Table of Contents

1.0 Introduction

Every friend, family member and colleague are more connected now than ever before with social media and messaging apps such as WhatsApp, Instagram, and Facebook. We all know how convenient it is to communicate with people who may not be in the same room as us, especially when those people are close friends, but what about the people who are not close friends? Imagine a hypothetical situation where John leaves his friend’s house an hour before a thunderstorm is predicted to hit. John glances back at his friend's house and notices that a patch of roof shingles are flapping in the wind. John arrives at his own house and sends his friend a message to warn him that he should do his best to secure the shingles on his house before the thunderstorm arrives. This would save John's friend time and money that his friend might otherwise spend on repairing excess damage caused by shingles flying off his roof as soon as the thunderstorm hits.

We have all likely been in a similar situation where we needed to send pertinent information to the people closest to us to help them in specific situations. This could be as simple as reminding a forgetful friend that the tax due date deadline is approaching soon. This type of caring communication should not be limited to close friends and family but should be made an option for businesses that emphasize care and compassion for their customers such as State Farm.

State Farm as a company and the insurance agents that they employ are responsible for providing friendly and helpful service to all of their customers. Since the company's inception, State Farm has focused on ensuring that their insurance agents provide service to their clients as if agents were personal friends with each client. Since State Farm employs over 19,000 insurance agents and handles over 84 million insurance policies, it is impossible for each agent to build a personal relationship with each client. What is absolutely attainable is for our team to increase the quality, value, and ease of communication between State Farm agents and their clients. Similar to the story about John, we envision building a platform that will allow agents to easily send text messages or emails to their clients based on the client's location. In the hypothetical scenario, John saw that his friend's house was vulnerable to the thunderstorm and so John gave his friend a heads up about fixing the problem before it became bigger. This type of scenario could easily apply to State Farm agents. For example, if a State Farm agent learns of a predicted thunderstorm in an area prone to flooding, the agent could send a text message to all of their clients in this area to remind them to make sure they have sand bags placed around the parts of their home that could flood. If there is a nearby fire causing significant air pollution an agent could send a text to their clients in the affected area to let their clients know that they should prepare for a possible evacuation.

State Farm agents do not have a tool that allows them to send mass communications to their clients based on their clients addresses. Currently State Farm agents have tools to search for specific clients based on attributes such as birthdate or address, but this tool does not enable swift mass communications as agents still need to manually enter each client’s contact information into their computer to send them a message. This is time consuming and painstaking, especially when an agent needs to send the same message to 20 or more clients.

Our solution is to build a user based web application that allows agents to send texts and emails to groups of clients based on their clients personal attributes. The most important client attribute is their location. The focus of our web application will be a single page that will contain a digital map that displays each client's home address as a pin on the map. State Farm agents will be able to search for clients just as one would search for restaurants on google maps by panning around the city. Agents would also have the ability to select clients based on zip code or by dragging their mouse over the area of clients they would like to select. In addition to this digital map we will also implement a search bar that will allow agents to search for specific clients based on attributes such as age, location, or policy type. When an agent selects a client, that client will be transferred to a pool of selected clients. When an agent is ready to send their message to the selected pool of clients they type in their message, choose text or email and the message is sent in the selected format.

2.0 Technological challenges

In order to implement features such as a modern web application interface, a digital map, and a backend database that integrates easily with State Farm's databases we will need the appropriate tools such as a website front end framework, an easy to use Geographic Information System API, a backend web framework, and a proper database.

2.1 Front End

Although a usable web application can be built with html, css, and javascript using a proper front-end framework will ensure consistency and responsiveness on our website. A useful front-end framework will supply our team with styled buttons, textboxes, and pop ups which will ensure that each section of our web application looks similar. Besides the cosmetic features of our site, a front-end framework will reduce our team's development time which would otherwise be spent creating custom UI elements as modal, pop ups, textboxes, and buttons will likely be provided by our front-end framework.

2.2 GIS API

Besides implementing a modern, beautiful, consistent, and responsive web application user interface we will also need to find a digital mapping tool that will allow us to display an agent's clients and allow agents to select clients interactively with their mouse on the map. Our chosen GIS API will need to provide behavior similar to google maps which allows users to search addresses and click on points of interest in the specified area.

2.3 Backend Framework

No matter how beautiful or feature rich our website is, managing a large web application's development can get messy which is especially true if a backend framework is not implemented. Without a backend framework forcing developers to organize their code in a structured format, it will be hard to find specific files and debug our website. This is why a backend framework is necessary as it will provide our code structure and enable maintainability and readability amongst our code base. A backend framework will also provide us with tools which will allow our team to easily manage users of our web application and their passwords among many other common web application features our product will need.

2.4 Database

No website is complete without a proper database to store, retrieve, and save user information. We will need a database that is the same or extremely similar to the database that State Farm uses which is MongoDB. Our database will also need support for our chosen backend as our backend will be the sole arbiter between the database and our website's user interface.

3.0 Technology Analysis

Choosing the correct technologies to suit our project's needs is not a simple task. Choosing the most used and mainstream technologies can lead to headaches down the line since our project has specific needs. Choosing the most appropriate technologies will save our team time and will make development easier.

3.1.0 Front-End Framework Analysis

3.1.1 Introduction

The front-end GUI of an application is one of, if not the most important parts of the software. This aspect of an application not only determines how usable an application is, but also several other things such as representing the quality of the application, functionality, and the friendliness to the user. This is an integral part of our software as the users of our software are insurance agents that do not have the technical skills in computer science.

3.1.2 Desired Characteristics

In order to deliver the best front-end possible, we have outlined below some of what we think are the most important qualities of the technology we would look to use in our development.

- **Easy-to-use Interface:**
  We want a user interface that is easy for users to both understand and to use in practice. This means that our UI should be clean and layout should be optimized with the intended users in mind.

- **Back-end Interfacing:**
  Another big part of our front-end is its ability to contain and support our back-end functions and operations. This means that our maps API should work well with the implementation of our front-end and the fields that the users fill in should correspond well to database queries. Communication between our front-end operations and back-end operations is extremely important and should work seamlessly.

- **Professional Look:**
  Going hand-in-hand with an easy-to-use interface, we also want our UI to look clean and professional. This will also help with the overall look and organization of the UI.

- **Ease of Implementation:**
  Ideally, we would like the implementation process to be as painless and hassle-free as possible. This would reduce development time of the UI and allow the team to focus on the more complicated parts of the project which we deem to be the backend functionality.

