Title: Technological Feasibility Analysis

Date: November 2nd, 2021

Olympic Developers

Peter Galvan, Camden Hortline,

Jason Gaglione, and Collin Rampata

Sponsor: Sean H. Anthony

Mentor: Melissa D. Rose

Overview: In this deliverable, the team explores a series of feasibility questions and provides viable solutions for overcoming these challenges through a technology analysis.

Table Of Contents

Pages

1. Introduction

Overview

The Olympics have been the pinnacle of professional athleticism for over a century. The games were designed to bring together athletes from across the world as a demonstration of national solidarity. In order to bring pride to their prospective countries, these athletes train as much as they can within the 4 year span in which the summer games are held. The goal of HYPO2 is to offer a fair, competitive edge to athletes who are training during off seasons. Athletes attend HYPO2 high-altitude training camps to increase endurance and improve overall health. The purpose is to train and support these athletes so they can excel in world-class competition.

Problem

While HYPO2 has been extremely successful, the current business model still faces a few challenges relating to client communication and scheduling. HYPO2 Staff communicates with their customers through individual text messages and emails which often requires constant, redundant work. This process for HYPO2 and their employees is more complex than it needs to be. For example, when a HYPO2 client wants to change a scheduled activity, they first ask HYPO2 if the change is possible, leading HYPO2 to cross reference data from all of the camps, facilities and services. If the change is approved, a master PDF must be manually updated for the client and communicated with HYPO2 staff through email. If the change is denied, there is a negotiation of other options and the process begins again. This could lead to using valuable camp time to plan rather than to train. The variability of training camps alongside the lack of a standard for automated communication has led HYPO2 to focus more on company operation rather than camp performance.

Solution Vision

The goal of this project is to improve on the existing workflow by creating a software system that allows employees to effectively communicate with clients, create training schedules, and receive notifications of changes in scheduling or client requests. HYPO2 envisions a communication system that benefits employees and clientele alike thus promoting training camp productivity. The proposed solution includes a web application that offers a client portal allowing coaches to access and change their schedules with ease. Alongside an admin portal that allows HYPO2 staff to access and update training camp information such as activities, costs and team schedules. Any use within these portals will be communicated automatically to respective recipients through templates already used by HYPO2. The Olympic Developer’s solution creates an opportunity for employees to better provide real time schedules and up to date information in reference to HYPO2’s offered services. Overall, there are too many steps in communication at every level of a training camp. With the team's envisioned solution, these tedious steps will be eliminated so the process is straightforward and easygoing for both clients and staff.

2. Technological Challenges

The objective of this project is to provide a powerful website to HYPO2 that will take away the need for manually documented spreadsheets and pdfs. The creation of a website that will take care of managing data and producing pdfs with requested data will ease the workflow of the staff at HYPO2. Staff is not the only important part as the website will also need to provide coaches with access to training camp schedules as well as the ability to communicate with staff for requests of service. As the objective has been determined by the team, this does not come free of issues in development.

The Olympic Developers team have identified four key technological challenges in development. The first challenge presented is how data from clients and staff will be stored. HYPO2 works extensively with client data as it drives the structure of team camps and what will be needed, so the use of a database will help with organization and the speed of finding needed information. The next challenge pertains to how data and other needs will be communicated between clients and HYPO2. A common system will be helpful to keep communication consistent and efficient. Next, a host will be required for the website. The server location is essential for any website and will ensure that the website will be available at all times of the day. The last challenge is how will the user interact with the website. A frontend framework will be needed to build a user interface that can be easily navigated. With each challenge presented, each member of the team will perform research to find the best solutions using a rating system that will be formulated by the member assigned to that task. Requirements for technological challenges will go more in depth in the technology analysis section.

3. Technology Analysis

3.1 Database

Introduction

The amount of moving parts in HYPO2's operation grows exponentially when considering the variability of elite level training camps and providing services to multiple teams concurrently. Keeping track of all of these parts and maintaining a real-time accurate schedule of services manually is a sizable task. The process currently is to cross-reference multiple google sheets and update each individual sheet manually and communicate the changes respectively. In this system there is no connection between sheets even though every teams' training camp has a direct impact on every other teams' schedules. HYPO2 is looking for a repository that mimics the real-world concept of information that is distinct but related.

A database management system(DBMS) is needed that allows for tables to be related based on team specific information and to hold team contact information, camp information and pricing. The DBMS will also store a master schedule, team schedules and facility schedules. The database needs to be able to create users as there is information specific to clients. The database needs to grant different levels of access, The client portal will need to access schedule tables and be able to input into client information tables. The admin portal will need to have access and the ability to input information into all tables. This database enables the solution to have a centralized repository for everything HYPO2 and their clientele. Thus streamlining the management of the moving parts HYPO2 encounters daily.

Candidates

1. MySQL

MySQL is a free to use open-source relational DBMS. MySQL is most commonly used for database driven web applications specifically for applications that use the database only for transactions. A relational database will allow for the data within the application to be related by specified keys. MySQL offers user creation support with extensive options for password timeout, attempts and lockout to ensure data security. MySQL offers levels of access with 34 different privileges including SELECT and INSERT ON privileges that can be combined to create the levels of permissions needed for each respective portal.

2. PostgreSQL

PostgreSQL is a free to use open source object-relational DBMS. PostgreSQL is best used when complex queries and data analysis are required. An object relational database is a relational database but uses an object-oriented model, this would allow for data to be related by keys in the application but in an object-oriented syntax. PostgreSQL offers support for user creation with options for password encryption and timeout. There are 13 different permissions

including SELECT and INSERT that can be used in any combination to produce each portal's level of access.

