Weber State

MicroFund

A web-based app and database for Weber State’s Micro Fund team.

Table of Contents

What is Micro Fund…………………………………………………………………………………………………………..……3

Languages/programs……………………………………………………………………………………………………………..4

What is MVC?………..………………………………………………………..…………………………………………………….5

How is the database Managed?…..…………………………………………………………………………………………6

Roles………………………………………………………………………………………………………………………………………7

How to design the UI……………………………………………………………………………………………………………..9

What has been done in V 1.0? ……………………………………………………………………………………………….10

Where we left off……………………………………………………………………………………………………………………11

Breakdown of Visual Studio and MVC…………………………………………………………………………………….13

Miscellaneous …………….…………………………………………………………………………………………………………15

What is MicroFund?

Weber State MicroFund is a program that allows Entrepreneurs that are either current students of Weber State, or Alumni to apply for grants to further their businesses along. It is a process that starts with an application. If the application is approved, the entrepreneur can pitch his business idea to a panel of judges. Once the judges approve that business idea has potential the applicant is assigned a mentor that will help the applicant fine tune their business pitch if needed. They will also help the applicants attend workshops that will really make their pitch shine. Managers will oversee the work of these mentors. In its current state, this whole process is scattered between several different applications. one admin uses a personal computer for storage, Excel for managing the process, and a google doc form for the application process, among others

The goal for the project is to create a Web-Based program to manage the whole process from start to finish, not only for the admin of the MicroFund but all of the different Roles that are needed, including judges, mentors, and applicants. Instead of several different programs, the goal is to create one web-based application that will have a login, and different UI’s for the different user roles. All of the data will be stored in a database hosted by a secure server provided by Weber State. This unified application will create a cleaner experience for the applicants, but even more importantly will help the MicroFund Team spend less time trying to keep everything organized and more time helping these Entrepreneurs be successful.

What Language and programs are needed?

The project is created in Visual Studio. The languages we have used so far are:

-C#

-HTML

-CSS

-SQL

The repo is Managed by Brad Peterson and can be accessed using GitHub. The link for the Repo is here:

<https://github.com/bradleypeterson/Wildcatmicrofund>

The project uses MVC as our software design pattern. More information on MVC on the following page.

What is MVC?

We would highly recommend doing Microsoft’s tutorial on MVC as the first thing, if you don’t have any experience coding using this design pattern before. The link that most of us used to get familiar with MVC is:

<https://docs.microsoft.com/en-us/aspnet/core/tutorials/first-mvc-app/start-mvc?view=aspnetcore-3.1&tabs=visual-studio>

A screenshot of a cell phone

Description automatically generated

The basic idea of MVC is to split the logic into three parts.

First is the Model

-This is the applications data structure, it manages the data, logic and rules of the application.

Second is the View

-This is where all of the HTML code resides and is what loads when you run the program (Pretty straight forward part)

Finally, is the Controller

-The controller is what handles all of the commands for the model and view.

How is the database Managed?

The database is built from Entity framework core. It is code first meaning the database is built from the models and database context files contained in the Data folder. The database server connection string is contained in the appsettings.json file this should be managed as a secret in the future.

To create the database the dotnet ef command line tool is used. Follow this tool to install on the development machine.

<https://docs.microsoft.com/en-us/ef/core/miscellaneous/cli/dotnet>

-Command to create a migration. This will create a file in the migrations folder. This is generated code that will show what is going to be changed on the database server. This will show how the database will be upgraded or downgraded. Replace migration name with the desired name for the migration.

dotnet ef migrations add MigrationName

-Command to commit the changes to the database. This command will run the migration file from the previous step.

dotnet ef database update

When the server starts the files Program.cs and Startup.cs will start the database context. If the database is new with no data the DbInitializer.cs in the data folder will run and seed the database with the general data needed like application status.

What are all the User Roles and Responsibilities?

(See Lucid Chart Dashboard’s for visual representation)

Admin

-Manage users and assign user roles

-Run reports from the database

-Create new applications

-Edit existing applications

-Look at the status of all current applications

-Build a cohort

-View and add calendar events.

Applicant

-Create a new application

-update an existing application

-Create a returning application

-Edit Business information

-View calendar events

Mentor

-Edit a user profile

-Access the business dashboard

-Review applications

-Fill out mentor reports

-Access reports

-Edit Business phase.

Judge

-Review Applications

-Review Mentor Reports

-Fill out a business evaluation Scoresheet

-Access the business dashboard

-Access Reports

Intern

-Edit applicant’s profiles

-Review assigned applications

-Access Reports

-Access the business dashboard

Designing the UI

The project started with two different teams, the Infrastructure team and the design team. The design team has been using Lucid Chart to design all of the UI’s and storyboards for the program. The link for the Lucid Chart diagram is:

<https://app.lucidchart.com/invitations/accept/09b7e23b-e025-4efd-8313-3b2edbda0788>

Lucid Chart is a paid program, but if you sign up using your student email, you’ll have full access to all of the tools and can make changes as needed by the program.

