CMPT 475/477 - CS/IS/IT
Capping Final Project Paper

ABS Life Coach

Patrick Zambri, Bradford Harris, Marisa Proscia, Chris Siena, Jacob Levinson

Table of Contents

I. Introduction

A. Introduction

The 21st century is a social and technological marvel. There are infinite possibilities of interests and activities that individuals take part in that have pushed the evolution of this era into the complex yet spectacular being that it is today. Although there exists so many opportunities for people to discover and learn, sometimes this overwhelming amount of information is difficult to handle and individuals can lose sight on their goals, hobbies, and aspirations. The more popular options for coping with an unorganized lifestyle include therapists, counselors, and mentors. On the rise lately is a new form of personal goal tracking known as Life Coaching.

Life Coaches work with individuals and their families, while helping to keep track of their daily lives, what matters most to them, and their most valuable dreams and accomplishments. They aim to help people “push the pause button”, and help them organize important facets of their lives such as financial obligations, personal schedules, and goals by providing around the clock contact and advice. These people looking for some life direction will be assigned a specific Life Coach who will grow to become a part of their family, and be able to keep their client on track with living and happy and healthy lifestyle.

B. Executive Summery

We started working with ABS Life Coach because we wanted to create an outlet to assist in their client’s personal organization, and invent a place that will help the coaches work with their clients to promote a successful future for the clients and their families. We have created a system that allows coaches to connect with their clients and to store personal information about them and their families in an effort to help with the coaching process. Our product consists of a beautiful, mobile-first website that scales efficiently on large desktop screens as well as on small smartphone screens, a robust database system that allow for expansion as our company grows, and a strong server configuration that can accommodate for one thousand concurrent users. Security is very important to us, so we have implemented many industry standard security practices to ensure an impermeable web interface and database. We have also designed the system with the capability to license this product to other companies so that Life Coaches around the world can provide all around care for their clients.

The details of our development are listed in this report which include the steps we followed to set up our sever, develop the website, and maintain the database of valuable client information.

II. Analysis

A. User Requirements

Below are the functional and non-functional User Requirements our team has gathered from our initial client interview as well as the project brief.

Project Objective

1. Create a System for Life Coach to keep track of their clients and their clients lives/families/issues/goals/achievements/coping mechanisms/life events/hobbies/favorite foods/notes about meetings
2. Create this system with the ability to license out to other life coach companies (including basic customization).
3. Set a hierarchy tree for coaches, supervisors and admins.

Client Profile

1. Basic functions of the system include client scheduling, recording meeting notes, reminders, client phone calls, client profile organization, family tree, and remembering client life events.
2. A convenient view page for all clients with basic client info, a family tree and the most recent interaction with the client.
3. The data gathered on a client will also be gathered on their family members.

Coach Responsibilities

1. Coaches can view all clients, set appointments, edit client profiles.
2. Can schedule appointments with clients.
3. Add and maintain client info
4. Coaches can reassign coaches to clients

Supervisor Responsibilities

1. Supervisors are coaches & have coach responsibilities.
2. Supervisors can add & remove coaches.
3. Add / remove supervisors

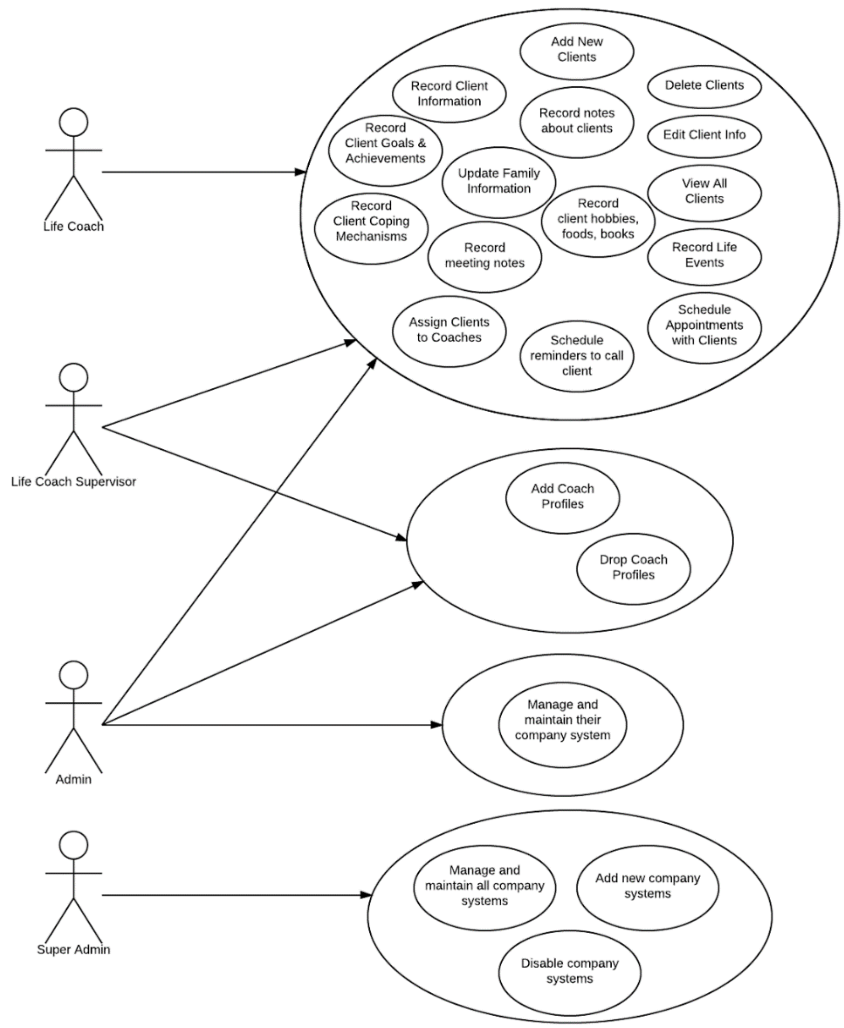
Super Admin Responsibilities

1. Make changes to maintain and update the system.
2. Manage the various components of the system.
3. Will be able to license product.

Technical Requirements

1. Security should be high priority as many of the clients are VIPs.
2. HTML5 & CSS3 will be used for a mobile-friendly website.
3. Handle 10 Million users, 1,000 concurrently.
4. System will be compatible with a LAPP server. (Linux Apache PostgreSQL PHP).
5. Website should run HTTPS.

B. Use Case Diagram



C. Use Case Documentation

Actors:

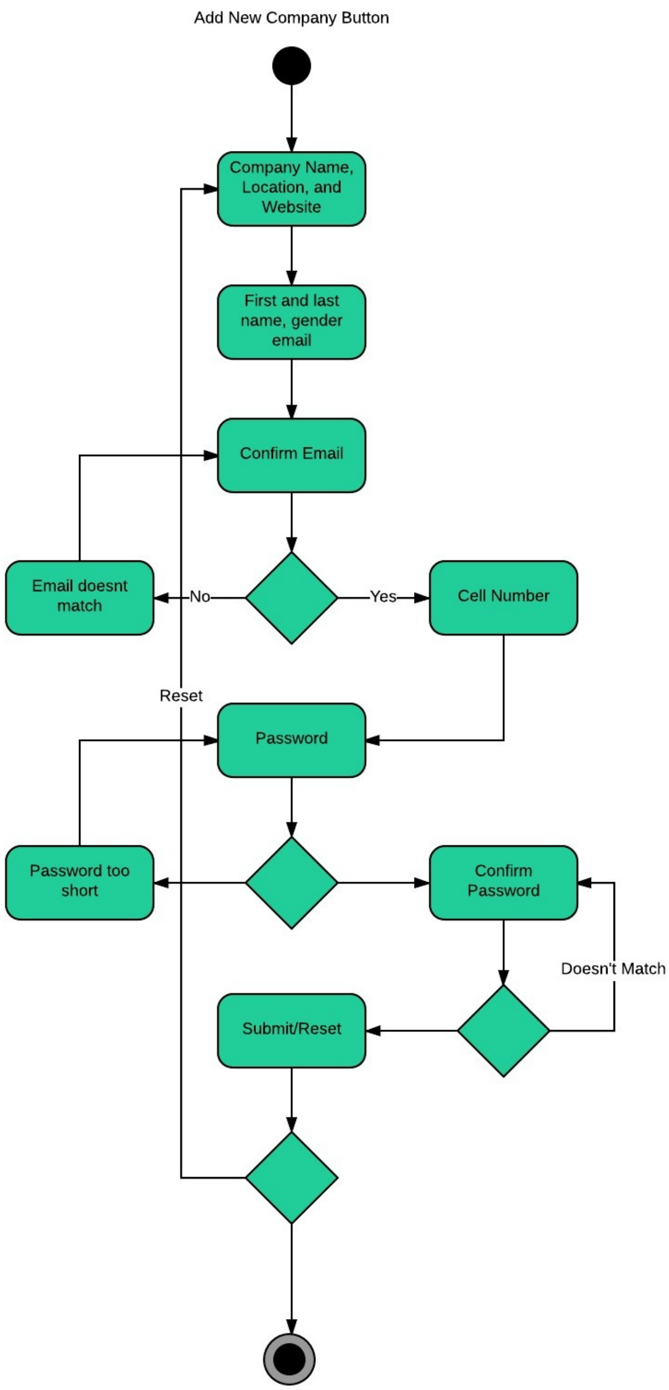
- Life Coach
    - Can add new, add/edit info about, add/update related family members, and drop/delete clients who are assigned to them.
    - Can view all clients.
    - Can record life events for clients.
    - Can record family information.
    - Can record client issues.
    - Can record goals & achievements of client.
    - Can record client coping mechanisms.
    - Can record client hobbies, favorite foods and books.
    - Can record meeting notes.
    - Can schedule appointments with clients.

- Call and record notes about clients.
- Can assign/re-assign clients to coaches.

- Life Coach Supervisor
  - Can complete all tasks coaches can complete
  - Can add and drop coach profiles.
- Admin
  - Can complete all tasks Life Coach Supervisor can complete
  - Can make edits to and maintain their company’s system
- Super Admin
  - Can make edits to and maintain all company systems
  - Can disable companies’ systems
  - Can add company system

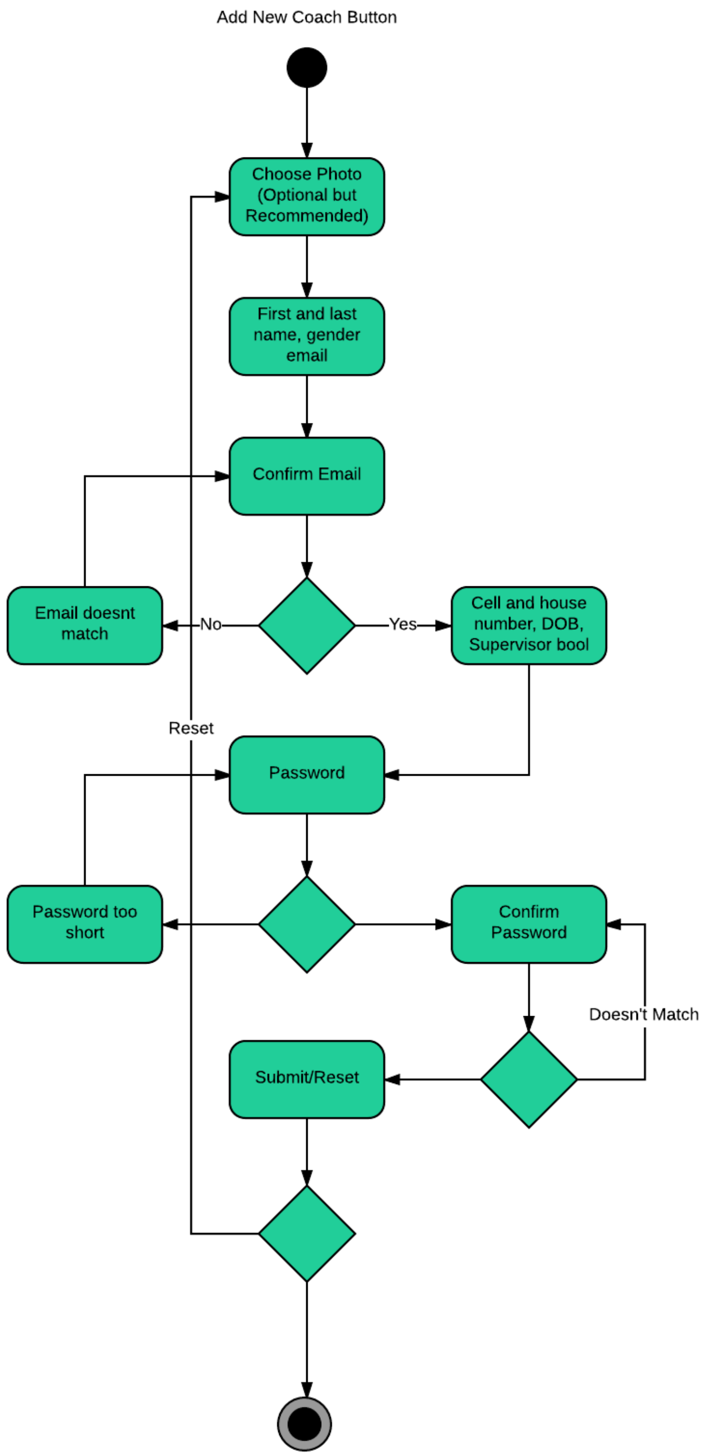
D. Activity Diagram and Documentation

Below User Interface Activity Diagrams can be found for most commonly used functionalities of our site.



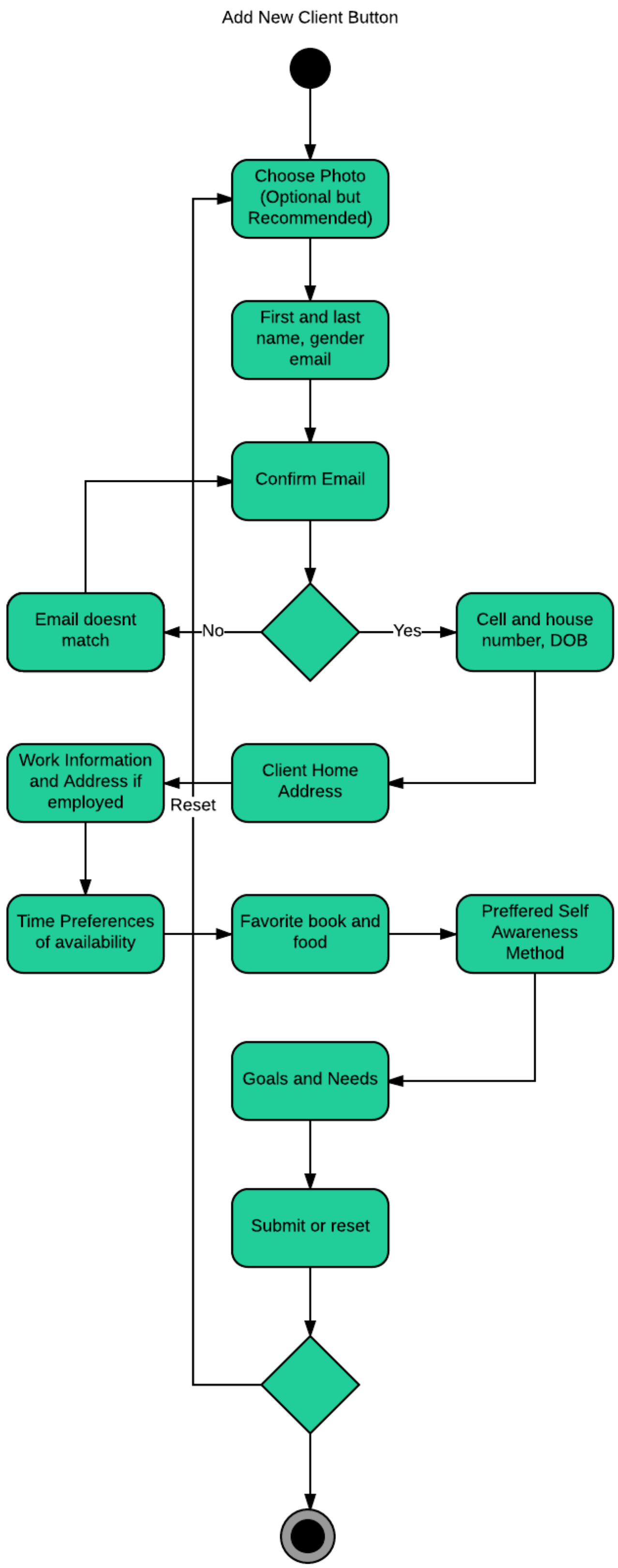
Add New Company Documentation

- Page Opens by clicking “Add Company” button
  - Form opens asking for Company name, location and site address
  - Form asks for information of site admin
    - First name, Last name, gender, email
  - Confirm email box will throw an error if email does not match
  - Enter admin’s cell number
  - Enter password for account and confirm password
    - Form throws error if password is less than 8 characters or does not match
  - User can either Submit form, or reset all fields



