System Design Document

# Code Crate

# Data Forge LLC.

# 4601 Mid Rivers Mall Drive

# Cottleville, MO 63376

# 7/7/2023

# Table of Contents

# 

[**Introduction 3**](#_heading=h.gjdgxs)

[**Purpose 3**](#_heading=)

[**System Overview 3**](#_heading=)

[**Design Constraints 3**](#_heading=)

[**Roles and Responsibilities 3**](#_heading=)

[**Project References 3**](#_heading=)

[**System Architecture 4**](#_heading=)

[**Database Design 6**](#_heading=)

[**Hardware and Software Detailed Design 6**](#_heading=)

[**System Security and Integrity Controls 6**](#_heading=)

# 

# Introduction

Some people use the same password for every account they have online. Some also use very simple passwords like “password” and “12345.” These bad ideas are frequently made when someone doesn’t want to constantly go to the “Forgot Password” section of the login page every time they go to log in. These people are why Data Forge LLC. has created Code Crate.

# Purpose

Code Crate is designed to provide users with a secure repository to save the passwords they use across the internet on desktop and laptop browsers, tablets, and smartphones. THis service will allow users to look up their saved account credentials from anywhere an internet connection is available.

# System Overview

Code Crate is a web application accessed via web browser that acts as a repository for user login credentials. Users will be able to create, lookup, modify, and delete information they save to the service. The system is secured using proprietary encryption methods to ensure data privacy. Account security makes use of two-factor authentication at sign-in to thwart unauthorized access attempts.

# Design Constraints

Code Crate is only supported on Windows 10 and newer. At this time, Code Crate is not supported on Macintosh or any distro of Linux.

# 

# Roles and Responsibilities

Connor Clawson (Project Manager)

* Manages what needs to be done, in order, and on time
* Helps design the backend for the project
* Creates the documents needed for the project
* Sets up meetings for the team to discuss future plans and issues

Bailey Ducommun (Full-Stack Developer)

* Develops the backend using the CSV files
* Develops the frontend using Blazor, HTML, CSS, and other UI libraries
* Develops the services needed in order for the frontend and backend to communicate
* Does bug testing for the project

Ryan Schoonover (Front-End Developer/Graphic Designer)

* Helps Develop the front-end UI components via HTML and CSS
* Goes through and makes sure our application works with mobile devices
* Creates and designs logos and other images to be used in the project
* Does bug testing for the project

Phuong Anh Nguyen (Back-end Developer)

* Develops the backend using C#
* Assists with documentation

# Project References

Code Crate makes use of the following libraries:

CsvHelper library (<https://joshclose.github.io/CsvHelper/>) | Used for our backend purposes

Microsoft QuickGrid, Version= 0.1.0-alpha.22351.1 | Used to create custom tables

Blazored.FluentValidation, Version= 2.1.0 | Used for form verification

Blazor.Bootstrap, Version= 1.9.2 | Used for all UI components

# System Architecture

*Hardware:*

The Code Crate design is based off a local host, which has the following components:

1. Local Host Desktop
   1. Motherboard
   2. CPU: 4.5ghz
   3. RAM: 32GB
   4. 1x Network Adapter
   5. 3 TB SSD
2. The Router in the location of the Local Host
   1. Asus 1200

*Software:*

The Code Crate design is based off of Blazor WebAssembly. This assembly gives us the ability to create custom components in which users will enter data. The software architecture is designed to store, manipulate, and validate all incoming data from the user. Some custom components that we have created in our architecture include:

1. The Login Component
   1. This component will take user input and that data will go through a validation check to see if that user exists in the database.
   2. This has a register button to go to the register component
   3. Once the check is true, the user will move onto the dashboard component
2. The Register Component
   1. This component has multiple entry fields on a modal
      1. Username
      2. Password
      3. Confirm Password
      4. Email
   2. These entry fields are stored into our database after it goes through a validation process to make sure bad data doesn’t enter the database.
3. The Dashboard Component
   1. A table will be shown and this is where ALL the users’ credentials will be placed.
   2. This also has other buttons to route you to the correct pages.
   3. This page represents our Create, Read, Update, and Delete.
4. The Add/Edit Component
   1. This will bring up a modal with multiple fields for the user
      1. Application
      2. Username
      3. Password
   2. This has a password generator if the user wishes to create a strong password
      1. Gives different length options
5. There are other components that assist these components
   1. The modal to display error messages
   2. The navbar component
   3. The footer component and much more

# Database Design

Figure: Entity Relationship Diagram

# 

The database consists of two tables; UserAccount and PasswordLog. The UserAccount table holds the login credentials for the Code Crate service. PasswordLog stores the data provided by users. Each primary key is assigned by the DBMS and is inaccessible to users. Each record in the PasswordLog is tied to a unique user with a shared UserID. The UserID and PassID fields cannot be modified by the user through any external means.

# Hardware and Software Detailed Design

## Hardware:

No additional hardware design is required. Existing hardware detailed design can be found in the Code-Crate Scope Of Work Document, dated June 23, 2023.

## Software:

More details about the components and the logic behind them are as follows:

1. The Login Component:
   1. This takes user data, that data goes through our User Account Validation check which is one of our custom made services, once the credentials match what is in the database, we grab the userID and are able to move onto the MainDashboard Component
2. The MainDashboard Component:
   1. With the userID we can go through our password log database and grab the credentials created by the current user. This data will display onto the table on initial start-up. We made services that assist the front-end receive the data needed for the user.
   2. The user can delete any credential, this will send the current passID to the service and the service will go to the DB to delete that credential. Once completed the table will refresh automatically.
   3. When a user adds a credential, that data is sent to the service to manipulate the data, a new passID gets created, then it is stored into the DB, once that is complete the table will refresh automatically.
   4. The same process with updating a credential, but no new passID is needed.

Services:

1. So far most of our services are custom made, we have a JSinterop Clipboard API as well
2. The UserAccount Service:
   1. This will handle ALL the data that involves the user accounts. It will route the data to the correct spots and manipulate the data before going to the database.
3. The PasswordLog Service:
   1. Basically the same thing as the UserAccount Service
4. The JSinterop Clipboard API
   1. This allows the user to copy the password with just a click of a button.

# 

# System Security and Integrity Controls

User access to data is limited to records tied to their account. This limitation is enforced in the user interface. Access to a user's account requires a valid username and password matching a record. at the account owners prerogative, they may pair a TOTP token to their account such as Authy or Google Authenticator. The back-end software lacks the means to perform database manipulation operations beyond what the front-end provides.

As for the records in the database, all data is encrypted using a proprietary cipher at the back-end. Encryption and decryption occurs at the backend only to ensure data security across the connection.Sponsor Acceptance

Approved by the Project Sponsor:

Date:

<Faculty Sponsor>

<Faculty Sponsor Title>

This free System Design Document Template is brought to you by [www.ProjectManagementDocs.com](http://www.projectmanagementdocs.com)