3. MongoDB

MongoDB is a document oriented NoSQL DBMS. MondoDB is best used when the data being held is unstructured and complex. A document oriented DBMS stores the data in JSONs rather than a traditional table of rows and columns. Relationships can be defined per document through creating reference documents or embedded documents. MongoDB supports both user creation and levels of access in the same process with 8 fields regarding access including FIND and INSERT as well as roles and password.

Rating System

1. Format

The team will be looking for DBMS that uses Tables with rows and columns. The data the team will be storing, processing and holding is structured and consistent. There is no need for the table to increase the amount of columns as every field is known before a camp begins. The DBMS will be rated either 0 or 10, 0 being a non table DBMS and 10 being a table DBMS.

2. Type of Database

In order to achieve the solution vision, the tables in the database need to be related. The data in the separate tables need to be made distinct by identifiers. Information about a team is not supposed to be accessed by any other teams. A relational database is needed to meet all of these requirements. The DBMS will be rated based on a scale of 0-10, 0 being non-relational DBMS, 10 being pure relation DBMS.

3. Querying from Multiple Tables

The database schema involves multiple tables as stated above and information from those tables need to be accessed at once. The team schedules and facility schedule tables need to be queried to determine open time slots available for changes to happen in a camp. The query needs to be able to cross-reference the tables instead of the staff member. The DBMS will be rated based on the complexity to query from multiple tables at once. 0 being the most complex and 10 being the easiest.

4. Familiarity

Working with a familiar DBMS will be an effective way for the team to not only save time in reducing the learning curve but allow the team to focus on the new aspects of the web app. The DBMS will be rated on a scale of 0-10, 0 being the most foreign syntax and operations and 10 being the most familiar syntax and operations.

Discussion

MySQL, PostgreSQL and MongoDB all have their own benefits but each have their own drawbacks as well. For the format of how each DBMS stores its data both MySQL and PostgreSQLuse tables earning them a rating of 10. MongoDB however stores data as documents earning a rating of 0 for this section. Both MySQL and PostgreSQL use a similar create command that defines the table's attributes. The biggest difference being PostgreSQLis it offers an inheritance field which defines how the table fits into the hierarchy based schema.

Looking into the type of each one of these databases. MongoDB is a document-oriented NoSQL database, NoSQL does not mean a non relational database, it supports relationships but not necessarily the same concept of relationships as a relational database management system(RDBMS). Instead of relating tables to one another through unique identifying columns, documents can be embedded into other documents to create a relationship. MongoDB supports every type of relationship between documents but the document-oriented aspect and the unique concept earn it a 6. Both MySQL and PostgreSQL are forms of RDBMS. MySQL is a pure relational database, meaning it offers table relationships through primary and foreign keys. Whereas PostgreSQL is an Object-relational database management system meaning it offers all that MySQL does pertaining to table relationships but also includes features like table inheritance. Features like this can be used extremely well under the right circumstances but for the purpose of this web app the relationships between tables is horizontal and not hierarchical. Thus earning MySQL a 10 and PostgreSQL a 9.

The ability to query from multiple tables or documents simultaneously is expected in modern day DBMS but the inherent features that come with querying data and ease of use is the deciding factor. In MongoDB for example, the $in operator is used in conjunction with the query standard of “db.DBNAME.find()” to pinpoint specific attributes that documents share and pull data from those documents. Instead of creating a primary or foreign key used in a RDBMS there would be a specific field in each JSON document that can be used as an identifier. This results in an unnecessary amount of steps and earns mongoDB a 7. Both MySQL and PostgreSQL use the same syntax to query from multiple tables, “SELECT attributes, FROM tables, WHERE condition”. The biggest difference being how to reference a table. In MySQL you use the table name whereas in postgreSQL you use the table type and its name. Considering both MySQL and PostgreSQL query using the same syntax and the only difference is a minor inconvenience, both earn a rating of 9.

Familiarity is going to be the biggest factor in deciding which DBMS as each one has been proven to work in a variety of use cases. Familiarity with the DBMS is going to allow the team a better opportunity to progress through issues if the system has been seen before. MySQL is the most familiar DBMS of the candidates, as all members have experience with creating databases using MySQL. The experience is not extensive and may lead to unforeseen issues but is a foundational understanding of MySQL implementation. MySQL earns a score of 8. MongoDB is the least familiar DBMS as it is an entirely new type of database and the syntax is unfamiliar to all members, thus earning MongoDB a rating of 4. PostgreSQL’s syntax is closer to MySQL as it follows the same concept of format but including object-oriented aspect is going to

lead to issues as no member has experience with an Object-Oriented DBMS. PostgreSQL earns a rating of 6.

Looking at each candidate through the lens of the rating scale. It is clear that MongoDB was the worst of the candidates as the optimal use-case, format and type were nearly the opposite of the solution vision requirements. Resulting in an average score of 4.25. Between MySQL and PostgreSQL it was very close, 9.25 and 8.5 averages respectively. The key distinguishing factors being familiarity and PostgreSQL's overall sophistication.

Table 3.1 (Rating For Database Candidates)

	Format	Type	Query Multiple Tables	Familiarity	Total
MySQL	**10**	**10**	**9**	**8**	**9.25**
PostgreSQL	**10**	**9**	**9**	**6**	**8.5**
MongoDB	**0**	**6**	**7**	**4**	**4.25**

Final Decision

Overall the database needed for the application is a simpler one that will be used for transactional processes. MySQL with the highest average rating will be the DBMS for the web application, as it covers format, type, querying and familiarity needed for the solution vision and is the best fit for HYPO2's use-case. The next step is to build a demo database schema and test inputting, querying and updating data and then to implement the schema into the communication API and the frontend-framework.

3.2 Server Location

Introduction

This problem is a crucial part of the web portal that will be created for the HYPO2 sports training center. This is where the portal will be operating from. The user will be able to come to the location and use the services the team will be creating for the program. The web server is the hosting location for services which will essentially be looking at the current schedule as well as allowing for modifications and cancellations of services. Cost, reliability, and Familiarity are the key proponents that will be decided on when coming to a conclusion on what will be most beneficial to solving this problem.

