Redesign of the British Columbia Amateur Softball Association Player and Team Management Website

by Joseph L. Pruner

B.Sc. COMPUTER SCIENCE HONOURS

in

Irving K. Barber School of Arts and Sciences

(Computer Science)

THE UNIVERSITY OF BRITISH COLUMBIA
(Okanagan)

April 2019

I would like to give a special thank you to my advisor, Dr. Ramon Lawrence, whose guidance and expertise were invaluable throughout this process.

Abstract

This work built a website intended for use by the British Columbia Amateur Softball Association as a player registration and management system. Representatives of the association have user accounts that grant them access to their respective softball clubs, where they can register, edit and manage teams and players. Higher-level representatives have access to register, edit and manage specific organizations, districts, and clubs within the association, as well as users. Each user has a certain admin-level, which dictates exactly what they have access to. All data is stored in a relational database, allowing users to easily find players and teams based on specific search criteria, as well as add existing players and teams from previous seasons and tournaments to current ones. The website was created using the Laravel PHP framework. The presentation will demonstrate the website and discuss the process of its construction.

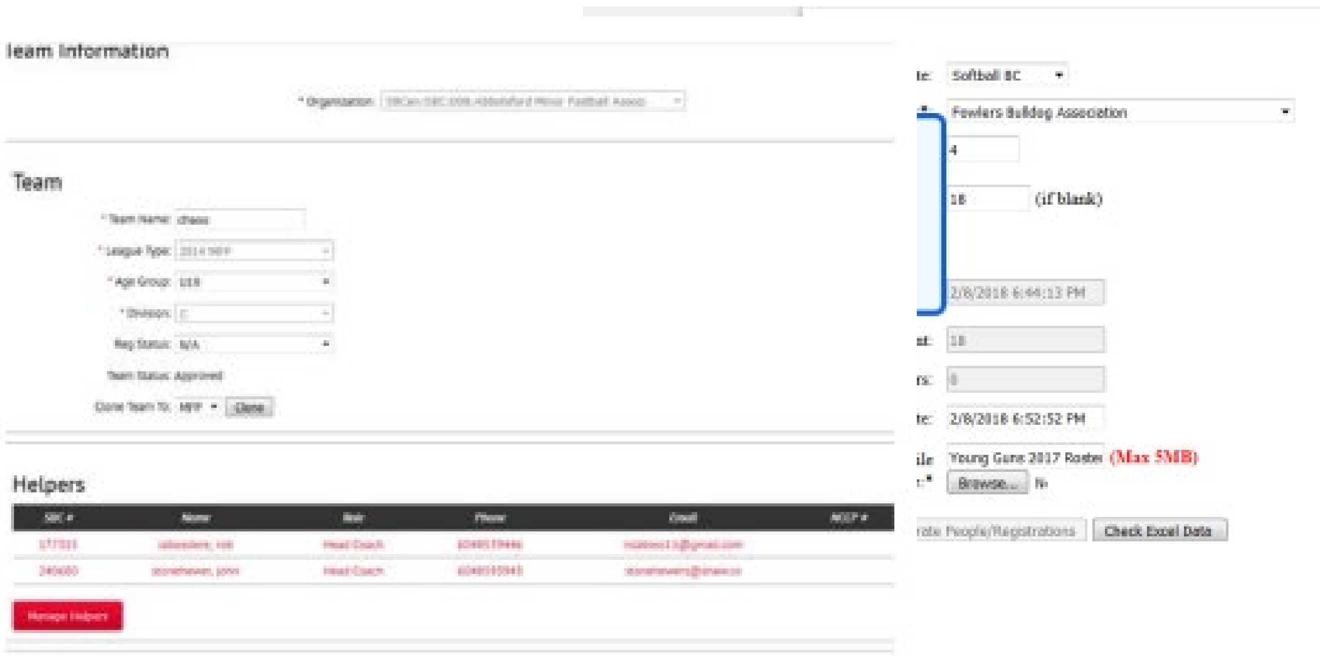
1 - Introduction

The British Columbia Amateur Softball Association (BCASA) player and team management system website is used to store thousands of player, team, season, club and tournament records. The users of the website are BCASA officials or appointed representatives. These users each have specific affiliations with specific softball clubs, geographic districts and larger organizations. When they login, they may only access and manage records from their respective affiliations.

Unfortunately, the website has not had any significant updates in many years. Its client-side design still reflects that of the early 2000’s, even though there has been significant research and progress made in the fields of human computer interaction (HCI) and user interface (UI) design. The website often presents information in a cluttered or confusing way, with menus and interactions that allow the user to display too much information at once. There are also various user interactions that require several steps to complete, but could be significantly reduced with proper page restructuring.

The same is true for the underlying server-side technologies driving the system’s core functionalities. There are instances where interactions result in a full webpage reload, redirection to a new page, or require the user to perform that same action multiple times. All of these could be eliminated with proper coding changes.

The combination of these external and internal hindrances can be the difference between a user having a frustrating, time-consuming experience, or a pleasant, streamlined one. The purpose of this thesis is to redesign and improve the BCASA website and management system by applying modern HCI and UI design concepts, as well as industry standard technologies to create fast and efficient core functionality.



Team Information

* Organization: [illegible]

Team

* Team Name: chess

* League Type: [illegible]

* Age Group: U18

* Division: [illegible]

Reg Status: N/A

Team Status: Approved

Clone Team To: MFP Clone

Helpers

SBC #	Name	Role	Phone	Email	NCCP #
[illegible]	[illegible]	Head Coach	[illegible]	[illegible]@gmail.com	
[illegible]	[illegible]	Head Coach	[illegible]	[illegible]	

Manage Helpers

Softball BC

Fowlers Bulldog Association

4

18 (if blank)

2/8/2018 6:44:13 PM

18

0

2/8/2018 6:52:52 PM

Young Guns 2017 Roster (Max 5MB)

Browse...

Check Excel Data

Figure 1.1 - The current Softball BC website style.

2 - Key Terms and Definitions [1]

Attribute/Field

The columns in a database table that describe specific characteristics of that entity type. (e.g. last name, address, cost, date etc.)

Active Person/Player

A person who has been added to a team and has a role.

The British Columbia Amateur Softball Association (BCASA)

This is the association that oversees the user, player and team registrations, across many BC districts and organizations.

Client

The web browser that a user is using to access the website.

Client-side

This refers to events or processes that happen within the client. (e.g. JavaScript is a client-side scripting language because it executes within the client.)

Content

The text, files, images, videos and anything else that is displayed or accessible to the user on a website.

Content: Dynamic

A website with content is created at run time by a program or script running on either the client or web server. (e.g. A user submits data in a form to the web server, and the server returns specific information from a database.)

Content: Static

A website that consists of only HTML pages that look identical to all users at all times.

Cascading Style Sheets (CSS)

CSS is a style language used to create the visual presentation of a web page through hundreds of attributes describing color, font, size, position etc.

Database

A structured collection of data stored in a computer with an interface allowing other programs to retrieve and store information from it.

Entity

A person, place or thing which is modelled and stored as records in a database table (e.g. Team, Player, Club).

Graphical User Interface (GUI)

A visual representation of an operating system or program where the user does not have to type commands to interact with it. It allows the user to interact with it through clicking on buttons, menus, images, windows etc. (e.g. Windows, Mac OS, Ubuntu)

Hypertext Markup Language (HTML)

HTML is a web development language that is semantically used to create the structure and content of a web page. It consists of elements that dictate where content will be placed.

Inactive Person

A person who is not on a team.

Operating System (OS)

The foundational computer environment that the user interacts with, where programs exist and are executed, and files are organized and accessed.

Person

An active or inactive person.

PHP: Hypertext Preprocessor (PHP)

PHP is a programming language primarily used for web development. It runs on a web server and is used for creating the logic of how the server should handle incoming and outgoing data.

Query

A structured language request for specific information that the database can interpret and respond to. SQL is used in this project.

Record

A single row in a database table.

Secure Shell (SSH)

Secure shell is a cryptographic network protocol that allows the user to securely connect to network services on another computer. It is typically used to remotely access the command line of another computer, and execute commands.

Structured Query Language (SQL)

A computer language used to query a database with human readable syntax.

Schema

A database schema is a defined structure and set of rules that the data in the database is constrained to. It dictates which data types are expected for each column, the relationships between tables, and the overall organization of the data.

Table

A data structure consisting of rows and columns representing a single entity type. Each row is an individual instance of that entity, and each column describes a certain characteristic of the entities.

Team

A collection of active people.

3 - Preliminary Design and Planning

3.1 - Database Overview

Almost all websites rely on a dedicated database to store and manage their content. It is the foundation of any website that expects to offer robust content to many users in a fast and efficient way. The optimal database for the BCASA management system is a relational database, because all the tables within the system rely, or relate to another table in some way. For example, a team consists of active people, so the active people will "belong" to a team. Within the database, the teams table is said to have a "relationship" with the active people table. This means that each active person has a unique piece of identifying information (an id number) which can be used by other tables to specifically refer to that exact person. Each team will also have a unique id, so all the people that have been added to a certain team, will have that team's id in their record. If using a non-relational database, there would be more redundancy that slows down search time. Using a relational database, the team id links directly to the players that belong to that team, significantly increasing search time, and allowing more complex queries. [1]

3.2 - Database Design

Designing the database is an important first step, as it is the foundation of the management system's core functionality. This started by creating mock-up designs in a modeling software called Astah. It shows the initial tables, attributes and relationships, as well as notes about the design. The tables are represented by the yellow rectangles, and the lines represent the relationships between them.

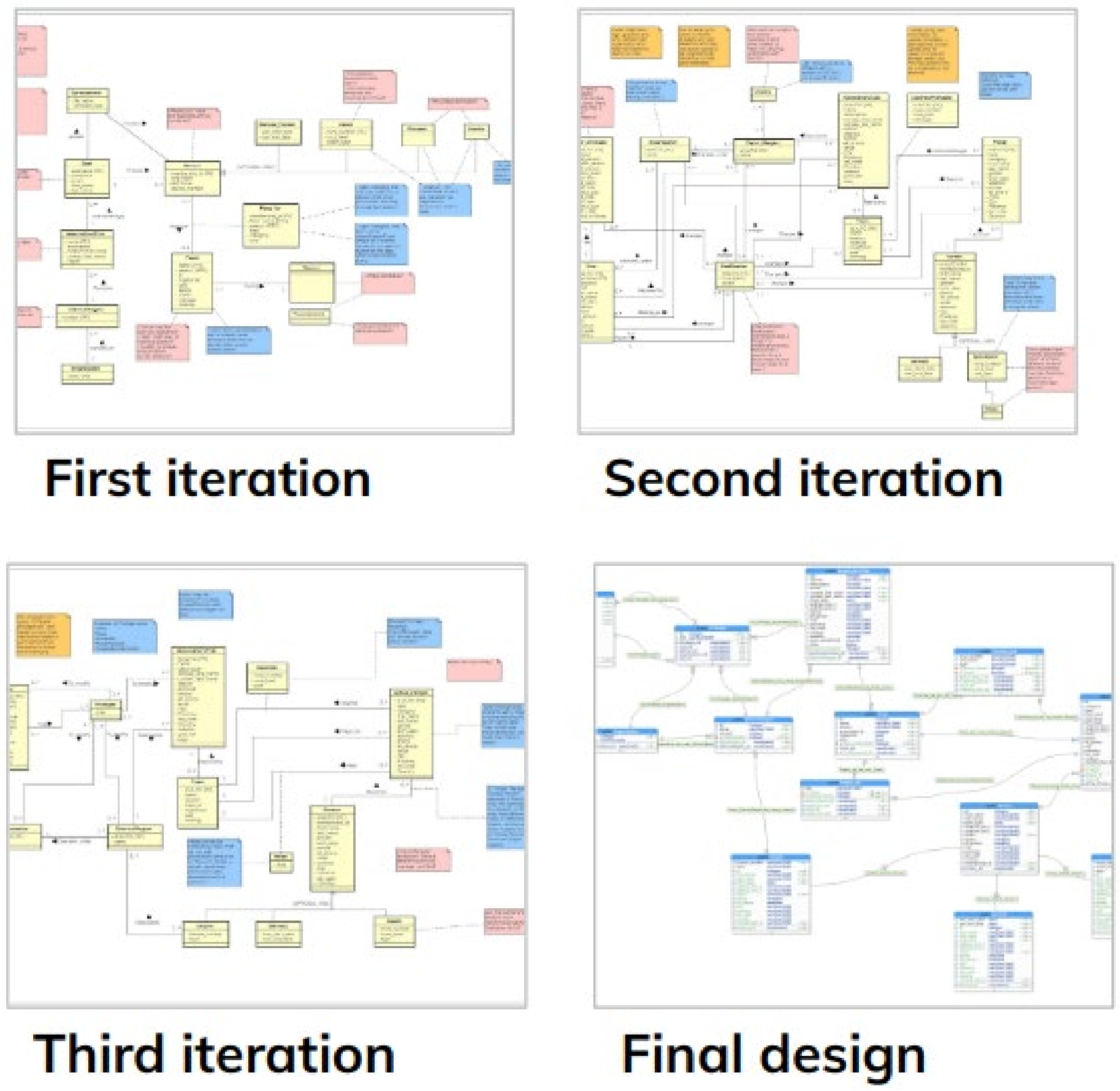


Figure 3.2a - The four iterations of database design

The database design was an iterative process and seen in Figure 3.2. After the first design, which was based on the trivial assumptions of what this kind of database would need (players, teams etc.), the model was reviewed and then adjusted accordingly. These adjustments are usually creating new tables for previously undiscovered entities, splitting a table into multiple tables to reduce redundancy, or removing and adding relationships.

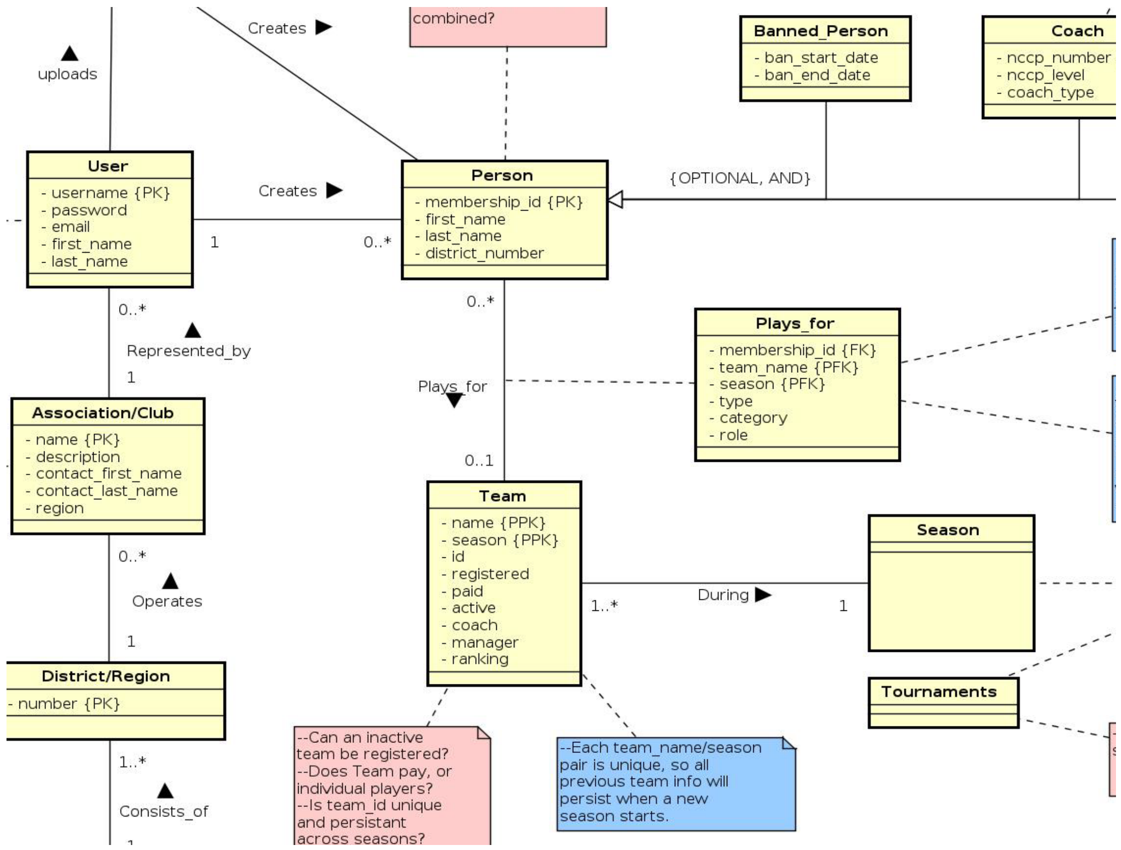


Figure 3.2b - A section of the first iteration. Both active, and inactive people exist in the person table

For example, in Figure 3.2b, the only table that stores non-user people information is the person table, regardless of whether that person is currently an active player on a team or not. This causes a problem for a couple of reasons:

1. ***Cannot maintain a history of active people's attributes across seasons.*** When a person who has previously been on a team begins a new season on a new team, with a new role (i.e. player to coach), the person table will be updated with this new information. Their previous seasons team and role attributes will be overwritten, and there will be no way to search for which

team a person has played on, or which roles they've had in the past. This also applies to any other updated information in the person table. Since searching for people's previous attributes is useful functionality, users should be able to do this.

2. ***Illogical relationships will be created.***
   If both active and inactive people exist in the same table, the person table will have relationships to other tables that don't make logical sense for inactive players to have, and vice versa. An inactive person, by definition, will never be a coach, a manager, or on a team, so they should not have any relationships with those tables. This will lead to visual confusion, and the potential for database query mistakes via convoluted SQL statements.

