eLoan Project Documentation

# Project Management

The role designation for the eLoan Project was as follows:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Armin | Anshul | Andrew |
| **Database** | | | |
| Database Schema | ACI | RA | ACI |
| Database Implementation | ACI | ACI | RA |
| **Client** | | | |
| Decision Making Script | ACI | RA | ACI |
| Acquisition Backend | ACI | ACI | RA |
| Account Management Backend | RA | ACI | ACI |
| UI | RA | ACI | ACI |
| **Post-Completion** | | | |
| Presentation | RA | RA | RA |

# Functional Specifications

The eLoan project is intended to provide users with a place to apply, receive, and manage loans. This web app will provide users with a graphical user interface to operate within their browser where they will input a series of information used to create profiles, apply for loans, and manage them. The web flow for the app is as follows:

Login -> Create an account -> Registration

Login -> Proper login submission -> Home (Index)

Registration -> Login Here -> Login

Registration -> Proper Registration Submission -> Login

Home(View) -> Apply -> LoanApp(View)

Home(View) -> My Loans -> Loans(View)

Home(View) -> My Account -> Account(View)

All of these views are managed within the HomeController class.

The web app should read information from HTML forms and apply them to the CRUD implementation to translate information to database entities.

The Database was designed with 7 tables in mind. These correlated to different parameters required for the user to maintain a loan account as well as create a new one.Diagram

Description automatically generated

# Technical Specifications and Implementation

## Architecture and Programming

The eLoan project is built using the ASP.NET Model-View-Controller architecture. The MVC architecture would be applied to allow for better delineation of roles within the architecture. The primary programming language was C# for the backend connections and logic.

## Create-Read-Update-Delete

The backend of the interaction with the database is written using CRUD. This was done with the intention of being able to consistently and uniformly enter and manipulate data within the database without over complication as well as accurately read entities from the Models within MVC.

## SQL Server

The data storage solution used was MySQL Server hosted by the Loyola Chicago CS Dept. This allowed for remote connection through the GlobalProtect VPN and easy access to the data without having to carrying it within the application or store it locally for the user. Queries to the database were handled by the Microsoft Entity Framework Core package except the DDL, which was written by a team member initialize the database with the correct tables.