Recreational Aviation Foundation Project

Joshua Anderson, Lenin Lewis, Gregory Fulbright

Gianforte School of Computing, Montana State University

CSCI 483R: Interdisciplinary Project Instruction

Dr. Clemente Izurieta

Spring 2022

I. Introduction

The Recreational Aviation Foundation (RAF) is a 501(c)(3) non-profit based in Bozeman, Montana. The mission of the RAF is to preserve, improve, and create airstrips with recreational value on behalf of the 500,000 general aviation pilots in the United States. One of the ways that the RAF does this is through a product they have known as the Airfield Guide. The Airfield Guide provides an easy-to-use aviator's atlas to backcountry airstrips, complete with photos, videos, and detailed information about airstrip features and amenities. The issue is that the Airfield Guide does not work if the user does not have an internet connection. Not having the internet is problematic for pilots who want to use the Airfield Guide because a majority of the airstrips and locations on the Airfield Guide do not have an internet connection. It is now our goal to help the RAF solve these problems of data accessibility for pilots in flight.

II. Qualifications

Joshua Anderson

1992 Greatview Drive, Kalispell, MT 59901
(406) 309-5317 | joshua_k_anderson@outlook.com

SUMMARY

Hard-working, organized, and passionate computer science major seeking opportunities to enhance my knowledge and experience in the field of software development/engineering.

EDUCATION

Montana State University
Currently a Senior - GPA: 3.87
B.S. Degree in Computer Science
Expected Graduation Spring 2022

WORK EXPERIENCE

*see back for detailed work history

MSU Residence Life
Assistant Community Director
July 2021 – Present

Zoot Enterprises
Microservices Intern
May 2021 – July 2021

MSU Residence Life
Resident Advisor
July 2020 – July 2021

MSU School of Computing
Teaching Assistant & Tutor
August 2020 - Present

Famous Footwear
Assistant Manager
July 2017 – July 2020

AWARDS & HONORS

- Dean's Honor Roll – Fall 2019, 2020 & Spring 2021
- President's Honor Roll – Spring 2020

REFERENCES

Katie DeVore (253) 320-4808
MSU Residence Life Area Coordinator

Heidi Yerkes (406) 300-6179
Famous Footwear Store Manager

Chris Dull (406) 579-1400
Zoot Vice President of Microservices

TECHNICAL SKILLS

Programming Languages
Java, Python, C, and SQL

Programming Tools
Visual Studio, IntelliJ, WebRules, vim, PuTTY, GitHub

Other
Linux, HTML, CSS, LaTex, Microsoft Office Applications, and Google applications

PERSONAL SKILLS

- High level of leadership and collaboration capabilities
- Strong problem-solving and critical thinking skills
- Communication skills in all methods are excellent
- Extremely organized and effective at time management

COURSE WORK

Computer Science
- Joy and Beauty of Data
- Data Structures & Algorithms
- Programming with C
- Software Engineering
- Computer Systems
- Data Mining
- System Administration
- Computer Science Theory (Fall 2021)
- Database Systems (Fall 2021)
- Advanced Algorithm Topics (Fall 2021)

Mathematics
- Calculus 1 & 2
- Multivariable Calculus
- Differential Equations
- Linear Algebra 1 & 2
- Intro to Statistics

Gregory Fulbright

312 10th Ave. North, Lewistown, MT 59457
406-366-3425 | gregory2fulbright@gmail.com

Montana State University student studying Computer Science with a minor in International Business. Career goals and interests include data analysis, especially in design and development. Possesses passion for working with others, excels in communication, and takes pride and ownership of work.

EDUCATION

Montana State University, Bozeman, MT.
Expected graduation: Fall 2022
Computer Science Major
International Business Minor.

SKILLS

Proficient in HTML, CSS, Python, Java, C, C#
User Experience (UX), User Interface (UI), Human-Computer Interaction

WORK EXPERIENCE

Resident Advisor • *Montana State University* 05/2020—present
University Student Housing, Bozeman, MT

- Resource for students living in the residence halls on campus
- Encourage and foster mental, academic, social, and physical well-being.
- Address and report any violations of university policy that are observed

Housing Ambassador / Courier • *Montana State University* 05/2020—present
University Student Housing, Bozeman, MT

- Introduce students and families to the residence hall and its services, facilities, and staff
- Make deliveries across campus and handle confidential student files
- Participate in data collection and survey response in accordance with departmental needs

Scaler • *Wheat Montana Bakery and Deli* 06/2020—07/2020
Three Forks, MT

- Prepare ingredients according to formula for mass bakery production
- Record and report use of ingredients and aid in facility sanitation and quality control

Student Trainee (Administrative) • *Bureau of Land Management* 06/2017—01/2020
Lewistown Field Office, Lewistown, MT

- Enter and record data into government fiscal, personnel, and travel programs
- Assist staff in field (GPS mapping, soil samples, fence construction, range monitoring, paleontological and vegetation plots survey)
- Manage reception area, greet customers, complete sales

AWARDS, RECOGNITIONS, & INVOLVEMENT

- MSU Bookstore/Auxiliary Services Student Scholarship (2022)
- MSU Dean's List (Spring 2019; Fall 2021)
- Montana State University Spirit Squad (fall 2018-spring 2020) (Rookie of the Year 2019)
- Fork and Spoon Bozeman volunteer
- Knights of Columbus member (2017-present)
- American Legion Boys State Delegate (Outstanding Citizen of Model City; June 2017)

Lenin N. Lewis

1188 104C Opportunity way
Bozeman, MT 59715
(406) 691-0988
lenin.n.lewis@gmail.com

EDUCATION

Montana State University: Bozeman, MT Expected graduation: May 2022
Bachelor of Science, in Computer Science and Psychology Status: Senior

SKILLS

Technical: Proficient skills in Java, Python, C, UX design, human factors design, computer security, data structures, discrete structures, Git/Github, algorithmic design, and Microsoft Office skills. Psychology skills in human social cognition, behaviors, abnormalities, and annotating medical documents.

