetings with client provided feedback on our progress with these plans and designs, which also helped with understanding the overall system. For the second semester, this project followed an iterative implementation process, with each iteration separated by client meetings. The first couple iterations involved implementing a few major operations of the system and acquiring feedback to be updated for the next iteration. The remaining iterations consisted of implementing the last requirements and additional functionality requested by the client. Following the iterative process standard allowed for the client to see progress and make changes earlier on in the development. To maintain and track our system, we used a private Github repository.

Testing

There were four different types of testing used for developing the Danfoss Auditing System: Integration Testing, Operational Profile Testing, Regression Testing, and Beta Test.

Integration Testing

With every push to the repository we tested to make sure that new functionality works as expected on other devices or browsers. Our client uses Internet Explorer and Safari for mobile, so ensuring the system works for these is essential. Shown below in Figure 7, is a way to test the responsive layout of the system and quickly see how it looks on an iPad device. In addition to this, tests were run on an Android tablet to provide similar testing results. This device also became our central testing device for the beta test discussed later.

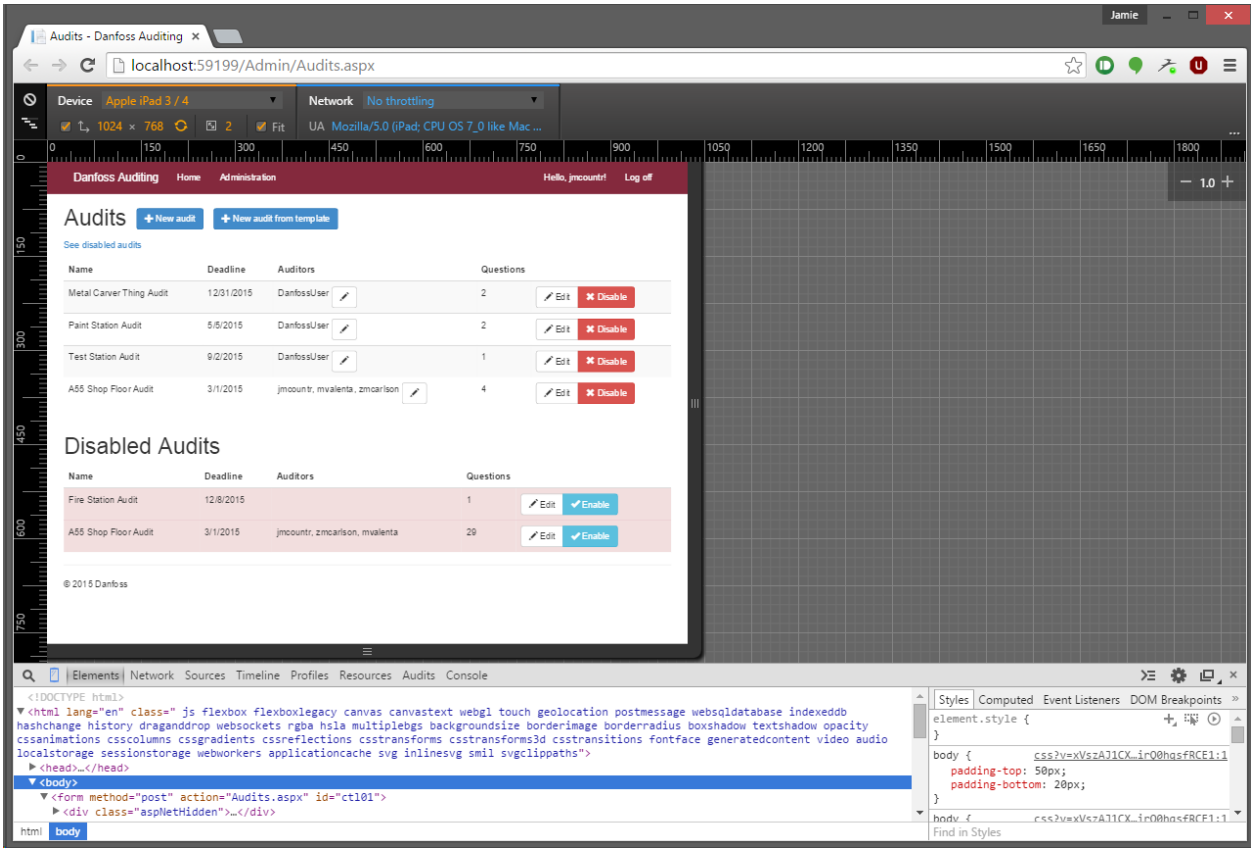


Figure 7 - Google Chrome Developer

Operational Profile Testing

With every iteration of the development process, tests were run to make sure that operations a typical user would have, run smoothly. These followed the guidelines of the use cases mentioned before. Since these use cases cover most of the functional and non-functional requirements, it is essential that these operations work correctly. An example is an auditor creating an audit from the template. Tests were made to make sure that an audit with the

correct questions are created in the database without harming other data such as the template itself.

Regression Testing

Every few major commits or iterations involved testing previous operations still work as expected. Sometimes changing the database schema or some business logic within WCF caused other pages to behave incorrectly, so testing these became essential. Tests included both operational profile and integration testing to make sure these operations still work as well as on other devices and browsers.