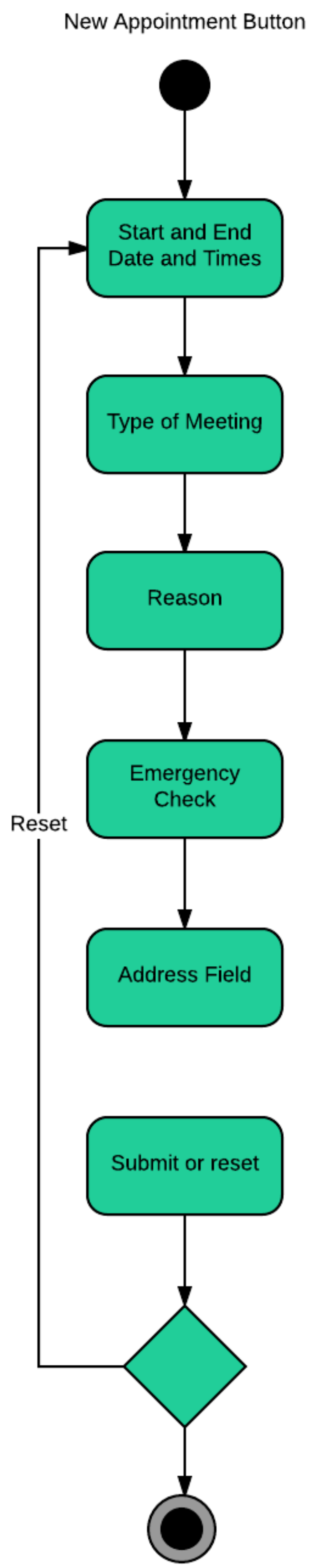
<u>Add New Coach Documentation</u>

- Page Opens by clicking “Add New Coach” button
  - Form opens asking for Coach’s Photo
  - Form asks for personal information of coach
    - First name, Last name, gender, email
  - Confirm email box will throw an error if email does not match
  - Enter coach’s cell and house numbers, Date of Birth and whether or not they are a Supervisor
  - Enter password for account and confirm password
    - Form throws error if password is less than 8 characters or does not match
  - User can either Submit form, or reset all fields



<u>Add New Client Documentation</u>

- Page Opens by clicking “Add New Client” button
  - Form opens asking for Client’s Photo
  - Form asks for personal information of client
    - First name, Last name, gender, email
  - Confirm email box will throw an error if email does not match
  - Enter admin’s cell and house numbers, Date of Birth, address, work info and address
  - Enter time of availability preferences, favorite book and favorite food
  - Enter preferred method of self awareness and goals/needs
  - User can either Submit form, or reset all fields



Add New Appointment Documentation

- Page Opens by clicking “New Appointment” button
    - Form opens asking for Start and End date and time
    - Drop down box of type of Appointment
        - Phone, office, home, out
    - Reason for Appointment
    - Check box for if the appointment is an emergency
    - Address of meeting

III. Plan

A. Project Plan

Below is the project plan created during the beginning stages of the project. As development progressed the plan has been updated to reflect any changes or updates. Following the plan is a description of our development process, including milestones, updates and various roadblocks we may have encountered throughout the project.

Week	Requirements	Development	Testing
1	Gather Requirements	Go over project overview and client expectations. Create questions based on project overview.	
2	Gather Requirements	Organize weekly updates with team to ensure requirements are me, and productivity is high. Ensure detailed documentation is completed weekly as development continues.	
3	Create use cases Create Database File Homework 1 due	Create Use Case Diagram & documentation. Create User Requirements based on client question panel and project overview. Create PHP file that has connection string & function call for Database. Continue Documentation and Requirement Gathering.	
4	Create website framework. Create website design/UI prototype. Create ER Diagrams. Homework 2 due	Create basic website in PHP. Create website design/UI prototype mockups. Create ER Diagram & Documentation Create IT Requirements. Set up virtual environment. Apply design standards to website design/UI prototype mockups. Implement UI mockups in website prototype.	
5	Begin work on basic functionality of website and features Homework 3 due	Finalize website UI mockups. Complete wireframes Update Use Case & Documentation per feedback from Week 4 Update User Requirements per feedback from Week 4 Compare mockups to client specifications.	
6	Final Project Plan & UML Diagrams	Research SSH problem Work on Database Design for demo Week 7 Update ER Diagram & Documentation per feedback from Week 6 Update IT Requirements per feedback from Week 6	

		Update Wireframes per feedback from Week 6 Continue work on website functionality and features.	
7	Database Design Draft Due	Create database design draft for demo in Week 7 Populate database with mock data Update ER Diagram & Documentation per current Database Design Update Wireframes per feedback from Week 6	
8	Initial Peer Reviews Due	Individually Complete first round Peer Reviews. Revisit database architecture and prepare prototype for production. Complete Encryption & authentication functionality Update Use Case Diagram & Documentation per feedback from Week 7 Update ER Diagram & Documentation per feedback from Week 7 Update Database Design per feedback from demo in Week 7	Test overall website functionality.
9	IT Requirements Draft Due	Complete coach adding functionality Decide on & Research CSS Framework Revisit database architecture and prepare prototype for production.	
10	Complete Database Prototype Homework 4 Due	Complete PHP Encryption & Password Encryption Continue adding mock data to Database Update IT Requirements per feedback from Week 9 Update ER Diagram and Documentation per feedback from Week 9 Continue Bootstrap research Ensure work is being completed on time in order to be ready for the demo in week 11.	
11	Demo of Prototype prepared	Continue adding mock data to Database Complete basic design & functionality of site for demo (ie adding clients/coaches, view client profiles, search clients & UI for other features) Provide a demo of a functioning prototype website.	
12	Prototype Testing and Client User Validation Testing and Documentation	Migrate site to Google Server Work on Family Tree implementation Complete IT Documentation Test website functionality and connectivity to server database. Verify UI elements are functioning properly.	Be able to fully test the prototype.
13	Continue testing Client Visit	Continue testing from week 12. Continue work on Family Tree Implementation Create dynamic Calendar	Continue user validation testing and make updates to testing as required.

14	Draft Documentation Completed Draft Final Paper Complete Draft Presentation	Add Remaining Site Functionality Update ER Diagram & Documentation per database changes during development Update Wireframes per UI & site functionality changes during development. Load Balance Site Complete Project Paper draft Begin Drafting Presentation.	Complete Load Balancing
15	Final Paper, Documentation, and Peer Evaluations	Complete Remaining Site Functionality Complete Minor UI Updates for final demo Complete Documentation for Final Paper Complete Presentation for final demo	Mock load testing, due to inability to load balance on Google Server.
16	Section Wide-Presentations		

B. Development Schedule

Project development began with gathering User Requirements and questions for our client. The questions we asked our client related to the functional and non-functional requirements our team gathered from the project overview. This process continued for three weeks. Following week 3, our team created Use Case diagrams and began documentation for our Database design and IT Requirements. In addition, our virtual environment was set up allowing us to begin development. By week 5, a functioning site was implemented to our server instance. Wireframes and User Interface, or UI, designs were created to help forecast the flow and functionality of the site for our client. Though there was trouble SSHing into our server during week 5, this problem was resolved by week 6 and closed. Database design continued, in an effort to prepare for the database demo on week 7. As the Database design was completed for week 7, our Entity Relationship, or ER, diagram and documentation were updated to reflect any changes made from our original designs. In addition, our Database was populated with mock data.

Week 8 began the beginning stages of site development, with Encryption and Authentication being implemented. During encryption implementation using MCrypt, roadblocks were hit. However, after research of the MCrypt PHP function, MCrypt was properly installed and configured on our Ubuntu 14 server. As we reached the middle of our development time frame, documentation continued to be updated per client feedback and the development process as a whole. During week 9, the ability to add new coaches to our system and login in with authentication had been implemented. In addition, Bootstrap had been chosen for our CSS framework. Bootstrap became our framework of choice due to its mobile first approach and responsive User Interface. Upon week 10, PHP encryption was complete as well as password encryption within our database. In addition, the ability to add coaches and clients had both been implemented with connections to the database. For our first demo, during week 11, many basic design elements and functionalities to the site were added. For our first demo we were able to successfully show the functionalities of: adding clients and coaches, viewing client profiles and searching for clients. In addition, UI elements were visible on the site for a calendar scheduling tool, full client profiles, a family tree and full landing page. Lastly, additional Mock data had been added to our database to allow for a proper site demo. While preparing for our first demo we found that Apache is case sensitive. Many of our files were returning errors stating that they were no longer on the server. After debugging we found that this error was due to file names being in different cases. Minor filename changes corrected this error.

During week 12, our site was migrated to the Google Server instance provided to us. In the initial migration, there were problems with external Postgres connections to our server. The solution for this roadblock became SSHing into the server instance, instead of connecting to the database as we previously were. Our biggest accomplishment thus far occurred during week 12. At this point, each user's profile displayed a family tree. This family tree is dynamic and provides coaches with the ability to quickly see relationships among people in the system. Following the Google Server migration, final IT Production Instance Documentation was completed.

Development continued through week 13, with a dynamic calendar being added to the site. In addition, development continued on the Family Tree. With the release of PHP 7.0, our instance of MCrypt PHP Function was no longer supported. By week 14 majority of our site functionality had been completed. This included: image uploading, address editing, profile editing, calendar functionality and the ability to add appointments, password resetting, declaring recently contacted clients, adding life events and adding notes. In addition, various documentation had been updated to reflect these updates. As the end of our development time frame came near, load balancing was attempted. Unfortunately, due to the migration to the Google Server we were unable to properly load balance. Our migration to the Google Server also posed an issue with our PHP mailer functionality. After extensive research, it can be noted that Google blocks PHP mailer on their servers. Because of this the functionality does not operate properly on our site. However, the functionality is present in our code in the event that PHP Mailer becomes available.

During the final week of our project, the last of the site functionalities have been added. This includes: the ability to add relationships among people, supervisor permissions, admin permissions and superadmin permissions. In addition, minor UI changes have been made to make the site more mobile friendly. Lastly, documentation has been finalized and our presentation completed.

Overall, many of the roadblocks faced during our development period were resolved with research or consulting the client and our professor. Our biggest lesson learned during the entire process was to always ask for help. Though many issues can be resolved over time, often asking for assistance speeds up the process and ultimately is more efficient.

IV. Cost/ROI

A. Application Development Costs

Students, similar to interns for a company, are essentially free labor. We complete task that we are assigned to us in an efficient manner and do not impact the company budget. Winning right? We decided it would be useful to research several tools that could assist us in creating a cost for this product. These costs will range from the beginning of development and prototype phases, extend into the implementation and production phase, and even provide insight as to what our running costs will be up to a year after the installation of the product. Of course, it is to be assumed that these prices are estimates, and given a real production environment with more resources available, the prices may differ.

The two main estimation tools we used are called “Estimate My App” and “How Much to Make an App?” These are both websites that allow us to select specific details pertaining to our product and arrive at a solid estimation for the development phase of our product. Links to these pages will be supplied following this summary.

The first tool we used, “Estimate my App”, opens up to a splash screen asking the user to select the size of the app they are intending to create. Small, medium, and large are displayed, and each is accompanied with a number of features that helps us categorize our product quantitatively. The category medium states that an application qualifies if it possesses a range of 6-9 major features. We selected this item and advanced to the next steps. The next question asked about UI design, and how complex or not we would like the product to be. Again, we selected the middle option, as the application does not need a bunch of crazy bells and whistles, but possesses a few UI elements that are far above a barebones framework. The next question asked us about user signups, and whether the user will be signing up with an email address, with social media platforms, or with custom domains. The next several questions related to user generated content, dates and locations, social engagements, and billing & eCommerce. The last three questions are a bit more technical and asked us what our preferences were regarding administrative analytics, internal APIs, and security. After completing the questionnaire, the website provided us with the quote $47,700 for our design and development phases.

We wanted a second opinion to ensure an accurate and reasonable price, but also to gauge an estimate for any future implementations of the application. The other resource we used, “How Much to Make an App?” was very similar to our first resource in that it asked similar questions regarding the requirements of our application. After selecting similar preferences on this website, we were returned with the quote of $59,400, significantly higher than the original.

Averaging these two quote yields the quote to lie around $53,000. Something very interesting that we noticed during this process was that the “Estimate My App” resource actually supplies a timeframe as well, and told us that there would be 3 weeks of design work for this application, followed by 18 weeks of development days. Although this is a very rough estimate for the development cycle of this project, it is imperative for us to be able to consider even a rough budget for this assignment. Knowing our budget can help us better organize our developers and designers to provide a more polished and efficient final product.

B. Application Operational Costs

Knowing the running costs of the product is vital to our success in terms of budgeting and planning for upkeep. We again wanted to look through a collection of resources that could assist in our process of figuring out how much this application will cost within its first year of use. We found two resources that use the same general rule of thumb when calculating the cost of maintaining an application. AppPress and FierceWireless both state that the industry standard equation for calculating an applications annual maintenance cost is 20% of its initial development costs. In our project, we estimated these costs to be about $53,000. Taking 20% out of that leaves us with an annual maintenance bill of $10,600.

We wanted to be as accurate and realistic as possible with these estimates so we completed more research to find other popular calculations for maintenance costs. We found several more links that address this calculation from a different perspective. According to ComputerWorld and Appster, the average initial development cost is only a third of the total maintenance cost after two years. For this application, that comes out to about $159,000. If we cut that price in half, we get the price of maintenance for a single year, $79,500.

Although the change between these two estimations is extremely dramatic, these tools provided us at least upper and lower bounds for an average of our upkeep costs after a year.

C. Cost Analysis Resources

Further information about the tools used for cost analysis can be found at the following links:

https://estimatemyapp.com
http://howmuchtomakeanapp.com
https://www.computerworld.com/article/2501477/mobile-apps/chief-mobile-officer--a-job-title-now-timely-.html
https://www.appsterhq.com/blog/mobile-app-maintenance/
https://www.app-press.com/blog/whats-the-cost-to-maintain-an-app
https://www.fiercewireless.com/developer/maintaining-app-critical-to-its-overall-success

V. Infrastructure Design

A. IT Requirements

Below are the IT Requirements decided upon by our team to effectively deliver an efficient product for our client.

1. **Server Platform** (for each "server" required) - Ubuntu Linux 16.04
    - **1.1. Physical system requirements**
        - 1.1.1. Storage capacity - Be able to handle 10 million users. - 1000TB
            - 1.1.1.1. Compared to other web applications, we came up with the capacity of 1000TB. Using Facebook as an example, we took their photo storage capacity and divided it in half due to the demand of our program. Though we will not host as many photos as Facebook, we felt this was a good gauge.
            - 1.1.1.2. As our final production instance is a Google server which was provided to us, the current setup is unable to support this many users. Our system however, is flexible enough to grow, and support this many users when the appropriate resources become available.
        - 1.1.2. Speed requirements / response time parameters
            - 1.1.2.1. none specified by client
            - 1.1.2.2. Scalability plans - Ability to grow to handle 10 million users in 2 years.
                - 1.1.2.2.1. With this expected growth rate, we will expand our storage at 50% of our capacity. At that point, the system may begin to slow down, and increasing the storage capacity will ensure accessibility to the system.
            - 1.1.2.3. Being a Google server, we can leverage the many scalability tools that Google provides, allowing us to grow freely.
        - 1.1.3. Be able to handle 1,000 concurrent users -
            - 1.1.3.1. This number of concurrent users was specified by our client.
            - 1.1.3.2. Our system will have the power to handle 1000 coaches logged in and documenting clients at the same time once load balancing has been implemented. We will provide allocated backup space for this scenario.
            - 1.1.3.3. Currently, with only one server available to us, we are unable to properly load balance.
    - **1.2. Virtual system requirements**
        - 1.2.1. OS to be supported
            - 1.2.1.1. System will be designed for users on both mobile and desktop

1.2.1.2. The system will be contained in a web browser so users can access it on any platform

1.2.2. Number of images expected - One main server, and we will provide 2 backups for redundancy.

1.2.2.1. We will provide 2 backups for redundancy. This allows for there to be a backup of the backup, in the event something fails.