Candidates

1. AWS (Amazon Web Services)

AWS is a widely-used cloud platform provider. AWS tailors to emerging technologies and companies so that they can use the program in a way that allows them to create, explore, and share what they want on a large scale. The web service is described as being flexible and focused on the security of its consumers. With 52% market share, AWS is the highest shareholder for all cloud computing services. What comes with this is a highly reputable service that gains familiarity, not only with the Olympic Developer's team, but with the majority of people that use cloud computing as well. AWS also offers a one-on-one customer service support model that matches experienced engineers with users. Although, in order to access this, it comes at a "Premium Support" price that depends on usage. For a general purpose type of service, AWS's pricing model lays out $0.154 an hour.

2. Microsoft Azure

Azure is similar in the way that it offers a widespread platform for users to operate from. The Azure system likes to focus on the idea of "choice" and options for its consumers. Azure supports open-source frameworks and is feasible in all languages. Currently, Azure contains 21% of the cloud computing market share value. The team is less familiar with this service which might make it harder to find coinciding resources. After research, it was discovered that Azure is somewhat known for low-quality customer support. When looking at the general purpose pricing, Azure markets at $0.166 an hour.

3. Google Cloud

Google Cloud mentions its promise to avoid vendor lock-ins. This cloud service also offers multi- and hybrid-cloud opportunities for its customers. One of the important aspects of the Google Cloud is the fact that it puts a priority on management control easy-to-navigate processes. Although the market share of Google Cloud is 18%, based on research, the Cloud has the most detailed documentation on their product. For customer service, Google Cloud offers a

standard support plan that costs $29 a month. The Cloud's pricing model for general purpose is $0.156 an hour.

Rating System

1. Cost

The team will be looking, not for the cheapest available solution, but the best solution for the warranted dollar amount to keep the site up and running.

2. Reliability

The team knows that a good website is one that is up for as much time as possible. Reliability is not just about how often the site stays up, but also includes the support that the web service team provides in times of incidence or issue (customer support).

3. Familiarity

Familiarity is very valued when picking a web service provider. The team will go through times when there is a problem within the web service. The more familiarity by the team, client and users using the web services will prevent these issues. If the team runs into an issue, if there is overall familiarity by the users this will result in quicker fixes by researching for a solution to the problem at hand.

Discussion

Amazon Web Services, Microsoft Azure and Google Cloud are very comparable to each other. When it comes to the pricing models of each service based on a general purpose plan AWS is ahead with the cheapest cost per hour at $0.154. Behind that, Google Cloud costs $0.156 an hour. Finally, Microsoft Azure costs an hourly rate of $0.166. They all offer a 12-month free trial and the billing models are competitive with one another. AWS and Google Cloud offer better up-time reliability in comparison to Microsoft Azure. According to online resources, Microsoft Azure experiences a more significant amount of outages (Microsoft Azure: 1934 hrs). In reference to this, Amazon Web Services did manage to slightly overcome Google Cloud in minimal outages per hour (AWS: 338 hrs/Google Cloud: 361 hrs). While it is not a huge difference in numbers, it is one worth taking into account when deciding which option is the best to go with. Also, most customer services provided by each cloud service company are marketed at a premium price. For Google Cloud and Azure, the prices are listed out and for AWS they are less up front and the price is not given until you know the usage of your product. Overall, the group and client has more historical reference and prior experience working with AWS. As well as AWS having the majority of cloud service market share of 52%. This gives an understanding that there will be more information on AWS from users than other cloud services. With such an important task, comfortability with the cloud-based service should also be named as an important factor for use.

Table 3.2 (Rating For Server Location Candidates)

	Cost	Reliability	Familiarity	Totals
AWS	10	9	10	9.66
Microsoft Azure	8	7	7	7.33
Google Cloud	9	8	8	8.33

Final Decision

With all of these factors taken into consideration, the group has decided to use Amazon Web Services as the main cloud service. AWS offers the group comfortability, reliability and a proper program environment for the project at hand. The group has made the decision that AWS is the best fit for both the group members and client. The next step for the group will be to set up a free-trial account with Amazon Web Services. The team will then input the code into the web-based application and begin configuring the application in order to tailor it to what the team needs.

3.3 Communications

Introduction:

When managing Olympic athletes, there is a multitude of digital paperwork that must be completed to ensure that they are receiving the proper training. This includes intake forms, health records, schedules, and lifestyle plans. In HYPO2's case, these forms must all be filled out manually, and relayed back to the customer through email or text message. This process is cumbersome, and repetitive as it often requires lots of the same information to be inputted. Furthermore, information about the customer is spread out across multiple platforms making querying data difficult. A process such as this could benefit greatly from a centralized web portal that could send and receive messages.

Communication between the client and his customers is a significant part of HYPO2's business . The team finds it especially important to choose a viable tool that will allow the client to carry out this process within the web portal. Fortunately, there are a wide variety of application programming interfaces (API's) available that simplify this process significantly. These technologies will allow us to build pre-existing functionality into the application - which will help us solve the problem. The team needs an API that will allow the web portal to take previously inputted data and compose pre-configured text messages or emails. HYPO2's administration can send these pre-templated messages to customers which will increase work efficiency. Moreover, the programming interface needs to be able to parse customer messages for useful information. Data such as the customer's name, association, and budget can be stored and utilized to aid in message customization. Providing unique, personalized automated messages will ensure that HYPO2 maintains a healthy relationship with its customers while still avoiding repetitive tasks.