Workplace: Outgoing, welcoming personality, problem solver, open minded, team player, professionalism, multitasking, works well under pressure, attentive, conducive, and creative

EXPERIENCE

Rankin Elementary School: Kalispell, MT *Intern* August 2021 – Present

- Shadowing in a Social Emotional Behavioral Cross Categorical and Cross Education Category
- classroom to support children socially through the school year
- Emphasizing with different learning techniques and creating online learning resources with supervisor
- Spending one on one time with students doing crafts to work with social and motor skills

Gianforte School of Computing: Undergrad Research Assistant March 2020 – July 2020

- Audio recorded, simulated, doctor-patient conversations, to assist in natural language processing
- Annotated the conversations to identify the psychology-based information
- Communicated effectively with fellow researchers from a remote setting

American Computer & Robotics Museum: Bozeman, MT *Tour Guide*August 2019- September 2020

- Guide numerous people through the benchmarks of technology
- Creating a welcoming atmosphere to people of all ages, by adapting tour based off interests
- Demonstrated people management, by handling multiple groups simultaneously

Gianforte School of Computing and Center for Mental Health Research and Recovery
March- September 2019

- Undergraduate Research Assistant at Montana State University
- Preparing gold standard literature data for mental health projects by annotating PubMed abstracts
- Sufficiently handles self-management, hours are self-constructed

ACHIEVEMENTS

- Montana State University President's and Dean’s list for the Norm Asbjornson College of Engineering

III. Background

For the Recreational Aviation Foundation (RAF) Project, we were assigned the task of modernizing and individualizing the current software used by the RAF. To do this, we first had to understand the current system that is being used on the backend. The current backend code is in an older language called Active Server Pages (ASP). ASP is a server-side technology introduced in 1998 by Microsoft, that is used to connect to a database containing all the airstrip information ("Asp and ASP.NET tutorials," 2021). From there, we set out to understand the new technology that we will be working with or more specifically, content packs and iOS (Apple's operating system). Content packs are a tool created by ForeFlight, a popular application used by pilots, that allows the transfer and implementation of data onto ForeFlight from users. The biggest problem that the RAF is facing with the current system is the offline-ability of their software, and this can be somewhat remedied by ForeFlight, which allows pilots to download information on the airfields prior to takeoff. At this time, ForeFlight only works on iOS. We will be ensuring that all of our improvements to the system work with the iOS design. Next, we planned a way to let users choose what specific files they want to download with content packs, although it was not implemented based on the RAF's discretion. This selection process design was as simple as letting pilots select however many airstrips they would want, up to thirty.

In the process of innovating the Airfield Guide with the addition of content packs, we found ways to implement the design changes without corrupting security systems already in place. We decided to use C# to screen scrape the HTML on the Airfield Guide webpages which allowed us to get the individualized information for desired airstrips without actually connecting to the database server. Having no connection to the server means that there are quite literally no vulnerabilities with the database with this new addition.

IV. Work Schedule

Figure 1

1st Semester CSCI 482R Work Schedule

Monday	**Tuesday**	**Wednesday**	**Thursday**	**Friday**
Team Meeting 4:00 PM - 8:00 PM	Team Meeting 1:00 PM - 3:00 PM		Class 2:05 PM - 2:55 PM	Team Meeting 4:00 PM - 6:00 PM

Figure 2

2nd Semester CSCI 483R Work Schedule

Monday	**Tuesday**	**Wednesday**	**Thursday**	**Friday**
Group Meeting 11:00 AM - 2:00 PM	Group meeting with RAF 4:00 PM - 5:00 PM		Class 10:50 AM - 12:05 PM	Group Meeting 11:00 AM - 2:00 PM

