****



**Project Proposal**

**For**

**Prototype Database with User Login & Friendly UI/UX**

**University of Nevada, Reno - IS 495**

**Prepared by Brianna Avalos, Colin Rogers,**

**Will Williams, Sal Rahimi, Tien Che**

**September 15, 2022**

**Table of Contents**

[**1**](#_heading=h.gjdgxs) **Executive Summary 3**

[**2**](#_heading=h.30j0zll) **Statement of Problem 3**

[**2.1 Background 3**](#_heading=h.1fob9te)

[**2.2 Importance of Problem 3**](#_heading=h.3znysh7)

[**2.3 Problem Description**](#_heading=h.2et92p0) **4**

[**3**](#_heading=h.tyjcwt) **Project Objectives 4**

[**3.1 Objective of the Project 4**](#_heading=h.3dy6vkm)

[**3.2 Constraints and Limitations**](#_heading=h.1t3h5sf) **5**

[**4**](#_heading=h.4d34og8) **Technical Approach 5**

[**4.1 Solution Concept**](#_heading=h.2s8eyo1) **5**

[**4.2 Performance of the solution**](#_heading=h.17dp8vu) **6**

[**5**](#_heading=h.3rdcrjn) **Product Owner Awareness 6**

# Executive Summary

<

1. In a world today where everything is data dependent, it is essential to maintain a reliant and easy accessible database. Our goal is to help out an organization that does not have the knowledge to set up a reliant database. We are hoping to improve the efficiency in any organization that chooses to work with us. Ideally a small local organization such as a library or small school.
2. The various topics we are addressing in our project all revolve around working with data. This will be a prototype of creating tables based upon the different types of data handed to us relevant to the organization. Creating correct unique identifiers for each table as their Primary Keys. Then relating tables with each other by the use of Foregin keys. Once all our tables are set up and we have all the data stored in the right places we begin our integration onto the web app. This web application will be run off the Microsoft’s ASP.NET framework to make the integration process easier. We will have different User login tables which will store credentials as well user roles for certain Users as needed. The interface should allow its users to queue through their database and access any information they are looking for at ease.

# Statement of Problem

This project is necessary because many small or brick and mortar companies still keep hard copies of records and data. This project would allow those smaller companies to eliminate redundant, corrupted or inaccurate data from their records and store it somewhere safe. By execution of this project the organization would be able access and query their data fast, allowing them to make more accurate business decisions with minimal errors.

## Background

There are many organizations, especially those with little to no income and resources, who struggle with upkeep and organization of their records. This has led to redundant information, errors, and loss of confidential information. This puts not only the business’s reputation in danger but information of the client. This ultimately affects the operations of the organization, usually ending with the demise of businesses.

## Importance of Problem

Redundant information takes a toll on the overall vitality of not only the business but its income. Redundant information creates more error, thus making it difficult for businesses to accurately comb through data and make accurate predictions. Many companies have tried to solve the issue by creating teams of employees that check each other's work. This may reduce the errors but it does not ensure that all information is accurately curated.

## Problem Description

In this day and age the amount of data and information is so vast that it can be difficult to track, sort and analyze without the help from databases, systems and AI. It is especially difficult for businesses that still keep track of their records and data physically. Not only does this method of record keeping prevent the business from making accurate and sound decisions, it also decreases the income that the company could be earning using web based databases. At this point in time many companies use Excel as a means to keep track of their records but Excel does not allow users to find redundant data that has varying characters or differing record information. Excel can’t help the company make sound decisions since it lacks the capability to perform complex queries and searches of data/records. Because of these characteristics of the program organizations lose money and time trying to find the data then analyze it. By implementing a web based database for companies to input and sort their data companies would run more efficiently.

We need to create a login page for the specified business to get into the database. The database would then have the current data logged into the system but allows the employees to add, revise and filter data as needed for the current business goal.

# **Project Objectives**

The objective of this project is to allow our client to be able to access their database at ease. We want to build a reliant and easily accessible database for an organization that is not familiar with working through an SQL nor any knowledge on how to query through data. Our objective is to create the front end of a Web Application that will allow users of the organization to simply login and browse through their database at ease. We will incorporate this using Microsoft’s ASP.NET framework. This framework will allow the users to query and navigate through many different tables of data in which our group will create for them based on the organization's data as well as integrate it into the front end of the Web Application we create for them. Each user in the organization will have their own credentials with certain limitations and constraints based on different roles of the Users, which will only allow them to whatever data they are limited to seeing. Not every User will have access to all the data. Once a User is logged in they should be able to query any data or access any data they need.