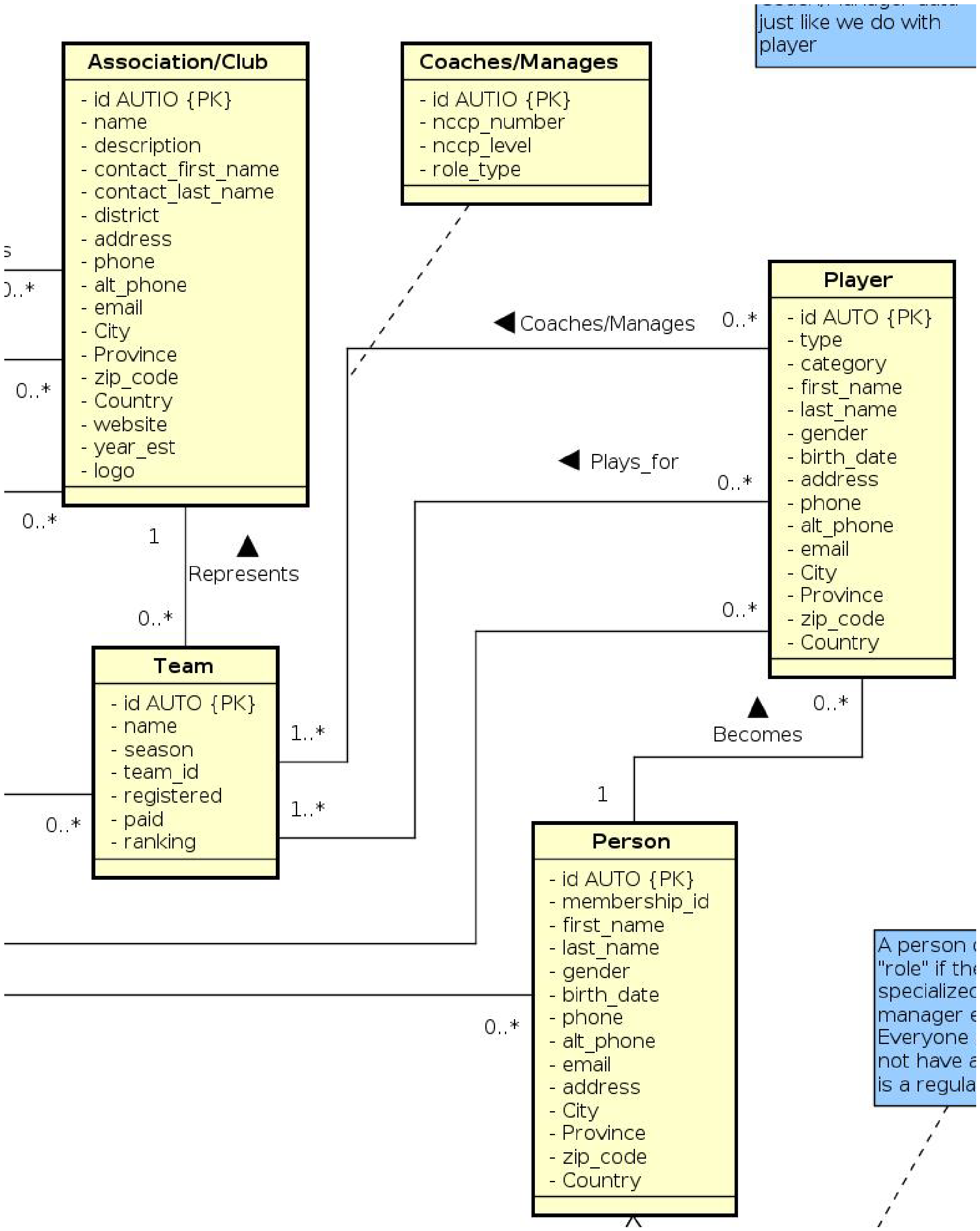
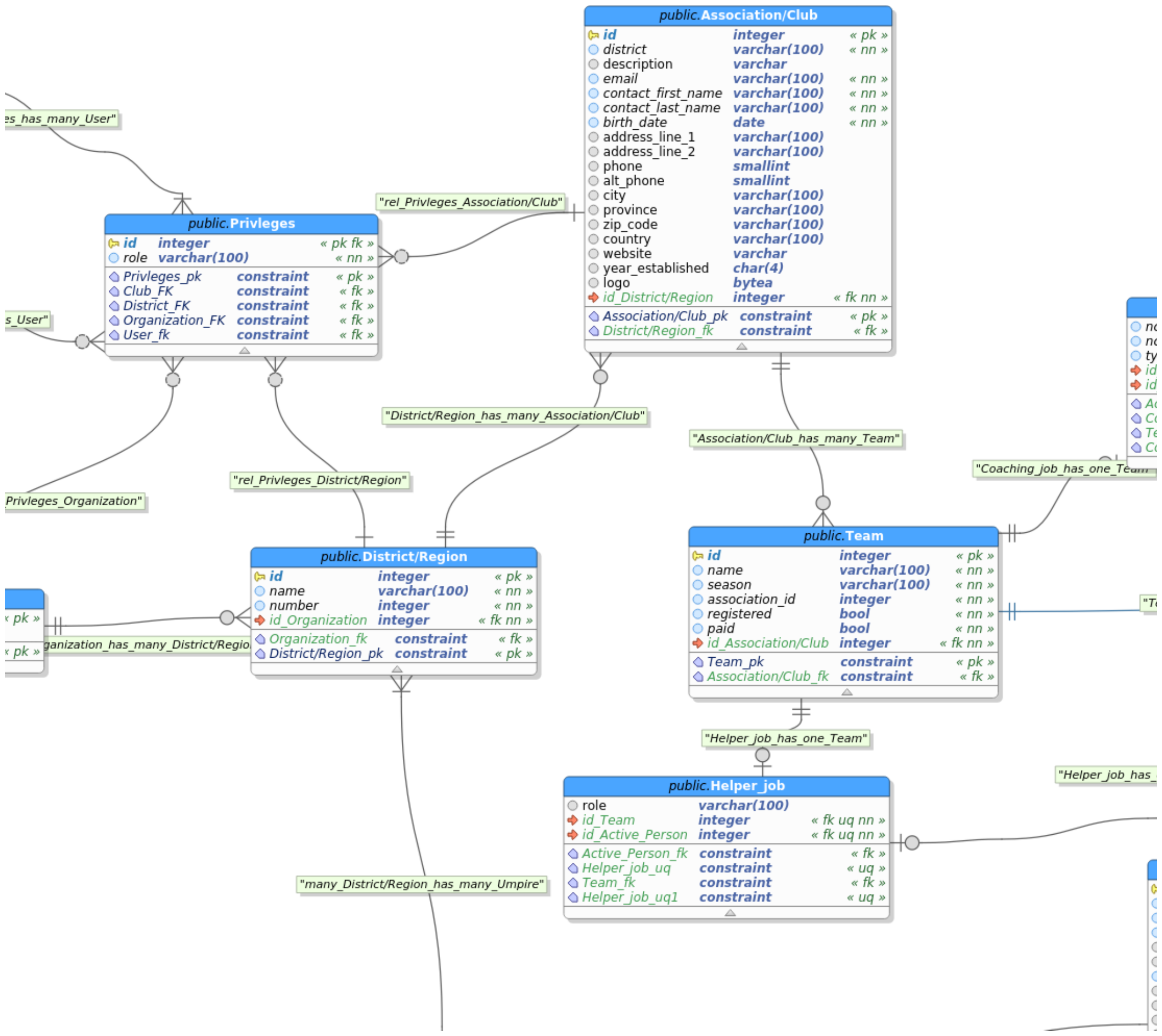


Figure 3.2c - A section of the second iteration. Note that there is now a Player (active person) table.

These problems are resolved in Figure 3.2c, which is the second iteration of the database model. For problem 1, having a dedicated active person/player table allows the database to add a new player record whenever the season changes, instead of overwriting that person's prior attributes. Now the database can simply search by a specific season, and see the exact, entire roster of players for that season.

For problem 2, the Person table no longer has relationships with the Team, or Coaches/Manages table (off screen relationships irrelevant). It only consists of logical relationships. An inactive person becoming a player directly reflects a real life scenario. With each iteration, similar problems were discovered and resolved.

The final design of the database was created using PGmodeler. This tool enforces the real rules and structure of an SQL database, and can generate the SQL code to create the designed database. This was very useful, because when converting the basic Astah diagrams into PGmodeler models, it would present errors, indicating that there were Astah tables that had invalid attributes, or relationships that violated certain constraints. This added another layer of refinement and correction to the database model.



public.Association/Club
id integer « pk »
district varchar(100) « nn »
description varchar
email varchar(100) « nn »
contact_first_name varchar(100) « nn »
contact_last_name varchar(100) « nn »
birth_date date « nn »
address_line_1 varchar(100)
address_line_2 varchar(100)
phone smallint
alt_phone smallint
city varchar(100)
province varchar(100)
zip_code varchar(100)
country varchar(100)
website varchar
year_established char(4)
logo bytea
id_District/Region integer « fk nn »
Association/Club_pk constraint « pk »
District/Region_fk constraint « fk »
public.Privleges
id integer « pk fk »
role varchar(100) « nn »
Privleges_pk constraint « pk »
Club_FK constraint « fk »
District_FK constraint « fk »
Organization_FK constraint « fk »
User_fk constraint « fk »
public.District/Region
id integer « pk »
name varchar(100) « nn »
number integer « nn »
id_Organization integer « fk nn »
Organization_fk constraint « fk »
District/Region_pk constraint « pk »
public.Team
id integer « pk »
name varchar(100) « nn »
season varchar(100) « nn »
association_id integer « nn »
registered bool « nn »
paid bool « nn »
id_Association/Club integer « fk nn »
Team_pk constraint « pk »
Association/Club_fk constraint « fk »
public.Helper_job
role varchar(100)
id_Team integer « fk uq nn »
id_Active_Person integer « fk uq nn »
Active_Person_fk constraint « fk »
Helper_job_uq constraint « uq »
Team_fk constraint « fk »
Helper_job_uq1 constraint « uq »
"rel_Privleges_Association/Club"
"District/Region_has_many_Association/Club"
"Association/Club_has_many_Team"
"Coaching_job_has_one_Team"
"rel_Privleges_District/Region"
"Privleges_Organization"
"ganization_has_many_District/Regio"
"Helper_job_has_one_Team"
"Helper_job_has_"
"many_District/Region_has_many_Umpire"

Figure 3.2d - A section of the final database model complete with all attributes and relationships.

3.3 - Web Page Wireframe Mock-ups

After the database modeling was complete, the next step was to begin wireframing the structure, layout, and basic visual appearance of the web pages. A wireframe is a barebones outline with minimal detail, and it is very useful for efficiently testing different visual structures. The tool used is a program named Balsamiq. It comes preloaded with many common web page items such as text boxes, buttons, menus and image placeholders etc., and it allows an easy drag and drop workflow onto a blank canvas to create a web page mock up.

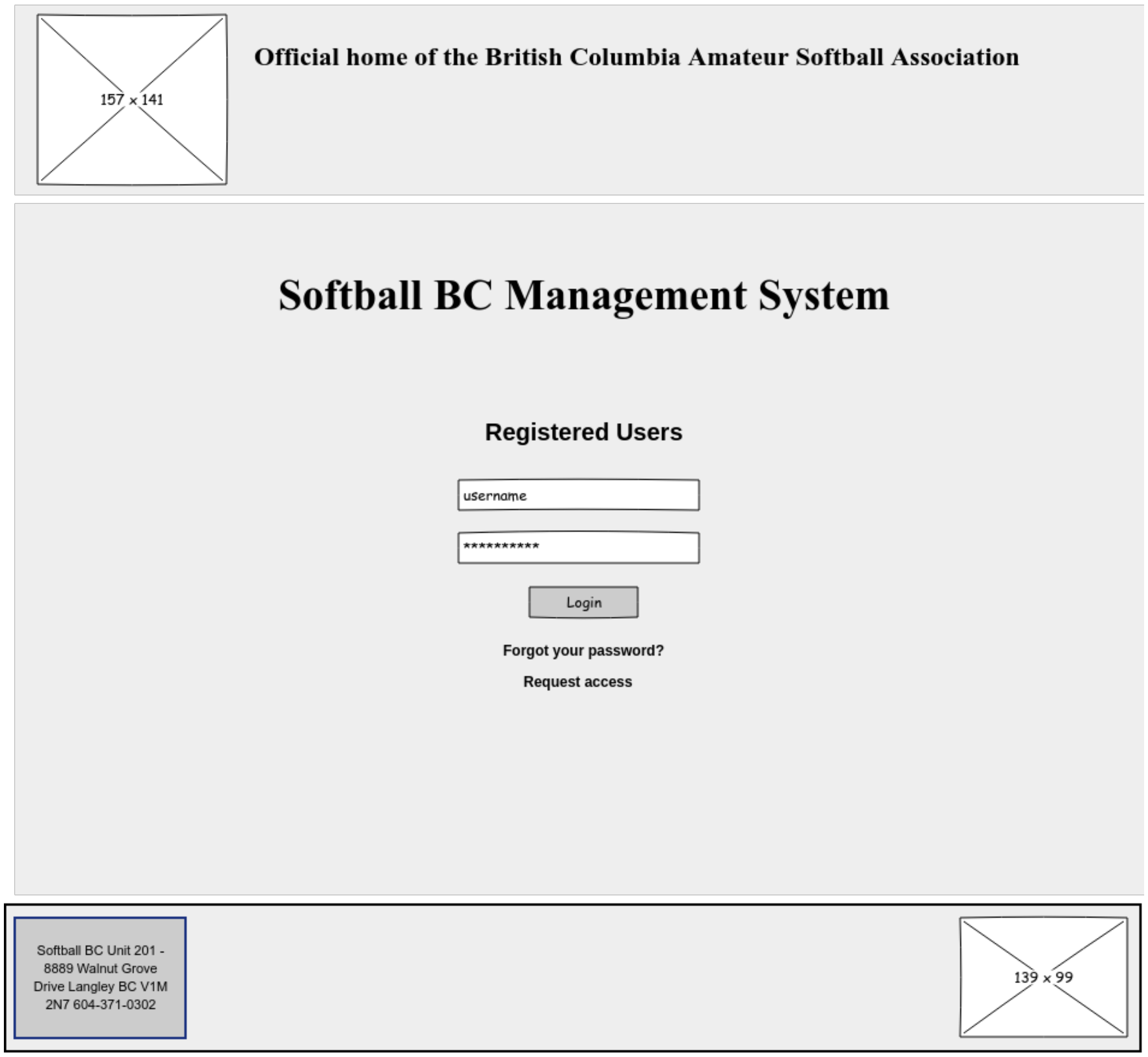
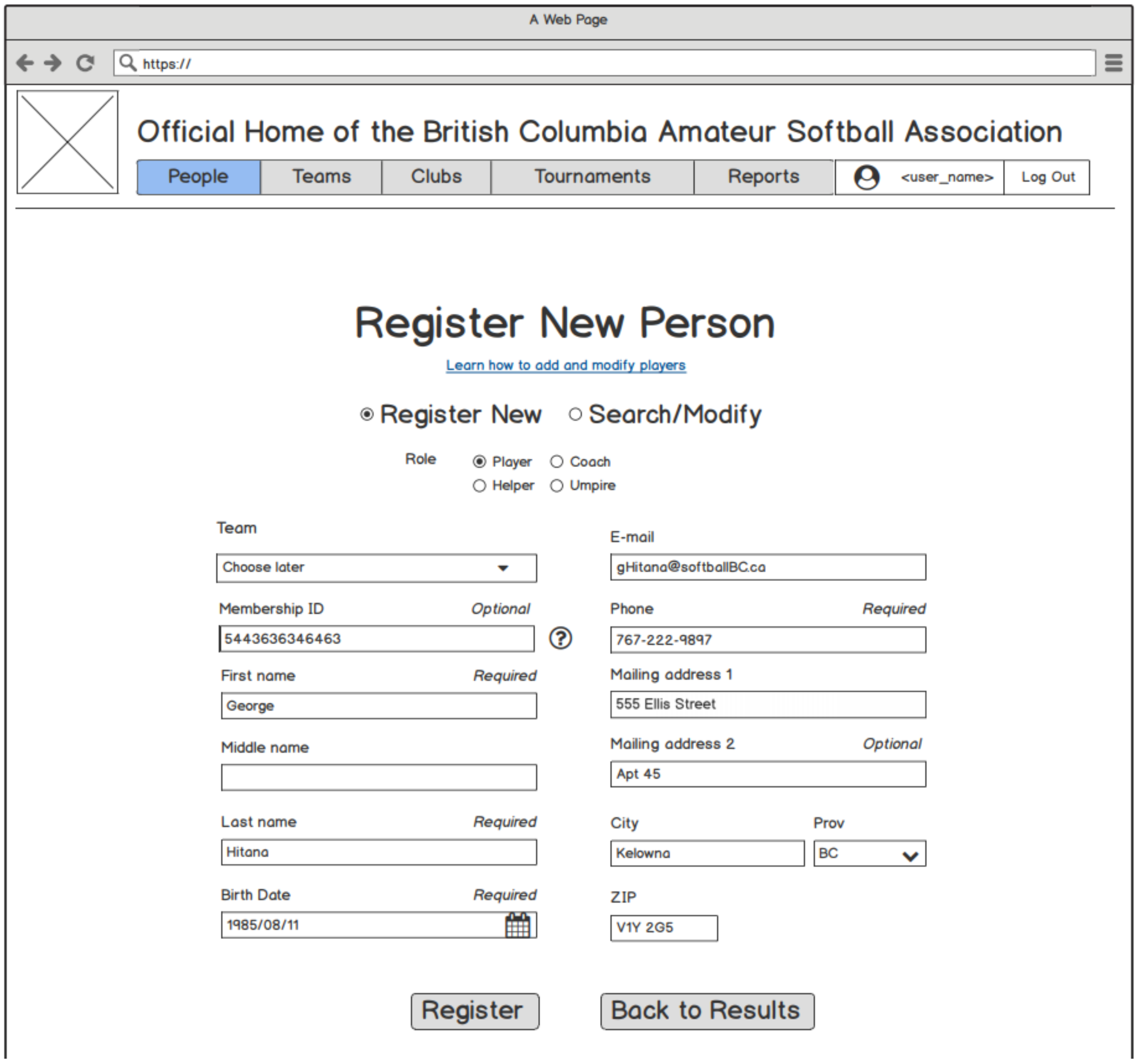

157 x 141
Official home of the British Columbia Amateur Softball Association
Softball BC Management System
Registered Users
username
****
Login
Forgot your password?
Request access
Softball BC Unit 201 - 8889 Walnut Grove Drive Langley BC V1M 2N7 604-371-0302
139 x 99

Figure 3.3a - The wireframe mock-up of the login page. Note that there is no color, or any visual flourishes, just the minimal functional structure.

Experimentation resulted in a structure and arrangement of user interactions items that balanced simplicity and functionality. Without using a wireframe program, it would be necessary to draw these by hand, in a drawing program, or code them, which would have taken much longer. This allowed for easy generation of many different pages, representing all the core functionality of the website. While the final version did not strictly adhere to these mock-ups, they were extremely useful in creating consistency in the structure and layout across web pages.



Register New Person

Learn how to add and modify players

[✓ Register New] Search/Modify

Role: [✓ Player] Coach Helper Umpire

Team	Choose later
Membership ID *(Optional)*	5443636346463
First name *(Required)*	George
Middle name	
Last name *(Required)*	Hitana
Birth Date *(Required)*	1985/08/11
E-mail	gHitana@softballBC.ca
Phone *(Required)*	767-222-9897
Mailing address 1	555 Ellis Street
Mailing address 2 *(Optional)*	Apt 45
City	Kelowna
Prov	BC
ZIP	V1Y 2G5

Figure 3.3b - Final wireframe mock-up of the register new person page.

4 - Development

4.1 - Tools

The main tool used to develop this website is the Laravel PHP Framework. PHP is a programming language that is used to create code that the web server will execute. It will handle operations such as retrieving data from the database and displaying it to the user, and storing user input into the database. A framework is a piece of software containing premade functions and shortcuts for a programming language. Frameworks condense long complicated code down into shorter, easier to remember code, and also assist in generating common website functionality such as a login system. Frameworks also enforce a specific design pattern to follow, and a structure to the project folders and files. Laravel easily integrates with CSS, HTML, and JavaScript, and with them all combined it is a powerful web development environment. [6]

The interactive development environment (IDE) used was Microsoft Visual Studio Code. This IDE is lightweight and fast, as well as having a large and active extension community. These extensions added additional functionality that such as code completion for all four languages used, automatic code restructuring, debugging, and additional support for Laravel Projects.