Figure 3

1st Semester CSCI 482R Milestones

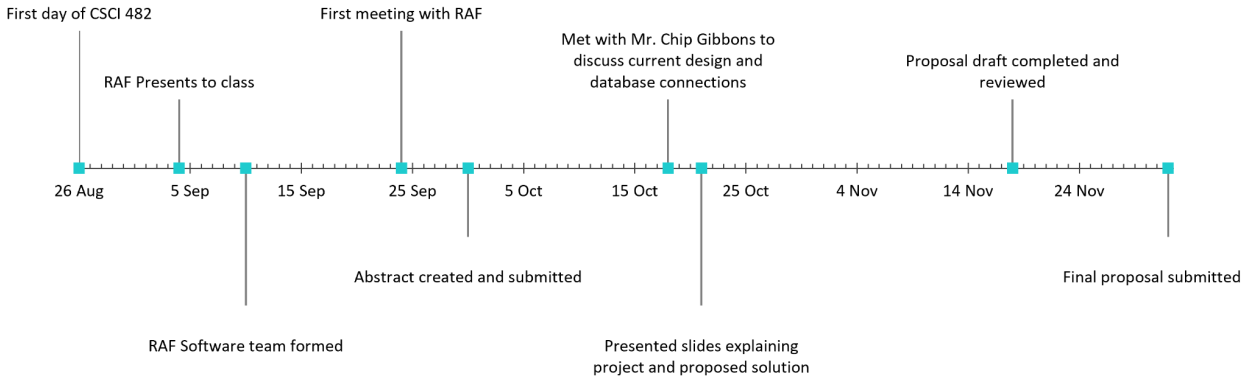
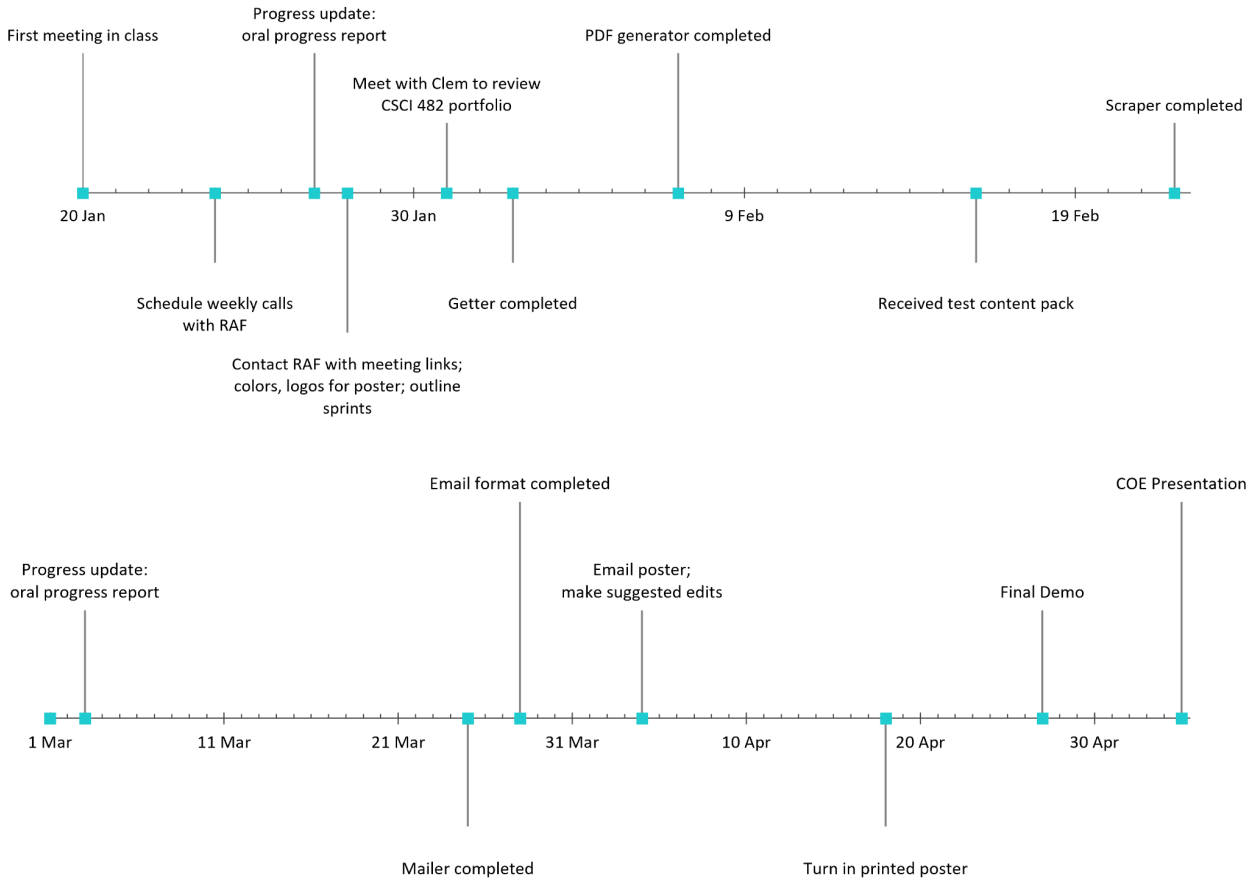


Figure 4

2nd Semester CSCI 483R Milestones



Scrum Life Cycle Approach

To best ensure that our group had a consistent goal and each member knew what their roles were, we decided to use the scrum life cycle approach. This framework emphasizes teamwork, accountability, and iterative progress towards a milestone (Lutkevich, 2021). Our structure was to start with what we knew and to establish through the presentations from the RAF, our meetings with RAF leadership, and our data analysis. From there, we were able to make new additions to our project, make weekly analyses of progress, keep each other accountable for assigned tasks, give feedback, and adjust milestones as necessary (Eby, 2021).

--- Roles & Responsibilities ---		
Josh	Lenin	Greg
● Diagramming ● Program Development ● Create an Email System ● Task Manager ● Quality Assurance ● JSON file Development	● Scheduling meetings ● Program Development ● Project Research ● Literature Revision ● GitHub Management ● KML file Development	● Program Development ● Presentation Design ● Design Revision ● Virtual Communications ● UX Design ● Email Development

V. Proposal Statement

Through the design process we implemented the functional, non-functional, performance, and interface requirements. Additionally, architectural design documents and development standards were referenced throughout the process. Our primary goal within the Recreational Aviation Foundation Project was to create customizable content packs to enable offline access to the Airfield Guide database within ForeFlight. Our functional requirements were creating downloadable, operational content packs that are extracted from the Airfield Guide and engaged through ForeFlight interface. Our non-functional aspects of the project aligned with creating a reliable system source for the RAF team to provide security, scalability, usability, and serviceability. In addition to being reliable, we wanted to preserve the users' security when they created content packs which we did by using a screen scraping method, rather than actually establishing a connection to the server. This project will be using the UML documents (class, use case, and component diagrams) to portray the architectural designs within the code. From all these requirements, we do expect that the code will perform at a certain level to meet the expectations of RAF now and for the next five years.