All of the lucid chart diagrams that have the ‘approved’ status have been shown and approved by the clients and we are modeling the HTML views based off of those approved views.

We started with Lucid chart and some of our early designs don’t match what the project looks like when you run the project in visual studio. We have slowly been updating the Lucid Chart to match as we go along. The important parts of Lucid Chart is that lucid chart contains all of the views that we want to implement.

If you shift + ctrl click on one of the buttons it will load the next view for you, so you don’t have to navigate through all of the pages in the grid view of lucid chart.

What are the clients expecting for version 1.0?

Version 1.0 is all of the features we want to have working for the client ASAP. It won’t be a fully functioning app that all the users will be able to use but is more so for the clients to get a feel on how the web-based app will help them improve their workflow.

Features that we have told the clients will be part of 1.0:

-Login page with the option to login with either google login or custom account creation

-Admin Dashboard with the following features:

-Edit profile (Change any users profile options)

-New Application (Create a new application)

-Reports (a few static reports)

-User lookup (assign user roles and get a list of all users and emails, Search function)

-Application Status lookup (see all current applications and have the ability to change its status) (not a final feature, just 1.0)

-Applicant dashboard

-Edit Profile (Change your personal details)

-User login (Every new user is defaulted to an applicant)

-New application (Create a new application and submit to a working database)

-Update application (Pull from the database all the application information and prefill the application form)

Where We Left Off

The Infrastructure:

-Right now the program only loads and runs locally, we still need to test it on a Weber State server, which Professor Peterson will provide.

-[ASP.NET Core Identity](https://docs.microsoft.com/en-us/aspnet/core/security/authentication/identity?view=aspnetcore-3.1&tabs=visual-studio): an API with which you can create Authorization and Authentication features in your application. Users can create an account and login with a username and password. ASP.NET Core Identity uses a SQL Server Database to store usernames, passwords, roles, and profile data.

What needs done for Identity: [Two Factor Authentication](https://docs.microsoft.com/en-us/aspnet/core/security/authentication/identity-enable-qrcodes?view=aspnetcore-3.1), [Account confirmation and password recovery](https://docs.microsoft.com/en-us/aspnet/core/security/authentication/accconfirm?view=aspnetcore-3.1) (need to configure email provider/sender), [Require Email Confirmation](https://docs.microsoft.com/en-us/aspnet/core/security/authentication/accconfirm?view=aspnetcore-3.1#require-email-confirmation) all still need added if you are wanting to use those features. Before production an email provider/sender needs configured.

The UI

-Update All of the dashboards to better reflect what they will really look like in the finished product. We have updated most of the dashboards, but a lot of the pages within the dashboard need to reflect those changes. (Specifically, the project has the row of buttons along the top, where Lucid chart has the buttons running down the left-hand side)

-We focused most of our meetings and work around the admin and the applicant user roles and dashboards. The other user roles have not been approved 100% by the clients which is why they are not part of 1.0

-Once the controller is finished the Views can be created in HTML for that controller. You can click Administrator along the top to see an example of what the other dashboards should look like.

-The beta branch has more CSS and look for V 1.0 then the master branch, but a merge of the branches may be needed sooner than later.

-The biggest part of the project that isn’t done, but probably the most important for V1.0 is the reports page for the admin. This page is difficult because a lot of the reports require all user roles to be fully functioning. We wanted to create a simple report page that will pull a few different SQL Reports from the database. The clients would like a very simple way of running the SQL queries with predefined reports that they can choose to run. There are two views in lucid chart for reports. One we created for simplicity, and one that will eventually need to be updated to be more finished with the other Roles.

-The final aspect of the UI that needs to be finalized is the Calendar, and how that will look and feel.

Break Down of Visual Studio File Explorer

(Insert screenshots and explanations here)

Miscellaneous

The Clients are Guy and Bob, they are very hands on and love to meet via Zoom to talk about the project and approve any work that has been made every few weeks.

Their contact information:

Guy: guyletendre@weber.edu

Bob: bobgruhler@weber.edu

A playlist of the recordings we had with the clients so far:

<https://www.youtube.com/playlist?list=PLqN9Ca7eU-YhuZ0quKr1hVDqocQYr0Udz>

About the Summer 2020 Team

Infrastructure Team

Alex Baldwin

David Kempke

Jesse Tice

Jonathan Morse

UI “The Fun Parts” Team

Jason Hansen

Colin Pytlik

Juan Santos

Brandon Beecroft