The virtual machine environment, Homestead, was used as the local web server containing the Laravel project, and is designed specifically for this purpose. When Homestead is launched, an Ubuntu 64 bit Linux operating system is created, fully preloaded and configured with all the software and dependencies a Laravel project needs. This Linux system operates as a local server to host the project. The project folder is linked and mirrored between the host OS and Homestead, so when one changes, the other changes. Development happens on the host OS like normal, by using an IDE to create and code the project files, but whenever the project is executed, it executes on the Homestead server. The purpose of Homestead is to ensure that regardless the host OS or host computer specifications that a developer is using, the Laravel project will execute on the *exact same* OS environment. Another benefit is that developers do not have to alter their host OS through many dependency installations. [7]

4.2 - Design Pattern

The design pattern that Laravel uses is called the Model-View-Controller (MVC) pattern. The purpose of MVC is to separate the application into three fundamental parts, the model, the view and the controller, which each representing a section that data can "flow" into. The structure of the project folders and files reflect this, which leads to a more intuitive, and easier to understand design process.

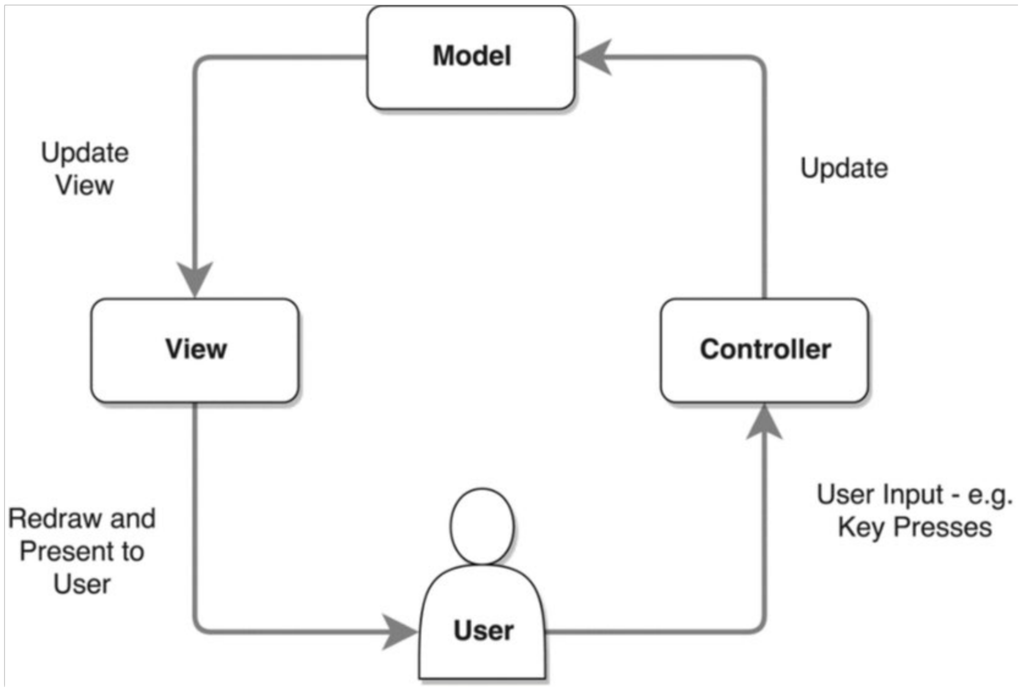


Figure 4.2a - A diagram of the MVC design pattern. [4]

- **Model:** This code contains the information that will be displayed or represented to the user in some way, and typically describes real-life things. In this case, it is the database containing all the softball information.
- **View:** This code dictates everything the user sees, and every interaction they have with the webpage. All the colors, positioning and style, as well as the logic behind every button click, form fill and dropdown menu is contained in the view code.
- **Controller:** The controller is like a bridge connecting the model and the view with each other. The user input from the view flows into the controller, and controller retrieves the appropriate functions to use to interact with the model and vice versa. For example, if a user types a team name into the search bar and clicks search, a request containing the team name gets sent from the user's browser (view) to the web server containing PHP code (controller). The web server will find the code that generates the SQL command for retrieving a team name, and it will send that SQL to the database (model). The database will return the raw team information back to the web server, which will generate web page containing that information, and send a response back to the web browser (view) where is displayed to the user. [1]

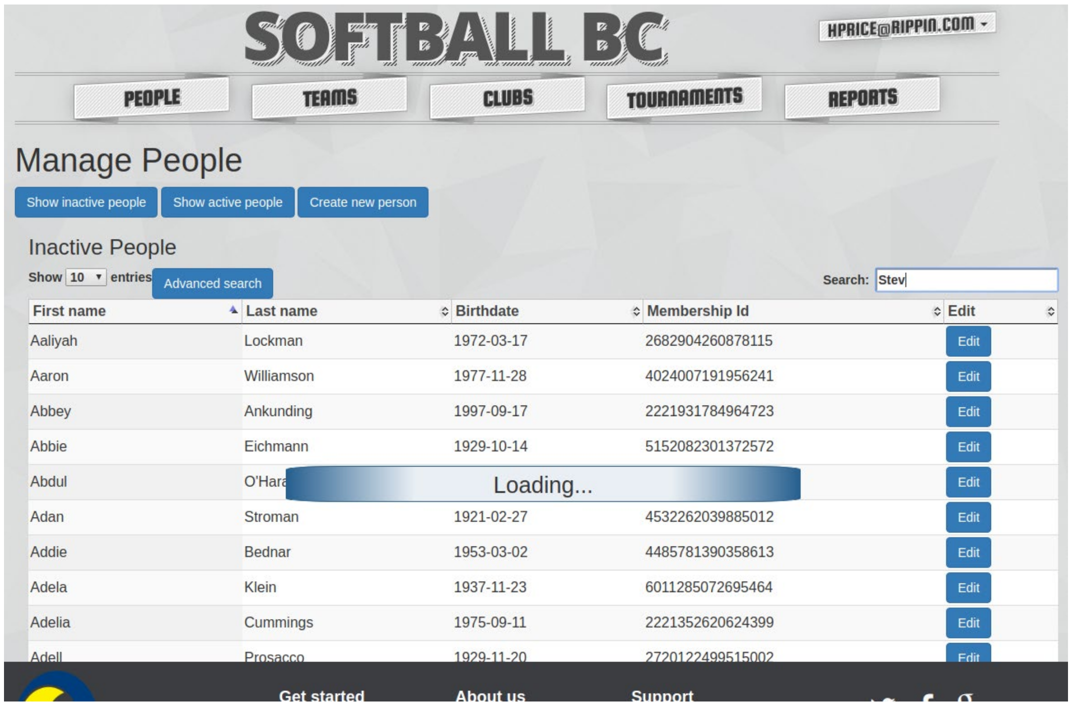
Using this model was very helpful in terms of compartmentalizing the structure of the project both physically and mentally. Whenever a new component needs to get developed, one only has to decide whether it was part of the model, view or controller. This somewhat narrows down the filetype, language and structure to use for the component.

4.3 - Applied Design Principles

While developing this website, Dr. Jakob Nielson's *Useability Heuristics for User interface Design* were applied to the creation of the user experience. Dr. Nielson is a Human Computer Interaction and User Interface researcher, whose contributions to the fields have become standard guidelines for the development of a user experience. These heuristics are a list of ten principles that are widely accepted as creating an optimal user experience for an application. They are called heuristics because they are not detailed rules that must be followed exactly, but general concepts that focus on avoiding situations where the user may become *impatient, confused or frustrated*. [2]

4.3.1 - Visibility of System Status

This heuristic is concerned with ensuring that the user is aware of the application's current state. State refers to the current action the application is performing, or if not performing an action, the input that the application is expecting. The user should never have to wonder what state the application is in. If data is being loaded, there should be a message displayed telling the user this, or if the application is expecting input, the possible interactions should be clearly labeled, or a message such as "Please enter a team name..." should be displayed. These displayed messages are considered application feedback. [1][2]



Manage People

Inactive People

Search: Stev

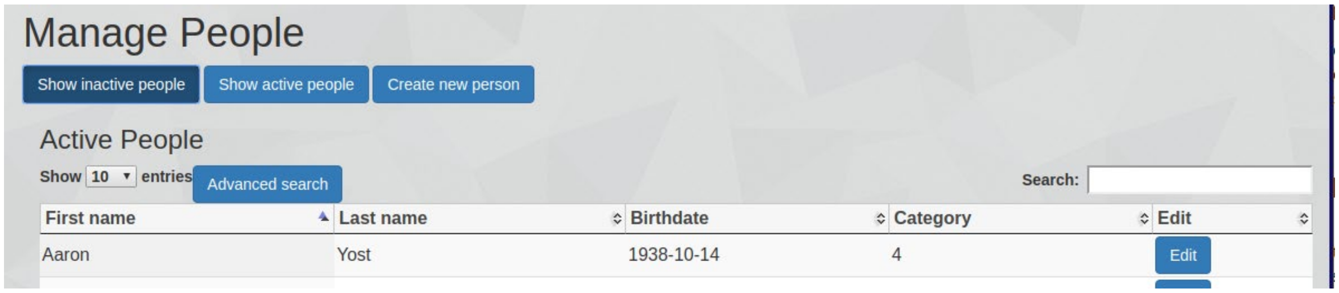
First name	Last name	Birthdate	Membership Id	Edit
Aaliyah	Lockman	1972-03-17	2682904260878115	Edit
Aaron	Williamson	1977-11-28	4024007191956241	Edit
Abbey	Ankunding	1997-09-17	2221931784964723	Edit
Abbie	Eichmann	1929-10-14	5152082301372572	Edit
Abdul	O'Hara			Edit
Adan	Stroman	1921-02-27	4532262039885012	Edit
Addie	Bednar	1953-03-02	4485781390358613	Edit
Adela	Klein	1937-11-23	6011285072695464	Edit
Adelia	Cummings	1975-09-11	2221352620624399	Edit
Adell	Prosacco	1929-11-20	2720122499515002	Edit

Figure 4.31a- Application search feedback in the form of a "Loading..." bar.

According to the findings in a 2004 study, after an interaction delay of more than two seconds, a user's attention begins to wander, and they may become impatient [3]. Since the database for this management system may potentially store tens of thousands of records, and involve complex queries combining many different tables, search times may take a few seconds or more. This makes it critical to include feedback to the user in some form of a "Searching..." or "Loading..." bar. If this feedback was not provided, a user may think that their search query was not registered by the system, or there is an unseen error, and they might try typing their query again, further delaying their results. The display of feedback is essentially telling the user "I acknowledge your interaction, and am now retrieving data." [2]

4.3.2 - Flexibility and Efficiency of Use

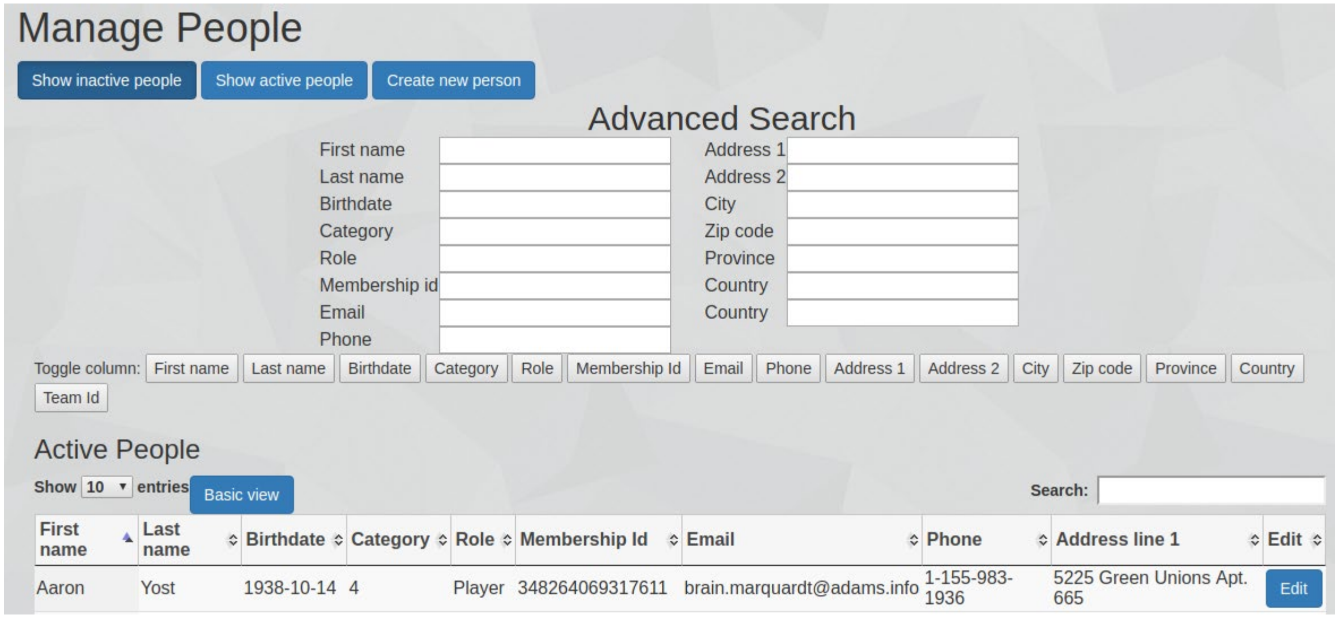
This heuristic focuses on providing optional interactions for both novice and advanced users. There will be many different users of various computer skill levels using this website, and the goal is to make sure they can all interact with it in a way that is comfortable for them. The novice user should not be overwhelmed with too many options or settings to choose from, because they may prefer the simplest way to accomplish a task even if it takes them longer. The advanced user will be more comfortable, and prefer interacting with lots of options and complex settings, as it will save them time in their current task. [2]



First name	Last name	Birthdate	Category	Edit
Aaron	Yost	1938-10-14	4	Edit

Figure 4.32a - Both basic, and advanced search options are available to the user

In the above figure, both basic and advanced search options are available to the user in an unobtrusive way.



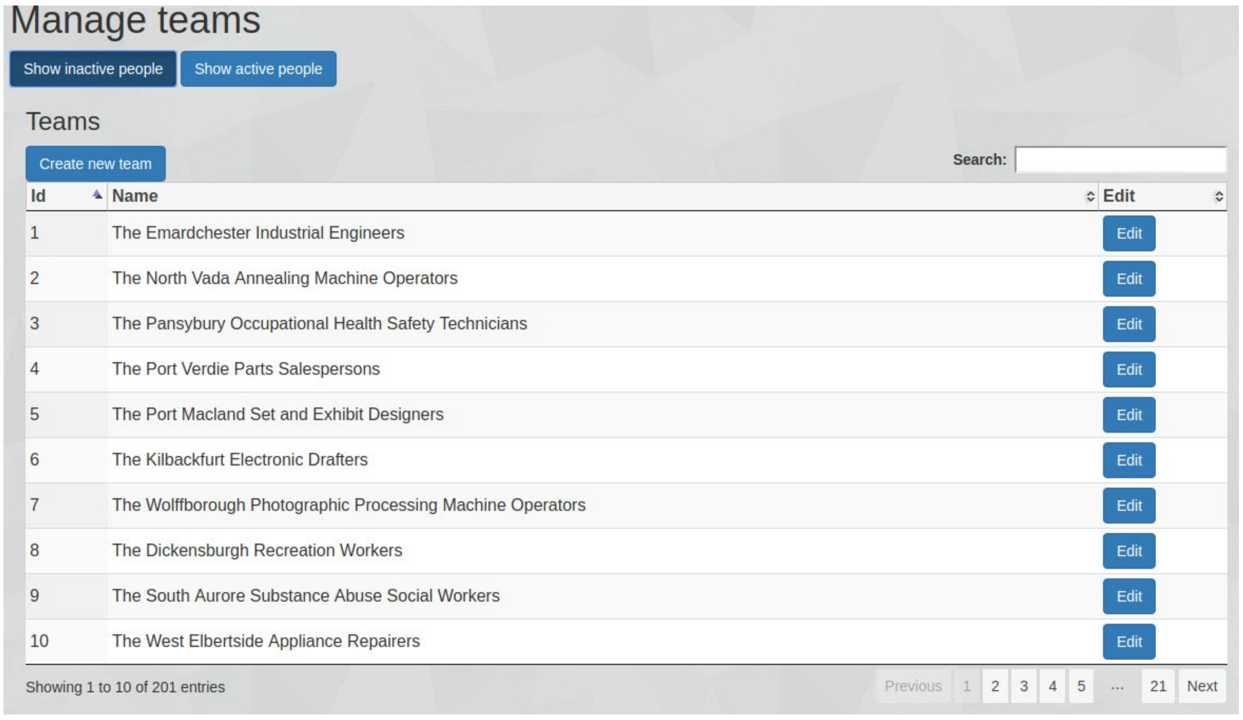
First name	Last name	Birthdate	Category	Role	Membership Id	Email	Phone	Address line 1	Edit
Aaron	Yost	1938-10-14	4	Player	348264069317611	brain.marquardt@adams.info	1-155-983-1936	5225 Green Unions Apt. 665	Edit

Figure 4.32b - After the advanced search button has been pressed.

The advanced search feature has many different options to interact with. If the novice user was immediately presented with these advanced options, or if it was the only search option, they may become confused and frustrated. The novice user may feel more comfortable scrolling through basic search results to find what they're looking for, even though it takes longer. The opposite is true for the advanced user. Only having the basic search option available may frustrate them because they want to perform a more complex search, which will be time consuming to perform through a basic search. [2]

4.3.3 - Aesthetic and Minimalist Design

The goal of this heuristic is to prevent displaying unnecessary, or excess information to the user. At any given moment throughout the user's experience interacting with the website, only the immediately important, relevant information should be the focus of their attention. For example, the system should prevent the user from opening too many menus or windows at once, and having all of them compete for space on the screen, leading to confusion or frustration. The user does not always know what they're doing when interacting with a website, so there should be some gentle boundaries set in place to prevent them from creating these messy situations, while not making them feel constrained. [2]

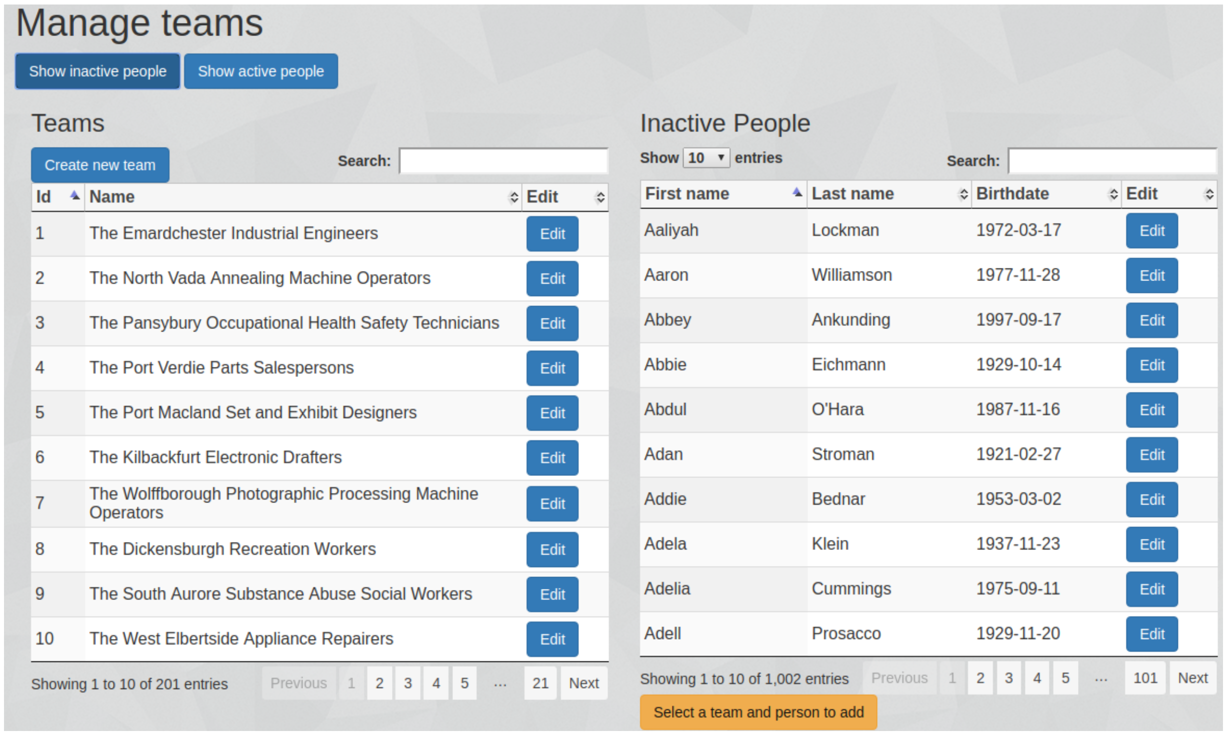


Id	Name	Edit
1	The Emardchester Industrial Engineers	Edit
2	The North Vada Annealing Machine Operators	Edit
3	The Pansybury Occupational Health Safety Technicians	Edit
4	The Port Verdie Parts Salespersons	Edit
5	The Port Macland Set and Exhibit Designers	Edit
6	The Kilbackfurt Electronic Drafters	Edit
7	The Wolffborough Photographic Processing Machine Operators	Edit
8	The Dickensburgh Recreation Workers	Edit
9	The South Aurore Substance Abuse Social Workers	Edit
10	The West Elbertside Appliance Repairers	Edit

Showing 1 to 10 of 201 entries

Figure - 4.33a - The team page.