Candidates

1. Twilio

Twilio is an expansive web service that offers a variety of different APIs and software design kits (SDKs). These libraries allow businesses to integrate Short Message Service (SMS) and Email capabilities into their existing platforms in an easy-to-use format. The advantage to using a service like Twilio is that it is specifically tailored towards developers, so there is comprehensive documentation for each of the API's the service offers. The API's offered by Twilio are free to implement, however they are subject to the company's Messaging Policy if messages are sent over their chat channels. Some of the services this company provides that could be used in the web portal include:

- **Twilio Lookup API -** Twilio offers multiple levels of security, including this convenient API that can validate phone numbers, detect line type, and prevent fraud. The team would use this API for user verification when the web portal receives new contact information.

Having an option to increase security is important especially when HYPO2's business deals so closely with sensitive health records.

- **Twilio SMS API** - This programming interface allows businesses to send and receive secure SMS text messages that can be structured to match a specific template. Designing a feature that could send automated text messages and emails would greatly reduce the time needed for the client to communicate with his customers. For instance, sending out a team schedule could be done automatically once the team has arrived at the camp. Another implementation could include sending out a running cost estimate through SMS messages as the customer spends more money at the training camp.
- **Twilio Message API -** This feature can work in tandem with several other API's, including the previous one. It allows for account owners to query and manage metadata from incoming text messages. This powerful tool could be utilized by HYPO2's web portal to extract customer information such as their name, association, budget, and a variety of other useful data. Furthermore, the team can load the parsed data into the web portal's database should it need to be searched at a later date.

2. Telesign

Telesign is a service that is used by leading online service providers to ensure secure account creation and user interaction. The company prides itself in offering one of the safest APIs for peer-to-peer interaction. There are multiple levels of verification that can be implemented, and messages sent over the company's channels are safe and secure. Users can choose between free and paid plans depending on the expected call volume. Documentation for the API can be purchased for a small fee on Telesigns website. They offer a wide variety of developer tools that include:

- **Digital Identity API** - This is the service's unique selling point. This API allows businesses to conduct a risk assessment for incoming phone numbers in an effort to detect fraudulent activity. Simply with a phone number, this interface can convey information like the phone carrier, device type, risk score, and "last used" date. This would be especially helpful in protecting HYPO2's web portal from scammers. The client would receive notifications directly to his phone that would help him decide whether he wants to continue business with the new contact. Messages that are not meeting a certain risk score can be filtered out to avoid wasting the clients time.
- **Messaging API -** This messaging API would allow the client to send and receive messages over SMS, Rich Communication Services (RCS), and WhatsApp. Messages can be designed to match a pre-configured template and can be scheduled to send at specific times. HYPO2's web management portal can utilize this programming interface to send out automated messages too. Once a team is registered with HYPO2 they can start receiving scheduled messages depending on their date of arrival. This could include training plans, travel accommodations, and cost estimates - which could all be assembled into preconfigured templates.

3. Mailchimp Developer

Mailchimp Developer is a service provided by Mailchimp, an all-purpose web development company that allows its customers to create websites and market existing businesses over the internet through their marketing platform. Mailchimp Developer is meant specifically for developers who are looking to implement a MailChimp API onto a pre-existing website. They supply their customers with an API that could prove beneficial in the teams solution:

- **MailChimp Marketing API** - This application programming interface is designed specifically for pre-templated automated emails. These can be sent out on a scheduled basis and can be customized to fit a variety of needs. In HYPO2's instance, MailChimps Marketing API can be used to send information such as intake forms, weekly schedules, and other messages that need to be sent automatically or on a recurring basis.

4. Amazon Web Services

Amazon Web Services supplies some of the most reliable products in the industry, and many of the world's leading businesses utilize them. One of their products, Amazon Simple Notification Service, supports SMS notifications to over 200 countries. It supports peer-to-peer, and application-to-peer messaging over a secure and encrypted network. All of this can be achieved using the Amazon SNS API that is provided on the company's website. This interface comes equipped with a wide variety of functionality, like verifying phone numbers, sending automated messages, and managing subscribers. Moreover, the API is designed with developers in mind, making its implementation simple. There is a vast amount of documentation on the official website that outlines how to properly integrate Amazon SNS into an existing codebase. It comes with many pricing options as well as free versions to test a proof of concept. This would be a highly beneficial API to use with the client's web management portal since it covers many of the issues HYPO2 is facing. This interface would allow us to send out automated messages that contain the initial intake forms for HYPO2, as well as the following messages that contain the price estimates and payment information.

Rating System

1. Message Customization

Message customization is vital to ensuring that HYPO2's transition to an automated messaging system does not interfere with the current customer experience. The team needs an API that supports multiple modes of communication, for example, SMS and Email. Also, the programming interface must support the creation of message templates so that HYPO2 can send its digital paperwork to its customers. Furthermore, the API must also support message scheduling so that date-specific notifications can be sent to update users on current events like team schedules and travel accommodations. An ideal API that would carry all three of these tasks out would be considered a 10. For every feature the API can not support, three is subtracted from the total.

2. User Verification

The client should feel confident that the customer he is contacting is an interested buyer and not a scammer. This is why the application programming language the team selects should support user verification to decrease the threat of frauds and scams. Also, The conversations between HYPO2 and its clients often contain sensitive health information so it is absolutely crucial that a secure connection between the two parties is established. This means that the communication channels are held on secure, encrypted servers from a reputable company - ensuring confidentiality between HYPO2 and its customers. A perfectly rated API (10/10) will include functionality for verifying phone numbers and emails, and be hosted on a secure server owned by a reputable organization. For every feature the API fails to support, five points are subtracted.

3. Documentation

It is important to pick an API that has complete documentation on its functionalities. This will ensure that features like message scheduling and user verification are implemented correctly into the web portal. Having an API that is easy to understand from a developers perspective will increase readability of the code and help the team develop a more robust solution. The client stressed the importance of maintaining a positive customer experience and developing a concrete solution is the main way of accomplishing this task. An API that includes free, complete documentation receives a perfect 10 in the documentation category. An API that offers complete but costly documentation will receive a five. Lastly, an API containing neither of these components will receive a zero.