1.2.3. Compatible with LAPP server

1.2.3.1. Our server is currently running a Linux, Apache, PostgreSQL, and PHP setup.

1.3. Connectivity

1.3.1. Network considerations - security, uptime, accessibility, private cloud over public cloud

1.3.1.1. See appropriate section of IT Final Production Implementation Documentation

1.3.2. Interconnection to what other systems - database & server connections

2. Reliability

2.1. Service Level Agreements

2.1.1. Uptime requirements - 99.99%, no specific requirement but we are aiming for system to be up at all times

2.1.1.1. We would like our system to be up for as long and as consistently as possible. This allows for the most efficient work environment for our client.

2.1.2. **Response time requirements** - Minimal response time for efficient use - The system will be fast so that it can handle very quick processes such as recording notes during a phone call which require attention to detail and the ability to move quickly throughout the interface.

3. Recoverability

3.1. Where are things backed up? How often?

3.1.1. Data will be backed up to a secondary server.

3.1.1.1. This is to have redundancy and ensure backups are available in the event of a failure.

3.1.2. Data will be backed up once a day, later in the evening to avoid backup complications while users are using the system

3.1.2.1. This level of frequency will allow for client's data to be secure in the event of a failure, allowing for the most recent images to be accessed. In addition, the backups will take place later in the evening to provide minimal disruption in daily work flow.

3.1.3. Super Admins have access to this data

3.2. **Access to backups** - Only the super admins will have access to backups and the ability to restore from them.

3.3. What data is transient and doesn't need to be stored longer term?

3.3.1. All data should be stored long term, unless a Super Admin removes it. If a client is no longer active, their information and data relevant to them will be stored for 5 years from date of deactivation. While a client is active, their information and data relevant to them will be compressed every 5 years. This will allow for the data to be stored long term, without providing much effect on the storage capacity.

4. Security and Privacy

4.1. Database

4.1.1. Access controls by userid / roles

4.1.1.1. Each Super Admin will have access to their company's database and the information pertaining to their company specifically.

4.1.1.2. Only Super Admins will be able to access Database information, for security purposes.

4.1.2. Update vs. Access

4.1.2.1. Each time a coach or supervisor uses the system they will be updating Database information.

4.1.2.2. Super Admins can access the Database information using their unique login ID.

4.2. Account information

4.2.1. User data

4.2.1.1. Personal / registration - Client profile information will be gathered after an in person, or phone interview with a coach.

4.2.1.2. All users will have a profile with personal information and notes. These profiles will include images.

4.2.1.3. User data within the system will be secured and only accessible to coaches once they authenticate themselves.

4.2.1.4. User data within the database will be secured and only accessible to Super Admins once they authenticate themselves.

5. **Maintenance**

5.1. Planned down time requirements

5.1.1. Database maintenance

5.1.1.1. Weekly-Bi Weekly database maintenance/upkeep

5.1.1.1.1. Database maintenance and upkeep will be conducted weekly or bi-weekly, on as as needed basis. This will allow for the database to stay up to date.

5.1.1.2. Major maintenance / updates on an as needed basis

5.1.1.2.1. As major changes are made to the system or maintenance is required, such updates will be conducted on an ass needed basis. These routines will be conducted in the evening to provide minimal work-time disruption.

5.1.2. Times of year when IT does maintenance

5.1.2.1. Weekly - Bi Weekly Minor Updates and bug fixes

5.1.2.2. Monthly - Bi Monthly Major updates and bug fixes

5.1.2.3. Maintenance/Updates will be scheduled during companies off periods, avoiding peak times and service outages.

5.1.3. Times of year when the systems are not available?

5.1.3.1. System should be available all year round, as biweekly and bimonthly updates are designed to ensure proper operation and minimal down time.

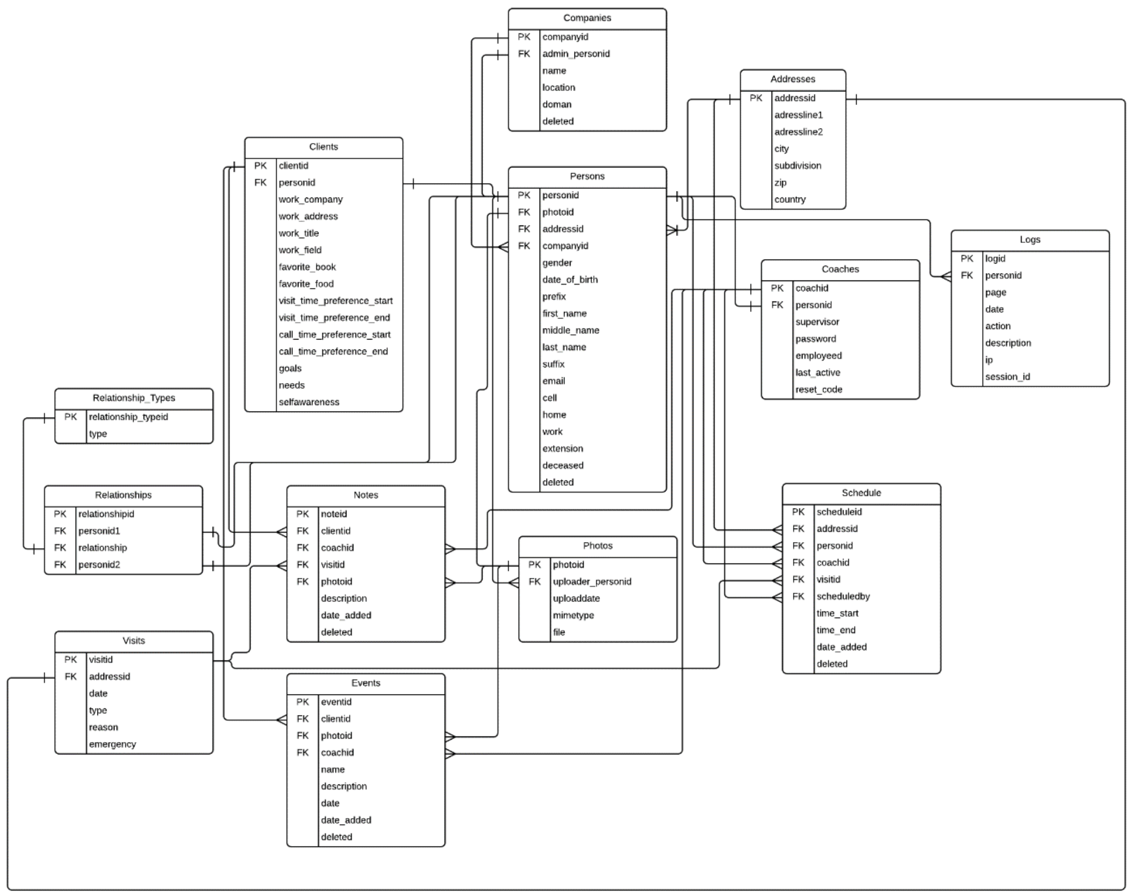
B. IT Documentation

Further IT Documentation can be found at Appendix I at the end of this document. Appendix I includes documentation for the Production Instance of Ubuntu 16.04 running Apache2, PHP, and PostgreSQL on the Google Server Instance provided to us. Installation instructions as well as various resources used throughout the setup process of the final production instance.

VI. Application Design

A. Database Documentation

1. Entity Relationship Diagram



2. Entity Relationship Documentation

Currently a PostgreSQL database is implemented in our system. Database tables include: addresses, clients, coaches, companies, events, logs, notes, persons, photos, relationship_types, relationships, schedule and visits. Further explanation and documentation of fields and relationships are listed below.

Addresses

- addressid - Each address has a unique ID. Addressid is referenced by clients as work_address, visits, schedule and persons.
- subdivision - Address state or province.

Clients

- clientid - Each client has a unique ID. Clientid is referenced by events and notes.
- personid - The personid of the client, referenced from persons.

- coachid - The coachid of the coach assigned to said client, referenced from coaches.
- work_address - The addressid of client’s work address, referenced from address.
- visit_time_preference_start - Timestamp of client’s preference for visits, start time.
- visit_time_preference_end - Timestamp of client’s preference for visits, end time.
- call_time_preference_start - Timestamp of client’s preference for phone calls, start time.
- call_time_preference_end - Timestamp of client’s preference for phone calls, end time.

Coaches

- coachid - Each coach has a unique ID. Coachid is referenced by events, notes, schedule as scheduledby, schedule as coachid and clients.
- personid - The personid of the coach, referenced from persons.
- supervisor - Boolean value to determine if a coach is a supervisor.
- employeed - Boolean value to determine if a coach is currently employed.
- last_active - Timestamp of last time coach has accessed system.
- password - Unique coach password to access online system.
- reset_code - Unique verification code used to allow for resetting of coach password via email.

Companies

- companyid - Each company has a unique ID. Companyid is referenced by the persons table.
- admin_personid - The personid of the admin of said company, referenced from persons.
- deleted - Boolean value to determine if a company has been disabled by Super Admin.

Events

- eventid - Each life event has a unique ID.
- clientid - The clientid of the client in which said life event relates to, referenced from clients.
- photoid - The photoid of the photo relative to said life event, referenced from photos.
- coachid - The coachid of the coach that documents said life events, referenced from coaches.
- date - Timestamp of when said life event occurred.
- date_added - Timestamp of when said life event was recorded by a coach.
- deleted - Boolean value to determine of a life event has been deleted.

Logs

- logid - Each log has a unique ID.
- personid - The personid of the person interacting with the system, referenced from persons.
- session_id - Unique ID for specific session, relative to the user’s session ID.

Notes

- noteid - Each note has a unique ID.
- clientid - The clientid of the client in which said note is referring to, referenced from clients.
- coachid - The coachid of the coach that is documenting said note, referenced from coaches.

- visitid - The visitid that is in relation to said note, referenced from visits.
- photoid - The photoid of a photo that is relative to said note, referenced from photos.
- deleted - Boolean value to determine if a note has been deleted.

Persons

- personid - Each person has a unique ID. Personid is referenced by clients, schedule, relationships as personid1 and personid2 and photos as uploaderid.
- photoid - The photoid of said person’s identification photo, referenced from photos.
- cell - Cellphone number of said person.
- home - Home phone number of said person.
- work - Work phone number of said person.
- extension - Work phone number extension of said person.
- addressid - The addressid of said person’s home address, referenced from address.
- companyid - The companyid of the company in which said person is a part of, referenced from company.
- deleted - Boolean value to determine if person has been deleted.

Photos

- photoid - Each photo has a unique ID. Photoid is referenced by persons, events and notes.
- uploader_personid - The personid of the coach that uploaded the photo, referenced from persons.
- uploaddate - Timestamp of when photo was uploaded.
- file - longblob, holds raw data from the file.
- mimetype - varchar(255), holds the file type for correctly displaying the information.

Relationship_types

- relationship_typeid - Each relationship type has a unique ID. Relationship_typeid is referenced by Relationships.
- type - relationship type (ie: marriage/partner, cousins, siblings, parent/child).

Relationships

- relationshipid - Each relationship has a unique ID.
- personid1 - The personid for person one of relationship, referenced from persons.
- personid2 - The personid for person two of relationship, referenced from persons.
- relationship - The relationship_typeid for said relationship, referenced from relationship_types.

Schedule

- scheduleid - Each appointment scheduled has a unique ID.
- addressid - The addressid of the address in which the appointment will take place, referenced from addresses.
- personid - The personid of the client in which the appointment is for, referenced from persons.
- coachid - The coachid of the coach in which the appointment is scheduled for, referenced from coaches.
- scheduledby - The coachid of the coach that scheduled the appointment, referenced from coaches.
- visitid - The visitid in which the appointment relates to, referenced from visits.

- deleted - Boolean value which determines if the appointment has been deleted.

Visits

- visitid - Each visit has a unique ID. Visitid is referenced by notes and schedule.
- date - Timestamp of visit.
- addressid - The addressid of the location of the visit, referenced from addresses.
- type - references the type of visit, ie. phone call or in person.
- emergency - boolean value that will indicate whether or not the visit was an emergency.

3. Database Mock Data

To generate mock data for our server, Mockaroo was used. Mockaroo has many different categories of data available to generate, including street addresses, cities, countries, and postal codes as shown below. We generated data for a variety of our tables such as the clients table, addresses, notes, events, and most importantly relationships to get our family trees working. Further documentation of Mockaroo can be found at Mockaroo.com.

Sample SQL Statements

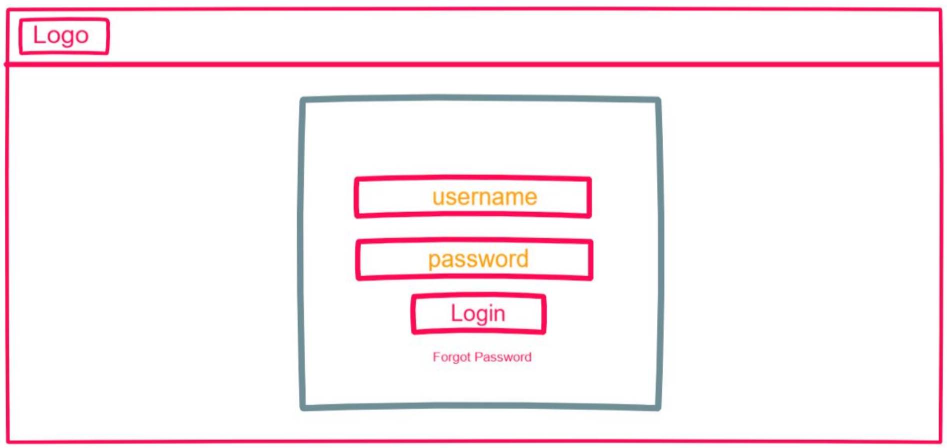
Below are sample SQL statements used to inject mock data into our database. These statements are examples of how mock addresses were added.

```
insert into addresses (adressline1, city, subdivision, zip, country) values ('0610 Thierer Drive', 'Washington',
'District of Columbia', '20392', 'United States');
insert into addresses (adressline1, city, subdivision, zip, country) values ('40 Merrick Park', 'Lexington',
'Kentucky', '40581', 'United States');
insert into addresses (adressline1, city, subdivision, zip, country) values ('803 Manitowish Way', 'Duluth',
'Minnesota', '55811', 'United States');
insert into addresses (adressline1, city, subdivision, zip, country) values ('82 Forster Terrace', 'Toledo',
'Ohio', '43615', 'United States');
insert into addresses (adressline1, city, subdivision, zip, country) values ('1 Westerfield Circle', 'Jamaica',
'New York', '11499', 'United States');
```

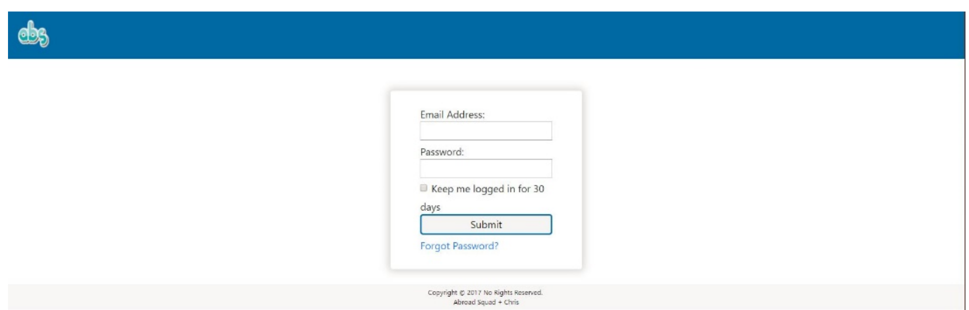
B. User Interface Design

Below are wireframe outlines and screen shots of our live site to depict website functionality to it's fullest. These views include both desktop and mobile views.