When a user navigates to the teams page, this is the default page that loads. It is not yet known what action the user is going to perform, but it will involve a team in some way, so the list of teams is the main focus of the page.



Manage teams

Show inactive people | Show active people

Teams

Create new team

Search:

Id	Name	Edit
1	The Emardchester Industrial Engineers	Edit
2	The North Vada Annealing Machine Operators	Edit
3	The Pansybury Occupational Health Safety Technicians	Edit
4	The Port Verdie Parts Salespersons	Edit
5	The Port Macland Set and Exhibit Designers	Edit
6	The Kilbackfurt Electronic Drafters	Edit
7	The Wolffborough Photographic Processing Machine Operators	Edit
8	The Dickensburgh Recreation Workers	Edit
9	The South Aurore Substance Abuse Social Workers	Edit
10	The West Elbertside Appliance Repairers	Edit

Showing 1 to 10 of 201 entries   Previous 1 2 3 4 5 … 21 Next

Inactive People

Show 10 entries   Search:

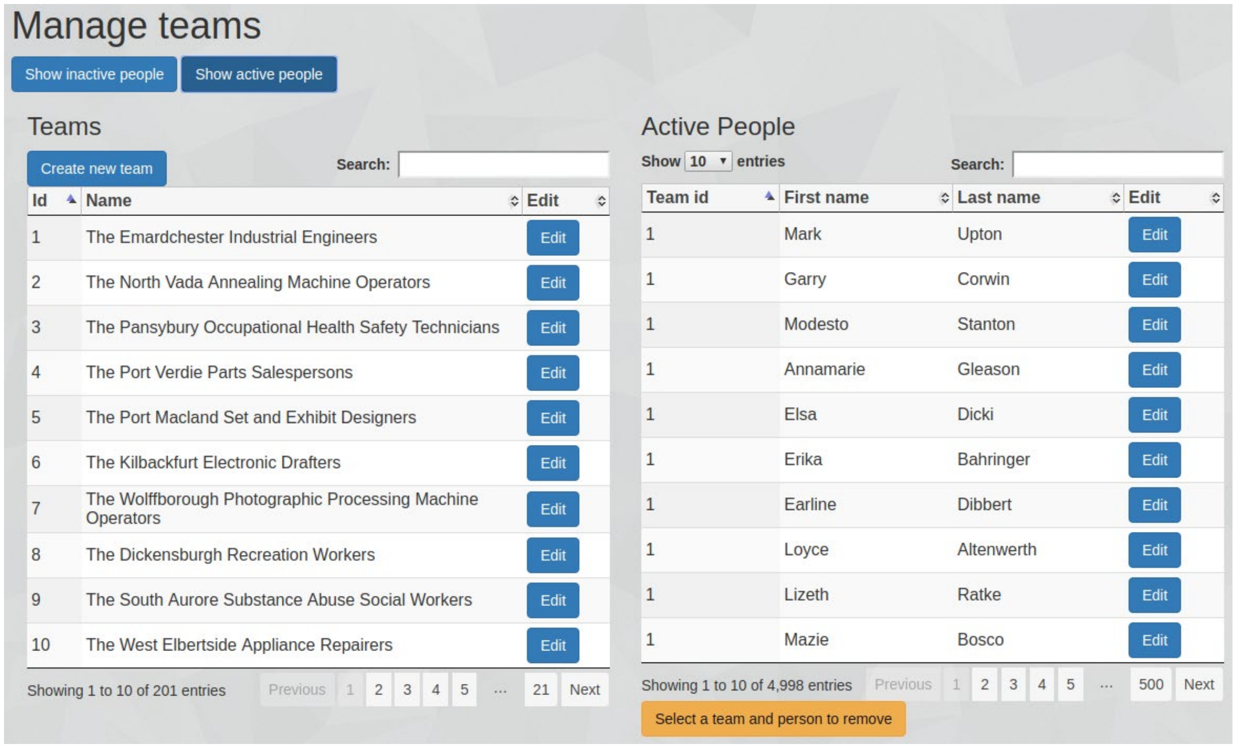
First name	Last name	Birthdate	Edit
Aaliyah	Lockman	1972-03-17	Edit
Aaron	Williamson	1977-11-28	Edit
Abbey	Ankunding	1997-09-17	Edit
Abbie	Eichmann	1929-10-14	Edit
Abdul	O'Hara	1987-11-16	Edit
Adan	Stroman	1921-02-27	Edit
Addie	Bednar	1953-03-02	Edit
Adela	Klein	1937-11-23	Edit
Adelia	Cummings	1975-09-11	Edit
Adell	Prosacco	1929-11-20	Edit

Showing 1 to 10 of 1,002 entries   Previous 1 2 3 4 5 … 101 Next

Select a team and person to add

Figure 4.33b - The "Show inactive people" button has been clicked.

After clicking the "Show inactive person" button, the inactive person information appears on the right side, compacting the team information so they both share the space equally. It is appropriate for them to share the focus because the functionality of this page is to add and remove *people* from *teams*.


Manage teams

Teams

Id	Name	Edit
1	The Emardchester Industrial Engineers	Edit
2	The North Vada Annealing Machine Operators	Edit
3	The Pansybury Occupational Health Safety Technicians	Edit
4	The Port Verdie Parts Salespersons	Edit
5	The Port Macland Set and Exhibit Designers	Edit
6	The Kilbackfurt Electronic Drafters	Edit
7	The Wolffborough Photographic Processing Machine Operators	Edit
8	The Dickensburgh Recreation Workers	Edit
9	The South Aurore Substance Abuse Social Workers	Edit
10	The West Elbertside Appliance Repairers	Edit

Showing 1 to 10 of 201 entries

Active People

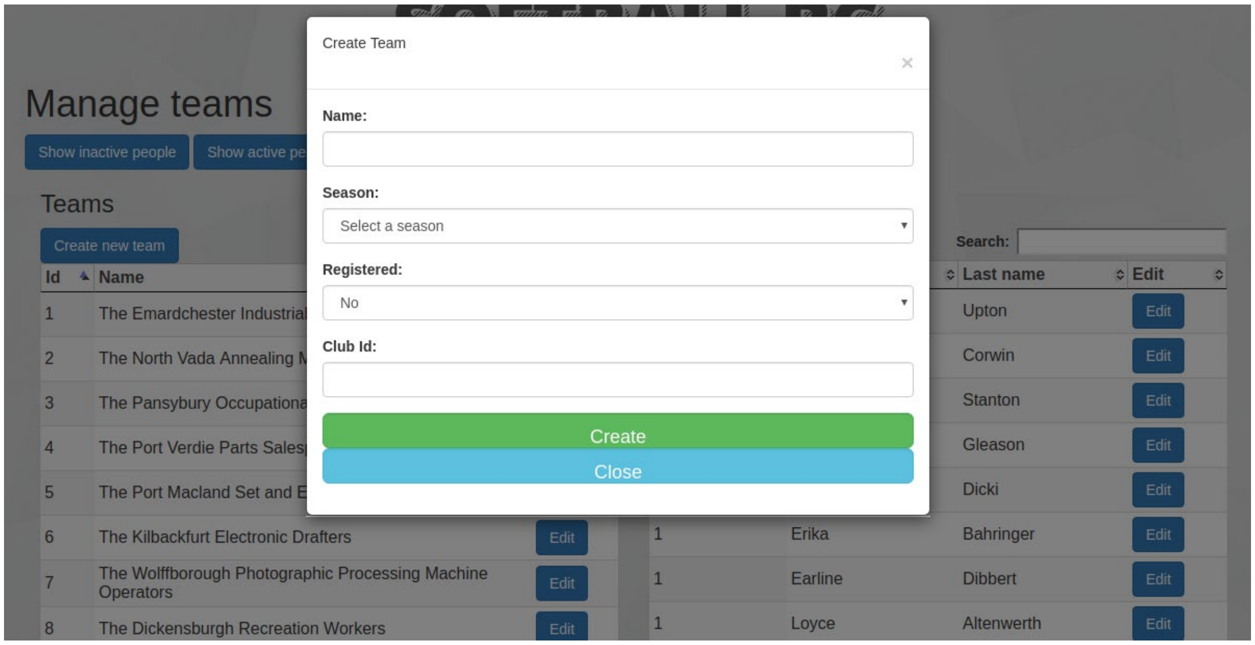
Team id	First name	Last name	Edit
1	Mark	Upton	Edit
1	Garry	Corwin	Edit
1	Modesto	Stanton	Edit
1	Annamarie	Gleason	Edit
1	Elsa	Dicki	Edit
1	Erika	Bahringer	Edit
1	Earline	Dibbert	Edit
1	Loyce	Altenwerth	Edit
1	Lizeth	Ratke	Edit
1	Mazie	Bosco	Edit

Showing 1 to 10 of 4,998 entries

Previous 1 2 3 4 5 … 500 Next
Select a team and person to remove

Figure 4.33c - After clicking the "Show active people" button, it replaces the inactive people.

If the inactive people information is being displayed, and the "Show active people" button is clicked, the active people information will replace the inactive people information on the right. This is because there is no added benefit or functionality to displaying all three windows at once (team, active and inactive people). It would result in irrelevant information on the screen, since the user can only interact with a single person type at once in any meaningful way. It may also create clutter and confusion for the user. Note that once the people buttons are deselected, the team information returns to filling the full width, since it is now the only focus.


Manage teams

Show inactive people | Show active pe

Teams

Create new team

Id	Name	
1	The Emardchester Industrial	
2	The North Vada Annealing M	
3	The Pansybury Occupationa	
4	The Port Verdie Parts Salesp	
5	The Port Macland Set and E	
6	The Kilbackfurt Electronic Drafters	Edit
7	The Wolffborough Photographic Processing Machine Operators	Edit
8	The Dickensburgh Recreation Workers	Edit

Search:

		Last name	Edit
		Upton	Edit
		Corwin	Edit
		Stanton	Edit
		Gleason	Edit
		Dicki	Edit
1	Erika	Bahringer	Edit
1	Earline	Dibbert	Edit
1	Loyce	Altenwerth	Edit

Create Team

Name:

Season: Select a season

Registered: No

Club Id:

Figure 4.33d - After clicking the "Create new team" button.

Lastly, when the user clicks the "Create new team" button, a window is displayed in front of all other information. Once displayed, the functionality of this form no longer relies on the user clicking any other previous content, so it is grayed out. After "Create" is clicked, the window disappears, and the grayed out information is returned to its normal color as the focus of the user.

4.4 - Underlying Technologies

4.4.1 - AJAX

The most frequently used functionality of this website is the ability to efficiently search for data stored in the database, and display it in an easy to understand fashion. One of the main concerns regarding the current Softball BC website is that the retrieval and display of information is done in a synchronous manner. This means that when the user retrieves information from database via the web server (i.e. a team name search), response containing the information will be returned and displayed as an entirely new webpage.

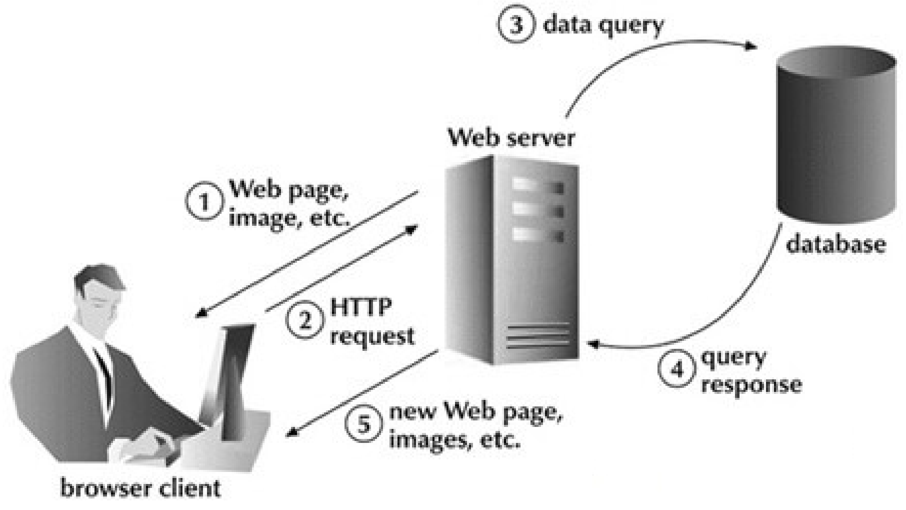
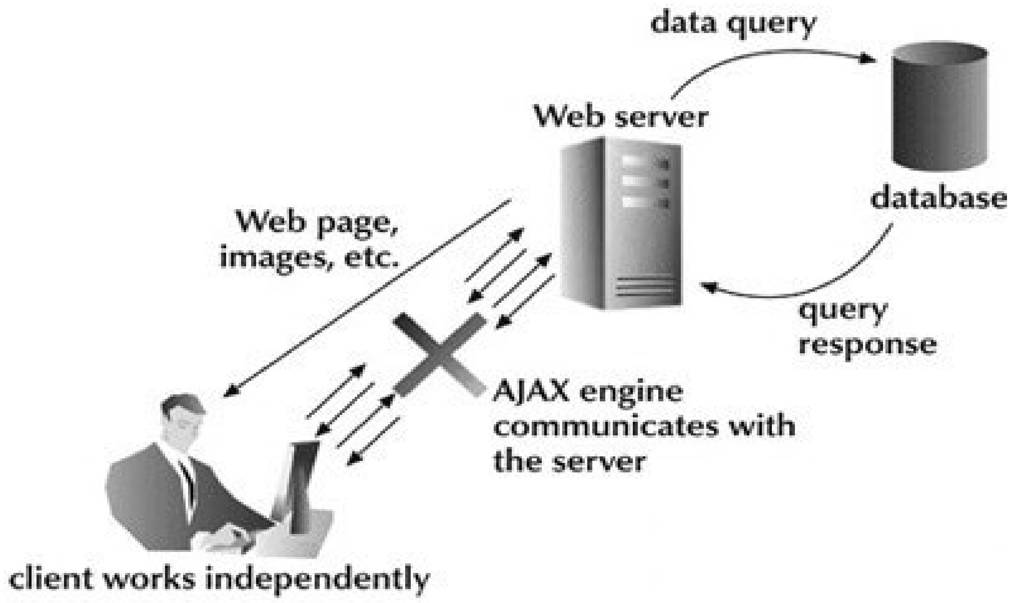


Figure 4.41a - A diagram of a synchronous HTTP request. [5]

It is called synchronous because the action of data retrieval (web server), and data display (client browser) are synchronized to the same internal clock, or computer processing time. This means that one must happen before the other. They cannot simultaneously execute. Therefore the team search page cannot be continuously displayed at the same time the information is being retrieved, so the search page must be reloaded before it can display the results. [1]

While this may seem like a small issue, given the number of requests needed to search, filter, and navigate through thousands of rows of data, it quickly adds up to wasted time and bandwidth. This is because when the page reloads to display the retrieved data, instead of only updating the results section of the page, it updates the *entire* page, even content that has *not* changed. This results in completely unnecessary requests that consume time and bandwidth. [1]



data query
Web server
database
Web page, images, etc.
query response
AJAX engine communicates with the server
client works independently

Figure 4.41b - Asynchronous HTTP request. [5]

The solution to this situation is to use a development approach called AJAX. This stands for Asynchronous JavaScript and XML, and refers a set of programming languages and techniques that enable the client to send and receive requests and responses asynchronously. This means that the client's browser is no longer synchronized with the web server, so it may perform simultaneous actions that are hidden from the user. JavaScript is used to interact with the data from a request, and then manipulate HTML elements and CSS to display the information. [1]

4.4.2 - PostgreSQL

PostgreSQL is a relational database management system. It is program that defines, creates, and manages a relational database, as well as providing a way to create, retrieve, update and delete data. I chose to use PostgreSQL for a few reasons:

1. **PostgreSQL is free and open-source:** Free is great, I prefer open source software because I believe it is generally more secure and stable. Open source means that anyone can view and change the source code for the software without permission from the original authors. Because of this, the possibility of bugs and errors being discovered and fixed is more likely compared to a piece of proprietary software. A proprietary software company may have upwards of one hundred developers working on a large project, while a piece of open source software could potentially be used and inspected by thousands of developers of varying skill level, increasing the odds of bugs and vulnerabilities being found.
2. **Laravel supports PostgreSQL:** Laravel comes with PostgreSQL drivers and configuration files, so it was very easy to get Laravel interacting with the database.
3. **PostgreSQL supports procedural languages:** This allows the use of an adapter provide database interaction from other programming languages such as Java, Python or R. I was hoping to have a chance to explore this when working on advanced functionality (such as data season data analysis using Python Pandas), but unfortunately the time constraints did not allow me to.
4. **PostgreSQL is fully ACID compliant:** See section 4.4.3 below.

4.4.3 - ACID

An event that causes a database to internally change in *any way* is called a "Transaction". This could mean that a piece of data has moved in, or out of a database, or it could also refer to any internal event such as the database restructuring or optimizing itself. Everything is a transaction essentially. Transactions are the most basic unit of a database operation, so it is critical that they perform in such a way that all database functionality is fully supported. [8]

ACID stands for Atomicity, Consistency, Isolation, and Durability, and are properties that all transactions should have. [8]

- **Atomicity:** If you are sending an active person record to the database for storage, and the connection gets interrupted before the transaction is completed, it would not be very useful to store a partial record. Even worse, that player record may become corrupted and permanently lost. Making transactions "Atomic" means that they can either successfully complete, or not complete at all. There are no partial transactions. This ensures that if a transaction is interrupted, no data will be lost or corrupted.