## Objective of the Project

1. Create database
   1. Build structure for database
   2. Implement rules for database

2. Apply Microsoft’s ASP.NET framework

a. Design the front end for the webapp interface

b. Create User Logins with valid credentials stored in database

3. Allow Users to queue through the interface with ease

1. Get rid of any redundant data in our database
2. Assign each table in the database with Primary Keys
3. Make relationships with foeign keys

## Constraints and Limitations

Our team faces several constraints and limitations for this project that are listed below:

1. **High Complexity:** A database Management System is quite complex as it involves creating, modifying and editing data. If the user is not familiar with technology it may be hard to maneuver around the database
2. **Price:** Creating and managing a database is quite costly compared to physical records
3. **DataBase Failure:** If data is inputted incorrectly or if the code is not valid the database will reject the new/edited data.
4. **Damage:** Damage to the database affects virtually all applications programs.
5. **Employee training required:** In order to reduce problems later down the road everyone should be taught how to move around the database.

# **Technical Approach**

We plan to use Microsoft SQL Server for creating the backend of the database; using this specific program will allow us to easily create a prototype relational database for the purposes of this project. Using SQL, we’ll need to create a number of tables, and columns within those tables, pertaining to important information that the business using this would need to functionally perform daily operations. SQL Server also offers excellent integration with the rest of the Microsoft ecosystem, most notably Excel.

Using Excel, we will be able to compile our fictional data, then import and populate it into our tables using SQL Server’s import wizard. Personally, in past experience this process is much more effective than hard coding in all of the data using SQL.

As for our front-end, UI/UX development, we plan to utilize Microsoft’s ASP.NET framework to construct a small web app, consisting of a login screen, as well as a query menu that will allow for users to be able to access, view, and modify their records. The main goal of this web app is to give users an interface that they are more familiar with navigating, opposed to having to write SELECT statements to manually query their data. We made the choice to use the ASP.NET framework because of its seamless integration with SQL Server, due to both programs being within the Microsoft ecosystem.

In order to best cooperate with each other on this project, we’ll have to split our work between the group members, i.e. some people are doing front-end, while others are doing backend. For us to best keep all of our files organized we’ll utilize a Github repository. Using Github is much more effective than a Google Drive or other online file storage resources because of its audit log that shows what’s been changed following each commit, making it easier to go back and make adjustments as needed.

## Solution Concept

Our solution is to develop a prototype database within Microsoft’s SQL Server, that would include tables containing records necessary for a business to carry out everyday operations. We will also be developing a web app in C# using Microsoft's ASP.NET framework, to serve as a friendly UI/UX for users to interact with the database. Upon first opening the web app, users will be presented with a login box, requiring a username and password (because we don’t want just anyone to access this info). If there is a user matching the given credentials, they will be granted access, and can begin querying the database, based on specific search parameters. In the instance where a user enters credentials unrecognized by the system, one of a few error messages will appear beneath the login box:

* If user enters their credentials and the username is found, but the password does not match, they will receive the message *“Incorrect password, please try again”*
* If user enters their credentials but the username is not found, regardless of the password entered, they will receive the message *“User not found, please try again”*

Because this project is a prototype, for ease of explanation, the example of a school will be used. The web app will be able to be accessed by student’s, parents, and teachers, to view and query information from the database. Like a student could look up the classes they’ve been assigned for a particular semester, along with the names of the teacher and class, and if a textbook is required. Alternatively, a parent could use this tool to query for a list of their student’s teacher’s contact info, if they needed to reach out to them regarding any concerns. For example, we’d need to make tables for the following:

* Student Table
  + StudentID, LastName, FirstName, SSN, BirthDate, Address
* Parent/Emergency Contact Table
  + ContactID, LastName, FirstName, Phone, Email, Address
* Teacher Table
  + TeacherID, LastName, FirstName, Phone, Email
* Class Table
  + ClassID, Subject, Name, RoomNumber, NumberOfStudents, BookAssigned (yes/no)

## Performance of the solution

Our first quality measure of performance is to ensure that the database will be able to store any relevant information that the business using it will need to store. Our second quality measure of performance is to determine if the database is being used effectively in an attempt to reduce/remove all instances of data redundancy. This measure will be achieved by being sure to only store information in a single place, and then referencing it using a foreign key if needed. An example of this would be storing the name of a teacher in both the *Class* table and the *Teacher* table, rather than simply calling the specific Teacher’s ID in the *Class* table. Along with mitigating instances of data redundancy, our third quality measure of performance would be that we hope to see a reduction in the amount of important documents (birth certificates, social security information, etc.) that go misplaced due to human error, now that this information will be stored within the database.

# **Client Awareness**

\*\*Can be left blank because Igor is standing in as our client.