- **Maintainability:**
  Lastly, we need our UI to be maintainable and easy to update. This is so we can easily make changes and update our UI as needed without too much overhead and time being spent in that part of the project. This is also good for later in the life of the project if, for instance, we need to add functionality.

3.1.3 Alternatives

- **AngularJS**
  AngularJS is a JavaScript-based framework that is used by many to develop web applications. It is an easy to learn framework as it is very similar to common web

development apps such as HTML, JavaScript, and CSS. It is also backed by Google which means that upgrades and support is provided by Google engineers. This framework is also open-source and thus provides developers with an even greater insight into AngularJs' functionality. It also has support for real-time testing which makes debugging and error-checking much simpler. A very important aspect of AngularJS to our project is its cross-browser compatibility.

- **Bootstrap**

  Bootstrap is a UI development framework that is based in HTML5 and is incredibly simple to use. Requiring almost no knowledge of even the most common web development languages (HTML, CSS, etc.), Bootstrap features a host of premade templates that can be edited easily. Because it exports your design to CSS and HTML files, it is also easy to tweak settings in the code itself if you want to customize further then the built-in editor will let you. Like AngularJS, Bootstrap is also supported by many browsers. One con is that this framework doesn't have much in the way of back-end interfacing, which is a definite problem for our project.

- **Ionic**

  Ionic is an open-sourced UI toolkit that is primarily focused on the interface part of application development. It is very easy to use and doesn't introduce any new languages outside of basic web development languages like HTML and CSS, making it also easy to implement as we would not be spending any extra time trying to learn new languages. There is also a strong community of developers, making any learning curve even more nonexistent with the ability to ask and receive feedback on questions about the framework. Another great feature is that the Angular framework is integrated into Ionic, which makes JavaScript development even easier. While this option has a lot of pros, it is mainly geared towards mobile web app development, specializing in a sort of hybrid application that creates web pages using an instance of a mobile browser's web page. While this could be used to support our project, users say that the web application development aspect is lacking and usually generates more overhead than it is worth.

3.1.4 Analysis

AngularJS:

- **Web application compatibility**

  Since AngularJs is a JavaScript framework, it can easily implement all of the JavaScript requirements that our project has. It is also developed by Google with web app development specifically in mind. This makes it a great fit for our project as it provides a quick way to design the functionality of our UI. Testing is supported in real-time which helps with debugging and error checking.

- **Ease of implementation**
  As we mentioned above, the implementation is reasonably easy as AngularJs is based on common web development languages, so it is pretty easy for us to pick up as we are all familiar with such languages. It also uses declarative UI which reduces the need for us as developers to specify program flows and load orders. On top of all of this, there is a very wide community that is available to help answer questions, if we have any, and lots of experienced developers in that community.if

- **Maintainability**
  AngularJs comes with a CLI (command line interface) that helps tremendously with code consistency and reusability. This would allow our entire team to maintain our AngularJs code while not compromising the integrity of our code with inconsistent files. With this feature built in, maintainability becomes extremely easy.

- **Back-end support:**
  As this is a JavaScript framework, the back-end support is very expansive, covering all of the back-end requirements that our project's UI would need. This includes support for interfacing with our database, our embedded GIS, and of course the general functionality of our front-end UI.

- **Platform support:**
  AngularJs has support for many platforms and browsers which isn't surprising considering it is supported by Google. This is critical for our project as we don't want to limit our users on how they are able to access our application. With that said, AngularJs supports most if not all the current major browsers such as Firefox, Android, IOS, Safari, and of course Google Chrome.

Bootstrap:

- **Web application compatibility**
  Bootstrap works well with many websites and web applications. Being a simple front-end UI framework, it has many applications for creating the visual aspects of web applications which makes it a very nice choice for our front-end GUI. It's customizability also makes it stand out for our project as this means that even if we aren't able to find a template we completely like for our front-end, we can customize further by simply editing the corresponding HTML and CSS files directly.

- **Ease of implementation**

Bootstrap has one of the easiest implementations of any framework since most of the UI design that is incorporated requires no actual programming at all. While we will almost definitely be editing the HTML and CSS, it's nice that some of the workload of creating a front-end UI is offloaded, allowing our team to focus on more of the functionality of the project rather than worrying about debugging our HTML and CSS code.

- **Maintainability**
  Bootstrap also handles consistency extremely well, completely eliminating the use of different libraries between developers. This means that the framework essentially forces consistency automatically. It also doesn't matter which browser that you use since it supports that same functionality across many browsers. This maximizes maintainability as our team would always have a consistent front-end design as well as a simple and easy to edit framework to work with.

- **Back-end support**
  One major downfall of Bootstrap is that it has little to no interface with our back-end operations. Because of the languages it uses, communication and interfacing with our back-end would be extremely difficult if not impossible. This would be a huge disadvantage for our team because our project needs to support a considerable back-end system.

- **Platform support**
  Bootstrap has support for all major browsers, just as AngularJs does. As we've mentioned previously, this is imperative for our project to be as accessible as possible. Such browsers that the framework supports are Chrome, Safari, Explorer, and Firefox.

Ionic:

- **Web application compatibility**
  While it may seem at first glance that Ionic has most if not all of the features that our project would need, most of those features are strictly aimed at mobile app development. As such, there is actually not a lot of web application support in Ionic. This is disappointing as the feature set that you get with this framework is quite impressive, featuring HTML and CSS for front-end and Angular for back-end.

- **Ease of implementation**
  Implementation is easy with Ionic being that it doesn't introduce any new languages that a developer experienced in web development wouldn't already know. Supporting HTML, CSS, and the Angular framework, there isn't a large learning curve at all.

- **Maintainability**

Unlike both AngularJs and Bootstrap, Ionic does not have any native consistency tools. This means that we would have to find an outside source (such as GitHub) for version control and consistency across the code. While this is certainly doable, it is not ideal when our other two options have consistency systems included and dedicated to their frameworks. Its UI design is also not as simple as Bootstrap's and it supports Angular for its back-end without the version control making it the least maintainable of the three options.

- **Back-end support**
  As Ionic uses Angular as its back end, it has the same great back-end support that AngularJs does. It can also work with any JavaScript by just adding a script tag, a useful tool if we need to add any external JavaScript to work with our front-end.

- **Platform support**
  Finally, Ionic supports all major platforms just as the other two options do, an aspect that is consistent across all of the options chosen for review because of its importance.