Discussion

Figure 1.0

```
2
3   import com.twilio.Twilio;
4   import com.twilio.rest.api.v2010.account.Message;
5   import com.twilio.type.PhoneNumber;
6
7   public class Example {
8       // Find your Account SID and Auth Token at twilio.com/console
9       // and set the environment variables. See http://twil.io/secure
10      public static final String ACCOUNT_SID = System.getenv("TWILIO_ACC
11      public static final String AUTH_TOKEN = System.getenv("TWILIO_AUTH
12
13      public static void main(String[] args) {
14          Twilio.init(ACCOUNT_SID, AUTH_TOKEN);
15          Message message = Message.creator(
16                  new com.twilio.type.PhoneNumber("+15558675310"),
17                  new com.twilio.type.PhoneNumber("+15017122661"),
18                  "Hi there")
19              .create();
20
21          System.out.println(message.getSid());
22      }
23  }
```

The analysis of the communication problem began by exploring how the Twilio SMS API would be implemented. Figure 1.0 demonstrates this process in Java. The contents of the string “Hi there” will be sent to the recipient - but this can be altered to fit any pre-templated message. Additional numbers can be added using the “new com.twilio.type.PhoneNumber()” attributes. Of course these can be extended upon with the implementation of additional APIs offered by Twilio. It was concluded that the functionality offered by this company's products was outstandingly high. This service offers programming interfaces that support peer-to-peer and peer-to-application email and SMS communication. There is functionality for composing pre-templated messages, verifying phone numbers, and scheduling high-volume SMS messages. Since Twilio has support for the three key features highlighted in the rating system section, it scored a perfect 10 in the “Message Customization” category. On top of all this, Twilio comes with an extensive amount of free online documentation to explain each application programming interface. As a result, Twilio scores a perfect 10 in the “Documentation” category. Twilio also scores a perfect score in the “User Verification” category because it offers ways to validate new users and the company encrypts messages sent across its chat channels. However, Twilio may not integrate well with other aspects of the application. The web portal of customers will be hosted on AWS which means that all the information about its users will be in the Amazon Cloud. Twilio would store user data using a separate cloud service. Although this does not

impact Twilio's rating, trying to translate the user information between the two services could prove to be a nontrivial task so it is important to consider.

Further research was also conducted on the services offered by Telesign, which promised to offer exceptionally secure peer-to-peer and application-to-peer messaging. Security is at the pinnacle of this company's business model, and it is evident through the functionality of their API. There are multiple levels of verification that developers can choose to implement including Telesign's Digital Identity API which allows the company to score a perfect 10 in the "User Verification" category . Telesign's Messaging API also included functionality that could be used to send scheduled, pretemplated messages. However, this application interface does not include any support for Email messages, so it lost 3.33 points in the "Message Customization" category. Free documentation is included on Telesigns website for each API they offer, so full points were awarded in the "Documentation" category. This service also suffers from the same issue as the previous candidate, Twilio, in that it would be difficult to integrate with other parts of the web portal. Having all of the information under one service will be increasingly important during implementation when the team needs to write the code for these features. Having a separate database for user phone numbers specifically for sending SMS messages when all of that data is already in the AWS platform would simply not make sense from a practical standpoint.

Figure 2.0

```
import mailchimp_marketing as MailchimpMarketing
from mailchimp_marketing.api_client import ApiClientError

mailchimp = MailchimpMarketing.Client()
mailchimp.set_config({
  "api_key": "YOUR_API_KEY",
  "server": "YOUR_SERVER_PREFIX"
})

body = {
  "permission_reminder": "You signed up for updates on our website",
  "email_type_option": False,
  "campaign_defaults": {
    "from_name": "Mailchimp",
    "from_email": "freddie@mailchimp.com",
    "subject": "Python Developers",
    "language": "EN_US"
  },
  "name": "JS Developers Meetup",
  "contact": {
    "company": "Mailchimp",
    "address1": "675 Ponce de Leon Ave NE",
    "address2": "Suite 5000",
    "city": "Atlanta",
    "state": "GA",
    "zip": "30308",
    "country": "US"
  }
}
```

Mailchimp Developer offered a highly functional email messaging service designed to automate marketing content and receive payments. Above in Figure 2.0 is a typical Python

implementation of a mailing list using the MailChimp Marketing API. The “set_config()” function looks to be a powerful method that can utilize string inputs to update a user database. Designing the API in this way keeps the code highly readable which will be important in development. The API proved to be very user friendly, and there was expansive amounts of documentation offered on MailChimp’s Official Github that could help with its implementation. As a result, MailChimp Developer scored perfectly in the “Documentation” category. However, this service's API lacked when it came to message customization. Unfortunately, the API provided by MailChimp only supports email messages, and does not include SMS support. Confining ourselves to one method of communication this early in development could prove to be an obstacle during implementation should requirements change. Certain scenarios may also prohibit the use of email, so having another means of communication is important to ensure that HYPO2 does not lose any clientele. Since some of the main functionality the team needs for the web portal is not supported by MailChimp’s Marketing API, only 6.66 points were awarded in the “Message Customization” category. The company does specify that all communications that take place within their servers are encrypted, and fully secure. Also, user verification is handled automatically by MailChimp, so full points were awarded in the “User Verification” category.