Beta Test

On April 25th, a beta session was held with our client. The motivation was to have a user test run our system and provide feedback of our main feature: completing an audit. This was tested on a personal Android device that is similar to what our client would actually use. The auditor took the device out onto the shop floor and performed an audit, then provided feedback. Based on this test, the amount of time to complete an audit drastically decreased, improving this entire process greatly.

Appendix I: Operation Manual

Requirements

- Microsoft Visual Studio
- Microsoft SQL Server
- Microsoft IIS
- Entity Framework
- Active Directory

Installation

1. Open the solution in Visual Studio.
2. Alter the ConnectionString in App.config of the DanfossAuditingService project to point to the database that will be hosting the auditing data.
3. Open up the Package Manager Console and run the command “update-database”. This will create the tables for the project on the database.
4. Set the ConnectionString in “Settings.settings” file in the properties folder
5. Right click the DanfossAuditingService project and click Publish. Fill out the details of where you will be publishing the software to (the server that will be hosting the service and website).
6. Right click the Danfoss Auditing project and hit Publish. Again, fill out the details of the server that will be hosting the application.
7. From the server’s IIS, verify that the website and service are running.

Usage

Auditor

If you are an auditor, when you log in to the application you are able to see pending and completed audits assigned to you. Clicking on a completed audit will provide a view of the audit’s results, while clicking on a pending audit will allow you to actually perform the audit, one question at a time.

Administrator

If you are an administrator, you capability is extended to not only view completed and pending audits assigned to you (if applicable), but also able to create, edit, and disable audits. Beyond that, you are also capable of viewing results of all audits, creating and editing areas, and providing other users with administrative access.

Creation of Audits

Prior to actually creating an audit, there are a few steps that must be taken care of. These are defined Areas that the audit will be associated with and auditors assigned to the area. Once this is complete, you are ready to create an audit.