3.1.5 Chosen Approach

Below is a table outlining our scores for each of the reviewed frameworks:

Technology Name:	AngularJs	Bootstrap	Ionic
Web application compatibility (scored out of 10):	10	10	3
Ease of implementation (scored out of 10):	9	10	9
Maintainability (scored out of 10):	8	10	5
Back-end support (scored out of 10):	10	1	10
Platform support (scored out of 10):	10	10	10

Figure 1: Table displaying the score for each desired characteristic for each alternative.

For our final decision, we threw out the Ionic framework as while the feature set was good, it unfortunately was geared more towards hybrid and mobile apps and not the web application our team is designing. This is reflected in figure 1 in the "Web application compatibility" category. That left AngularJS and Bootstrap which, after some further research, we found could work in tandem. We found that many developers use Bootstrap for the front-end UI implementation and AngularJS as their functionality and back-end implementation. This creates a very appealing blend of a simple and clean looking UI and a powerful back-end functionality that our team is looking for (see figure 1). It supports our two main goals for the UI; having a user friendly and appealing UI for our clients to interact with as well as a powerful and functional back-end that can support our implementation of the functional requirements or our project.

3.1.6 Proving feasibility

Our team has already worked with Bootstrap in the setting up of our team website and has had a good experience with the framework. However, to test our choice of two frameworks working in tandem, our team will be working on a prototype UI for our web application very soon to make sure that this implementation fits our project well.

3.2.0 GIS Analysis

3.2.1 Introduction

Our web application is largely based around the ability for State Farm agents to easily view and interact with a graphical map in order to efficiently identify clients within a specific geographical region; The State Farm agents should be able to draw, or otherwise, select a region on the map manually / interactively. There are plenty of GIS(Geographical Information System) frameworks / APIs that can easily display a map in a web application, but extracting the specific data from an interactive map, to be used in our backend, can complicate things and make deciding on a framework more difficult. Many GIS APIs involve very costly call limits that could quickly create a massive amount of overhead, especially considering our web app is designed to be used by thousands of users concurrently.

3.2.2 Desired Characteristics

The desired characteristics of our GIS framework cover the “fundamentals”, and are rather straightforward; We will be considering three main characteristics: tools and features, price point, and community support / ease of use.

- **Tools and Features**

The interactive graphical map will be the center stone of our web application, the map needs to be fast, responsive, and easy to use for State Farm agents. The map framework that we choose should have plenty of tools / methods provided for us to build out the interactive graphical map in a way that is conducive to the workflow of the agents that will be using it. For example, our client has requested that they be able to “outline” certain regions on the map in order for the agents to easily select demographics, then notify them of various things based on pre-selected queries. The GIS framework can be thought of as two parts, the map visualization, and the geocoder; Plenty of GIS include both of these features, but the geocoder typically requires the use of an API and as I mentioned before, the use of APIs is far from cheap. The geocoder is what takes an address / location and turns it into geographical coordinates, Latitude & Longitude. The opposite of this process is called reverse geocoding, which is typically included as well. Geocoding in some capacity is essential to the operation of our application, the State Farm agents are going to be consulting the map for client locations constantly.

- **Price Point**
  We would like to be able to achieve this while maintaining as little overhead in cost as possible, preferably for no cost at all! So, while it is entirely possible to keep this as an “all in one” package, splitting this portion into two pieces and finding the best alternative for each is seemingly the best way to approach the issue in order to reduce overhead and minimize the use of APIs, which typically cost hefty monthly fees in order to use.

- **Community Support / Ease of Use**
  Once we begin development of our project, we want to hit the ground running and waste as little time as possible learning new software and figuring out how things work. If we can choose an alternative that has plenty of community support and well written documentation, this will significantly decrease the amount of time spent getting familiar with the new framework / API. Whether it be Youtube tutorials or forum posts about pesky issues, it will all help us keep the ball rolling once we begin the integration process.

3.2.3 Alternatives

- **Leaflet + Nominatim**
  Leaflet is the self-proclaimed leading open-source JavaScript library for mobile-friendly interactive maps. This is an incredibly lightweight library, at a measly 39KB of JavaScript, and provides plenty of mapping features in order to create maps that suit any given project. Like many others, this library utilizes “Open Street Map” which is the leading open source mapping database. Leaflet was initially released in 2011 and has spent the years since being honed as an open source software library; Some of its popular use cases are in Pinterest, Flickr, and FourSquare.

Nominatim is a bit different from the others here, it is not a GIS, but rather a tool that can be used to support a GIS, such as Leaflet. This is a free geocoding API that also allows you to download and use it's framework hosted from your own server if you have a particularly large geocoding request load. Many GIS APIs utilize this as a supplementary API because it is open-source and free!

- **Google Maps API**
  The Google Maps API is an incredibly robust and feature rich web service that is in a league of its own when compared to other similar GIS. Developed and maintained by Google, this API service provides pretty much everything you'd imagine from a geographical interface system. This is typically the first thing that comes to mind when on the subject of web mapping, since Google has been paving the way for many years now; But, as you can imagine with such a prevalent name, comes a very large cost with it.

- **ArcGIS**
  ArcGIS is a family of client software, server software, and online geographical information system services developed and maintained by Esri, another very prevalent name in the web mapping field. ArcGIS has been around since 1999 and began its roots as a command line GIS tool for manipulating data. I heard about this one through NAU since it is supposedly offered for free with a licence acquired through the university.

3.2.4 Analysis

Leaflet + Nominatim:

- **Tools and Features**
  I've done some minor experimentation with Leaflet and all of the features and methods are relatively straightforward and provide plenty of functionality in order to create an interactive map suitable to our needs. The graphical map out the box looks very nice and also has the ability to be customized if need be. Although size is not a major concern, Leaflet is incredibly lightweight which reduces the size of our project overall and keeps unnecessary bloat to a minimum. Nominatim is one of the most popular free open source geocoders that is used in many GIS; The API provides one free call per second, but they give you the option to download the entire framework on to any given server so you can make these calls as many times as your own server can handle. This would be very beneficial to our project considering that we are designing this with thousands of concurrent users in mind. This is an incredibly capable combination and really shines in terms of affordability.

- **Community Support / Ease of Use**

Leaflet is rather popular and widely used which has led it to have a bit of a community surrounding it, this is great for finding useful forum posts or the occasional YouTube tutorial. Although the documentation is well written and provides sufficient enough information, it does feel a bit dry in comparison to something like Google Maps API documentation.