- **Consistency:** A "Database state" is like a snapshot of the entire database and all its stored information in any given moment. A database is said to be in a "Consistent state" when there are no violations of the enforced database schema. A database that enforces consistency ensures that a consistent state is maintained at all times before and after any transactions.
- **Isolation:** A database can perform many different transactions at the same time. It is important that these concurrently executing transactions do not alter the database in such a way that other transactions are affected. This could lead to unpredictable results and possible data corruption. A database that enforces isolation ensures that transactions are managed in this.
- **Durability:** If a database suddenly loses power, or suffers a non-data related internal component failure, the stored data should still exists after the database is restored. A durable database is one that permanently stores the effects of any completed transaction regardless of any failure.

These four properties ensure robust core database performance, allowing it to operate in a predictable and stable manner when handling queries and executing internal functions. Most importantly, the database can be recovered in emergency situations where power is lost, or non-data components of the database become unstable or corrupt. [8]

5 - Walkthrough and Core Functionality

5.1 - Login and Create New Team

The first page the user sees is the login page. Since the new registration process will be handled personally by a Softball BC representative, we will assume the user already has an account.

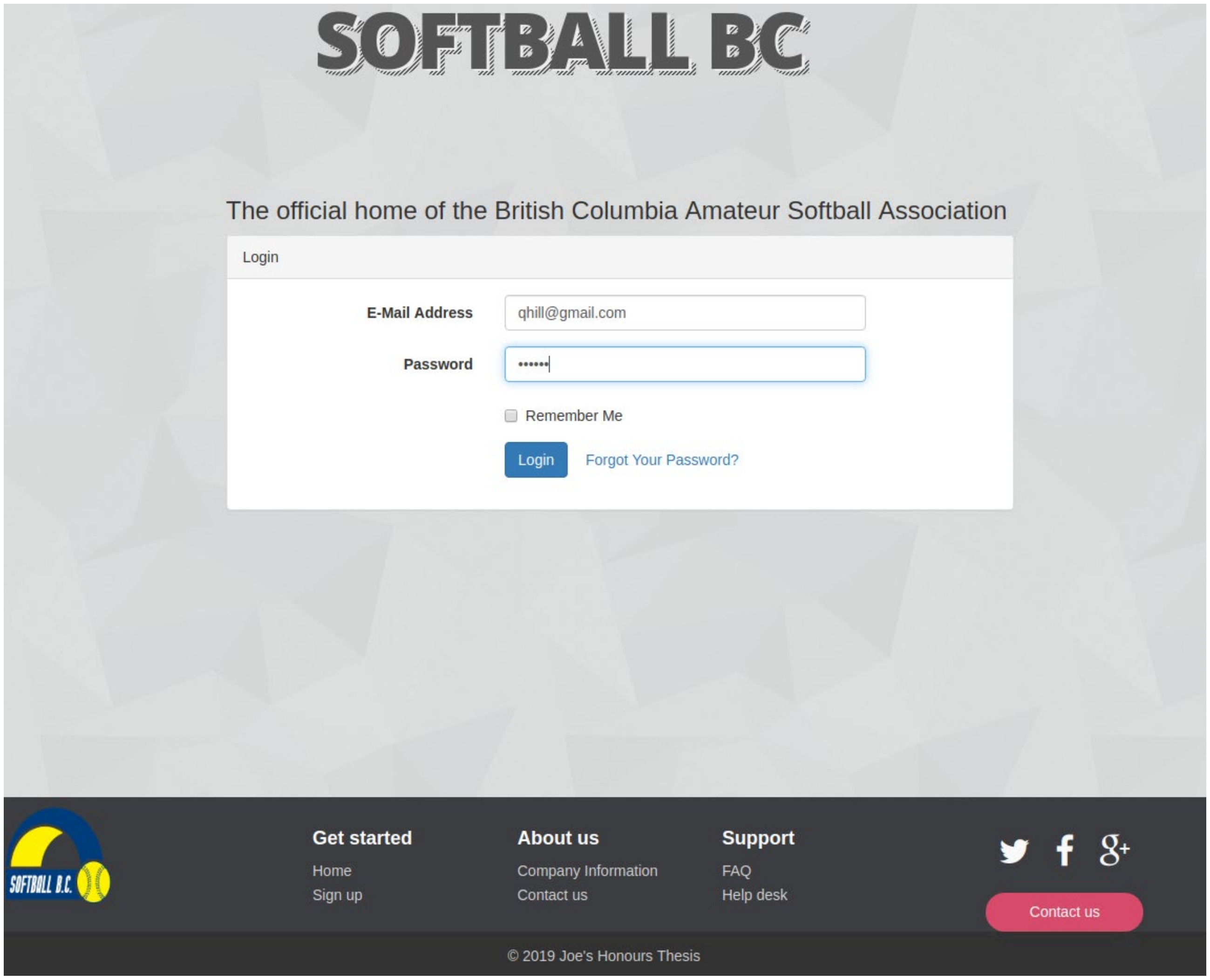
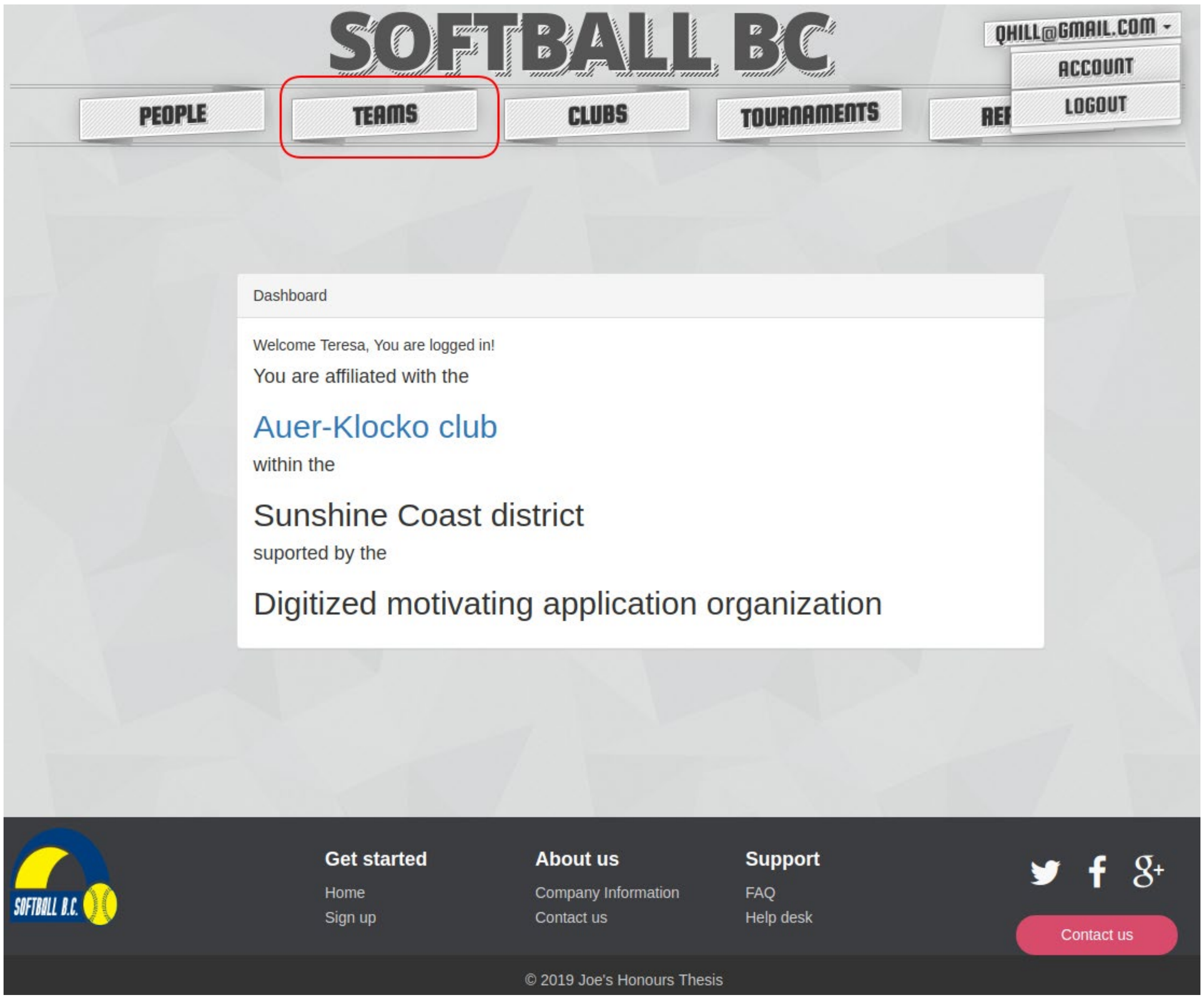


Figure 5.1a - The user logs in.

After they have logged in, the database will retrieve that user's record and populate a personalized welcome page displaying their club, district and organization affiliations. The user navigation menu, as well as the user drop down menu are now available at the top. First, the user wants to create a new team and add players to it. They click on the "Teams" button in the user navigation bar and then click the "Create new team" button. After they fill in the team information, they click the "Create" button, and a new team record is stored in the database in the "teams" table. A message will display telling the user the team was successfully created.



Dashboard

Welcome Teresa, You are logged in!

You are affiliated with the

Auer-Klocko club

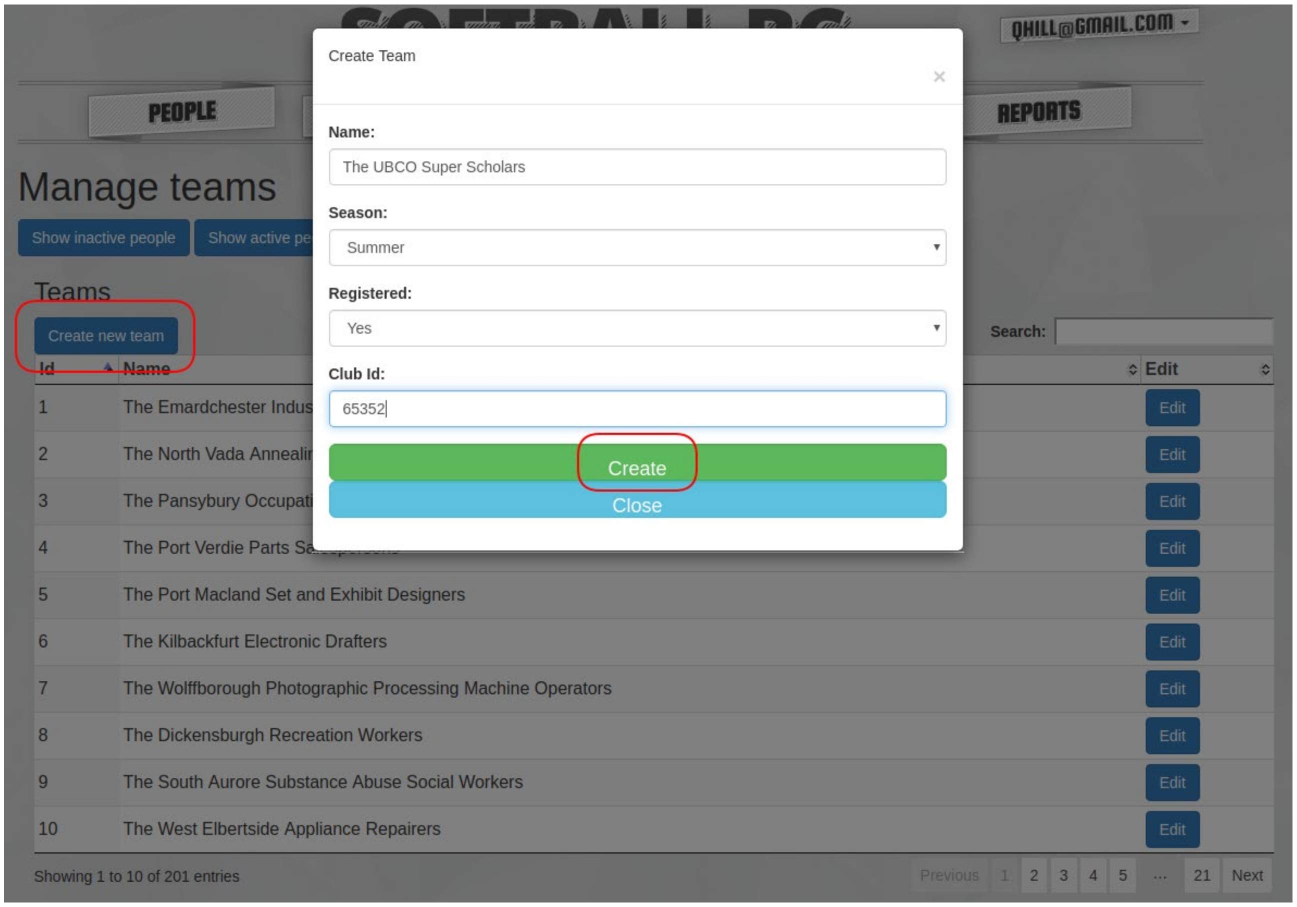
within the

Sunshine Coast district

suported by the

Digitized motivating application organization

Figure 5.1b - User welcome page.



Manage teams

Teams

Create new team

Create Team

Name: The UBCO Super Scholars

Season: Summer

Registered: Yes

Club Id: 65352

Create

Close

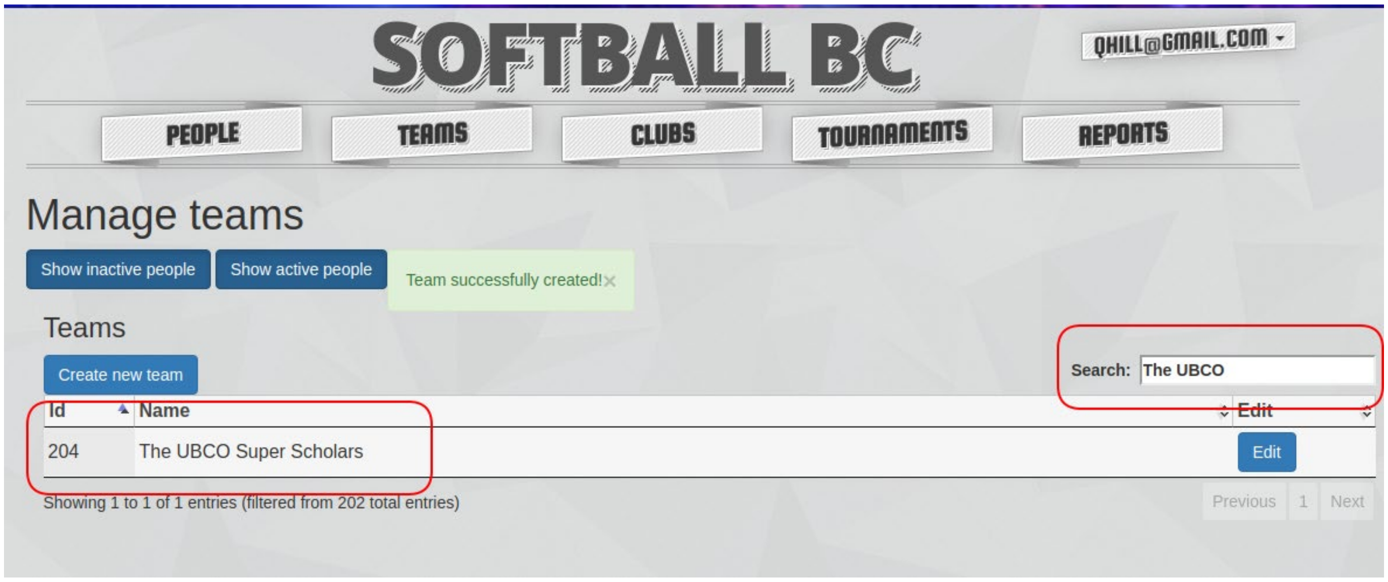
Search:

Id	Name	Edit
1	The Emardchester Indus	Edit
2	The North Vada Annealir	Edit
3	The Pansybury Occupati	Edit
4	The Port Verdie Parts Sa	Edit
5	The Port Macland Set and Exhibit Designers	Edit
6	The Kilbackfurt Electronic Drafters	Edit
7	The Wolffborough Photographic Processing Machine Operators	Edit
8	The Dickensburgh Recreation Workers	Edit
9	The South Aurore Substance Abuse Social Workers	Edit
10	The West Elbertside Appliance Repairers	Edit

Showing 1 to 10 of 201 entries

Figure 5.1c - Team creation.

Now to access their new team, the user searches for it in the search bar and it appears in the results.



Manage teams

Team successfully created!

Teams

Search: The UBCO

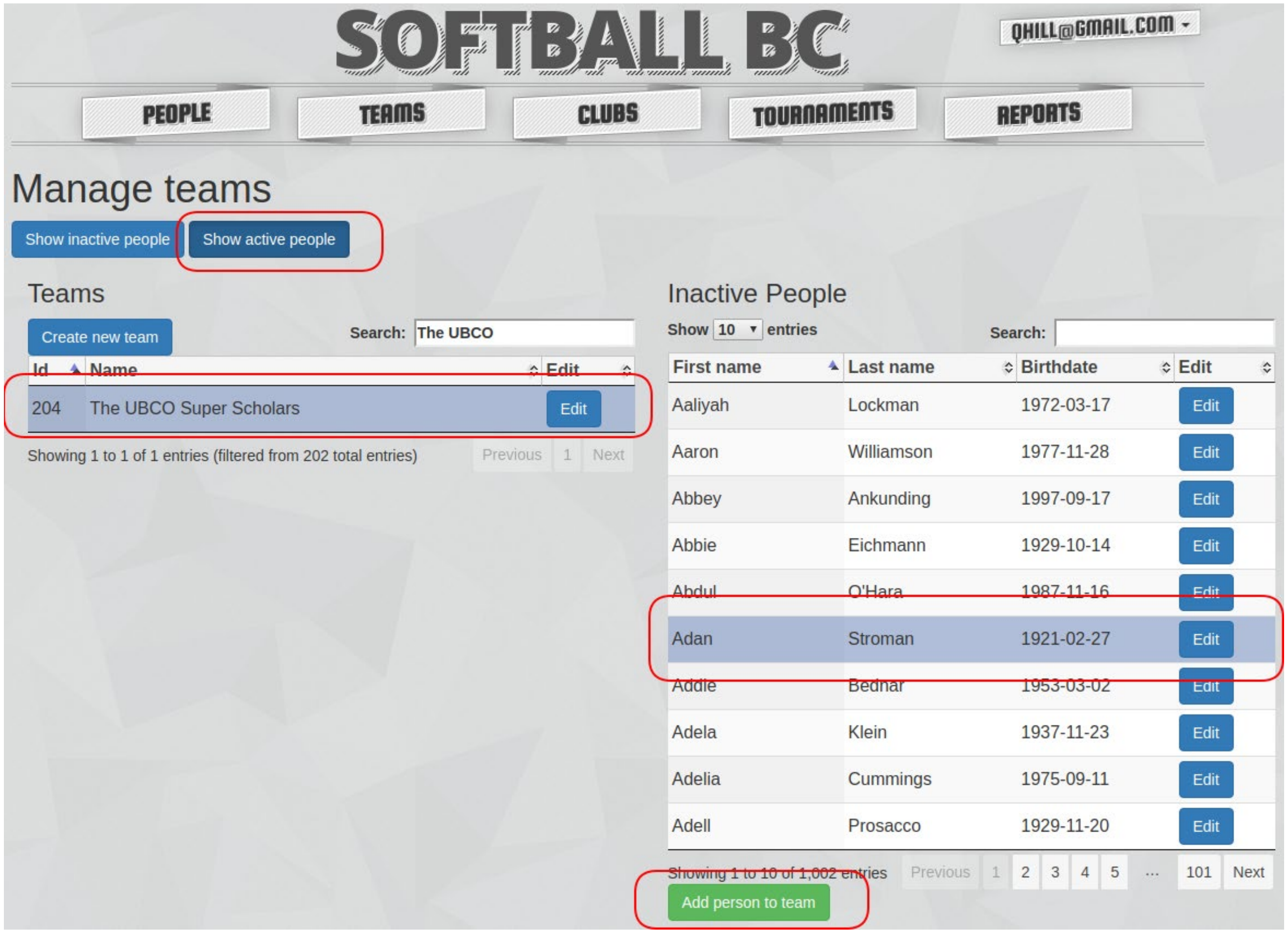
Id	Name	Edit
204	The UBCO Super Scholars	Edit

Showing 1 to 1 of 1 entries (filtered from 202 total entries)

Figure 5.1d - The newly created team has been successfully stored in the database.