The RAF did not want us to implement the front end side of things which would have allowed the user to access the Airfield Guide and select the map that they need, in almost like an online pick-and-choose, to then create their individually-customized content packs. We still think this may be the most user-friendly way to implement how content pack creation can occur within the user's experience.

We will be using scrum development in our project to get work done effectively while keeping the stakeholders' needs met each step of the development. We will be using five tools to complete this project: Visual Studio Code, Discord, WebEx, and GitHub. Visual Studio will be

used for coding and debugging our content packs. We will download and us the .NET Framework and C# onto Visual Studio. Then Discord, Webex, and GitHub will be used to communicate and collaborate with the RAF and within the team to increase ease of sharing code, information, and ideas.

VI. Methods

Figure 5

Class Diagram

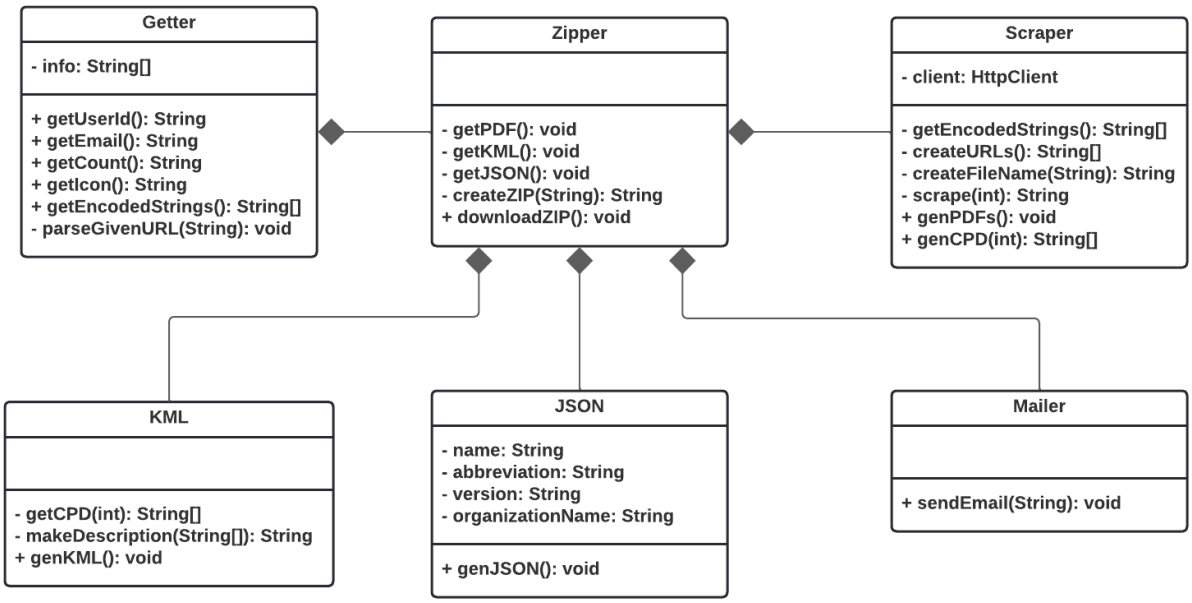


Figure 6

Use Case Diagram

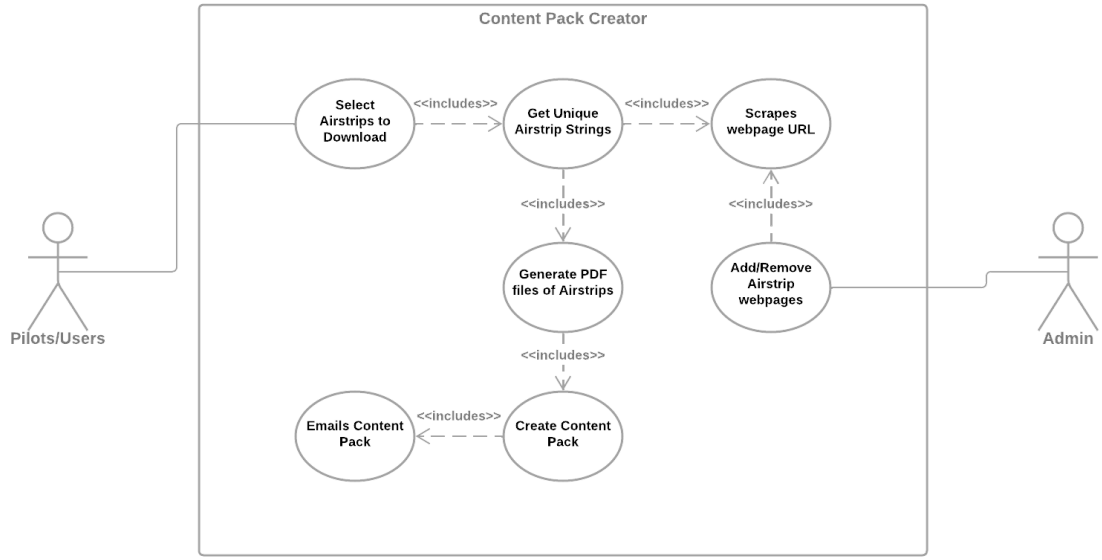
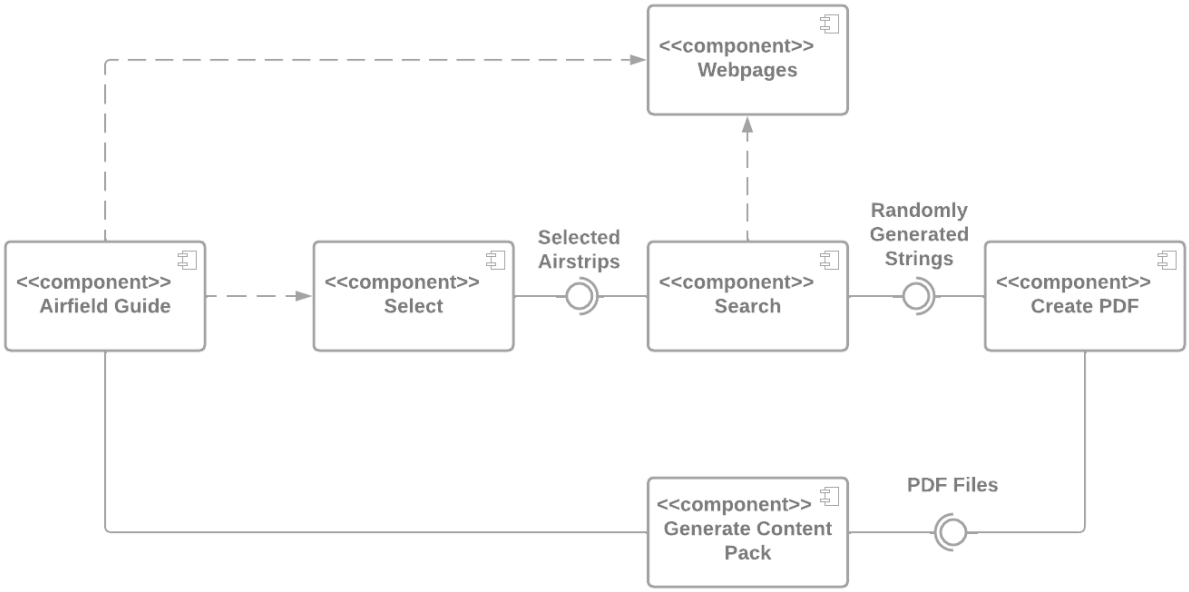


Figure 7

Component Diagram



Design Pattern

The design pattern we decided to use for this project was a Private Class Data design pattern. This design pattern ensures that data is only mutable once and is accessible only through getters (Shvets, 2018). This is crucial to have because we do not want attributes to be changed or assigned to the wrong values when they shouldn't be. This can happen when multiple users are trying to run the code at the same time, thus why a Private Class Data design is necessary.

Trade-Offs

Throughout the design process of this project, we ran into many times where we had to make decisions that lead to different trade-offs. Here are some of those decisions:

1. **Full Database SQL vs. Screen Scraping:** At first we decided we would use a full database SQL to access all of the airfields. We could have gone the route of doing everything completely in SQL and using the current databases in place to get the information that we want to include in the content packs. There were a couple problems with this solution. One, the majority of our group had never seen SQL before. This would have been a big learning curve for all of