- **Price**
  One of the biggest advantages of Leaflet + Nominatim over the other alternatives is that it is entirely free and open source. Keeping the overhead in cost to little, or nothing, is very important for our project and our clients. Do note, although this is entirely "free", due to the nature and scale of our project we would more than likely need to host our own server with the framework on in order to make API calls for geocoding and the like. Still, this beats the alternative of paying upwards of hundreds of dollars a month to use a 3rd party GIS API service.

Google Maps API:

- **Tools and Features**
  This API is the most popular option among all of the alternatives; Google is a massive corporation that has an enormous amount of resources at their disposal which leads to software and web services that go above and beyond. The Maps API has an abundance of functionality and top tier GIS features, they would easily cover our needs and then some.

- **Community Support / Ease of Use**
  Google Maps is one of the most widely known GIS API systems across the world, the API is finely documented with plenty of examples and code snippets, not to mention all of the support and tutorials you could also find online via forums and YouTube; If you were to ever run into an issue, you are nearly guaranteed that there will be some form of help / solution already waiting for you on the internet. When learning anything new in regards to software, this kind of support is incredibly useful.

- **Price Point**
  This could easily be our one stop solution to implementing our GIS, but of course there is a catch. With such amazing functionality comes an equally costly price point, to get access to any meaningful services with Google Maps API the pricing starts at $200 a month! This is pretty absurd for our intents and purposes, and State Farm is not expecting there to be any cost for this project.

ArcGIS:

- **Tools and Features**

ArcGIS is a solid option that has many different tools and features and has built a reputation for itself over the course of many years, this is always a plus since the more popular a library or framework is, the larger the amount of support and community there is; When you run into an issue inevitably, you can typically find the solution rather quickly! ArcGIS also has their own geocoder included with their web service, as you would expect from such a robust package. In all honesty, ArcGIS provides way more tools than would be needed for our project.

- **Community Support / Ease of Use**
  After browsing through the documentation for a bit, it seems that this alternative would not mesh well with our team work flow and would cause some hiccups in productivity while implementing and learning how to use it. Although it is a pretty popular GIS, it does not have much of a community following and there is not much to be found on the internet in regards to tutorials or forum posts, this could really hurt us down the line when we inevitably come across some issues.

- **Price Point**
  ArcGIS was mainly considered because we would have access to a license through NAU, but we must keep in mind that we are developing this software for our client, State Farm; I don't think they would be thrilled to find that in order to continue using the software that they would have to renew a pricey subscription through Esri, the creators of ArcGIS. So while we could technically build the app for free with ArcGIS, once we hand over the product to our client they would have to pay a pretty penny to maintain functionality of the web app.

3.2.5 Chosen Approach

All three of these alternatives are very capable of providing us with the necessary tools required to implement our desired features in our web application, as can be noted by the similar scoring in *Figure 2*. Google Maps API is great for many reasons and would easily knock this portion of our project right out of the park, but its largest con lies within the price point, it is simply too expensive to get access to the tools that would be needed to complete our software. ArcGIS falls into the exact same boat as Maps API, the features and tools are there, but the price point would be a huge negative for our client; At the end of the day, why would we pay many hundreds of dollars to accomplish the same goal that could be done for free? This is why the combination of Leaflet and Nominatim, the free and open source duo, is leading the conversation for our GIS needs. Leaflet does most of what ArcGIS and Google provides but in a much more conservative and conducive manner, suitable to our project.

Desired Characteristics	GIS Alternatives		
	Leaflet + Nominatim	Google Maps API	ArcGIS
Tools and Features (scored out of 10)	8	10	9
Price Point (scored out of 10)	10	4	4
Community Support / Ease of Use (scored out of 10)	7	9	7

Figure 2: Table displaying the score for each desired characteristic for each alternative.

All three of the alternatives scored similarly in the “Tools and Features” as well as the “Community Support” but Leaflet + Nominatim really outshine the rest in the price point characteristic, it doesn’t get better than free! It may now be quite as convenient and easy to use as something like the Google Maps API, but it does everything relatively well and after my short amount of time tinkering with the library I found that it is very capable. The community support is not the greatest for Leaflet, but the documentation is more than enough to cover any basic questions or issues that might pop up. All in all, Leaflet + Nominatim seems to be a very promising candidate for our web application considering our desired characteristics.

3.2.6 Proving Feasibility

While there has already been a small amount of demoing with our chosen approach, to further validate our findings we will begin to create a prototype of our desired interactive map and ensure that it can operate to our specifications.

3.3.0 Backend Framework Analysis

3.3.1 Introduction

Backend frameworks are an important part of building a secure, organized, and maintainable web application. Choosing a correct backend framework for a webapp is paramount to implementing the features required by a web application. For our purposes, our choice of backend framework needs to be able to support up to 20,000 concurrent users and securely manage their username and passwords.

3.3.2 Desired Characteristics

Picking a backend framework is central to a successful project since the backend that we choose will be the backbone of our entire project. Each backend framework is unique and has different features for different purposes depending on the type of web application that will be built using our framework. Characteristics such as convention over configuration, built in tools, speed, ease of use, and programming language are extremely important to consider when choosing a backend framework.

- **Convention vs Configuration**
  One of the most important facets of a backend framework is whether the framework emphasises convention or configuration. A framework that emphasises configuration leaves much of the functionality to be decided and created by the developers. This can lead to longer development times because developers may have to implement functionality that would likely be predefined by a framework that emphasizes conventions. The benefits of a configuration framework is that the developers have more control over how features are implemented. A framework that emphasizes convention over configuration will require less manual configuration to get site features up and running, but much of the site configuration is done automatically in black boxes which can make debugging difficult for new framework users. For our purposes a framework that emphasizes convention over configuration is desired because many common features that will be required to create our web app are already provided by the framework which will reduce development time by decreasing the amount of time our team will need to spend creating needed features such as session management, authentication, and routing.

- **Built In Tools**
  As mentioned above, many features such as user authentication, routing, and session management will be predefined by the framework we choose so our backend framework will need capable tools which for example can support thousands of concurrent users. We must also be able to securely receive and save user passwords which will be made easier by a competent backend framework. Since our project is time constrained, our backend framework will need robust and well documented functionality to implement required features which could take too long if our framework does not come with these tools.

- **Backend Framework Speed and Efficiency**
  To support thousands of concurrent users of our web application we will not only need capable tools to do so but our framework will also need to serve these users efficiently. Not all backend frameworks have the same performance given the number of concurrent users that will be using the web application. Although our web application will need to

support thousands of users, these users will likely not be sending requests to our server every second or even minute so a framework with moderate speed will be the best fit.