1. Desktop View

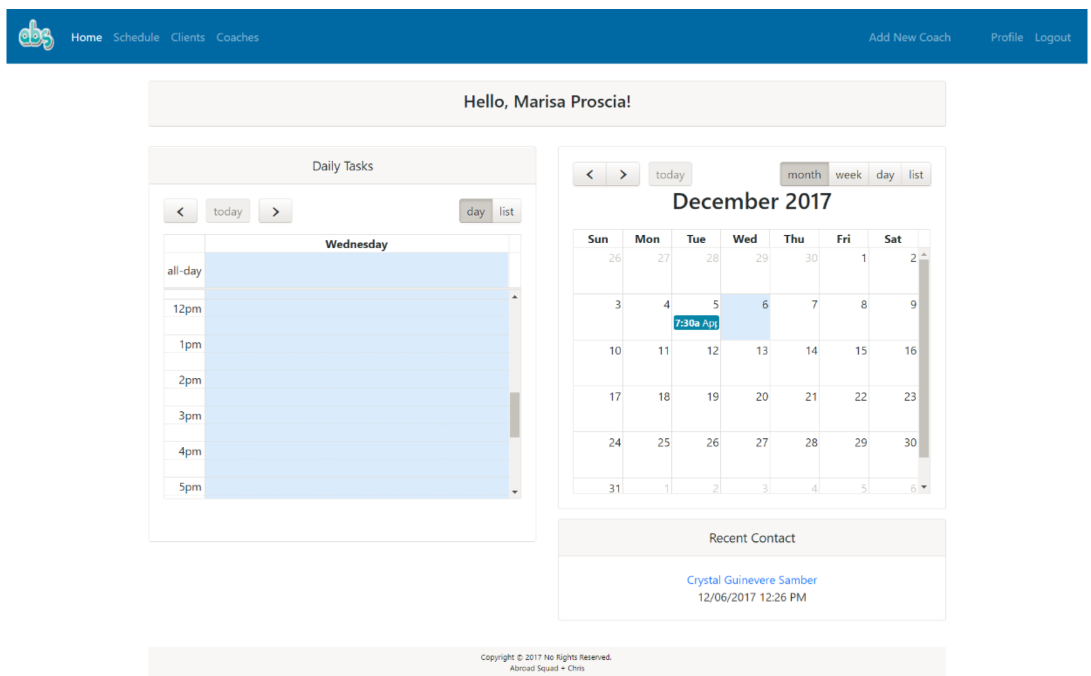


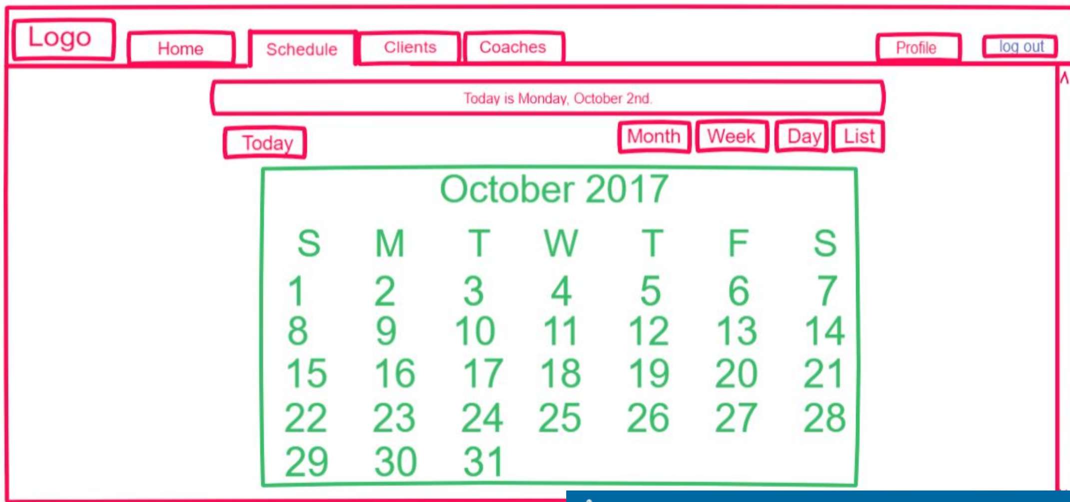
Login Page



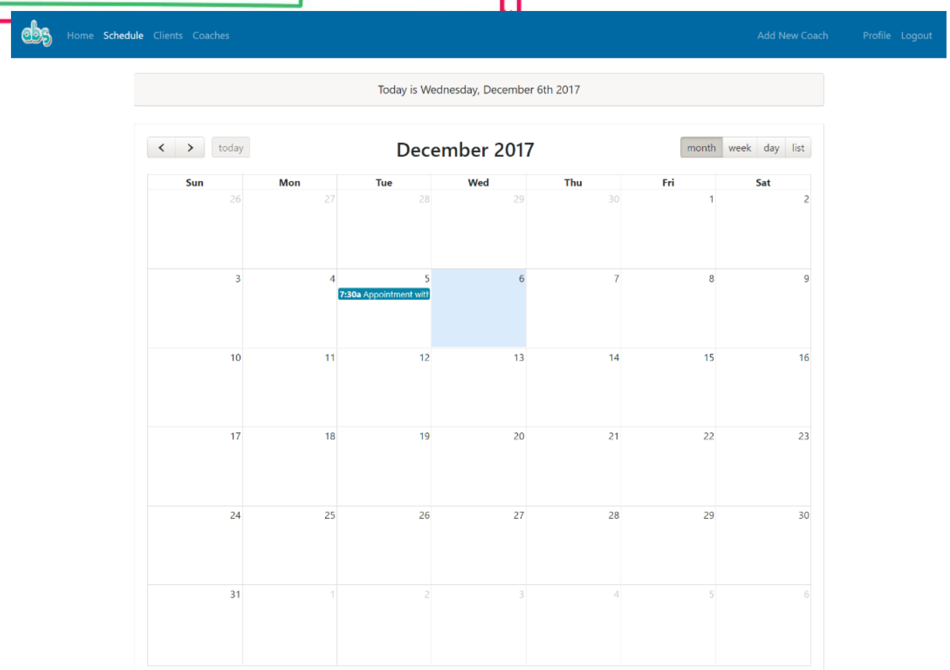
Home Page
Coach View

Home Page
Supervisor View

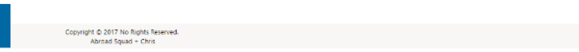


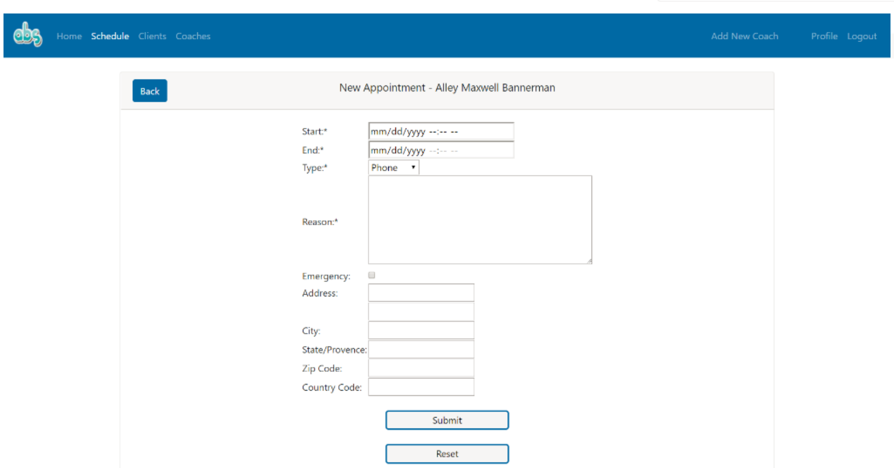


Schedule Coach View



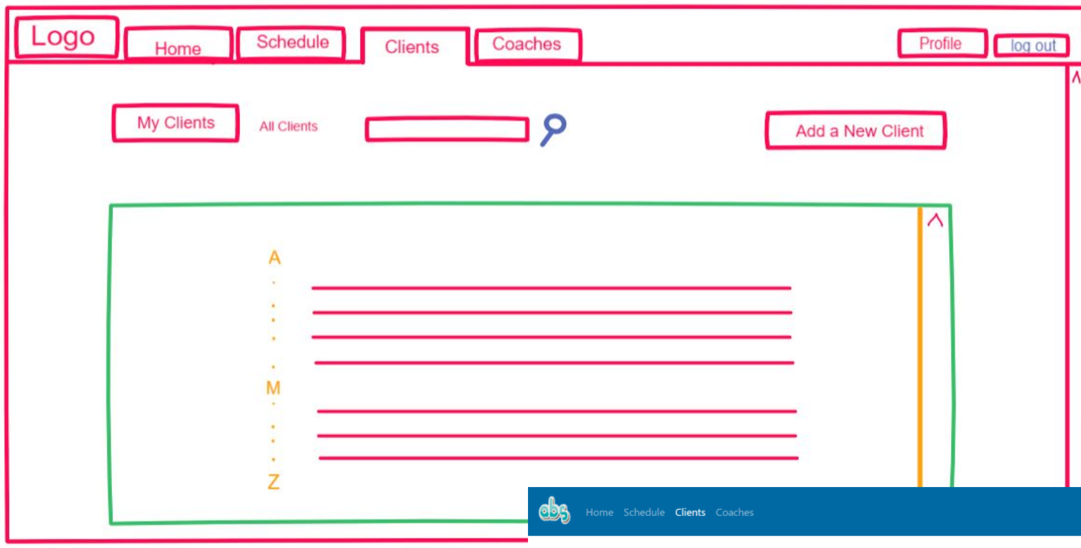
Schedule Supervisor View



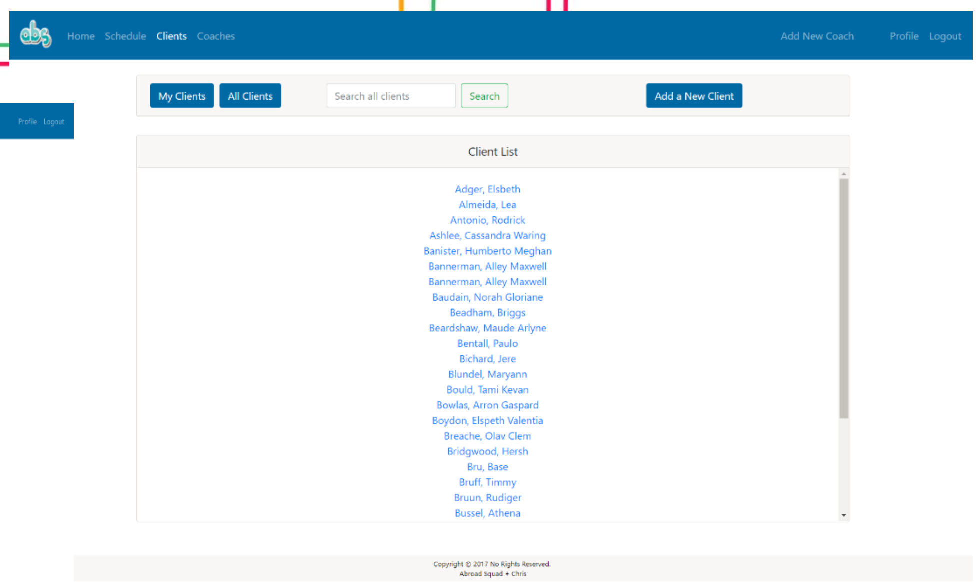


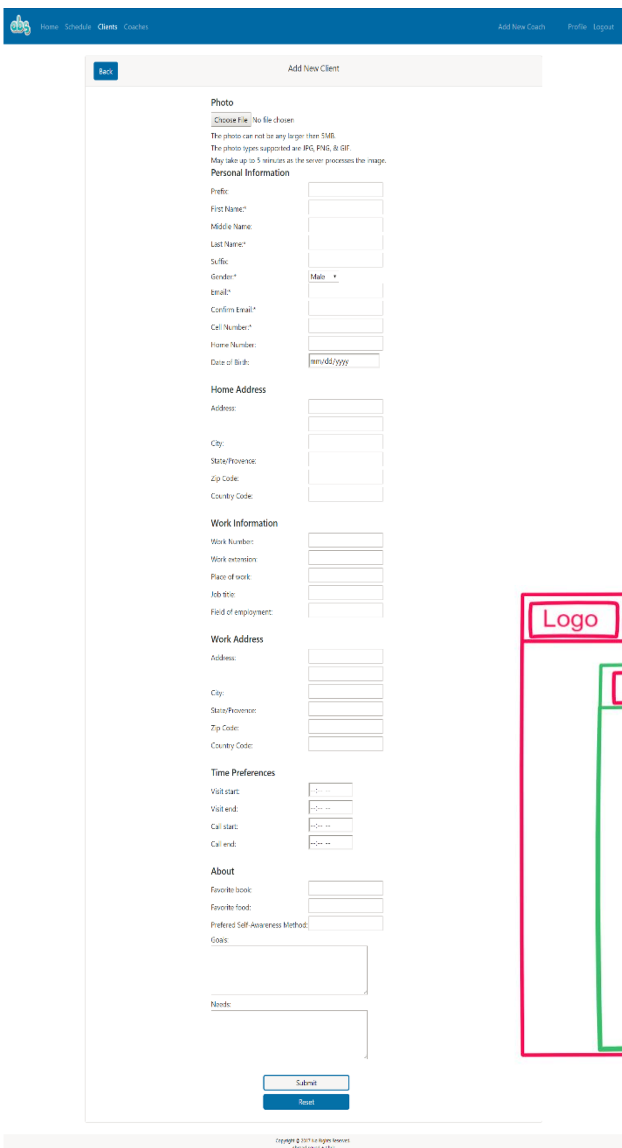
Add Appointment Supervisor View

Client Search Coach View