us. It seemed most effective to choose something we were more comfortable and experienced with. Two, going the SQL route would have been redundant for what was already created within the system. The current Airfield Guide already has a system in place that gathers the wanted information and puts it into a .pdf file for us, so it would be unnecessary for us to create the exact same thing that was already in place. When considering all of those points a different solution seemed like the best route for this project. During research we came across the Screen Scraping method. This method means that we more or less just take a screenshot of the .pdf file that is already generated. Using a bit of URL manipulation, we now could get the information that we needed without running through any connection or SQL hastles. This also means that majority of work could be done in C# which we all were much more comfortable with and would use the system already in place rather than recreate it.
2. **Scrum vs. Waterfall:** The other decision we had to make as a group was what our life cycle approach was to be in order to most effectively meet our milestones. Of the many potential methods, we determined that either Waterfall or Scrum would be best for our

desired outcomes. Ultimately, we ended up on Scrum due to the flexibility that it had over Waterfall (Eby, 2017). With the Scrum life cycle approach, we are able to work on different parts of the project at any time which was something the group was very fond of especially with this design process. Waterfall has a very structured process and clear requirements, but since we knew that there was potential for significant change as we got into the implementation process, we felt that Scrum had all the pieces that we wanted over Waterfall (Eby, 2017).

3. **Python vs. C#:** When we initially began planning how to approach this project, we were confident that the work could be done in Python 3.6. However, once the coding actually began, it was determined that C# would be a preferable language to effectively achieve our goals. The web host server that the Airfield Guide runs on does not currently support Python, so it could not be an effective language to use. C# does run on the Airfield Guide server, so it would make much more sense to implement that language.