- **Community Support and Ease of Use**
  Because backend frameworks can be complicated and unintuitive, we will need a backend framework that is widely adopted and has a large community of users for support. This means a mainstream framework will be the best choice as a popular framework usually has very strong support communities and resources that will assist us with any technical issues.

- **Backend Framework Programming Language**
  Lastly, our choice of backend framework dictates the primary language will be using to program the majority of our web application. This means we will want a backend framework that implements a programming language that is easy to learn, write, and maintain.

3.3.3 Alternatives

- **Django**
  Django is a high level backend framework that comes with many prepackaged tools to assist in web development. This framework is extremely popular and is known for supporting rapid development and scalability. Django is known for its 'batteries' included design philosophy which means many tools that a developer would need to create themselves are already included. The framework was written by Adrian Holovaty and Simon Willison and was released in 2005 and uses the immensely popular python scripting language. Django is also open source meaning we will have access to thousands of third party libraries. Adding to its credibility, Django is used by companies such as Spotify, Mozilla, Pinterest, and Instagram.

- **Ruby on Rails**
  Ruby on Rails is a high level backend framework that was influenced by frameworks such as Laravel, CakePHP, and Django. Ruby on Rails is also open source and provides most of the functionality a developer needs to create a robust web application. Ruby on Rails uses the Ruby programming language which is known for being easy to write and learn. Ruby on Rails was released in 2004 and was created by David Heinemeier. Rails promotes quality programming standards by using the MVC architecture and strict code conventions to create well organized and maintainable websites. Because Ruby on Rails is open source, there are thousands of third party libraries available for use which goes hand in hand with the immense documentation and online community support forums available for Rails framework.

- **Laravel**

  Like Ruby on Rails and Django, Laravel implements MVC design architecture. Laravel was released in 2011 by Taylor Otwell. Given its release date, Laravel is comparatively newer and less mature than Django and Ruby on Rails. Like Django and Ruby on Rails, Laravel comes preconfigured with many necessary tools for implementing modern websites. Laravel was designed with the goal of making website development easier and less time consuming although Laravel is considered a lightweight framework and comes with less prepackaged tools compared to Ruby on Rails and Django. Laravel uses the PHP scripting language as its primary programming language.

3.3.4 Analysis

Django:

- **Convention over Configuration**

  Django emphasizes configuration over convention which means much of our site configuration will have to be explicitly declared in our code rather than being done automatically like a convention leaning framework would do. Django also follows the model view controller architecture. In Django, URL routing is much more explicit and verbose compared to a framework such as Ruby on Rails where the user can quickly define a set of commonly used URLs in a single line. Django on the other hand uses regular expressions to declare website URLs.

- **Built in Tools**

  Django comes with a built-in tool called the admin interface which allows developers to view and change their websites database tables. This means we won't have to manually install an application to manage our database and we won't have to use the command line to edit our database as often which will save us time. Django also comes with many built in security features out of the box, meaning that if a web app is built in accordance with Django's standards, their application will be secure against many different types of attacks and exploits. Just to name a few, Django comes with built-in XSS, CSRF, SQL injection, and clickjacking protection. Besides site protection Django also comes with a built in user authentication and authorization system. This allows the use of tools to allow our team to manage user permissions, user groups, and sensitive information forms. Adding to Django's list of helpful user tools, is a system for hashing user passwords which comes with Django.

- **Backend Framework Speed and Efficiency**

Django is generally regarded as a fast and efficient framework compared to other popular backend frameworks. Django is also highly scalable as shown by apps such as Instagram, Bitbucket, and Spotify which use Django and have a constantly growing user base. Django is not a bottleneck when it comes to the amount of users it can support. The amount of users our web app will be able to support by using Django will be determined by our servers infrastructure and resources such as storage, cpu and ram.

- **Community Support and Ease of Use**
  There are many Django courses on Udemy and Youtube as well as an official tutorial located on the official Django website. Developers can set up a hello world Django app in less than a day however, Django is a large and intricate framework so it may take a few months to get experience with most features Django offers. This framework seems like it has a strong support base with many tutorials on the web to help new Django developers get started.

- **Backend Framework Programming Language**
  Django's primary language is python.

Ruby on Rails:

- **Convention over Configuration**
  Ruby is a convention over configuration framework meaning many of the tools and features we will use are provided by the Rails framework. The downside to a convention over configuration framework is that there is a lot of "magic" and automation in Rails which can make it difficult for new developers to debug and understand. The upside to this is that our team will not have to manually implement many features we will need since Ruby on Rails will provide it for us. For example, Rails will automatically create redirect methods after a new url route has been created. This is helpful because instead of hard coding a url into an anchor tag, we can simply call the url redirect method which will return the url for the specific route.

- **Built in Tools**
  Ruby on Rails does not come with tools for user authentication but there are 3rd party libraries which support user authentication such as devise. For our purposes we will need to rely on 3rd party Ruby on Rails libraries such as Devise in order to implement user authentication. There is plenty of documentation on Devise, but we will still need to spend time setting it up to work with our web application. It is not clear if this is a weak point of the Rails framework as we would still need to set up user authentication in Django even though Django comes with authentication tools out of the box. Other authentication libraries for Rails include Omniauth and doorkeeper. Luckily, Ruby on Rails comes with built-in methods for defending against SQL injection, XSS, CSRF, and

clickjacking. Ruby also encrypts cookies by default. Similar to Django, the best way to build a secure Ruby on Rails web application is to abide by programming standards set by the Ruby on Rails framework.

- **Backend Framework Speed and Efficiency**
  Ruby on Rails is a scalable framework as shown by popular websites such as Basecamp, Airbnb, and Github. Scalability is something that will depend on our server infrastructure and hardware more so than the Ruby on Rails framework. Since our user base is almost guaranteed to be less than 20,000 users, Ruby on Rails should be a perfect fit for our needs.

- **Community Support and Ease of Use**
  Ruby on Rails has many different tutorials and community forums for support and learning. There is an official Rails tutorial provided by the Ruby on Rails website as well as a highly regarded tutorial created by The Odin Project. Ruby on Rails has also been around since 2004 so the Rails framework has had a long time to mature and there are a plethora of questions and answers on the Ruby on Rails Stack Overflow forums.

- **Backend Framework Programming Language**
  Ruby on Rails uses the Ruby programming language.

Laravel:

- **Convention over Configuration**
  Laravel is a convention over configuration framework that focuses on simplicity and is known for allowing developers to create web applications faster than most other PHP frameworks. Laravel was inspired by the Ruby on Rails framework and functional programming languages. Many PHP frameworks are extremely verbose and complex while Laravel goes against the grain and emphasizes simple and dynamic coding practices.