5.2 - Add Player to Team and Remove Player from Team

The user now wants to add people to the new team, so they click on the "Show inactive people" button. Now they click "The UBCO Super Scholars" team to select it, select an inactive person from the list, and then click the "Add person to team" button. That person will then be removed from the "Inactive People" list. The user then repeats this until they have added all the players they need to.



Manage teams

Teams

Search: The UBCO

Id	Name	Edit
204	The UBCO Super Scholars	Edit

Showing 1 to 1 of 1 entries (filtered from 202 total entries)

Inactive People

Show 10 entries

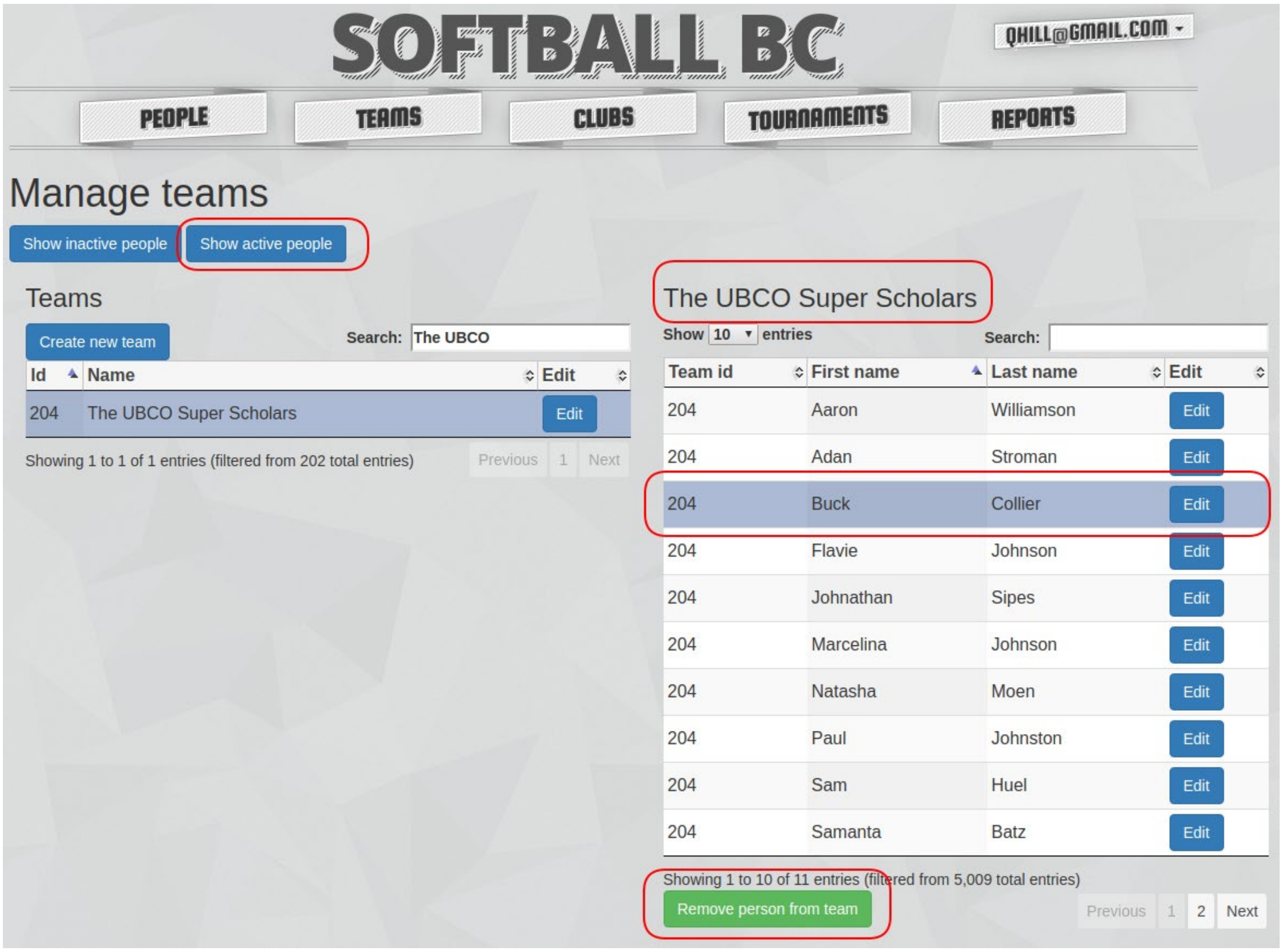
Search:

First name	Last name	Birthdate	Edit
Aaliyah	Lockman	1972-03-17	Edit
Aaron	Williamson	1977-11-28	Edit
Abbey	Ankunding	1997-09-17	Edit
Abbie	Eichmann	1929-10-14	Edit
Abdul	O'Hara	1987-11-16	Edit
Adan	Stroman	1921-02-27	Edit
Addie	Bednar	1953-03-02	Edit
Adela	Klein	1937-11-23	Edit
Adelia	Cummings	1975-09-11	Edit
Adell	Prosacco	1929-11-20	Edit

Showing 1 to 10 of 1,002 entries Previous 1 2 3 4 5 … 101 Next

Figure 5.2a- Adding an inactive player to a team.

Now, with the team still selected, the user clicks on the "Show active people" button to view the roster. Notice that the team name is at the top of the active player list. This means we are viewing the roster for that team. The user realizes that Buck had been permanently banned last season for spitting on the umpire, so the user removes Buck from the team by selecting him, and clicking the "Remove person from team" button. Buck will be removed from the active player list, and added back to the inactive player list.



Manage teams

Teams

Search: The UBCO

Id	Name	Edit
204	The UBCO Super Scholars	Edit

Showing 1 to 1 of 1 entries (filtered from 202 total entries)

Previous 1 Next

The UBCO Super Scholars

Show 10 entries

Search:

Team id	First name	Last name	Edit
204	Aaron	Williamson	Edit
204	Adan	Stroman	Edit
204	Buck	Collier	Edit
204	Flavie	Johnson	Edit
204	Johnathan	Sipes	Edit
204	Marcelina	Johnson	Edit
204	Natasha	Moen	Edit
204	Paul	Johnston	Edit
204	Sam	Huel	Edit
204	Samanta	Batz	Edit

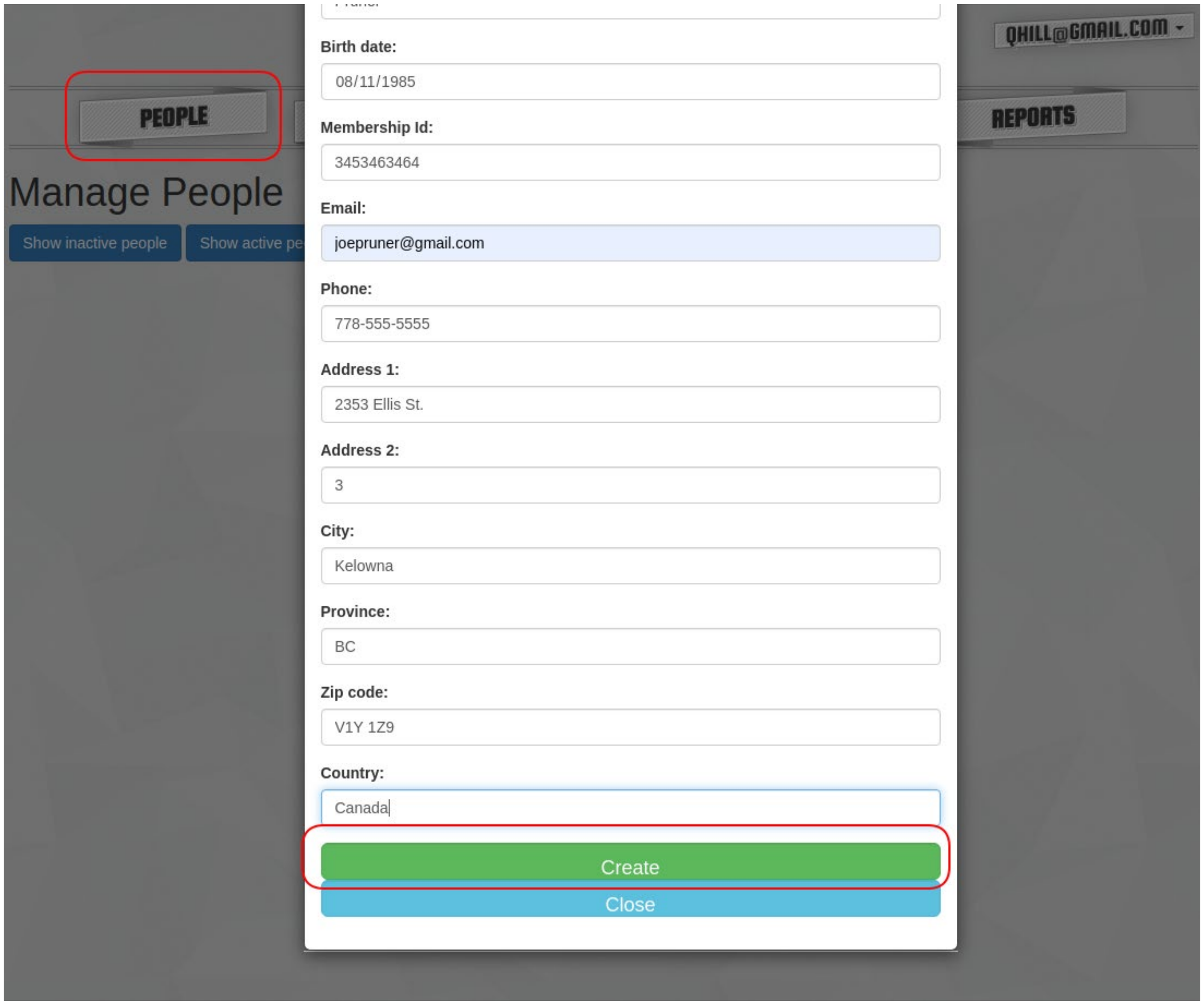
Showing 1 to 10 of 11 entries (filtered from 5,009 total entries)

Previous 1 2 Next

Figure 5.2b - Buck is being removed from the team.

5.3 - Create New Person

The user then wants to create a new person to add to their newly created team. The user clicks the "People" button at the top navigation bar, then clicks the "Create new person" button. The user fills in that person's information into the form that pops up, then they click the "Create" button.


QHILL@GMAIL.COM

PEOPLE

REPORTS

Manage People

Show inactive people

Show active pe

Birth date:

08/11/1985

Membership Id:

3453463464

Email:

joepruner@gmail.com

Phone:

778-555-5555

Address 1:

2353 Ellis St.

Address 2:

3

City:

Kelowna

Province:

BC

Zip code:

V1Y 1Z9

Country:

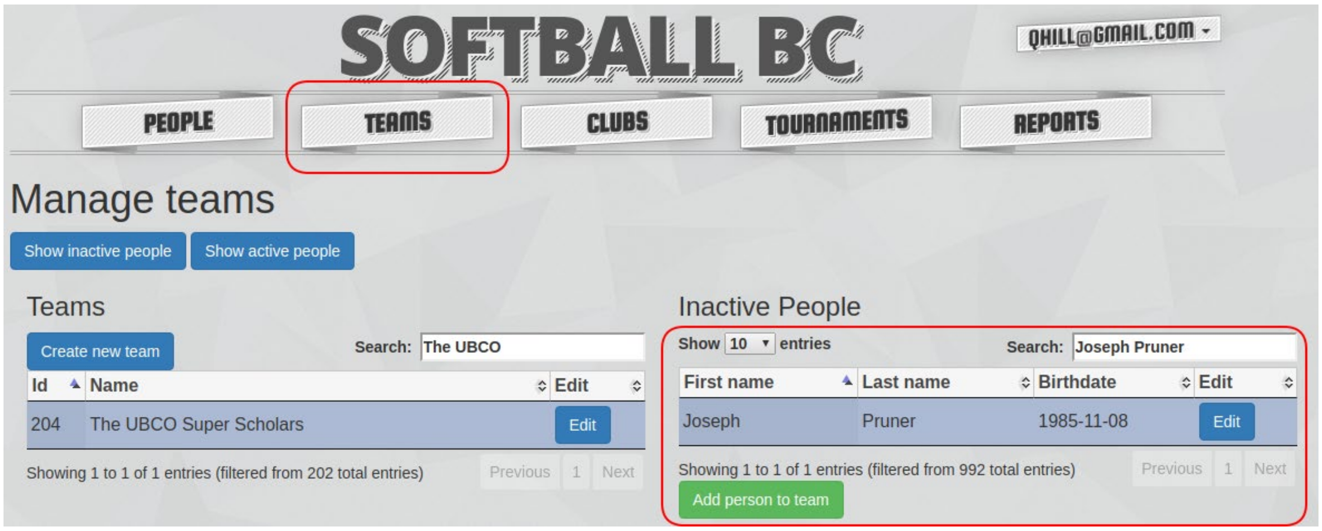
Canada

Create

Close

Figure 5.3a - Creating a new person.

The user then returns to the team page, searches for and selects "The UBCO Super Scholars" team, clicks the "Inactive person" button, searches for and selects "Joseph Pruner", then adds him to the team.



Manage teams

Show inactive people | Show active people

Teams

Create new team — Search: The UBCO

Id	Name	Edit
204	The UBCO Super Scholars	Edit

Showing 1 to 1 of 1 entries (filtered from 202 total entries) — Previous 1 Next

Inactive People

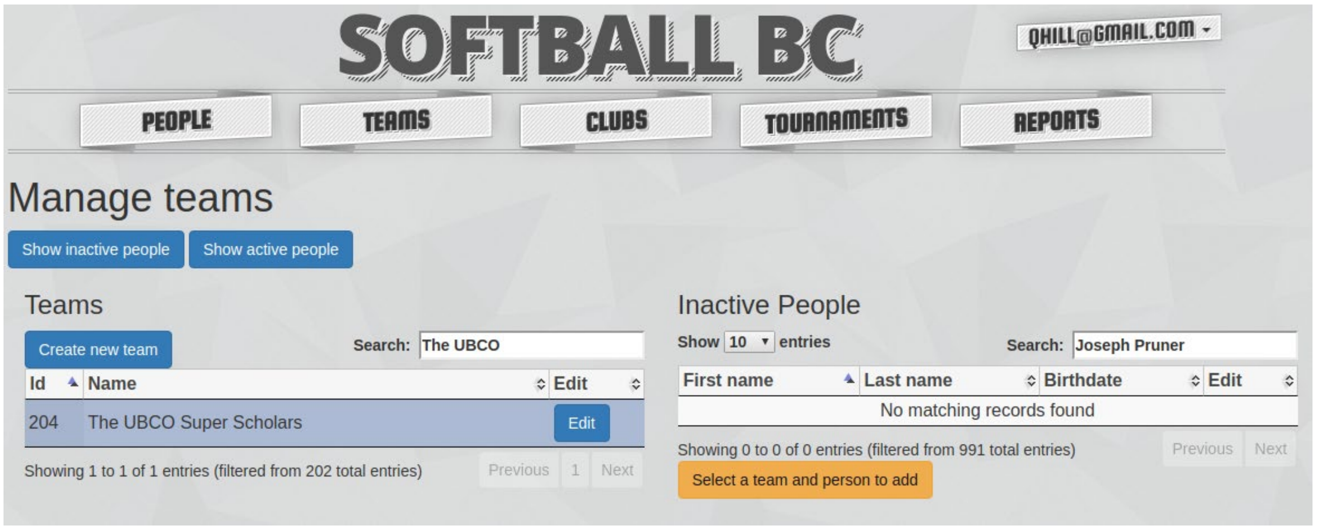
Show 10 entries — Search: Joseph Pruner

First name	Last name	Birthdate	Edit
Joseph	Pruner	1985-11-08	Edit

Showing 1 to 1 of 1 entries (filtered from 992 total entries) — Previous 1 Next

Add person to team

Figure 5.3b - The newly created person can be added to the newly created team.



Manage teams

Show inactive people | Show active people

Teams

Create new team — Search: The UBCO

Id	Name	Edit
204	The UBCO Super Scholars	Edit

Showing 1 to 1 of 1 entries (filtered from 202 total entries) — Previous 1 Next

Inactive People

Show 10 entries — Search: Joseph Pruner

First name	Last name	Birthdate	Edit
No matching records found			

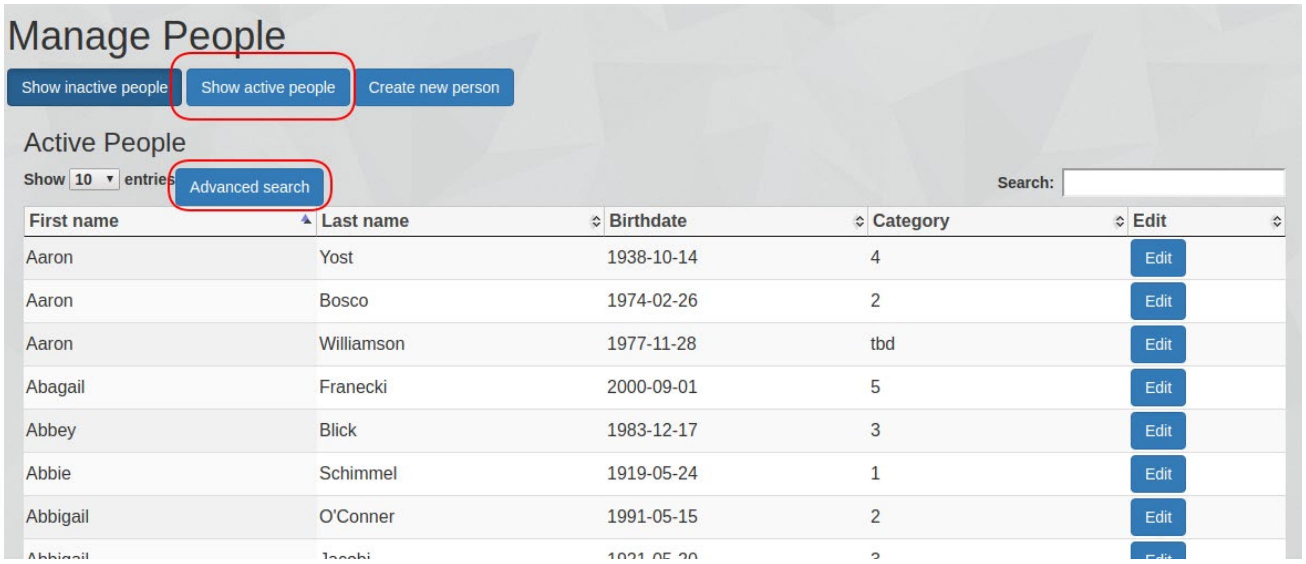
Showing 0 to 0 of 0 entries (filtered from 991 total entries) — Previous Next

Select a team and person to add

Figure 5.3c - After adding the new person to the team, they are no longer active.