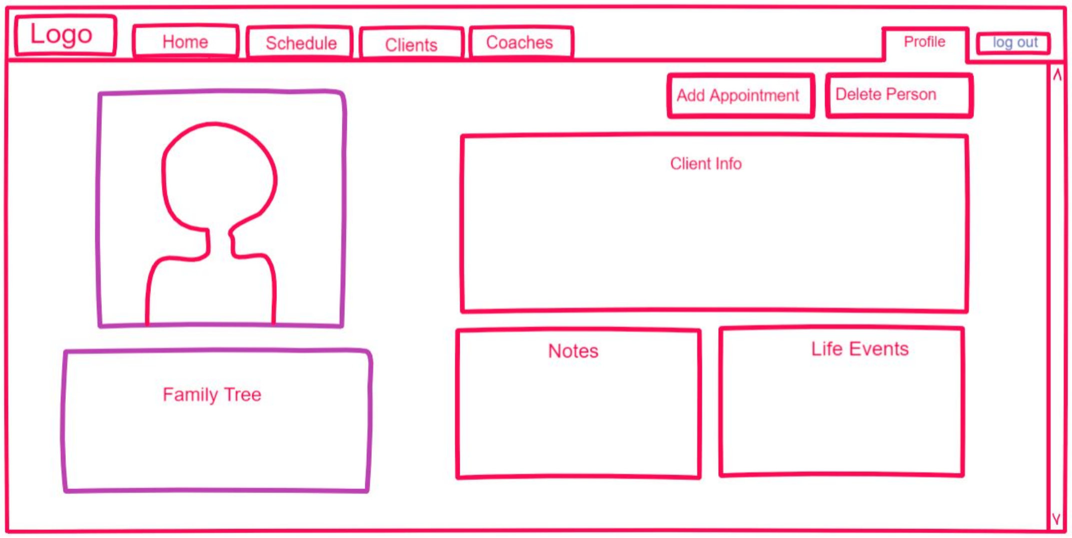
Client Search Supervisor View





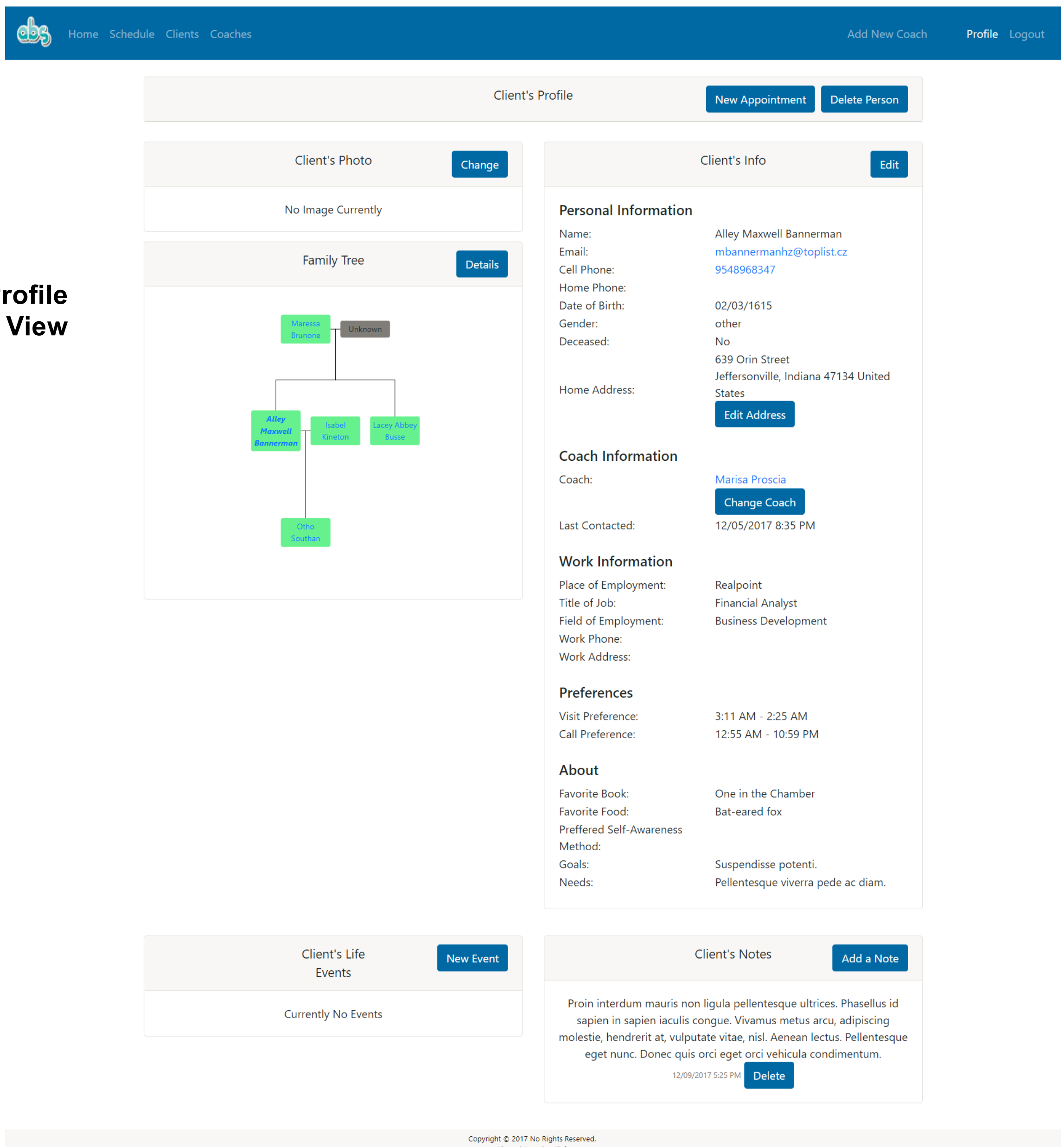
Add New Client

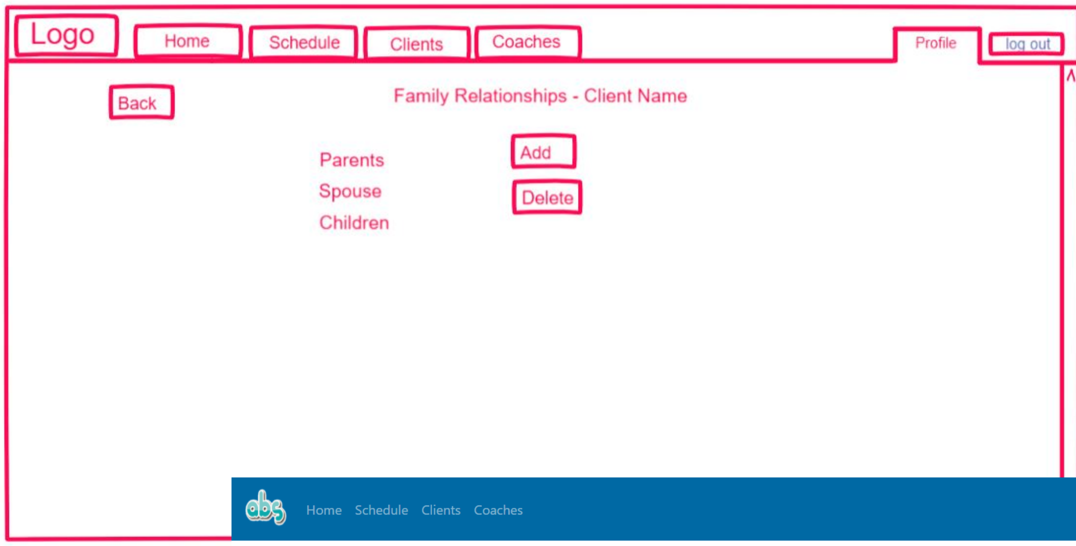
ABS Life Coach 27



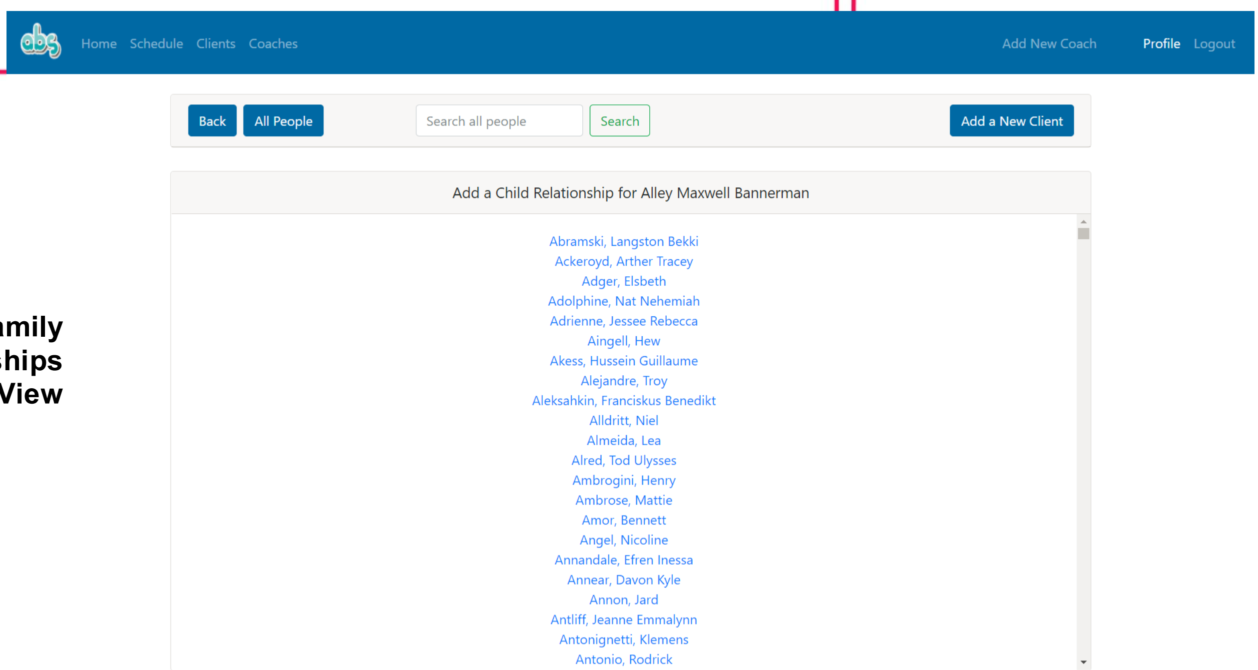
Client Profile
Coach View

Client Profile
Supervisor View

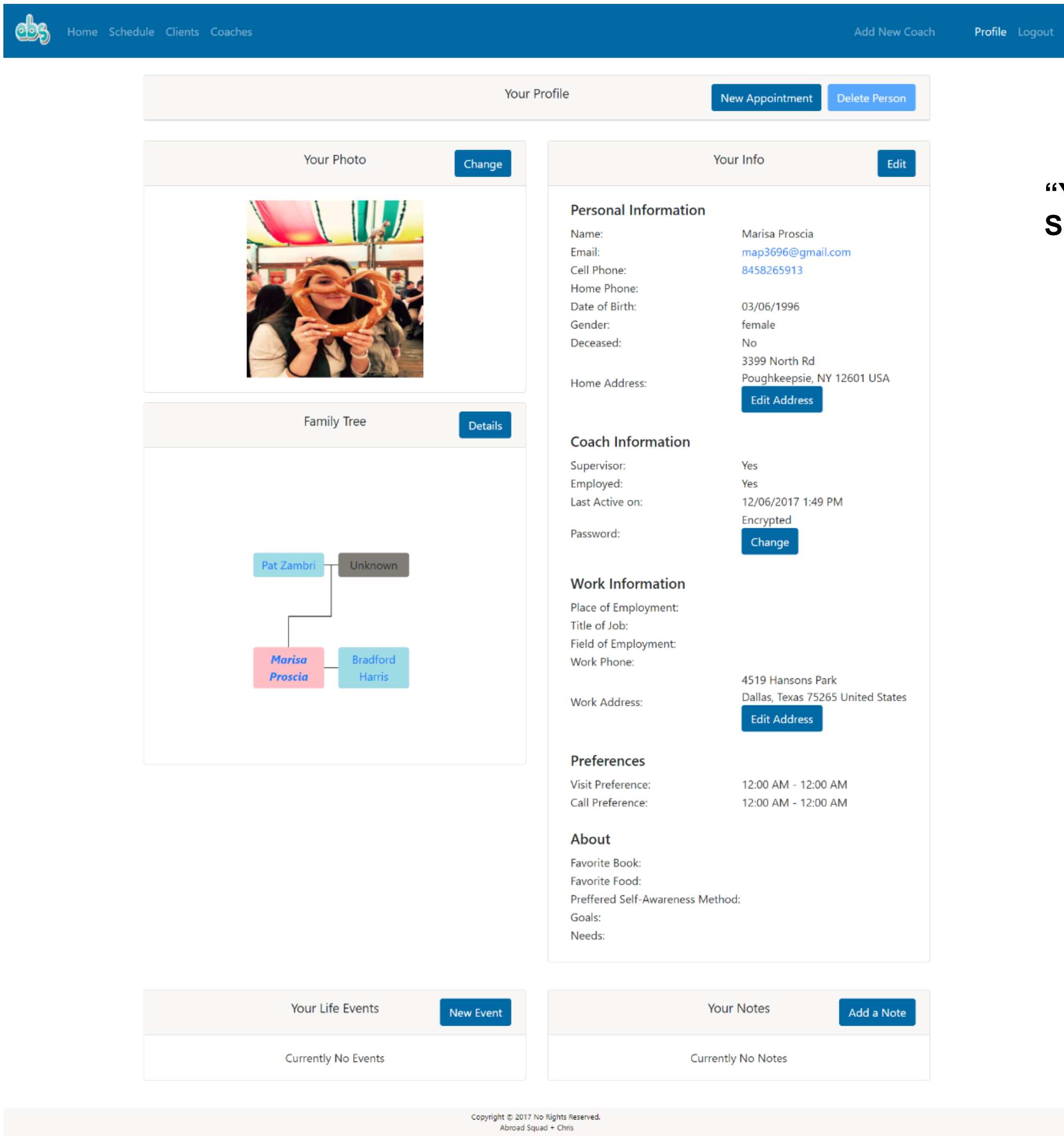




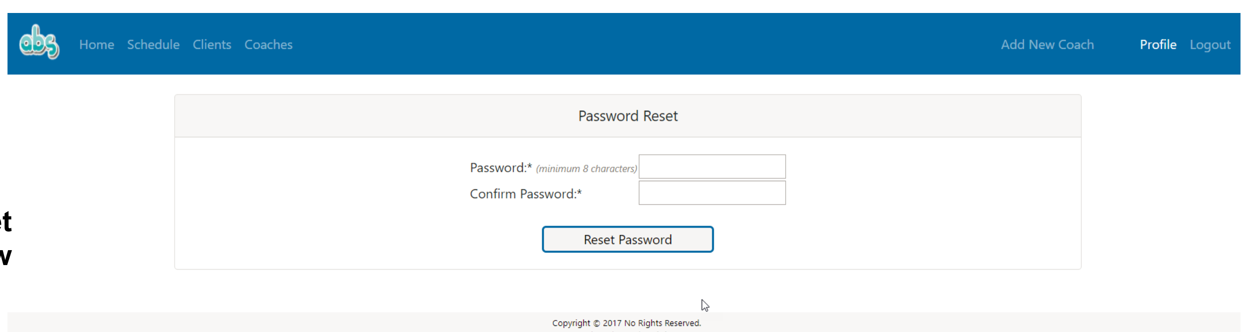
Viewing Family Relationships Coach View



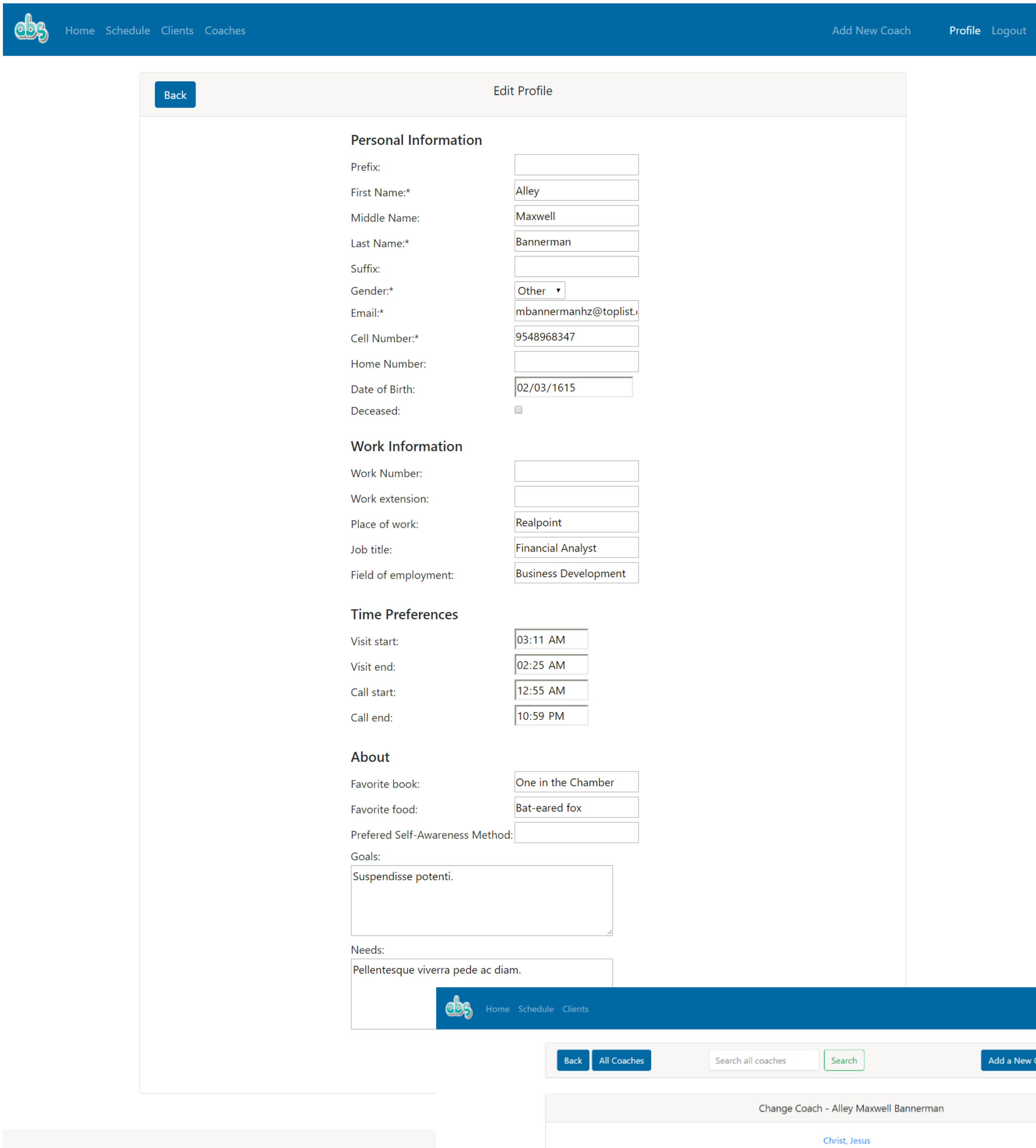
Viewing Family Relationships Supervisor View