- **Built in Tools**
  Laravel is considered a lightweight framework compared to Django and Ruby on Rails since Laravel has less built in web development tools. Laravel comes with built in user authentication and session management tools. Laravel also comes with many built in security features such as automatic encryption of cookies. Laravel also contains built-in security features that protect against SQL injection, CSRF, and XSS attacks.

- **Backend Framework Speed and Efficiency**
  Laravel is comparatively slower compared to other backend frameworks, but these benchmarks can be a bit biased and are not significantly slow enough to make an impact

on our project since our user base will only contain up to 20,000 users. Laravel is also capable of supporting more than 20,000 concurrent users which is more than enough for our needs.

- **Community Support and Ease of Use**
  Because Laravel uses the PHP programming language, all of our team members will need to first learn PHP before even attempting to start working with the Laravel framework. On the plus side Laravel is known for being one the easiest to learn mainstream PHP frameworks. Laravel's official website contains a tutorial and there are also tutorials on udemy and laracasts.com. Laracast is Laravel's official website for learning new frameworks and other web technologies which means the developers that know Laravel are creating tutorials for beginners to get started. Other than tutorials, Laravel has an active community on Reddit and Stack Overflow.

- **Backend Framework Programming Language**
  Laravel uses the PHP programming language.

3.3.5 Chosen Approach

Our web application only needs to support up to 20,000 users which made the choice between backend frameworks less about scalability and features and more so on ease of use. Since our entire team knows the Python programming language Django was a top contender while Laravel was immediately dropped from consideration since it would require the entire team to learn the PHP language. Besides the fact that Laravel uses PHP there are no distinct advantages to Laravel that outweigh the downsides of spending development time learning PHP. Ruby on Rails and Django were considered on similar terms since the Ruby programming language is easy to pick up for experienced programmers which narrowed down the decision constraints to ease of use and features. The Django framework is relatively easy to learn and will make start up time faster for our team since each team member knows Python, but the Django framework does not automatically handle as much website configuration compared to Ruby on Rails. On the other hand, Ruby on Rails does much of the website configuration for us but does so in black boxes which could lead to difficult debugging.

Desired Characteristics	Backend Framework Alternatives		
	Django	Ruby on Rails	Laravel
Convention Over Configuration (Scored out of 10)	5	10	10
Built In Tools (Scored out of 10)	10	7.5	8
Speed (Scored out of 10)	10	10	5
Community Support / Ease of Use (Scored out of 10)	6.5	6	5

Figure 3: Table displaying the score for each desired characteristic for each alternative.

Ruby on Rails and Django scored similarly in most categories except for convention over configuration and ease of use. In *figure 3*, Django received a score of 5 out of 10 in the convention over configuration category while Ruby on Rails received a score of 10 out of 10. Django received a score of 5 out of 10 because it may require more manual website configuration versus Ruby on Rails, but team members will not have to master the conventions of Django to understand what is occurring in the code since Django is much more explicit than Ruby on Rails. Although Ruby on Rails handles much of the site configuration for us, each team member would have to learn the Ruby language which would make it harder to use for our team specifically. This is the reason that Ruby on Rails received a score of 6 out of 10 while Django received a slightly higher score. After conducting research it became apparent that speed would not be an issue no matter the framework we chose because of the size of our user base which is why each framework received a perfect score in the speed category. Overall Django is the best fit for our team because team members will be able to focus on learning how to use the Django framework and will not need to worry about the intricacies of using a new programming language. Django will require more explicit web site configuration compared to Ruby on Rails but this is not likely to be an issue since our team is new to backend frameworks and will likely appreciate the explicitness of Django's configuration settings.

3.3.6 Proving Feasibility

To further validate our choice our team will work on setting up a simple Django web application on their own computers. If each team member is able to do this easily without major problems, Django will be our final choice for the backend framework.

3.4.0 Database Analysis

3.4.1 Introduction

Databases are one of the most essential technologies for companies. A database is an organized collection of structured information, or data, typically stored electronically in a computer system. They allow for a wide variation of usability and are key in functions relating to information storage and retrieval. The data can be easily accessed, managed, modified, updated, controlled, and organized. Most databases use structured query language (SQL) for writing and querying data. Databases assist with the creation and maintenance of records of customers and allow for a simple solution when needing to construct reports from these records.

3.4.2 Desired Characteristics

When choosing a Database we made sure to keep in mind the scope of our project and also the overall intent. Described below are some of the key characteristics we considered when choosing our database.

- **Size**
  When developing our solution; we will need a database that can support 1000's of State Farm Agents, and all their clients. The database will need to be constructed in a manner that allows for simple accessibility, modification, and retrieval. Information, such as; name, policy, age, location, etc will need to be recorded. Databases are designed to hold much larger collections of organized information. Databases allow multiple users at the same time to quickly and securely access and query the data using highly complex logic and language. Furthering, the reasoning behind the need for a concrete structured database.

- **Relevance**
  Continuing, the database will need to be integrated with current State Farm Technology and practices. Our clients have informed us that State Farm currently has databases constructed with MongoDB and DB2. Our solution will investigate these databases as they are already used by the client, however are not limited to these databases.

- **Flexibility:**
  Many databases are created with SQL which allows us to be flexible in our approach in choosing which database to use. SQL is a programming language used by nearly all relational databases to query, manipulate, and define data, and to provide access control.

- **Tools/Functions:**
  Another key characteristic about the database that we are going to look into are their functions. We want a database that allows us to easily import, export, update, and overall simple maintenance of the database. With this in mind we will be looking at the functions and tools that are offered within specific different databases.

- **Community Support:**
  Many databases will all do the same job. However some have more support and easier learning curves than others. We will be looking for a database that offers lots of support and has functions that are simple, yet effective.

3.4.3 Alternatives

- **MongoDB**
  MongoDB is an open-source document-oriented database that is designed to store a large scale of data and allow you to work with the data very efficiently. It is a cross-platform, document-oriented database that provides high performance, high availability, and easy scalability. It is classified under the NoSQL (Not only SQL) database because the storage and retrieval of data in MongoDB are not in the form of tables. It provides official driver support for all the popular languages like C, C++, C#, and .Net, Go, Java, Node.js, PHP, Python, Ruby, Scala, Swift, etc. State Farm currently utilizes MongoDb, as well as thousands of teams, such as: Verizon, Google, SEGA, EA, and many more companies.