5.4 - Advanced Search and Edit Person

The user now needs to take care of some administrative tasks. They have a list of people who have not been receiving the Softball BC newsletter emails. The people have provided their current, correct email addresses, and the user must verify them. The user clicks on the "People" button in the user navigation bar, and since the list contains all active people, clicks on the "Show active people" button. The user knows they will need an advanced search for this, so they click the "Advanced search" button.



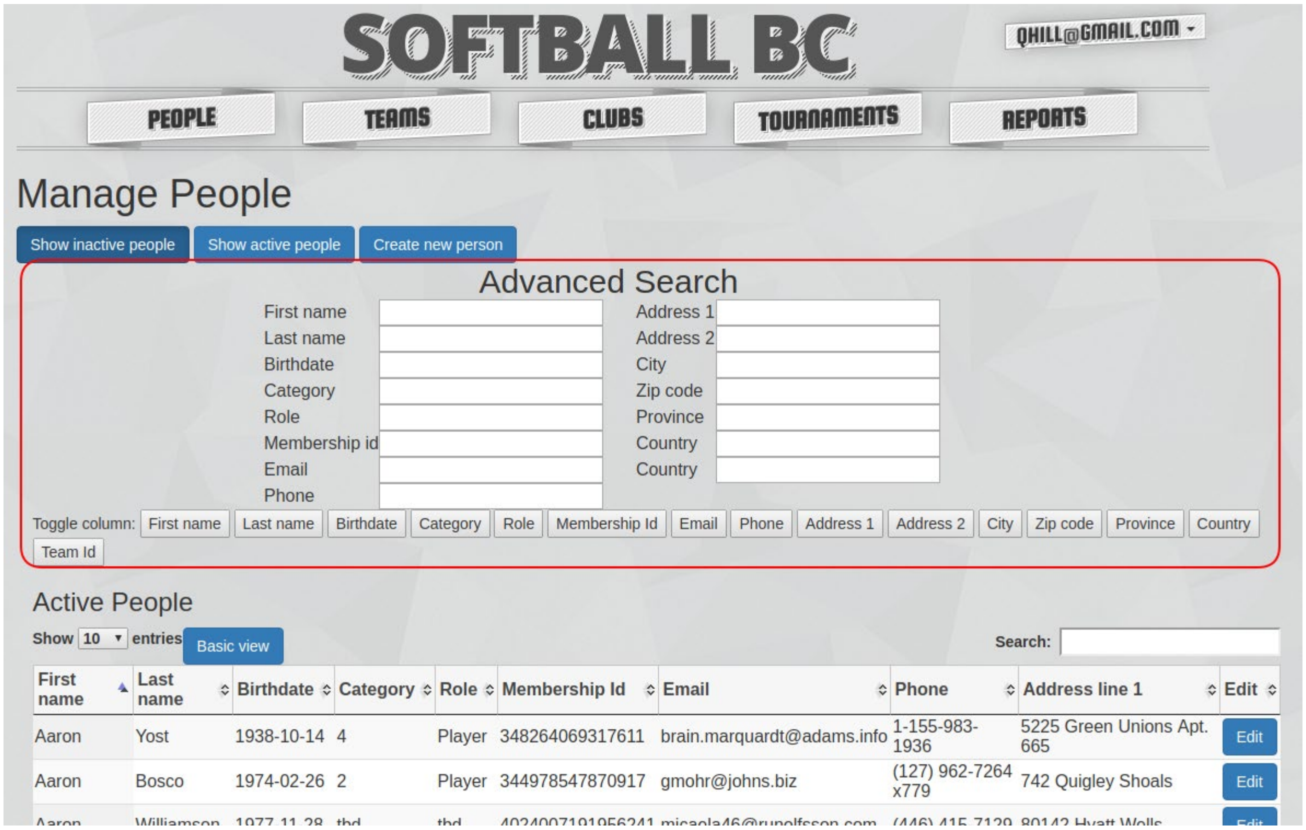
Manage People

Show inactive people | Show active people | Create new person

Active People

Show 10 entries | Advanced search | Search:

First name	Last name	Birthdate	Category	Edit
Aaron	Yost	1938-10-14	4	Edit
Aaron	Bosco	1974-02-26	2	Edit
Aaron	Williamson	1977-11-28	tbd	Edit
Abagail	Franecki	2000-09-01	5	Edit
Abbey	Blick	1983-12-17	3	Edit
Abbie	Schimmel	1919-05-24	1	Edit
Abbigail	O'Conner	1991-05-15	2	Edit



Manage People

Show inactive people | Show active people | Create new person

Advanced Search

First name		Address 1	
Last name		Address 2	
Birthdate		City	
Category		Zip code	
Role		Province	
Membership id		Country	
Email		Country	
Phone			

Toggle column: First name | Last name | Birthdate | Category | Role | Membership Id | Email | Phone | Address 1 | Address 2 | City | Zip code | Province | Country | Team Id

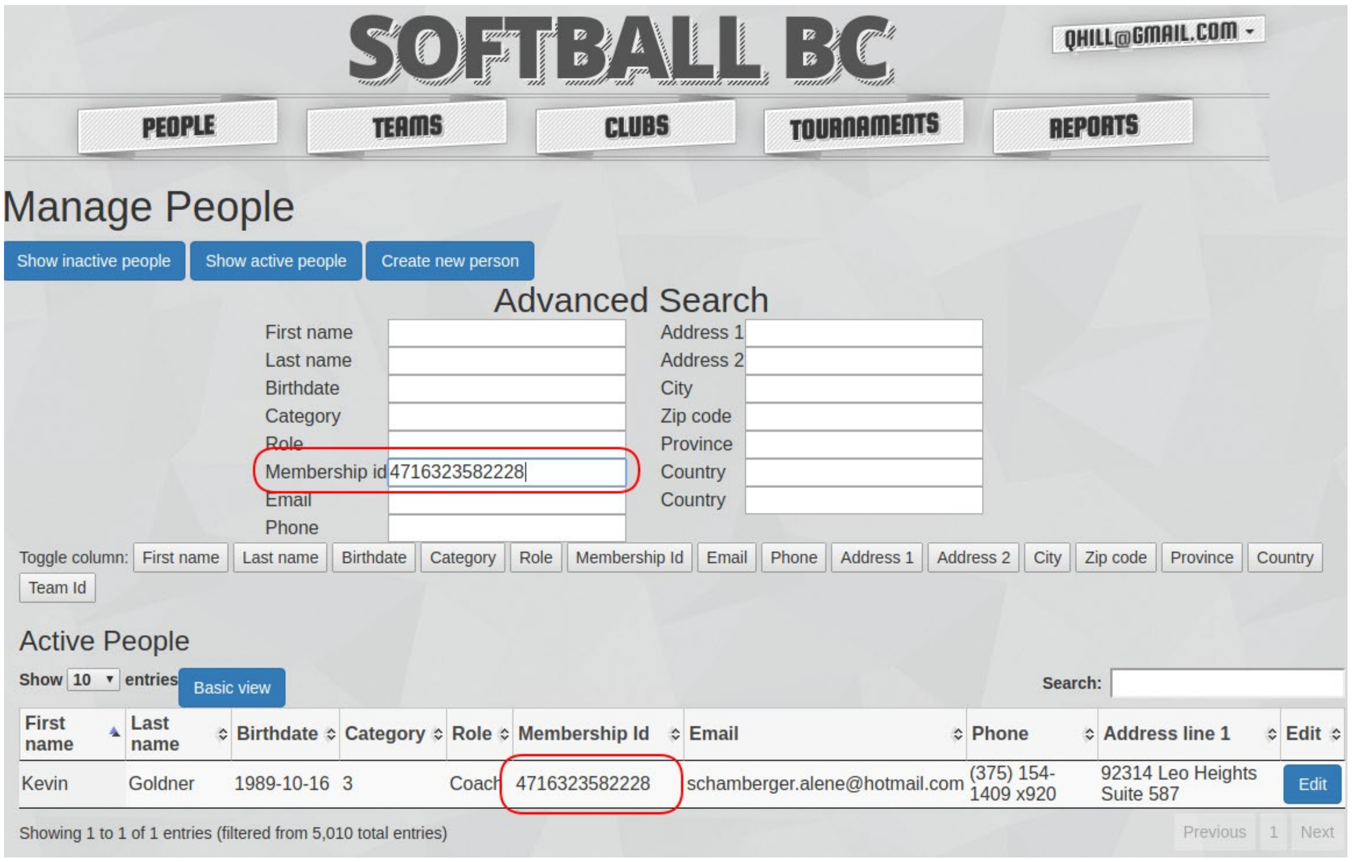
Active People

Show 10 entries | Basic view | Search:

First name	Last name	Birthdate	Category	Role	Membership Id	Email	Phone	Address line 1	Edit
Aaron	Yost	1938-10-14	4	Player	348264069317611	brain.marquardt@adams.info	1-155-983-1936	5225 Green Unions Apt. 665	Edit
Aaron	Bosco	1974-02-26	2	Player	344978547870917	gmohr@johns.biz	(127) 962-7264 x779	742 Quigley Shoals	Edit

Figure 5.4a - Navigating to the people advanced search options.

The user will search for people using their membership id, since it is a unique identifier for each person. Once the search results are returned, the user filters out all of unnecessary columns.



Manage People

Show inactive people | Show active people | Create new person

Advanced Search

First name		Address 1	
Last name		Address 2	
Birthdate		City	
Category		Zip code	
Role		Province	
Membership id	4716323582228	Country	
Email		Country	
Phone			

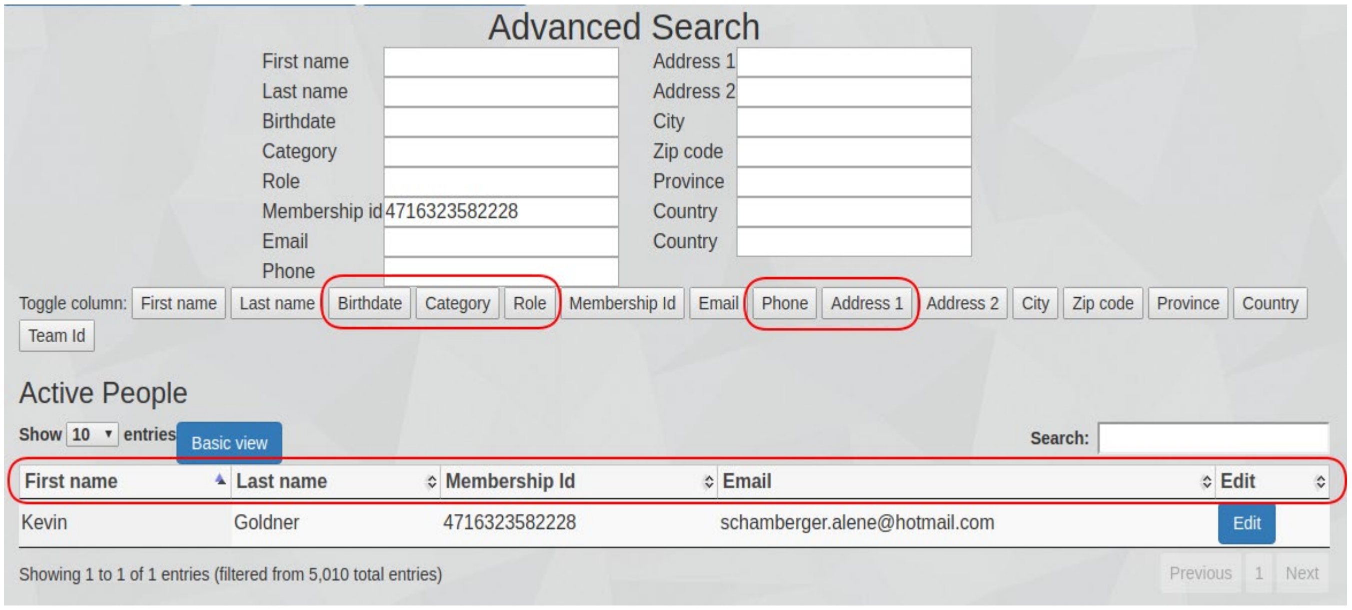
Toggle column: First name | Last name | Birthdate | Category | Role | Membership Id | Email | Phone | Address 1 | Address 2 | City | Zip code | Province | Country | Team Id

Active People

Show 10 entries Basic view Search:

First name	Last name	Birthdate	Category	Role	Membership Id	Email	Phone	Address line 1	Edit
Kevin	Goldner	1989-10-16	3	Coach	4716323582228	schamberger.alene@hotmail.com	(375) 154-1409 x920	92314 Leo Heights Suite 587	Edit

Showing 1 to 1 of 1 entries (filtered from 5,010 total entries) Previous 1 Next



Advanced Search

First name		Address 1	
Last name		Address 2	
Birthdate		City	
Category		Zip code	
Role		Province	
Membership id	4716323582228	Country	
Email		Country	
Phone			

Toggle column: First name | Last name | Birthdate | Category | Role | Membership Id | Email | Phone | Address 1 | Address 2 | City | Zip code | Province | Country | Team Id

Active People

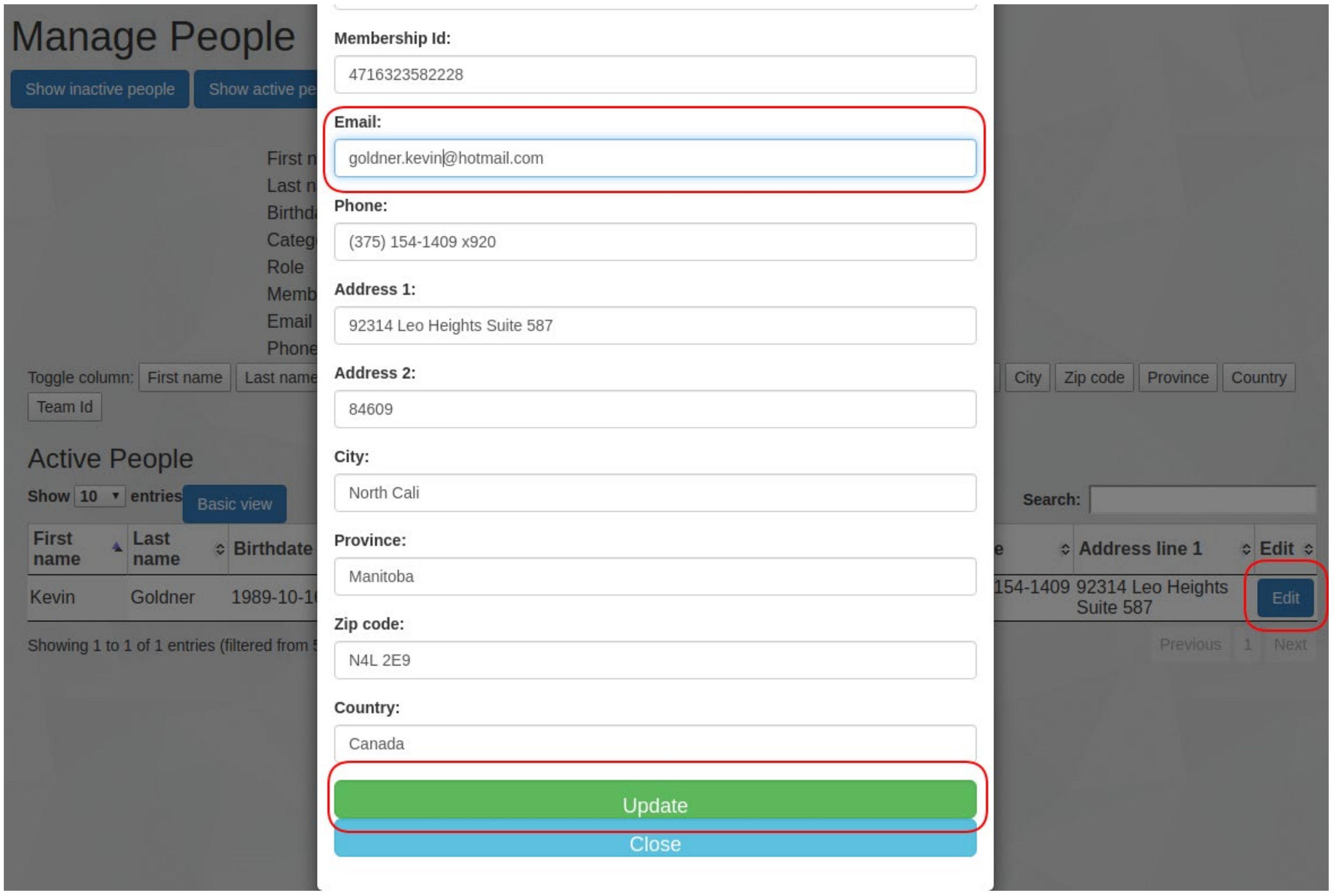
Show 10 entries Basic view Search:

First name	Last name	Membership Id	Email	Edit
Kevin	Goldner	4716323582228	schamberger.alene@hotmail.com	Edit

Showing 1 to 1 of 1 entries (filtered from 5,010 total entries) Previous 1 Next

Figure 5.4b - Advanced search and filtering.

The email is someone else's name, so it must be the mistake that was preventing Kevin from receiving newsletters. The user clicks the "Edit" button to edit Kevin's information, enters the correct email address and clicks "Update".



Manage People

Show inactive people | Show active pe

First n
Last n
Birthd
Categ
Role
Memb
Email
Phone

Toggle column: First name | Last name | City | Zip code | Province | Country | Team Id

Active People

Show 10 entries | Basic view | Search:

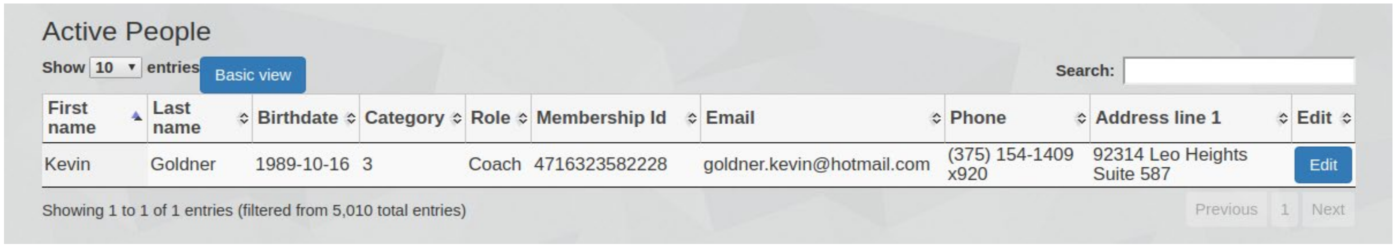
First name	Last name	Birthdate	e	Address line 1	Edit
Kevin	Goldner	1989-10-1	154-1409	92314 Leo Heights Suite 587	Edit

Showing 1 to 1 of 1 entries (filtered from 5

Previous 1 Next

Membership Id:	4716323582228
Email:	goldner.kevin@hotmail.com
Phone:	(375) 154-1409 x920
Address 1:	92314 Leo Heights Suite 587
Address 2:	84609
City:	North Cali
Province:	Manitoba
Zip code:	N4L 2E9
Country:	Canada

Figure 5.4c - Updating a person's email address.