VII. Expected Results

Upon completing the project working with the RAF, there are three primary objectives that we anticipate developing or creating. First, content packs can be created and customized to each user's specifications using the Airfield Guide. These content packs will be put together by selecting each desired airfield, up to twenty five airfields, in the Airfield Guide and then created by adding each file as a .pdf into a .zip folder. Each file will be listed lexicographically in the Navigation Data (navdata) folder combined with each airstrips information that will be added to a KML file within the customized content pack. Second, content packs will be sent to a pilot's email and can be downloaded after accessing the Airfield Guide. These packs will then be capable of being added directly to ForeFlight. Third, content packs can be accessed offline through the ForeFlight app for iOS.

VIII. References

Eby, K. (2017, February 15). *What's the difference? Agile vs Scrum vs Waterfall vs Kanban*. Smartsheet. Retrieved November 29, 2021, from https://www.smartsheet.com/agile-vs-scrum-vs-waterfall-vs-kanban.

Lutkevich, B. (2021, October 28). *What is Scrum?* SearchSoftwareQuality. Retrieved November 29, 2021, from https://searchsoftwarequality.techtarget.com/definition/Scrum.

Shvets, A. (2018). *Private Class Data*. SourceMaking. Retrieved April 26, 2022, from https://sourcemaking.com/design_patterns/private_class_data

Refsnes Data. (n.d.). *Asp and ASP.NET tutorials*. W3Schools. Retrieved November 29, 2021, from https://www.w3schools.com/asp/default.ASP.

Shvets, A. (2021). *Singleton*. Refactoring.Guru. Retrieved November 29, 2021, from https://refactoring.guru/design-patterns/singleton.

IX. Appendix

Scraper:

```
using iText.Html2pdf;
using iText.StyledXmlParser.Css.Media;
using iText.Kernel.Pdf;
using iText.Kernel.Geom;

public class Scraper {

   private static readonly HttpClient client = new HttpClient();

   // Makes call to get encoded strings associated with airfields
   private static string[] getStrings() {
       return Getter.getEncodedStrings();
   }

   // Generates URLs for airstrips
   private static string[] createURLs() {
       string[] encodedStrings = getStrings();
       string[] urls = new string[encodedStrings.Length];
       for (int i = 0; i < urls.Length; i++) {
           string tempURL =
"https://airfield.guide/generate_printable_briefing.asp?a=";
           tempURL += encodedStrings[i];
           urls[i] = tempURL;
       }
       return urls;
   }

   // Creates file name from HTML
   private static string createFileName(string HtmlBody) {
       string name = HtmlBody.Substring(HtmlBody.IndexOf("<!--ID:")+7);
       int end_name = name.IndexOf("-");

       name = name.Substring(0,end_name)+"-Airfield Guide";

       string id = Getter.getUserId();

       string path = @"../ContentPackGenerator/AirfieldsToZip/" + id + "/navdata/" +
name +  ".pdf";
```

```
        return path;
    }

    // Scrapes HTML from URL
    private static string scrape(int index) {
        try {
            string[] urls = createURLs();
            string responseBody = client.GetStringAsync(urls[index]).Result;

            // PDF Converter doesn't understand certain attributes, so they needed to
be replaced or removed
            responseBody =
responseBody.Replace("\n",string.Empty).Replace("\r",string.Empty).Replace("\t",string
.Empty);
            responseBody =
responseBody.Replace("<b>","<B>").Replace("</b>","</B>").Replace("Div","div");
            responseBody =
responseBody.Replace("css/site-font-css/style.min.css","https://airfield.guide/css/sit
e-font-css/style.min.css").Replace("css/filter-airports.min.css",
"https://airfield.guide/css/filter-airports.min.css");
            responseBody = responseBody.Replace("images/green-background.png\"
style=\"width:100%;
height:100%;\"","https://airfield.guide/images/green-background.png\"
id=\"green-header\"").Replace("images/logos/TAFG_150.png\"","https://airfield.guide/im
ages/logos/TAFG_150.png\"");
            responseBody = responseBody.Replace("images/yellow-background.png\"
style=\"width:100%;
height:100%;\"","https://airfield.guide/images/yellow-background.png\"
id=\"yellow-header\"").Replace("images/logos/TAFG_150.png\"","https://airfield.guide/i
mages/logos/TAFG_150.png\"");
            responseBody =
responseBody.Replace("images/logos/TAFG_150.png\"","https://airfield.guide/images/TAFG
_150.png\"");
            responseBody = responseBody.Replace("style=\"position:absolute; top:6px;
left:6px; font-size:30px;", "style=\"position:absolute; top:6px; left:6px;
font-size:30px; text-align:left;").Replace("style=\"position:absolute; top:6px;
right:6px; font-size:30px;", "style=\"position:absolute; top:6px; right:6px;
font-size:30px; text-align:right;");
            responseBody = responseBody.Remove(responseBody.IndexOf("Helvetica
Neue")-2,18);
            responseBody = responseBody.Replace("style=\"font-size:12px;",
"style=\"position:relative; top:1px; left:1px; font-size:12px; text-align:left;");
```

```
            return responseBody;
        }
        catch(HttpRequestException e) {
            string responseBody = "Error " + e.ToString();
            return responseBody;
        }
    }

    // Generates PDFs using PDF Converter
    public static void genPDFs() {
        string id = Getter.getUserId();
        string directoryPath = @"../ContentPackGenerator/AirfieldsToZip/" + id +
"/navdata";

        // Deletes directory if it already exists
        if (Directory.Exists(directoryPath)) {
            Directory.Delete(directoryPath, true);
        }
        Directory.CreateDirectory(directoryPath);

        // Generates PDF for each airstrip
        for (int i = 0; i < Int32.Parse(Getter.getCount()); i++) {
            string responseBody = scrape(i);

            string filePath = createFileName(responseBody);

            if (File.Exists(filePath)) {
                File.Delete(filePath);
            }
            FileStream fs = File.Create(filePath);

            ConverterProperties converterProperties = new
ConverterProperties().SetMediaDeviceDescription(new
MediaDeviceDescription(MediaType.PRINT));
            PdfWriter pdfWriter = new PdfWriter(fs);
            PdfDocument pdfDocument = new PdfDocument(pdfWriter);
            pdfDocument.SetDefaultPageSize(new PageSize(PageSize.A3));
            HtmlConverter.ConvertToPdf(responseBody, pdfDocument, converterProperties);
        }
    }
```

```
    // Generates Content Pack Data from HTML
    public static string[] genCPD(int index) {

        string responseBody = scrape(index);

        int start = responseBody.IndexOf("<!--CONTENT PACK DATA-->")+24;
        int end = responseBody.IndexOf("<!--END CONTENT PACK DATA-->");

        string cpdString = responseBody.Substring(start,end-start);

        string[] cpdTags = cpdString.Split("<!--",
StringSplitOptions.RemoveEmptyEntries);

        for (int i = 0; i < cpdTags.Length; i++) {
            string currentString = cpdTags[i];
            start = currentString.IndexOf(":")+1;
            end = currentString.IndexOf("-->");

            cpdTags[i] = currentString.Substring(start,end-start);
        }
        return cpdTags;
    }
}
```