- **DB2**
  Db2 is a line of data management products from IBM. It includes a well-known relational database management system (RDMS) that IBM introduced in 1983. The name DB/2 originally referred to IBM's shift from a hierarchical database model to the relational database model. Although DB2 was initially designed to work exclusively on IBM mainframe platforms, it was later ported to other widely used operating systems, including UNIX, Windows and Linux and now supports non-relational structures such as JSON and XML.

- **Oracle**

Oracle Database is the first database designed for enterprise grid computing, the most flexible and cost effective way to manage information and applications. Enterprise grid computing creates large pools of industry-standard, modular storage and servers. Grid computing is a new IT architecture that produces more resilient and lower cost enterprise information systems. With grid computing, groups of independent, modular hardware and software components can be connected and rejoined on demand to meet the changing needs of businesses.

3.4.4 Analysis

MongoDB:

- **Size**
  The maximum BSON document size is 16 megabytes. The maximum document size helps ensure that a single document cannot use excessive amounts of RAM or, during transmission, excessive amounts of bandwidth. To store documents larger than the maximum size, MongoDB provides the GridFS API, however this should be more than enough for our system. MongoDB is ideal for situations that involve big data, mobile and social infrastructure, user data management, and also content management and delivery.

- **Relevance**
  MongoDB is one of the world's most popular NOSQL databases today, with over 15 million downloads and counting. Our clients have also specified that they use this system within their infrastructure daily.

- **Flexibility**
  Another good thing about MongoDB is that it is a cross-platform database which means that it can be installed across different platforms like Windows, Linux, etc.

- **Tool/Functions**
  MongoDB also uses a schema-free document database. Data is grouped into sets that are called ‘collections’. Each collection has a unique name in the database and can contain different types of objects. Every object is also called a document or a list of key-value pairs. The value can be of three types – a primitive value, an array of documents or a list of key-value pairs. MongoDB’s documents are encoded in a JSON-like format, called BSON, which makes storage easy and is also lightweight, fast and traversable. MongoDB uses Mongo server and Mongo shell commands to fetch records or the information from the database (i.e. collections).

- **Community Support**

MongoDB has rapidly grown to become an ideal choice of databases for a number of reasons. MongoDB is an open source, document-oriented, NoSQL database. It provides high performance, high availability and automatic scalability. It also helps in data modeling and data management of huge sets of data in an enterprise application. On the MongoDB Website they have many free resources for learning the system, as well as across the web.

DB2:

- **Size**
  IBM Db2 Community Edition is a free to download, use and redistribute edition of the IBM Db2 data server, which has both XML database and relational database management system features. It is limited to four CPU cores, 16 GB of RAM, a database size of 100 GB, and no Enterprise support and fix packs. This free product contains plenty of storage for the product, however if implemented into State Farm, the upgraded version would need to be purchased.

- **Relevance**
  DB2 is a database product from IBM. It is a Relational Database Management System (RDBMS). DB2 is designed to store, analyze and retrieve data efficiently. The DB2 product is extended with the support of Object-Oriented features and non-relational structures with XML.

- **Flexibility**
  DB2 has APIs for Rexx, PL/I, COBOL, RPG, Fortran, C++, C, Delphi, .NET CLI, Java, Python, Perl, PHP, Ruby, and many other programming languages. The service can also be deployable on multiple cloud providers.

- **Tool/Functions**
  DB2 also supports integration into the Eclipse and Visual Studio integrated development environments. The system also allows for elasticity; Independent scaling of storage and computing so organizations can customize their data warehouse to meet the needs of their business.

- **Community Support**
  DB2 uses an autonomous cloud service that is supplemented by a DevOps Team that are on-call to handle unexpected system failures. The technology ensures stable and reliable performance making it great for managing highly concurrent workloads. IBM offers some resources for assistance and learning DB2, however besides IBM there are not many resources available.

Oracle:

- **Size**
  Many Oracle databases tend to be medium to large in size, ranging anywhere from 10 GB to 5 TB, with some as large as 5 TB to 20 TB or more. This is more than enough storage for our project.

- **Relevance**
  Oracle allows the full control of space used, so that hardware devices can be used efficiently. Continuing, the software can be ported to work under a number of operating systems. Applications that are developed using Oracle can be ported to other operating systems without requiring any changes.

- **Flexibility**
  Oracle allows for a number of concurrent users to execute a variety of applications concurrently. A number of applications can be run at the same time by using the software. Oracle allows the full control of space used, so that hardware devices can be used efficiently. Continuing, the software can be ported to work under a number of operating systems. Applications that are developed using Oracle can be ported to other operating systems without requiring any changes.

- **Tool/Functions**
  The Oracle Server allows for processing to be split into client and server application programs. Another feature of Oracle is that, to protect against unauthorized database access and use, Oracle provides fail-safe security features to limit and monitor data access. Oracle, also maintains a high degree of overall system performance. The database users do not suffer from slow processing performance.

- **Community Support**
  The Oracle website itself has over 200 tutorial video that are free to watch on how to use certain features. The videos range anywhere from 30 minutes to 2 hours. On top of this, many other online websites offer tutorials, however some do require a payment.

3.4.5 Chosen Approach:

Below is a table outlining our scores for each of the reviewed frameworks:

Desired Characteristics	Database		
	MongoDB	DB2	Oracle
Database Size (scored out of 10):	10	10	10
Relevance (scored out of 10):	10	8	9
Tools/Functions (scored out of 10):	9	7	8
Community Support/Ease of Use (scored out of 10):	10	5	9

Figure 4: Table displaying the score for each desired characteristic for each alternative.

After viewing the different databases, we removed the characteristic of Flexibility, as many of the databases offered the same characteristics. Another common aspect we found among the databases, was that all of them would be able to support the size of the data for our project. Next, we viewed the Relevance and found that MongDB was at the top for this category, due to its widely used range in the world and also that our clients already work on this database. This is why we scored it a 10 (see Figure 4). Continuing, the tools and functions were viewed with MongoDB having a wide range of simpler features. Lastly, we are looking into the overall ease of use and support from the community. We found that due to the wide amount of usage of MongoDB that there are a lot of references and assistance in learning this database. With all this in mind MongoDB scored the highest for our desired characteristic out of all the databases.

3.4.6 Proving Feasibility

Our team will look into getting a MongoDB database constructed. If the database is able to hold adequate data without maxing out storage, then we will continue with MongoDB because of the ability to integrate it with current State Farm Technologies and due to the desired characteristics it contains.