Active People

Show 10 entries | Basic view | Search:

First name	Last name	Birthdate	Category	Role	Membership Id	Email	Phone	Address line 1	Edit
Kevin	Goldner	1989-10-16	3	Coach	4716323582228	goldner.kevin@hotmail.com	(375) 154-1409 x920	92314 Leo Heights Suite 587	Edit

Showing 1 to 1 of 1 entries (filtered from 5,010 total entries)

Previous 1 Next

Figure 5.4d - The address has been successfully updated.

The user repeats this process for all the people who need updates.

5.5 - Edit User Account

Now the user wants to edit their own account information. In the top right it displays the current user's email address. It is a drop down menu, so the user clicks it, then clicks "Account". They then click the "Update account information" button.

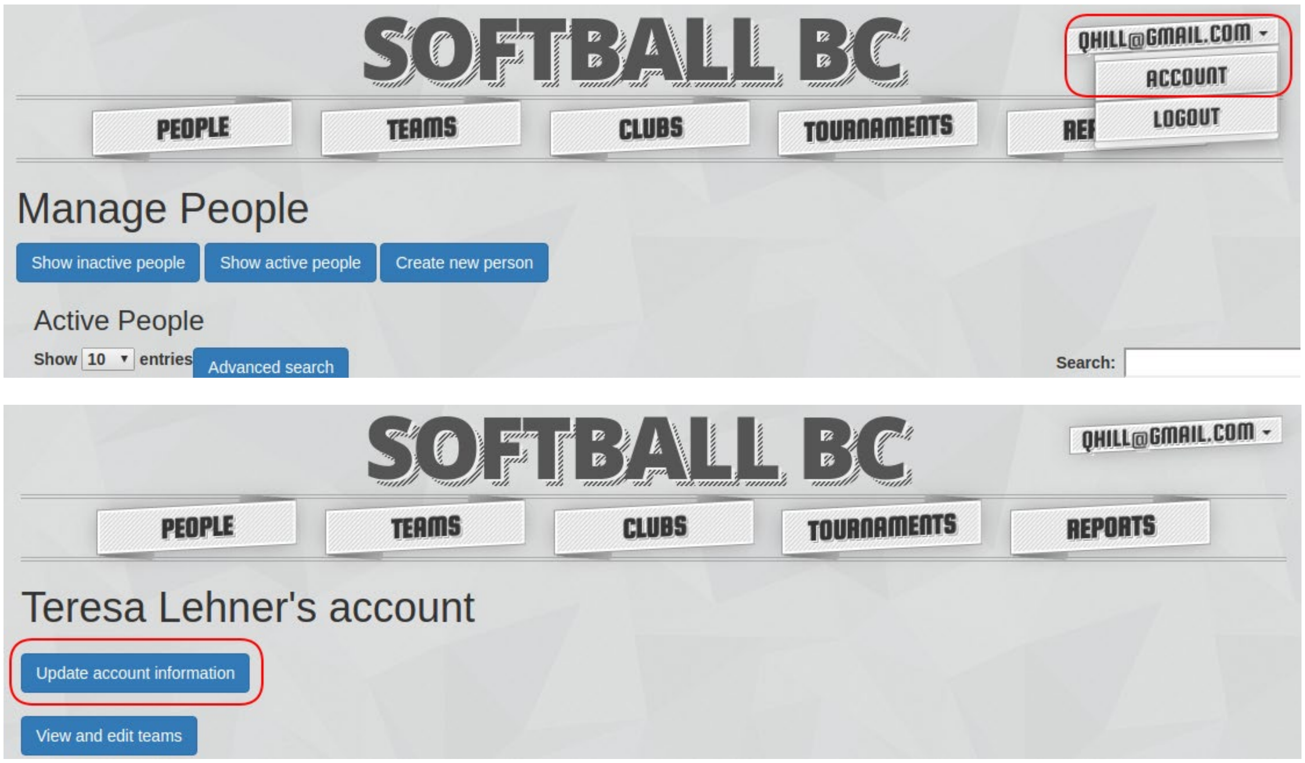
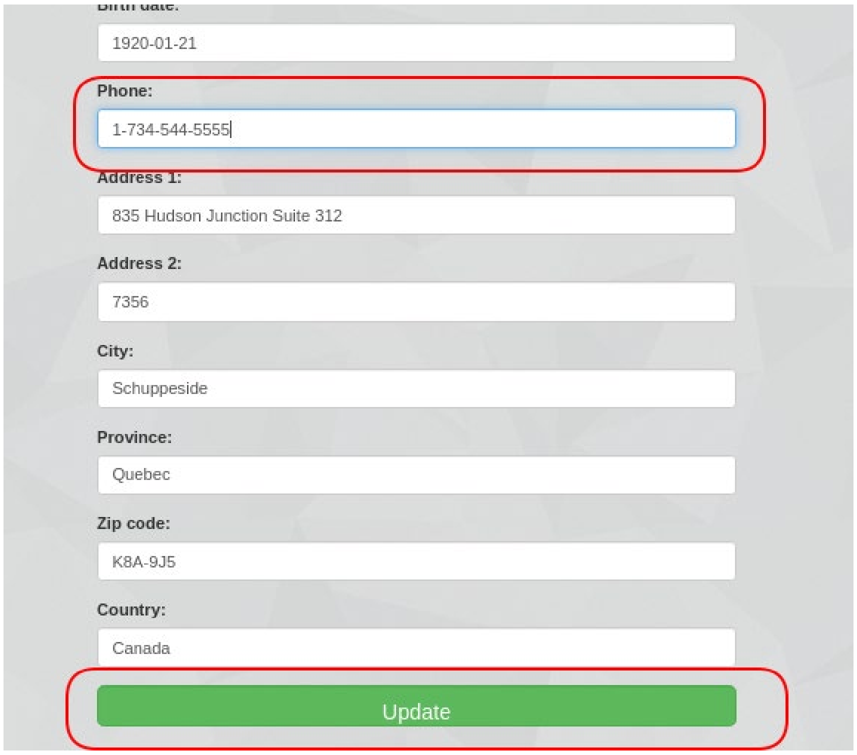


Figure 5.5a - Accessing the user account page to update user information.

The user recently got a new phone number, so they type it into the "phone" field and click update. Their account is now updated.


Birth date: 1920-01-21

Phone: 1-734-544-5555

Address 1: 835 Hudson Junction Suite 312

Address 2: 7356

City: Schuppeside

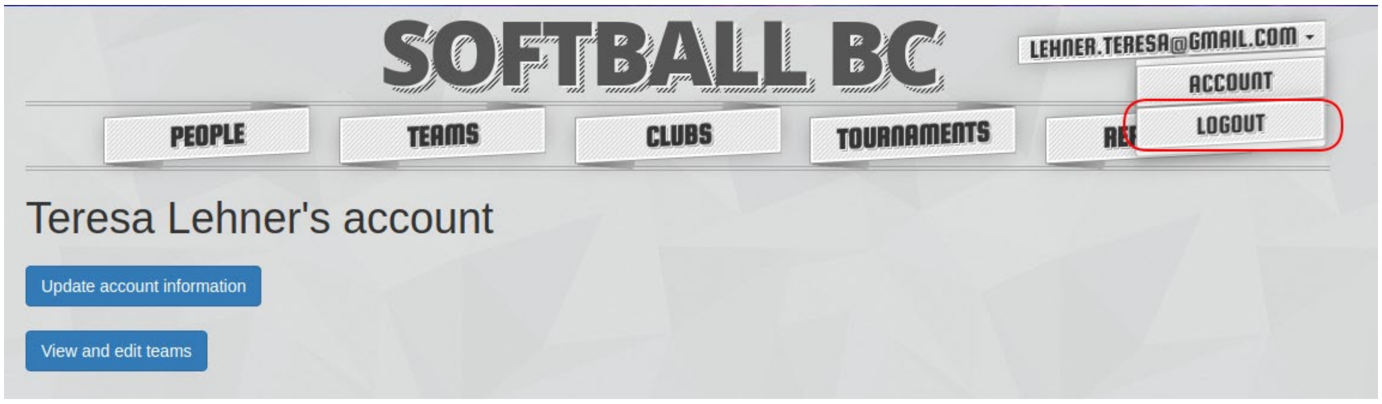
Province: Quebec

Zip code: K8A-9J5

Country: Canada

Figure 5.5b - Updating the users phone number.

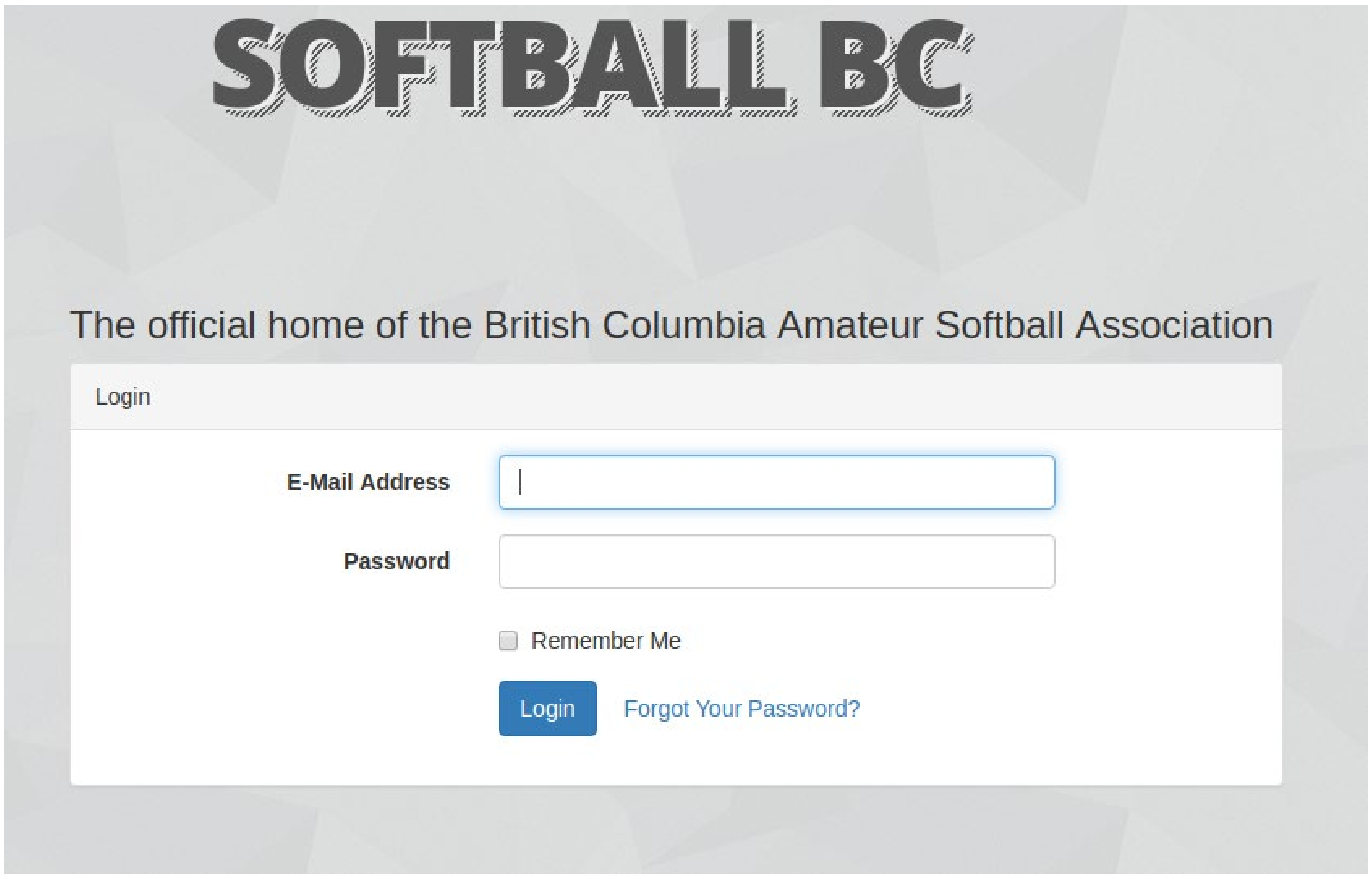
The user has completed everything they wanted to do, so they logout by clicking "Logout" from the top right drop down menu. They are logged out and taken back to the login page.



Teresa Lehner's account

Update account information

View and edit teams



The official home of the British Columbia Amateur Softball Association

Login

E-Mail Address

Password

Remember Me

Login Forgot Your Password?

Figure 5.5c - The user successfully logs out.

6 - Challenges Encountered

6.1 - Technological Challenges

Computers. Can't live with 'em, can't live without 'em.

Challenge 1:

- **Understanding Local Virtual Servers:** I've only briefly worked with virtual machines before, so the concept is still new to me. Homestead is a virtual machine operating as a server on the host machine. When the Homestead server launches, there is no GUI for the OS, just a command line that I can optionally SSH into. The server runs very quietly in the background otherwise. The concept of SSHing "into my own computer" really threw me off at first, and sometimes I'd completely forget that I had to interact with the server though an SSH'd terminal. When I wanted to install additional functionality packages to the virtual server, I would accidentally use a regular terminal and install it on my host. Things wouldn't work, and I'd think it was caused by something else, which wasted time. The same thing would happen when trying to run a file or command meant for the server. Those same files and commands existed on my host also, so the errors I'd get would sometimes send me on a wild goose chase. I have learned my lesson since then.

Challenge 2:

- **Debugging:** The default debug settings for PHP, Apache2 (host HTTP server) and Nginx (Homestead HTTP server) were surprisingly set to minimal output. This made it difficult to understand what was causing errors, or making things break. I didn't realize how much information I was missing until I got fed up with an unsolvable error, and learned how to configure everything to output detailed bug and error reports. I also learned where all the log files are. This saved me *so much time* moving forward. Easily the number one thing I wish I had known starting this project.

Challenge 3:

- **Learning Laravel:** Laravel had a steep learning curve for me, since I have never worked with a full-fledged language framework before. Even though it uses the Model-View-Controller design pattern, which seems straight forward, the pattern is represented by many different folders, and file types. I was not uncommon for a user interaction with the database to require six or seven different files to operate correctly. Understanding how all these files were sending data to each other, or calling functions took many hours of reading and watching tutorials. I now feel that I have somewhat of a grasp on Laravel, and am excited to continue learning and become more comfortable with it.

6.2 - Personal Challenges

Through the experience of developing this website, I became acutely aware of certain behaviours I possessed that hindered me through this process.

Challenge 1:

- **Over eagerness:** In the beginning, I was very excited and enthusiastic (and still am) about this project, and I was especially looking forward to the coding process. So much so that I began coding before fully exploring the development tools that were available to me. I had this big idea that I was going to free code everything from scratch, instead of using a supportive framework to help build and organize the core functionality, because it would be a more complete learning experience.
- **Result:** After spending weeks free coding components from scratch, making little progress, and struggling to get them to work properly, the reality of my current situation slowly dawned on me. This honours project is constrained to a specific time frame, and I am expected to produce tangible results at regular intervals. If I continued at my free coding pace, it would then be more accurate to call this project my retirement thesis. I wasted a lot of time working in a very inefficient way.
- **Lesson:** I will now take more time in the beginning assessing the limitations and constraints of the current project, as well as performing more in-depth research of the potential development tools. Most importantly, I will assume that everything will take longer, and be more difficult than I initially anticipate.

Challenge 2:

- **Poor Prioritization:** I would often get distracted, and caught up adjusting the minor aesthetics of a page, or unimportant details of a component, even when it did not affect the core functionality or user experience in any major way.
- **Result:** I spent hours tweaking insignificant code, when that time could have been spent improving core functionality.
- **Lesson:** I need to frequently check in with myself and ask "Is what I'm working on right now critical to the user experience or core functionality of the project?" If the answer is no, then I must move on to something more important.

Challenge 3:

- **Scattered Focus:** If I was developing a certain component, and suddenly got an idea for a different component, I would immediately switch over and work on the different component.
- **Result:** By frequently switching focus to different components, I would lose the momentum and train of thought I had built up, when lead to lost time trying to remember what I was working on before, and finding where I left off.
- **Lesson:** I now keep an organized document on hand where I can quickly type up the ideas I have for other components, without derailing my current focus.

Challenge 4:

- **Single-minded:** It was very difficult for me to ignore non-critical errors or bugs, even when it did not affect the core functionality or user experience in any major way. I became overly focused and would relentlessly pursue a solution to the bug or error.
- **Result:** I wasted hours trying to fix bugs and errors that the user would never notice. This time could have been spent of enhancing the user experience or core functionality.
- **Lesson:** Again, I need to frequently check in with myself and ask "Is this bug/error critically affecting the user experience or core functionality of the project?" If the answer is no, then I must move on to something more important.

7 - Conclusion

The process of developing a functional website that incorporates modern technologies and design principles was an immense learning experience, both on an academic and personal level. I have improved my knowledge and proficiency in SQL, PHP, HTML, CSS, and JavaScript, as well as learned a new framework, Laravel. This, combined with further experience performing database design, web page design, and project planning have given me more confidence and marketable skills as I approach the job market. The experience of using all these technologies together has given me a much deeper understanding of how the client, web server and database are truly interacting with each other, and solidified the information I learned in lectures.

My ability to manage, and be realistic about project expectations has been greatly refined through this process. I will approach the next project I work on more conservatively, planning with the expectations that things will go wrong and take longer than initially anticipated in mind. This played hand in hand with reinforcing how helpful maintaining a well organized and consistent project structure and workspace can be, as this streamlines the development process, and allows for better estimates of how long things take, or how complicated they might be.

Keeping within the time and production constraints of this project pushed me to my limit at times, but it was through these moments that I learned the most about myself, and I am very grateful that those experiences. I have learned so much about my own behaviour and tendencies which impact the development process at all stages, as well as other aspects of my life. Now that I am more aware of them, I can take further steps to mitigate their effects and become a more comfortable and confident developer moving into the future.

References

[1] R. Connolly and R. Hoar, Fundamentals of web development, 2nd ed. Boston: Pearson, 2015.

[2] J. Preece, Y. Rogers, and H. Sharp, Interaction design: beyond human-computer interaction, 4th ed. Chichester: John Wiley & Sons, Inc, 2015.

[3] F. F.-H. Nah, “A study on tolerable waiting time: how long are Web users willing to wait?” Behaviour & Information Technology, vol. 23, no. 3, 2004.

[4] Stojanovic, Vladeta. (2016). Streaming of High Resolution Aerial Photography Textures Using a Web-Based Client/Server Model on Mobile Devices. 10.13140/RG.2.2.36150.29767.

[5] “Synchronous vs asynchronous - javatpoint,” www.javatpoint.com. [Online]. Available: https://www.javatpoint.com/ understanding-synchronous-vs-asynchronous. [Accessed: 16-Apr-2019].

[6] T. Otwell, “Laravel Homestead,” Laravel. [Online]. Available: https://laravel.com/docs/5.8/homestead. [Accessed: 22-Apr-2019].

[7] Tutorialspoint.com, “Laravel Overview,” www.tutorialspoint.com. [Online]. Available: https://www.tutorialspoint.com/laravel/laravel_overview.htm. [Accessed: 21-Apr-2019].

[8] R. Elmasri and S. B. Navathe, Fundamentals of database systems, 7th ed. Harlow: Pearson Education, 2017.