Zipper:

```
using System.IO.Compression;

public class Zipper {

    // Makes call to generate PDFs
    private static void getPDF() {
        Scraper.genPDFs();
        return;
    }

    // Makes call to generate KML
    private static void getKML() {
        KML.genKML();
        return;
    }
```

```
    // Makes call to generate JSON
    private static void getJSON() {
        JSON.genJSON();
        return;
    }

    // Gets PDFs, KML, JSON and creates ZIP file
    private static string createZIP() {
        string startPath = @"../ContentPackGenerator/AirfieldsToZip/" +
Getter.getUserId();
        string zipPath = @"./../output/" + Getter.getUserId() + ".zip";

        getPDF();

        getKML();

        getJSON();

        // Deletes file path if it already exists
        if (File.Exists(zipPath)) {
            File.Delete(zipPath);
        }
        ZipFile.CreateFromDirectory(startPath, zipPath);

        return zipPath;
    }

    // Makes call to create ZIP file and then sends it via email to user
    public static string downloadZIP() {
        string zipPath = createZIP();

        Mailer.sendEmail(zipPath);

        return "SUCCESS";
    }
}
```

Getter:

```
public class Getter {

   // String array that will hold parsed url info
   private static string[] info;

   public Getter(string url) {
       parseGivenURL(url);
   }

   // Parse url and starts content pack generation
   private static void parseGivenURL(string url) {
       info = url.Split('&','?');
       Zipper.downloadZIP();
   }

   // Gets user id from parsed url
   public static string getUserId() {
       return info[1].Substring(2);
   }

   // Gets email from parsed url
   public static string getEmail() {
       return info[2].Substring(2);
   }

   // Gets count from parsed url
   public static string getCount() {
       return info[3].Substring(2);
   }

   // Gets icon from parsed url
   public static string getIcon() {
       return info[4].Substring(2);
   }

   // Gets encoded strings from parsed url
   public static string[] getEncodedStrings() {
       return ((info[5].Substring(2)).Split(',')).Where(x =>
!string.IsNullOrEmpty(x)).ToArray();
   }
```

```
}
```