4.0 Technology Integration

Each individual technology is not useful on their own until they are integrated into a single product. Our database will need to communicate heavily with our backend framework and our backend framework will need to utilize the features provided by our front end framework to create our final web application. This is not to mention that our GIS API will also need to function in tandem with our website's backend.

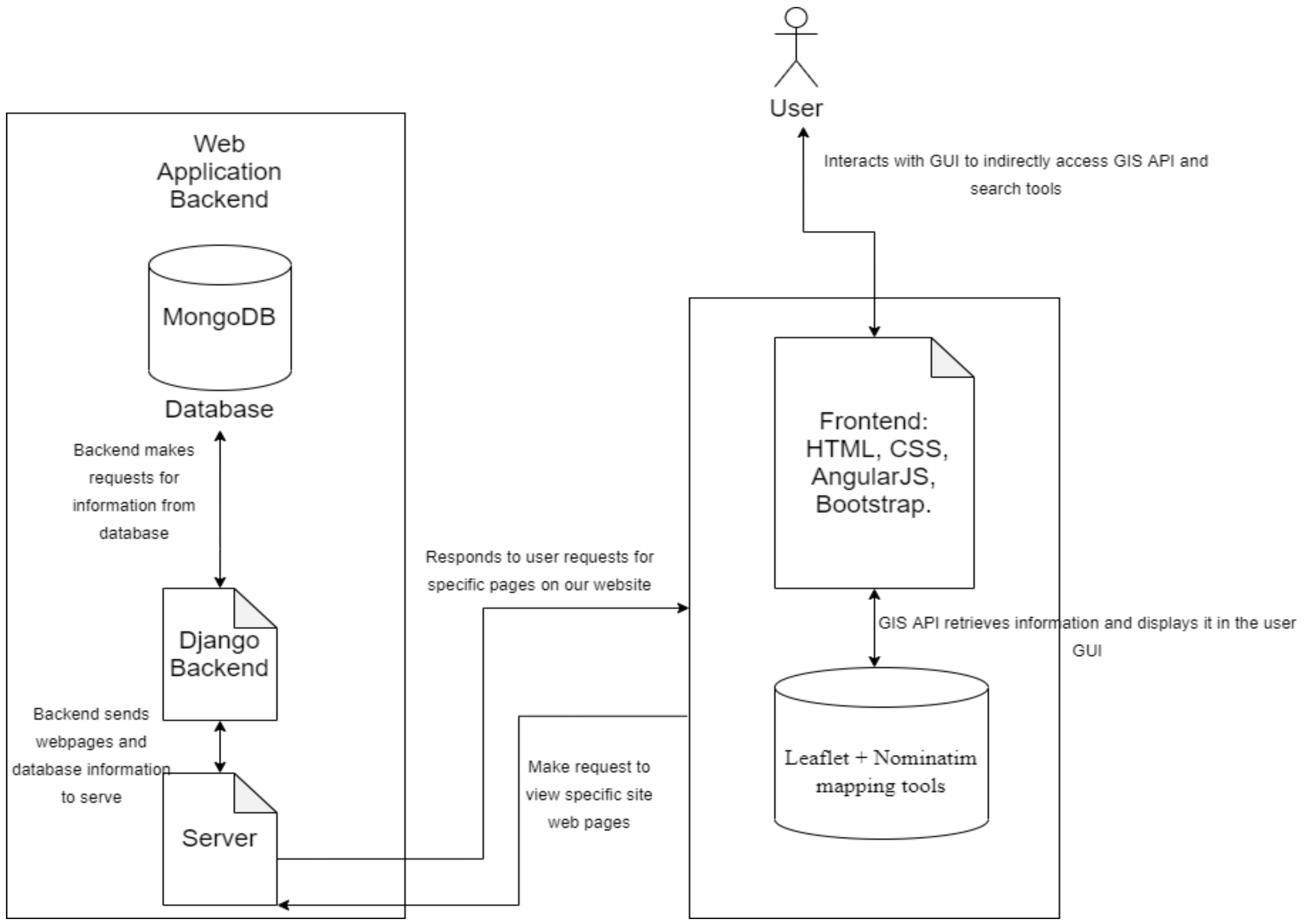
User
Interacts with GUI to indirectly access GIS API and search tools
Web Application Backend
MongoDB
Database
Backend makes requests for information from database
Django Backend
Backend sends webpages and database information to serve
Server
Responds to user requests for specific pages on our website
Make request to view specific site web pages
Frontend: HTML, CSS, AngularJS, Bootstrap.
GIS API retrieves information and displays it in the user GUI
Leaflet + Nominatim mapping tools

Figure 5: Table displaying the score for each desired characteristic for each alternative.

Our web application backend will consist of our server which should come bundled with the Django framework, a mongoDB database, and the Django framework. Our web application front end will consist of HTML, CSS, AngularJS, and Bootstrap as well as Leaflet and Nominatim. Each user will begin interacting with the web application through the front end. Users make requests to view specific web pages which the server intercepts and handles. The server will interact with the backend framework to supply the necessary webpages as well as any data needed to display the web page. If information from the database is needed to display the web page such as on a user login page, the backend framework will retrieve information from the database so that it is available in our web application front end. If the user interacts with our digital map, any requests to view specific addresses and locations will be made through the Leaflet and Nominatim javascript libraries in order to be displayed to the user.

5.0 Conclusion

As a team, we recognize that there are many challenges that we will face when implementing our project and we hope that by identifying and exploring solutions to those problems now, we will mitigate the time spent solving spontaneous challenges that may arise later. Our biggest concern when choosing the technology that will go into creating our project was how well they can work together. We are confident that we have chosen a strong set of technologies for our front-end, back-end, GIS system, and database system and look forward to prototyping and testing each aspect to make sure that they work as well together in practice as they do on paper. We hope to utilize these technologies in a way that allows us to build an effective web application that would not only be a large benefit to our client, State Farm, but also to organizations as a whole. By providing a smart, reliable, and highly interactive way to communicate more closely with their customers, our application can greatly improve the current methods of client contact and give our client and other organizations the tools they have been sorely lacking.

Our next step will be to set up a simple hello world Django website that is hosted on our own machines. From there our team will work on setting up AngularJS and Bootstrap so that it can be immediately used on our front end. Next, our team will create a mock database and practice pulling data from the database and using it on an example web page. Finally, we will implement the GIS mapping API on our practice website to show that we can make calls to the GIS server to plot a point on a map given an address.