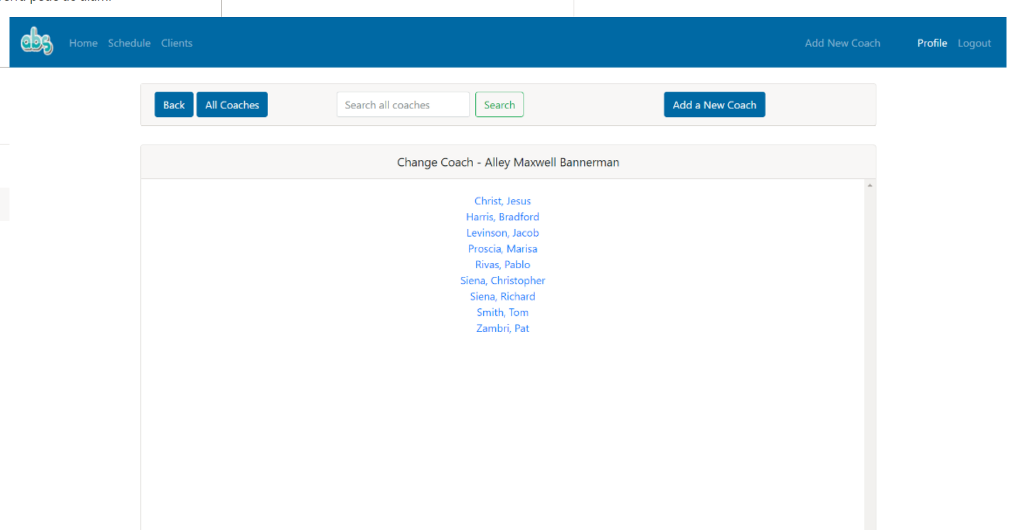
"Your" Profile Supervisor View



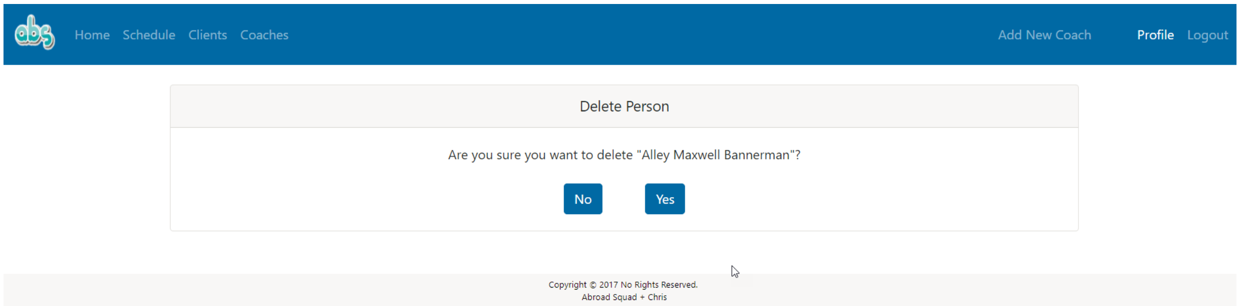
Password Reset Supervisor View



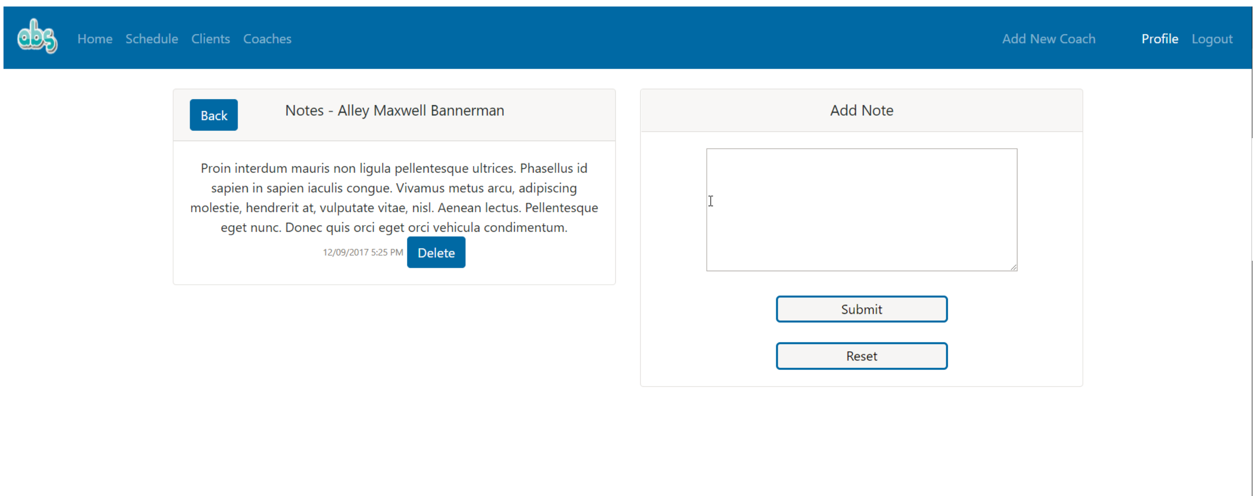
Editing Profile Supervisor View



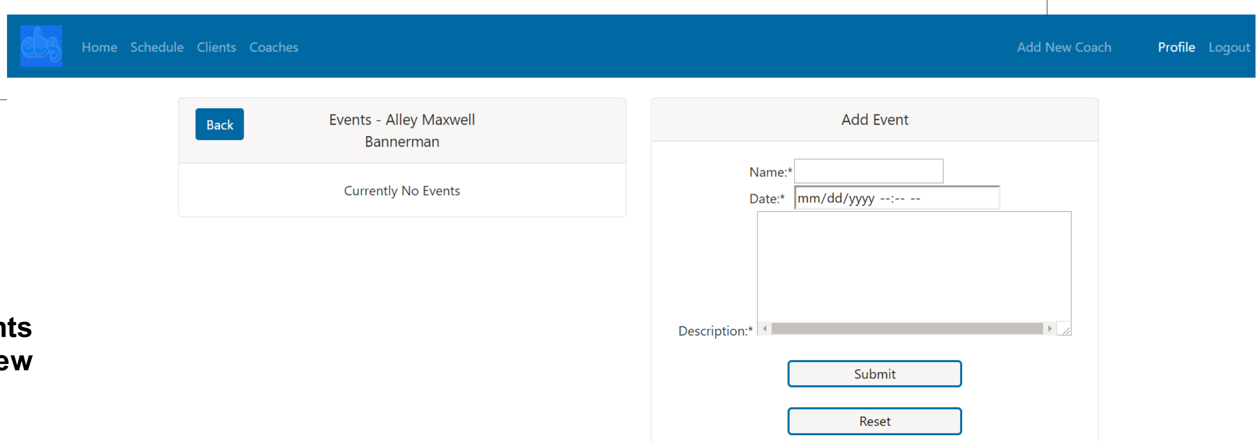
Re-Assign Coach Supervisor View



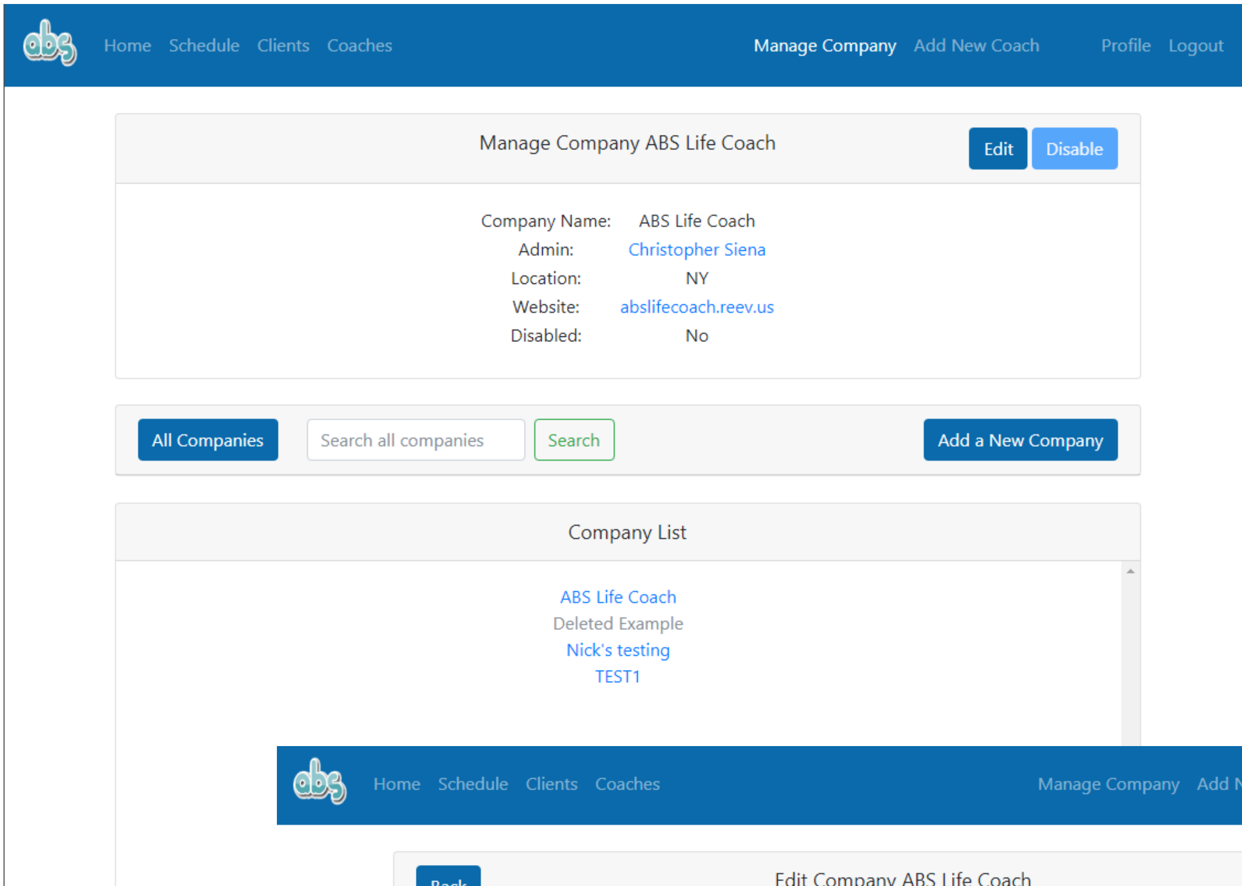
Delete Person Supervisor View



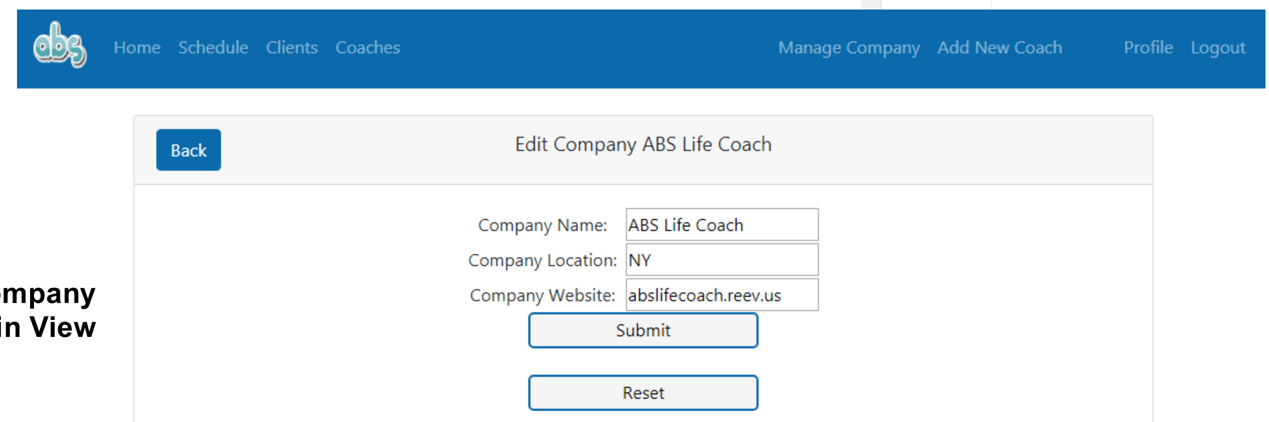
Notes Supervisor View



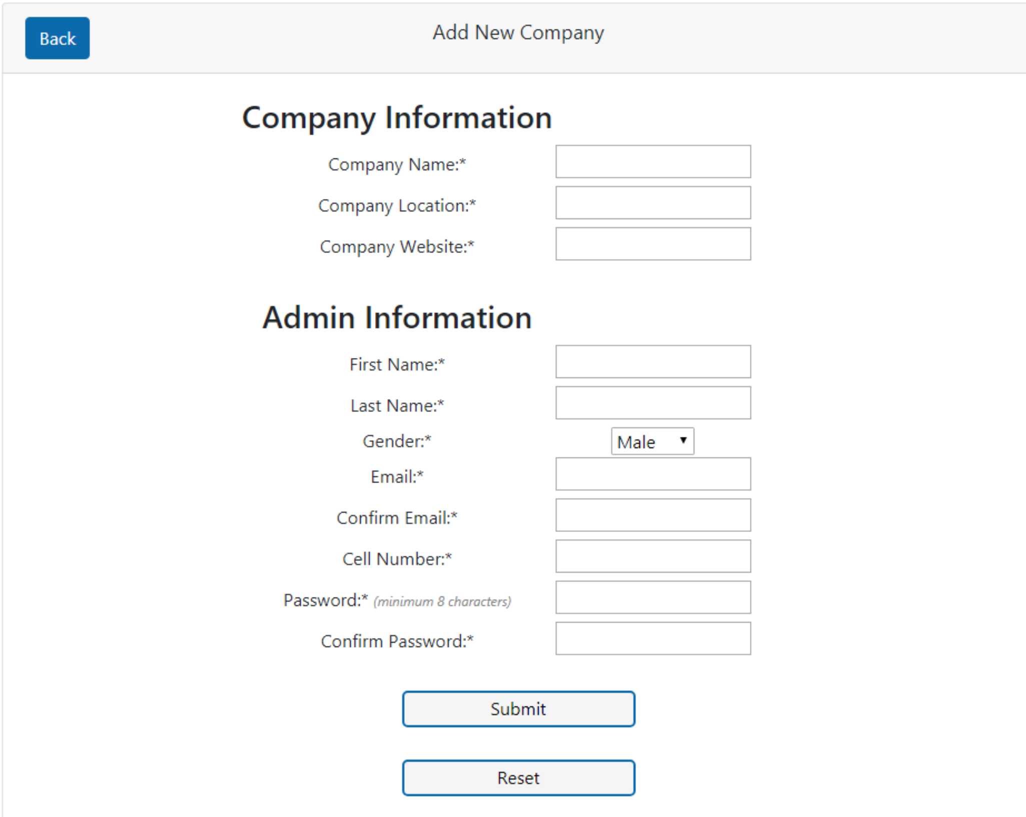
Life Events Supervisor View



Manage Company Admin View



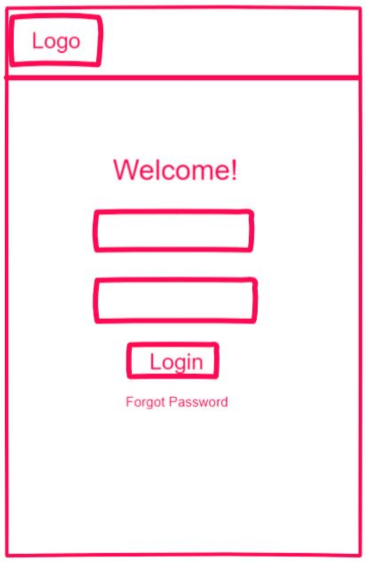
Edit Company Admin View

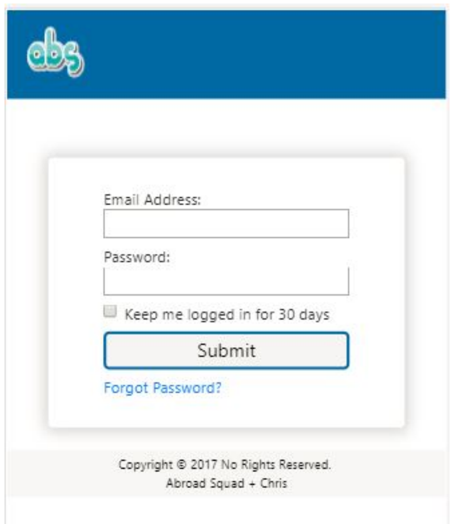


Add New Company
Super Admin View

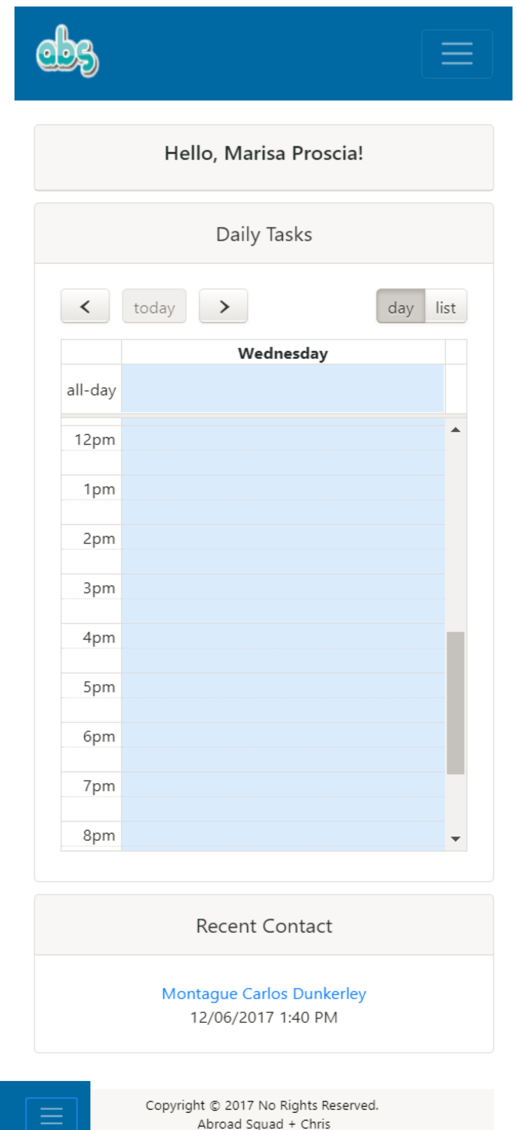
2. Mobile View

Login Page

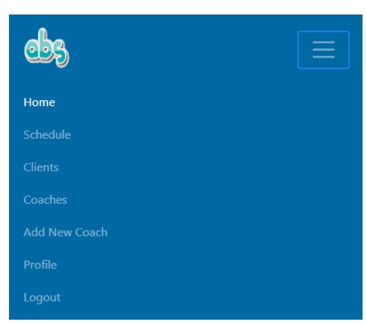




Home Page

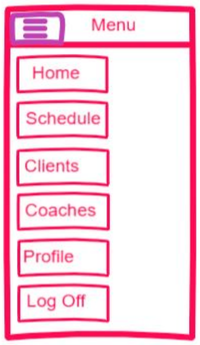


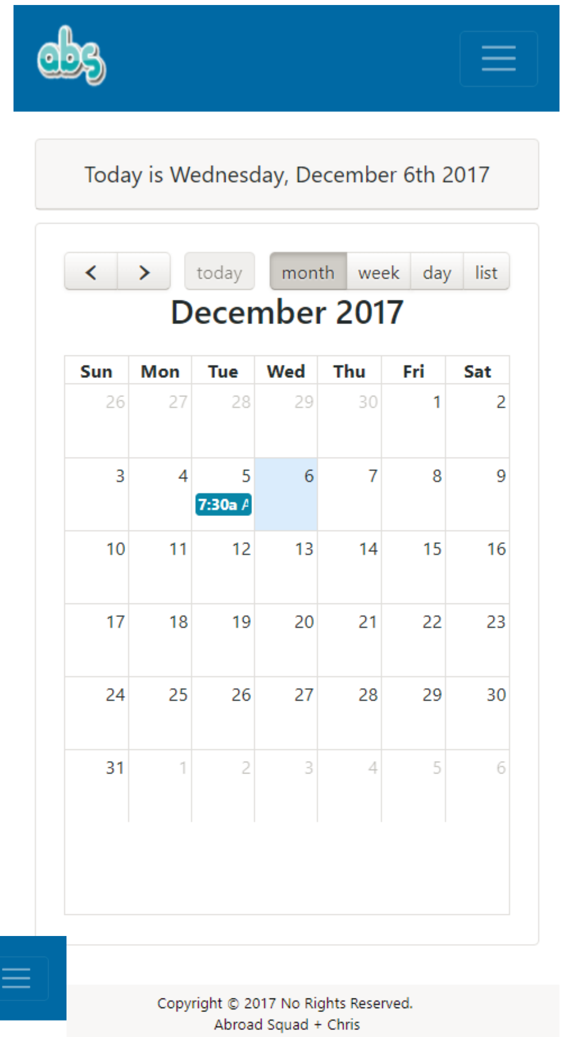
Home Page



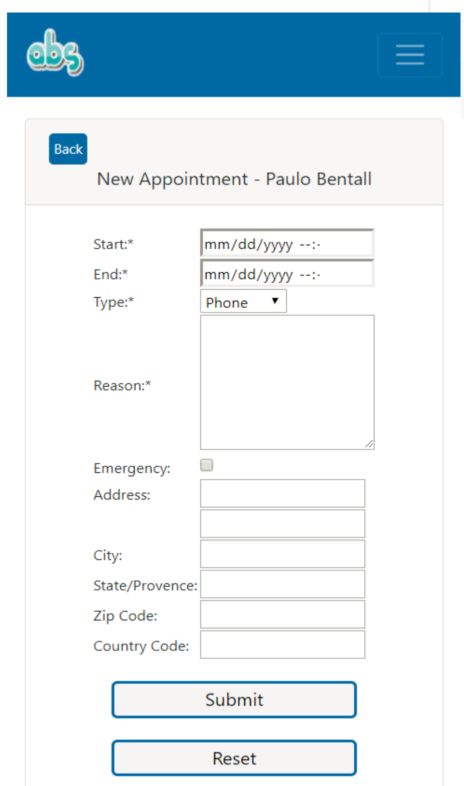
Hamburger Menu Supervisor View

Schedule

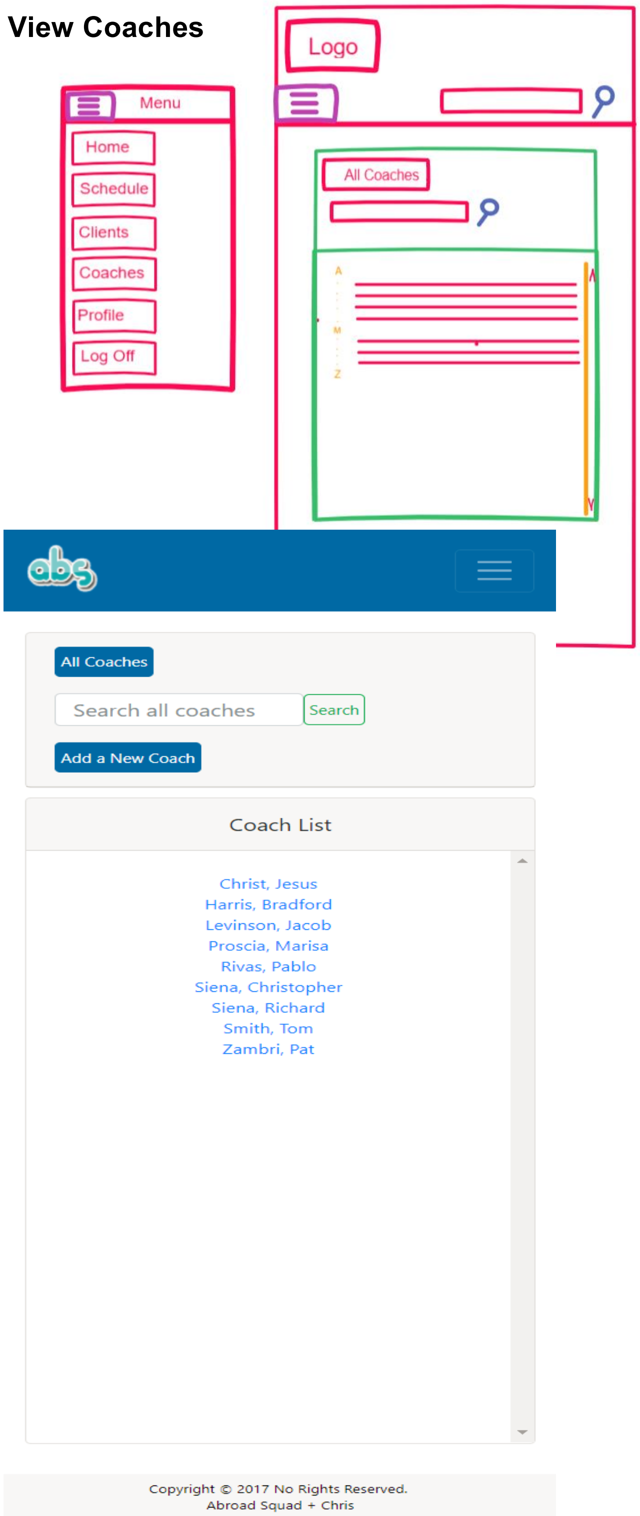




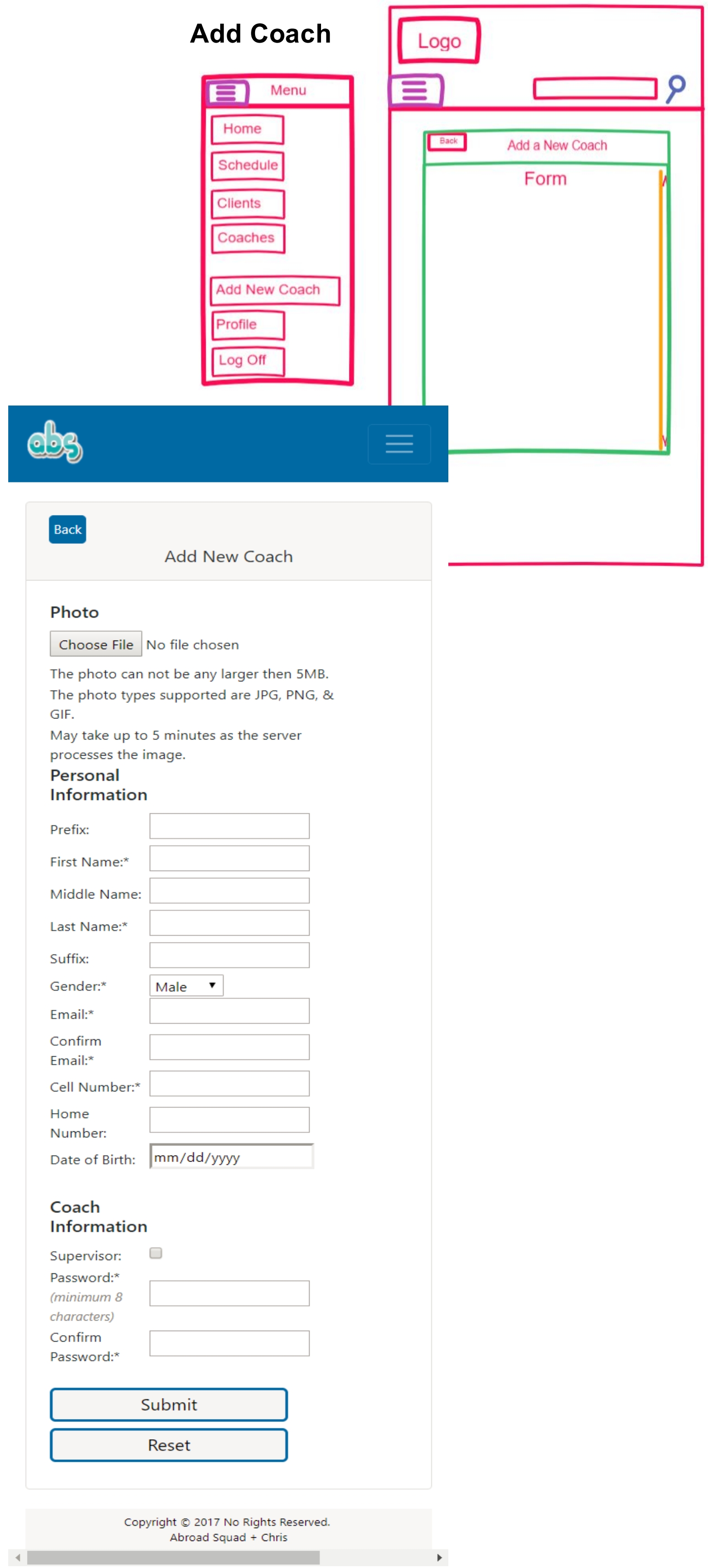
Add Appointment



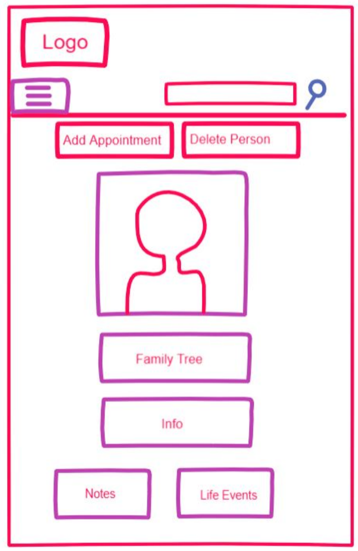
View Coaches

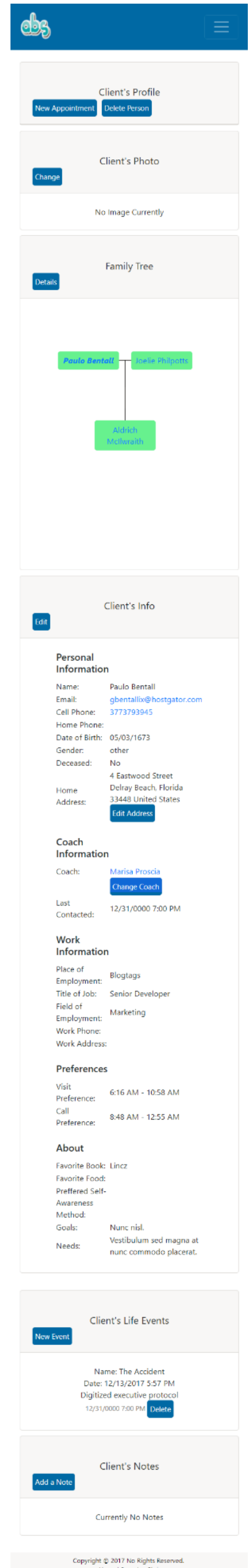


Add Coach

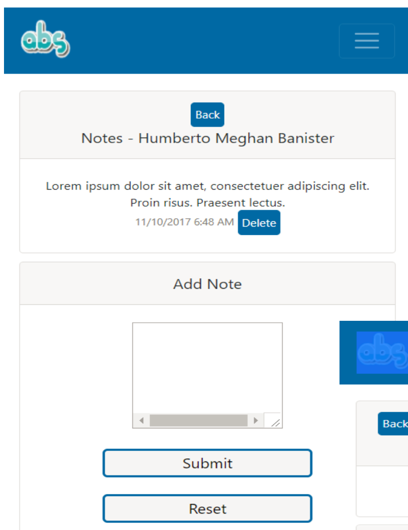


Client Profile

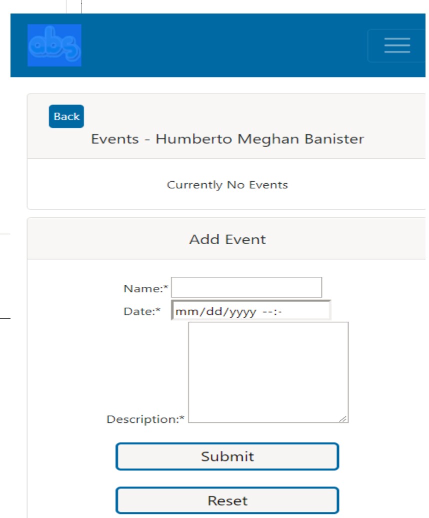


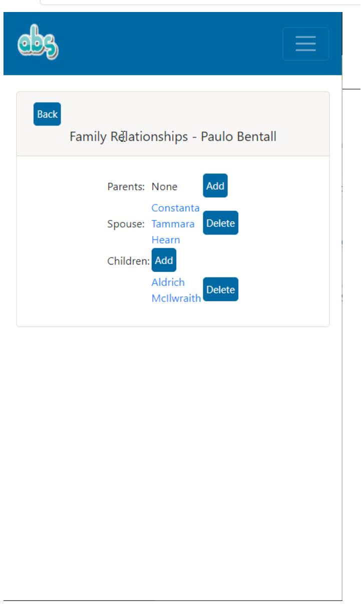


Notes

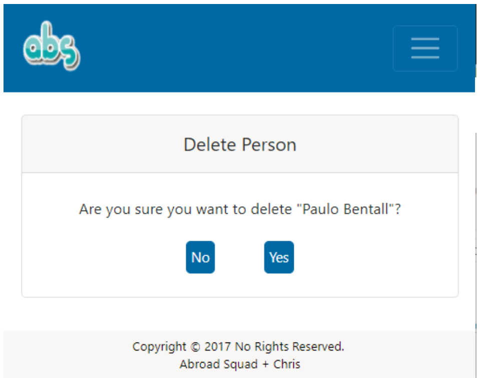


Life Events



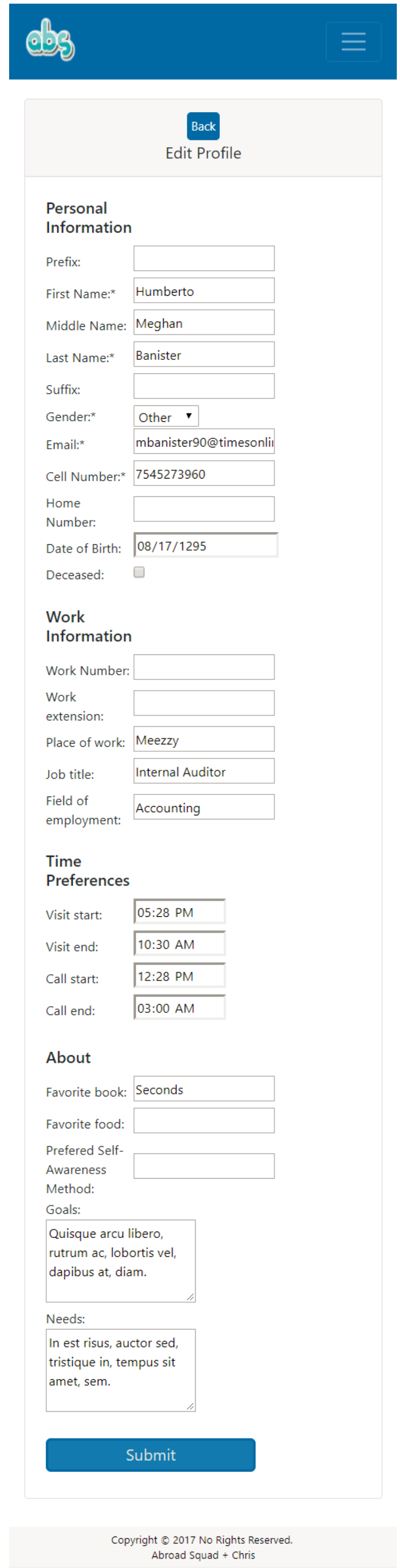


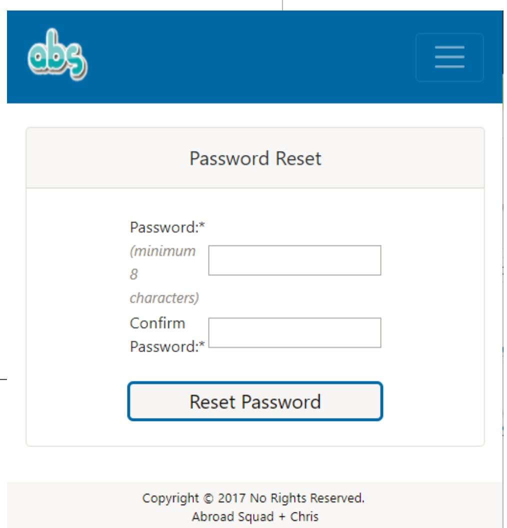
Viewing Family Relationships



Delete Person

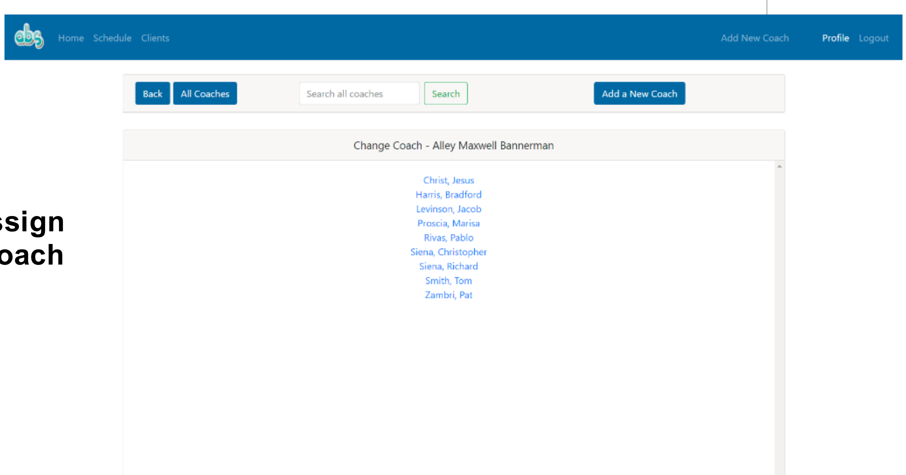
Edit Profile





Password Reset

Re-Assign Coach



3. Additional Resources

Bootstrap v4

As our CSS Framework, we have implemented Bootstrap v4.0.0. Bootstrap was chosen due to its mobile first approach and responsive user interface. In addition, a responsive grid system and extensive front-end component library allowed for a simple integration within our site as well as easy to use functionality to deliver the most efficient site for our client. To implement Bootstrap into our site, the Bootstrap v4.0.0 library was downloaded to our site repository. This package included the compiled CSS and JavaScript necessary to properly implement Bootstrap in our site. During our first implementation of Bootstrap, Bootstrap v3.3.7 was downloaded. However, shortly after beginning development Bootstrap v4.0.0 was released. In turn, Bootstrap v4.0.0 was downloaded to our site and Bootstrap v3.3.7 was removed. Further Bootstrap documentation can be found at https://getbootstrap.com/docs/4.0/getting-started/introduction/.

FullCalendar

As our dynamic scheduling system, FullCalendar has been implemented. FullCalendar was chosen due to it's ability to handle events and it's responsive user interface. FullCalendar provides extensive documentation, while being customizable and open sourced. FullCalendar provides all of the necessary files to properly implement the dynamic calendar into our site. These files include CSS files and JavaScript and jQuery files necessary for the site. To implement FullCalendar into our site, the FullCalendar package was downloaded to our site repository. Once implemented into our site, scripts were written to allow for appointment adding to a coaches calendar. In addition, appointment features were integrated into our database. Currently the schedule feature on our site allows for coaches to schedule appointments through a client's profile. This allows for accurate declaration of which client the appointment is for. After creating a new appointment, a coach's calendar is updated with the new information. Further FullCalendar documentation can be found at https://fullcalendar.io/docs/.

dTree v2.0.2

To include a dynamic family tree showing relationships among clients, dTree v2.0.2 has been implemented in our site. dTree is a library used to visualize data trees and available for use through a GitHub repository. Authored by Erik Gartner, through a MIT license, substantial documentation for dTree is available. Because of this dTree has been implemented into our site for use as our family tree program. To include dTree in our site, a compiled javascript file has been added to our site repository. In addition, dependencies of D3 v4.x and lodash v4.x have also been added. Changes have been made to the dTree implementation to better utilize the library for our specific project needs. Currently each person