KML:

```
using System.Xml;
public class KML {

   // Makes call to generate Content Pack Data
   private static string[] getCPD(int index){
       return Scraper.genCPD(index); // CPD = [id, facility, elevation, comm,
latitude, longitude, ownership, contact, facility-use, runway]
   }

   // Makes description field in KML file
   private static string makeDescription(string[] cpInfo){
       string description = "<![CDATA[<div style='width:700px;'><h2>";
       description += cpInfo[1] + " (" + cpInfo[0] + ")"; // Getting facility then id
       description += "<BR></h2><h3>Elevation: " + cpInfo[2] + "<BR>Comm: " +
cpInfo[3]; // Getting elevation and comm
       description += "<BR>Lat/Long: " + cpInfo[4] + " / " + cpInfo[5] +
"<BR>Ownership: " + cpInfo[6] + "<BR>";
       description += "Contact: " + cpInfo[7] + "<BR>Facility Use: " + cpInfo[8] +
"<BR>" + cpInfo[9] + "</h3></div>]]>";
       return description;
   }

   // Generates KML file
   public static void genKML() {
       using (XmlWriter writer =
XmlWriter.Create("../ContentPackGenerator/AirfieldsToZip/" + Getter.getUserId() +
"/navdata/RAF - Airfield Guide.kml")) {

           // Namespace Declaration
           writer.WriteStartElement("kml");
           writer.WriteAttributeString("xmlns", "x", null,
"http://www.opengis.net/kml/2.2");
           writer.WriteAttributeString("xmlns", "xsd", null,
"http://www.w3.org/2001/XMLSchema");
           writer.WriteAttributeString("xmlns", "gx", null,
"http://www.google.com/kml/ext/2.2");
           writer.WriteAttributeString("xmlns", "atom", null,
"http://www.w3.org/2005/Atom");
```

```
            // Constant Elements/Attributes
            writer.WriteStartElement("Document");
            writer.WriteStartElement("Style");
            writer.WriteAttributeString("id","permissiveIcon");
            writer.WriteStartElement("IconStyle");
            writer.WriteElementString("scale", "1.5");
            writer.WriteElementString("color", "ff00CC00");
            writer.WriteStartElement("Icon");
            writer.WriteElementString("href",
"http://maps.google.com/mapfiles/kml/shapes/placemark_circle.png");
            writer.WriteEndElement(); // Icon End
            writer.WriteEndElement(); // IconStyle End
            writer.WriteEndElement(); // Style End
            writer.WriteStartElement("Style");
            writer.WriteAttributeString("id","conditionalIcon");
            writer.WriteStartElement("IconStyle");
            writer.WriteElementString("scale", "1.5");
            writer.WriteElementString("color", "ff00eaea");
            writer.WriteStartElement("Icon");
            writer.WriteElementString("href",
"http://maps.google.com/mapfiles/kml/shapes/placemark_circle.png");
            writer.WriteEndElement(); // Icon End
            writer.WriteEndElement(); // IconStyle End
            writer.WriteEndElement(); // Style End

            // Airfields Information
            int counter = Int32.Parse(Getter.getCount());
            for (int i = 0; i < counter; i++) {
                string[] cpInfo = getCPD(i);
                writer.WriteStartElement("Folder");
                writer.WriteStartElement("name");
                writer.WriteStartElement("div");
                if (cpInfo[8] == "Conditional") { // Style depends on if airfield is
Conditional or Permissive
                    writer.WriteAttributeString("style", "background-color: #F0F014");
                }
                else {
                    writer.WriteAttributeString("style", "background-color: #14FF00");
                }
                writer.WriteString(cpInfo[0] + " - " + cpInfo[1]);
                writer.WriteEndElement(); // div End
```

```
                writer.WriteEndElement(); // name End
                writer.WriteElementString("visibility", "1");
                writer.WriteElementString("open", "0");
                writer.WriteStartElement("Placemark");
                writer.WriteStartElement("name");
                writer.WriteElementString("h1", cpInfo[0]);
                writer.WriteEndElement(); // name End
                writer.WriteStartElement("description");
                writer.WriteRaw(makeDescription(cpInfo));
                writer.WriteEndElement(); // description End
                writer.WriteElementString("visibility", "1");
                writer.WriteStartElement("styleUrl");
                if (cpInfo[8] == "Conditional") { // styleUrl depends on if airfield is
Conditional or Permissive
                    writer.WriteString("#conditionalIcon");
                }
                else {
                    writer.WriteString("#permissiveIcon");
                }
                writer.WriteEndElement();
                writer.WriteStartElement("Point");
                writer.WriteElementString("coordinates", cpInfo[5] + "," + cpInfo[4] +
",0");
                writer.WriteEndElement(); // Point End
                writer.WriteEndElement(); // Placemark End
                writer.WriteEndElement(); // Folder End
            }
            writer.WriteEndElement(); // Document End
            writer.WriteEndElement(); // kml End
            writer.Flush();

        }
    }
}
```

JSON:

```
using System.Text.Json;
public class JSON {

   public string? name { get; set; }
   public string? abbreviation { get; set; }
   public string? version { get; set; }
   public string? organizationName { get; set; }
   public static void genJSON() {
       var json = new JSON
       {
           name = "Airfield Guide Content Pack",
           abbreviation = "AG.V1",
           version = "1.0",
           organizationName = "RAF"
       };

       string fileName = @"../ContentPackGenerator/AirfieldsToZip/" +
Getter.getUserId() + "/manifest.json";
       string jsonString = JsonSerializer.Serialize(json);
       File.WriteAllText(fileName, jsonString);
   }
}
```

Mailer:

```
using MimeKit;
using MailKit.Net.Smtp;
using MimeKit.Utils;

public class Mailer {

   // Sends email with pre-generated message and content pack as attachment
   public static void sendEmail(String zipPath) {
       var message = new MimeMessage();
       message.From.Add(new MailboxAddress ("Airfield Guide",
"admin@airfield.guide"));
       message.To.Add(new MailboxAddress ("Airfield Guide User", Getter.getEmail()));
       message.Subject = "Test Email -- " + DateTime.Now;

       var builder = new BodyBuilder();
```

```
        // Email header banner
        var image = builder.LinkedResources.Add (@"./../images/footer-banner.png");
        image.ContentId = MimeUtils.GenerateMessageId();
        builder.HtmlBody = string.Format (@"<img src=""cid:{0}"" width=""500""><p>Thank
you for using the Airfield Guide in union with the Recreational Aviation Foundation
(RAF).<br><br>To learn more about the RAF, visit <a
href=https://www.theraf.org/'>TheRAF.org</a></p>", image.ContentId);

        // Content pack attachment
        builder.Attachments.Add(zipPath).ContentDisposition.FileName = "RAF - Airfield
Guide.zip";

        message.Body = builder.ToMessageBody();

        using (var client = new SmtpClient()) {
            client.Connect("smtp.gmail.com", 465, true);
            client.Authenticate("rafairfieldguide@gmail.com","RAFemail123");

            client.Send(message);
            client.Disconnect(true);
        }
    }
}
```