The application programming interfaces offered by Amazon Web Services were unparalleled compared to the other products that had been researched thus far. There were several solutions that could be implemented with the API provided by this company , including Amazon SNS, which supports SMS and Email messaging and scheduled SMS messages. Furthermore, AWS offers a “message delivery retry policy”, which checks to see if a message was sent successfully and resends it if it failed. Having a robust system to onboard clients is vital if HYPO2 wants to capture and retain customers. Thus AWS earns a 10/10 in the “Message Customization” category. Every function included in the API is well-documented on the AWS website, as well as potential bugs that could be encountered. All of this information is free to the developer as well. Having a thorough set of instructions will prove to be highly beneficial when it comes time to implement Amazon SNS. This is how this service scored perfectly in the “Documentation category”. AWS also excels in security, offering end-to-end encryption over several strategically placed servers. Amazon SNS has the ability to utilize the Amazon Virtual Private Cloud, which gives users the ability to send SMS messages without traversing the public internet. Being the monolith that is Amazon, security is arguably guaranteed, and there are a variety of other features like phone number verification that can also be implemented. Thus, Amazon SNS also scored a perfect score in the “User Verification” category. Lastly, the research concluded that Amazon SNS would integrate well with other aspects of the web application. AWS is already being utilized to host HYPO2’s web portal. By using an interface that is built by the same company, it ensures compatibility between the two systems. This will also assist in data aggregation since all of the user data will be on a single platform.

Table 3.3 (Rating For Communications Candidates)

	Message Customization	User Verification	Documentation	Totals
Twilio	**10**	**10**	**10**	**10**
Telesign	**6.66**	**10**	**5**	**7.22**
MailChimp	**6.66**	**5**	**10**	**7.22**
Amazon Web Services	**10**	**10**	**10**	**10**

Final Decision

After conducting extensive research into the most appropriate service to utilize it is clear that AWS is the highest performing candidate and the best option to implement moving forward. It offers a plethora of programming interfaces like Amazon SNS that will allow us to solve the communication issues.Furthermore, AWS also offers the safest option since conversations are carried out over its encrypted servers. Amazon also offers the most extensive documentation for its APIs. The analysis proved why the application programming interfaces were all superior to its competitors. Now that Amazon SNS is the preferred API for the application, it can be tested by sending and receiving text messages through Amazon's Web Hub using functionality provided by the interface. A series of phone numbers can be supplied to the functions which will then be used to establish a secure line of communication between the application and the peer. Moreover, the team can supply the functions with faulty inputs like fake phone numbers to test the secureness of Amazon SNS. Then, the process of successfully extracting user data from text and email messages can begin.

3.4 Frontend Framework

Introduction

As this project will be a web application, having an efficient front-end framework is essential for a fast and interactive web application. Having a time efficient user interface will allow the user to navigate their needs without unnecessary steps. HYPO2 lacks a user-friendly interface as their primary way of operation is through Excel sheets. Implementing an efficient user interface will help HYPO2 with work efficiency by cutting down time to complete tasks and locating needed information.

Candidates

1. React

React is an open-source framework developed by Facebook and uses Javascript as its programming language. React allows the ability to create an interactive user interface while keeping the learning curve low. React uses the virtual direct object model, which improves the speed of a website by updating only those elements that need to be updated. By using React to build a single page application, it allows for seamless page-viewing as pages are updated to the user's needs rather than changing pages and the information displayed. This is necessary for athletes as time is important and must be used efficiently. Along with pertaining to user needs, a framework that is easy to learn is desired to ensure that learning will not be longer than developing.

2. VueJS

VueJS is a framework that utilizes javascript to build web interfaces and single-page applications. Vue.js utilizes virtual document object model (DOM) to allow for updating the elements that need to be updated rather than updating all elements, resulting in faster rendering of a web page. The learning curve is not large and has detailed documentation as it has an active community to maintain functionality. Vue.js would benefit the most in the sense that it is a lightweight framework that is the easiest to learn. As this project is not large in the sense that it will rarely exceed hundreds of users, Vue is a great choice because of its expandability on products that may seem complete. In the early stages of this project, a heavy-weight framework may not be needed as there will not be a need for many frontend layers.

3. Angular

Angular is an open-source framework developed by Google and uses typescript for its applications. Angular is good for building single-page applications, which allows data to be updated without refreshing the whole page. Two-way data binding is also a feature of the framework, which reflects updated data in real-time. With a large community, Angular is well documented, allowing issues to be resolved quicker than other frameworks. While the framework

allows many features, it does have a steeper learning curve due to its heavy-duty nature. Angular provides a powerful, heavy-duty framework that gives the widest selection of tools and resources. The templates that come with Angular give a jumpstart in developing the web application as they can be modified to the team's needs. Two-way data binding also gives the web application the opportunity to update the pricing of provided services to the user.

Rating System

1. Functionality

Functionality is important to ensure that all moving parts in a website work properly and work with each other. The framework must provide the tools to allow communication between frontend and backend as data will need to be displayed to the user. The framework should also allow authorized users to change data and update the displayed data.

2. Ease of use

Ease of use is needed for the development of the web application as more time should be spent developing rather than learning. Having detailed documentation will be necessary in providing answers to any issue that will come up during development. This will provide the best chance of delivering a great product on time. The framework will also need to use javascript as the team is most familiar with the language.

3. Performance

Performance is essential in getting the user from point A to point B as fast and seamless as possible. The main user audience will be athletes, coaches, and admins. These users are constantly working and need to be sure that everything they do is in a timely manner. The support for virtual document object model (DOM) will be beneficial to performance.

Discussion

When researching each framework, it was important to look at what each was capable of and what their strong points were, although it was helpful, it was also important to test each framework to get a better understanding of the three frameworks. To test each framework, base websites were created to implement basic elements such as images, links, and text boxes. These items helped identify issues and advantages with the three frameworks. While testing each framework, VueJS excelled in the ease of use category as it allowed for a traditional separation of HTML, CSS, and javascript files. However, VueJS also supports JSX just like React, which uses HTML and CSS inside of javascript. VueJS gives flexibility to the programmer on the way they structure code, giving it the edge over React. Angular lacked in the ease of use category because of its many needed components. Angular is a heavy-duty framework and has more moving parts to understand than VueJS and React.