profile displayed on our site includes a family tree, even if there have not been relationships declared for that person. Relationships can be added for parents, spouses and children of a client. Relationships are added through the use of our client directory. Ultimately, the Family Tree functionality provides coaches with the ability to quickly link to the profiles of related people within the system, as the family tree displays clickable links. In addition, our family tree functionality is currently nested within itself. For example, if a coach is trying to view the cousin of a client, they would first need to view a parent and grandparent to find the cousin that is related to the client.Further documentation of dTree can be found at https://github.com/ErikGartner/dTree.

VII. The Project

A. GitHub Repository

Our project repository can be found at https://github.com/Chris221/Life-Coach. In addition to our final code, updated documentation, diagrams and all weekly status reports can be found on our GitHub repository.

B. Live Web Demo

Currently, our system is live at https://abslifecoach.reev.us/.

C. Test Cases

Test Case #1 - User Permissions
Case: A coach that is not a supervisor would like to delete another coach from their company.
Instructions:

- At the top of each coach profile there is a “Delete Person” button. If a non supervisor tries to delete other coaches or if a supervisor tries to delete someone they are not allowed to, they will incur a greyed out delete button, not allowed to be clicked. The same will happen if you try to delete yourself.

Test Case #2 - Day to Day Operations of a Coach
Case: A coach would like to add a new client. Once they add the new client the coach would like to add relationship information for the client’s family members.
Instructions:

- To add a new client, the coach must go to the “Clients page” and click on the “add new client” button. You will be brought to a form to fill out which has asterisks next to the mandatory fields. After completing this, the client will be assigned to whatever Coach created the client, however supervisors can change this at any time. By going to a specific client’s page, a coach can set an appointment with that client, add life events, and notes. Coaches may also manage a client’s family tree by adding and deleting relatives and updating accounts depending on whether or not the person is deceased. In order to add a relative, the coach must make a client account for the relative and then search them up through the add relative function when editing the family tree.

Test Case #3 - Supervisor Functionality
Case: A Supervisor would like to add a new Coach to their company.
Instructions:

- Supervisors can add new Coaches from the Coach page, or by clicking the “Add New Coach” button at the top right of the page. It is very similar to adding a client, however only supervisors and admins can add coaches. When it comes to adding a new coach or clients, any duplicate emails will throw an error, as well as emails that do not match, passwords that do not match and passwords that are under eight characters.

Test Case #4 - Super Admin Functionality
Case: A Super Admin would like to add a new company instance.
Instructions:

- Super admins can add, edit, and disable companies. Adding a company will ask for the name, location and site domain of the company, as well as the information of a new admin account for that company. Once the company and admin are set up, the new admin can edit the information of the company and do everything a supervisor can do.

Test Case #5 - Admin Restrictions

Case: An admin would like to disable their company.

Instructions:

- Anyone under an admin does not have access to the manage company page, therefore they cannot edit or delete anything. The only person with the power to delete or add companies is the super admin. Admins and the Super admin will have a special view of the site which will allow them to go to the page to edit and manage companies, respectively. When attempting to delete their company, an admin will only be presented with the ability to manage the company's information. No delete option will be visible to them.

Test Case #6 - Malicious User

Case: A user attempts to use SQL Injection

- We have taken the correct precautions to make sure that SQL Injection attempts will be unsuccessful.

VIII. User Documentation