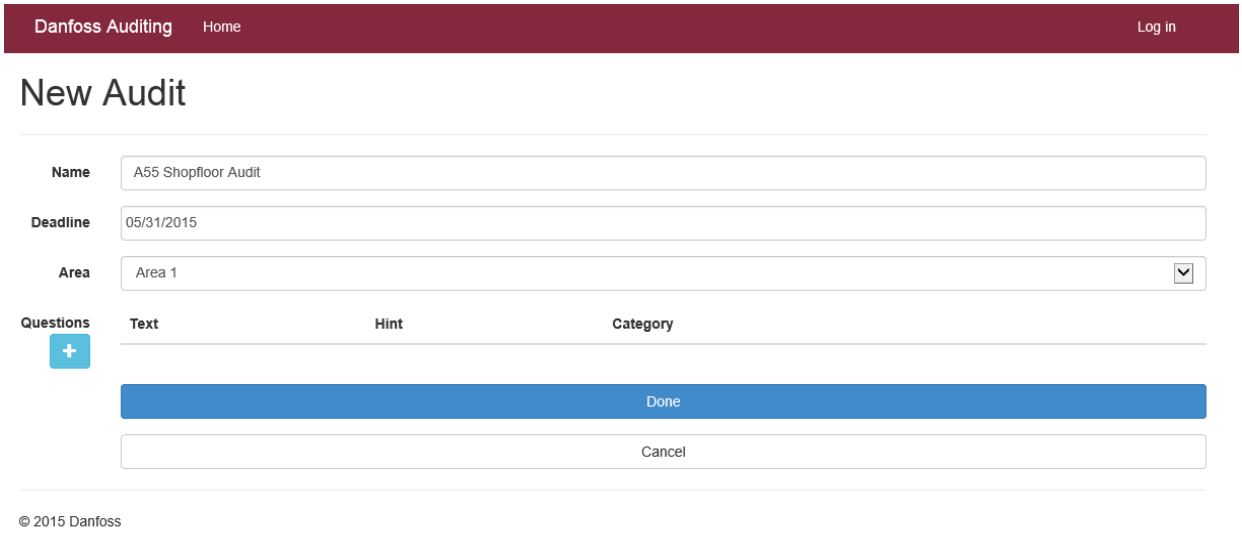


Figure 9 - Creating an audit

After the area information is already set, creation of an audit simply requires the definition of which area it should be associated with, and it will assign itself to an auditor. On top of this, the deadline of the audit will cause administrators of a designated area to be emailed upon non-completion, along with providing the assigned auditors emails reminder emails of upcoming audits.

Editing audits

From the main audits page, you can select a specific audit and add/remove questions, along with reordering the questions themselves. If need be, you are also capable of altering the name of the audit, the deadline, and the area itself. The edit page is the exact same as the creation page, with the exception that it's editing an audit rather than creating a brand new one.

Appendix II: Other Designs

MVC vs Web Forms

The initial design for the auditing system involved Microsoft's ASP.NET MVC framework, which was more familiar to us as a development pattern. Danfoss later made it clear that Web Forms was their preferred technology and should be used instead to be consistent with other internal applications.

Addition of WCF

At the beginning of the auditing system's development, each web page was interacting with the Entity Framework database abstraction. Danfoss recommended using WCF, which helps separate business logic from the presentation layer and allows other Danfoss applications to access auditing data if necessary.

User Roles and Authorization

Since the auditing system uses Active Directory to authenticate users, our original plan for user roles (such as administrators) was to grant permissions based on the user's groups in Active Directory. After discussing this with Danfoss it was determined that tracking user roles in-app would be better—adding users to groups in Active Directory involves a lot of red tape and takes too much time. A new page was added to the auditing system to add or remove administrators and store these role assignments in the application's database; users added to the administration role are then granted access to administrative operations when they log in.

Appendix III: Other Considerations

During the design process in (EE/CprE/SE 491), several additional features were discussed but never implemented due to time constraints, team knowledge, and task priority. Our client determined these as tasks to be done at the end if there was time, otherwise they could implement it themselves.

Upload pictures as part of hint

Initially, it was brought up in a few meetings to implement the ability to take pictures with an iPad and upload them as either answers to audit questions or in the hint area to direct more accurately to an answer. Methods of saving this information to the database was discussed but not implemented. It was decided by our client to be implemented by them for us to focus on higher priority tasks.

Automatically print / email PDF

In the last meeting, it was brought up by a user to implement a feature to automatically generate a pdf report of a completed audit, email it to auditors, and print it. This would automate the audit process and overwrite it even further allowing for faster auditing. Due to the time constraints and system access, it was determined that our client could implement this later.

SSRS

SSRS (SQL Server Reporting Studio) provides analytics of various parts of the system was discussed to be implemented in this system. It requires having a special server set up to handle queries made to it, similar to a database. Since this was pushed back until the end of the project, we determined it wouldn't be feasible to implement and test due to time constraints.

Lessons Learned

Working with ASP .NET was new for all of us so becoming familiar with it took awhile to get used to. Creating a database to store data and save it instead of just designing one was thrilling to us. Having regular deadlines to keep everyone on track with implementation would have helped with showing more progress to our client. With that, keeping up to date with our project work on Trello because there were some confusions on who would do what task. Lastly, asking for help on parts of the system that our client has implemented before with other systems would have saved us a lot of time.

Other

We all liked working with our client and had some enjoyable meetings. When showing them mockups of the system, one asked if it looks and different on the iPad and another responded saying it looks the same, but smaller. This is just an example of the kind of humor that went on sometimes while designing this system which made things more entertaining.