When looking at functionality, Angular ranks higher than React and VueJS. Both React and VueJS may require third-party libraries in further development, but Angular does not need any additions and comes with everything that will be needed. Angular serves as a complete framework, which provides the highest functionality as React and VueJS are not so completely out of the box. However, with React and VueJS, there is a wide variety of tools to import that increase the functionality of each framework.

Both React and VueJS are close when comparing performance as they both utilize virtual DOM and single-page applications. VueJS is very lightweight and when factoring in its support for virtual DOM and single-page applications, performance excels. React performs just as well and is more fitting for larger projects. Comparing React and Angular for larger applications, React has higher performance as Angular does not use virtual DOM and carries more components to work around.

Table 3.4 (Rating For Frontend Framework Candidates)

	Functionality	Ease of Use	Performance	Totals
React	**9**	**9**	**10**	**9.6**
VueJS	**8**	**10**	**9**	**9**
Angular	**10**	**8**	**8**	**8.6**

Final Decision

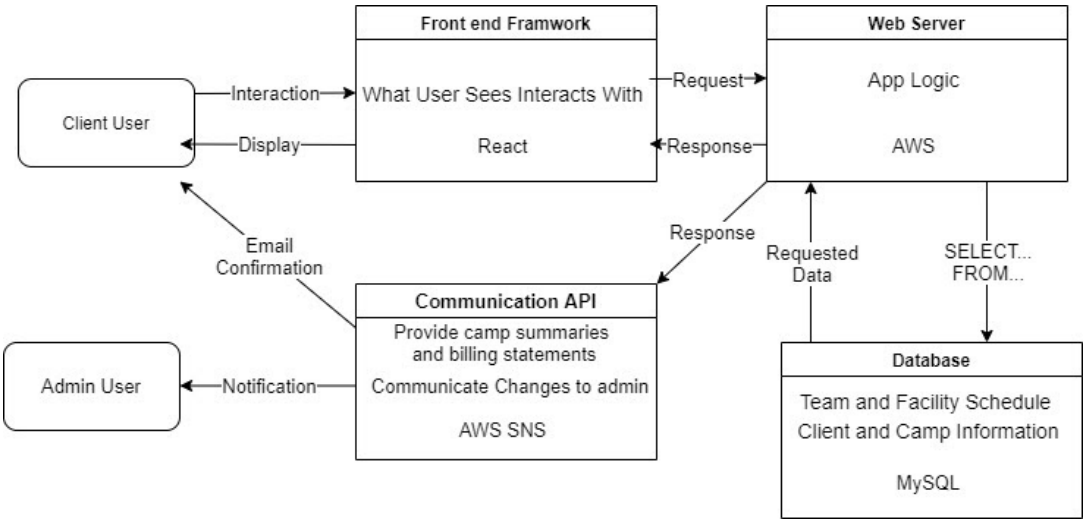
React will benefit this project the most because of its overall edge over Angular and VueJS. While VueJS came close, React did a lot of the same things as VueJS and then some. VueJS is a lightweight framework that would grant the fastest development time but is more fitting for smaller projects. Angular may have provided better functionality, it comes at a cost of performance and ease of use. React offers the best balance of the three categories comparable to Angular and VueJS as it has external tools that can provide functionality like Angular while also keeping performance and ease of use similar to VueJS.

The next steps in testing, the team will use React to create a single page web application to simulate real time workflow. The website will need a login screen and a screen that displays the information about the user and other needed items. From there, a calendar needs to be available to view their schedule and also make modifications. This will cover some of the immediate needs to the user and prove feasibility by showing functionality and the data needed.