Currently the following User Documentation is available to the client in print form. User Documentation will later be added as help documentation within the application.

A. End User Documentation

Each user of our system has a different level of permissions and functionalities available to them. Below is an outline of each level of permissions and the site functionality that corresponds to that level of permissions.

Coaches

- Edit Profile: Click "Profile" button on top right of page. Choose what part of your profile you would like to edit. This works the same for editing a client from the client's page.
- Add a Client: Click the "Clients" button on the top left of the page. Click "Add New Client" a d fill out the form.
- Editing, making an appointment with, and deleting a Client: Search the client from the "Clients" page and go to that client's profile. Choose which section you would like to edit or click delete button at the top to delete client. Creating an appointment can be done at the top of the Client's profile page.
- Editing Client's Family Trees: From the client's page, click the edit button above the family tree. Here you can choose to add a parent spouse or child. If a family member does not exist in the list, add a new client, then input into family tree.

Supervisors

- Add a Coach: Click Add new coach button on the top right of the page. Fill out the form and submit.
- Assigning a client to a Coach: Go to the client's profile page. Under "Client's Info" there will be a "Coach Information" section. Click the "Change Coach" button and select a new Coach from the list.

- Delete a Coach: On the Coach's profile page, click the delete button. This will bring you to a confirmation page. Click yes or no.
- For all other coach related questions, see the Coaches section.

Admins

- Edit Company Info: Click the manage company button on the nav bar. Click edit, and here you can edit the company name, location, and site domain.
- For all other questions see "Supervisors" and "Coaches" sections.

Super Admin

- Manage Companies: From the Manage companies page, you can add a company with the form, which will create a new admin profile. You can also edit and disable companies from this page.

B. Admin Maintenance and Code Documentation

Further documentation of the most current server instance and implementation can be found in Section V: Infrastructure Design. Information necessary to maintain and update the server instance can be found here as well. Further documentation of the most current code and database information relative to Front End Development can be found in Section VI: Application Design. Information necessary to maintain and update Front End Development of the system can be found here as well. Presently, our Database instance provides Logs for Super Admin to monitor site activity and provide maintenance or updates when necessary.

IX. Conclusion

Life Coaching is clearly more than just an assistant to daily tasks, or a therapist for when times get tough. They are a combination of it all, counselors, consultants, and even friends. The importance of these individuals is immense, as having a designated Life Coach has a huge impact on the way clients can change their lives for the better. Many people who use Life Coaches are executives and CEOs of companies, and they use Life Coaching to take their professional careers and businesses to the next level. There is also a wide range of "Blue Collar" workers who use Life Coaching to help with the advancement of their personal lives. Life Coaches are accountable, efficient, and experts in the field, and they want their clients to be able to reach their most seemingly unattainable life goals.

The development of the ABS Life Coach platform was completed by a skilled team of five specialized Computer Engineers. In order to ensure that the final product included all given specifications, much of the development process was surrounded around gathering client requirements. These requirements helped the team create a prototype for the website and database, and from there we're able to begin full development and eventually arrive at the polished product that you see here.

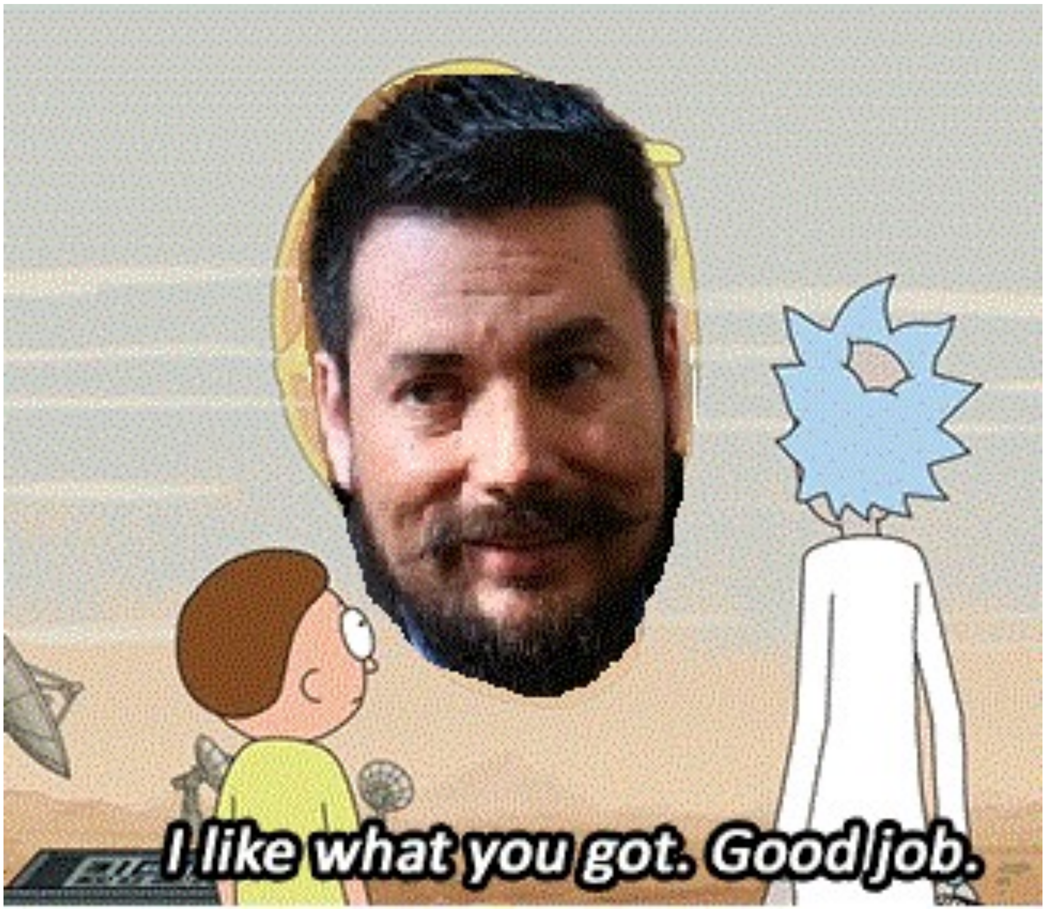
X. Appendix

A. Appendix I

Capping CS/IT/IS Project
CMPT 475-200

Documentation for Production Instance of Ubuntu 16.04 Running Apache2, PHP, and PostgreSQL on Google Server Platform

ABS Life Coach
Abroad Squad + Chris


I like what you got. Good job.