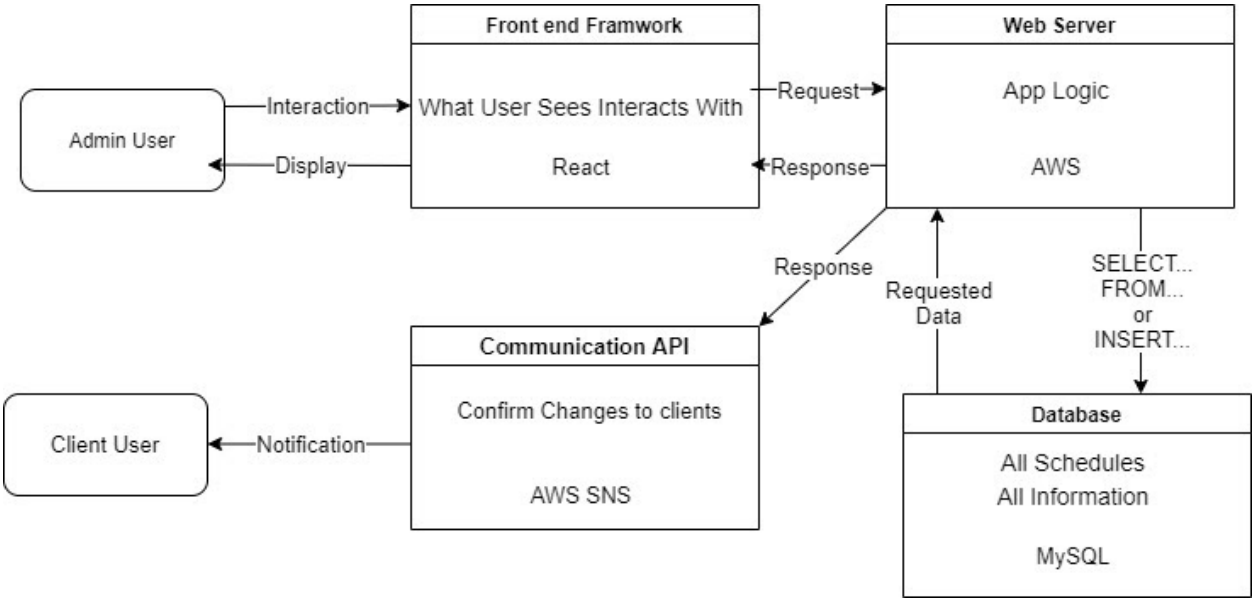
4. Technology Integration

In order to provide the client with a usable application that solves HYPO2's problems, each one of these technologies must work in tandem with each other. The application will be hosted on an AWS server to offer reliability in order to provide continuous client and staff access. A front-end framework, made with React, will be used to construct the user interface for the user and staff portal. This will provide clients the ability to interact or view their schedule as well as make service requests. The admin user interface provides a contact point between camp information alongside the master schedule to the HYPO2 staff in a time efficient manner. These portals will connect to the backend database using MySQL. All service and facility schedules will be stored in this database. The master schedule will be a table that represents the schedules of all teams to provide the application a repository that can be used to find available time slots based on current availability. The master schedule will be updated in conjunction with changes made to individual team schedules. Price estimates for specific services and facilities will be stored in this database to provide users with an initial quote for their requested camp schedules . Changes made throughout a camp will directly affect the running cost in real-time that can be used to offer transparency to HYPO2 clientele. Any and all changes an admin or a user makes will be communicated through the communication API. The communication API will pull specific data from the backend database and input the information into a respective template, representing daily schedules, billing statements or camp overviews, that will then be sent in an email or SMS.

Client Portal



Admin Portal



5. Conclusion

The team started by conceptualizing four main obstacles that could be encountered when creating a web management portal for HYPO2. The four topics that were included in the technological analysis were: Database, Server Location, Communications and Frontend Framework. A thorough study of each topic was conducted to find solutions to the problems that were found. The result of the first study concluded that the best database to choose for HYPO2's web management portal is MySQL since it is a relational database with an optimal format that is familiar with the team. The second study found that the best web host for HYPO2's management portal is AWS. This candidate would provide the team with reliable web servers that can be interfaced through a familiar platform. The results of the third study proved that Amazon SNS by AWS was the ideal API for handling customer communications with the management portal. This application programming interface was chosen because of its customizability, security, and ease-of-use. The fourth study concluded that the front end user interface is best designed with the React Framework. React was chosen because it utilizes virtual DOM and single page applications. It is also lightweight, and well-suited for large-scale projects. Looking forward, the team aims to draw on the knowledge found in the technological feasibility analysis to deliver a viable software solution. Now that the team has proposed solutions to the technological challenges, further work can be conducted to see how each area of the platform integrates with the other. A deeper understanding of the functional and nonfunctional requirements for HYPO2's management portal will be developed. Overall, the team has a concrete understanding of the project requirements as well as a decent understanding of the technologies required for the web portal. There is still much work to be done but everyday the team moves closer to devising a working solution. The web management portal that is being designed for HYPO2 will revolutionize the way the client operates his business.

6. References

"Amazon Simple Notification Service" *AWS*, AWS

https://aws.amazon.com/sns/?whats-new-cards.sort-by=item.additionalFields.postDateTime&whats-new-cards.sort-order=desc

"Angular Introduction: What It Is, and Why You Should Use It", *Sitepoint*, https://www.sitepoint.com/angular-introduction/

"AWS Azure and Google: What's the Cloud Security Difference?" *The State of Security*, 19 Jan. 2021, https://www.tripwire.com/state-of-security/security-data-protection/cloud/aws-azure-google-difference-cloud-security-standpoint/

"Comparing the Cloud Giants: Uptime and Reliability." *Linkeit*, https://www.linkeit.com/blog/comparing-the-gigants-of-cloud-uptime-and-reliability#:~:text=Of the three technology giants,AWS or Google Cloud Platform.&text=Of the two that recorded,registered by Google Cloud Platform

"Connectors and APIs." *MySQL*, https://dev.mysql.com/doc/index-connectors.html

"Communication APIs for SMS, Voice, Video and Authentication." *Twilio*, https://www.twilio.com/.

"Create a Single Page Application with React and React Router", *Split*, https://www.split.io/blog/react-router-feature-flags/

Mailchimp Developer, https://mailchimp.com/developer/

"React Features", *JavaTPointI*, https://www.javatpoint.com/react-features

Santis, Peter. "Ultimate Cloud Pricing Comparison: AWS vs. Azure vs. Google Cloud in 2021." *CAST AI*, 10 Sept. 2021, https://cast.ai/blog/ultimate-cloud-pricing-comparison-aws-vs-azure-vs-google-cloud-in-2021/

"TeleSign CPaaS, SMS & Phone Number Verification APIs." *TeleSign*, https://www.telesign.com/.

"The Good and the Bad of Vue.js Framework Programming." *altexsoft*, https://www.altexsoft.com/blog/engineering/pros-and-cons-of-vue-js/

"What is Angular?: Architecture, Features, and Advantages", *Simplilearn*, https://www.simplilearn.com/tutorials/angular-tutorial/what-is-angular

"What Is AWS." *Amazon*, The Board, 1917, https://aws.amazon.com/what-is-aws/

"What Is Azure-Microsoft Cloud Services: Microsoft Azure." *What Is Azure-Microsoft Cloud Services | Microsoft Azure*, https://azure.microsoft.com/en-us/overview/what-is-azure/

"Why Google Cloud." *Google*, Google, https://cloud.google.com/why-google-cloud

"Why use MongoDB and when to use it?", mongodb, mongodb.com. https://www.mongodb.com/why-use-mongodb

"Usability Reviews", Postgre.org. 7 June. 2012 https://wiki.postgresql.org/wiki/Usability_reviews

"Vue.js for Interactive Web Interfaces", *Vegibit*, https://vegibit.com/vue-js-for-interactive-web-interfaces/