

# COS 397: Computer Science Capstone I

---



**BirdSpotter**

**Critical Design Review**

Developed for: Dr. Cynthia Loftin

Developed by: Penobscot Development Group

# Executive Summary

BirdSpotter is designed to integrate machine learning into the analysis workflow for estimating the population of nesting birds using aerial imagery. BirdSpotter is built for use by biologists, to allow rapid population measurement without the added overhead of manual categorization. It allows users of the app to view data, as well as importing or exporting processed datasets. The project uses a machine learning algorithm created by members of the University of Maine Graduate program for the categorization of nesting birds.

BirdSpotter is being developed using agile methods in a modified V development model, with frequent client involvement from initial requirements specification to prototyping and validation. The development process is solidly into the prototype and revision phase, with a future emphasis on prototype testing and validation.

## Preface

We aim to have BirdSpotter be a solution to the difficulty that is currently required in getting through surveys of seabird population data. We would like for it to be a solution that can be widely utilized by biologists seeking to research the current state of some type of seabird, or simply for hobbyists that may find the information to be interesting. If everything goes well, it may also be possible to modify it slightly and use it as a more general map data display software. This project is being developed under the advice and guidance of Dr. Cynthia Loftin to best fit the needs that arise from surveying seabird data, and alongside machine learning that is being developed under the mentorship of Dr. Roy Turner. In this document, we will discuss in great detail the specifics of the BirdSpotter app that we have been developing, as well as how we plan to continue development going into next semester.

# Table of Contents

<b>Executive Summary</b>	<b>2</b>
<b>Preface</b>	<b>2</b>
<b>Table of Contents</b>	<b>3</b>
<b>List of Figures</b>	<b>4</b>
<b>List of Tables</b>	<b>4</b>
<b>Symbols</b>	<b>4</b>
<b>Summary</b>	<b>4</b>
<b>Introduction</b>	<b>6</b>
<b>Purpose</b>	<b>6</b>
<b>Method</b>	<b>6</b>
<b>Design</b>	<b>6</b>
Design Constraints	7
Responsiveness	7
Accessibility	7
Ease of Use	7
Security	8
Design Approach	8
Design Problems	8
Design Details	9
<b>Equipment</b>	<b>9</b>
Advanced Computing Group(ACG)	9
Libraries	9
<b>Tests</b>	<b>9</b>
Unit Test Plan	9
Integration Test Plan	10
Acceptance Test Plan	10
<b>Analysis</b>	<b>10</b>
Visualization	11
Accounts	11
Progress	11

<b>Conclusions</b>	<b>11</b>
<b>Recommendations</b>	<b>12</b>
<b>References</b>	<b>12</b>
<b>Bibliography</b>	<b>12</b>
<b>Acknowledgments</b>	<b>12</b>
<b>Appendix</b>	<b>12</b>
Software Requirements Specification Document	12
Software Design Document	12
User Interface Design Document	13

## List of Figures

- 1, Logical System Architecture, 9
- 2, Technological System Architecture, 9
- 3, View Decomposition Diagram, 10
- 4, Database Description, 11
- 5, Navigation View Diagram, 13

## Symbols

ACG: Advanced Computing Group at the University of Maine

AI: Artificial Intelligence

WCAG: Web Compatibility and Accessibility Guidelines

SRS: Software Requirements Specification.

UIDD: User Interface Design Document

SDD: Software Design Document

ACME: Automated Certificate Management Environments

MVC: Model-View-Controller

## Summary

BirdSpotter is a graphical interface for integrating and viewing machine learning and field survey data for rapid population estimation of colonial nesting birds. The creation of this interface will allow officials to quickly and effectively draw conclusions based on data provided from human and machine learning observations of bird species, activity, and location. BirdSpotter also aims to allow scientists and citizen scientists alike to view their own collected data and share it with others and the community as a whole if desired. Select users will also be able to queue analysis to be performed by a machine learning algorithm prepared by graduate student Alex Revello that is run on UMaine's Advanced Computing Group (ACG) cluster.

# Introduction

The current workflow for scientists concerned with the tracking and categorization of birds and their behaviors includes manual categorization and visualizing the results in ArcGIS software. This project is being performed in tandem with a graduate student project to automatically categorize birds and behavior using machine learning algorithms.

Furthermore, this project will allow for a more streamlined approach to viewing data in place of using ArcGIS. The current approach is limited to scientists who hold a subscription to the service and prevent citizen scientists from taking a proactive role in assisting with the categorization of birds and their behaviors.

## Purpose

The objective of this application is to better support bird spotters, specifically biologists, in their pursuit of tracking various species of birds. The program is designed to bolster collaborative data collection and analysis of sea birds through remote sensing. The hope of the application is to fill the gap in software that is both able to interface with an AI able to identify birds and allow for the consolidation of data in regards to bird nesting.

## Method

BirdSpotter is being developed using a modified version of the V development model alongside an agile development process. Sprints are conducted in 2-week intervals, with two standups and one scrum per sprint. The client is consulted at the end of each sprint, with a presentation of the document artifact from that step in the V model, as well as any features completed during the sprint. The project is overseen by two main roles, the Product Owner (Alex Feren) and the Scrum Master (Nick Kania). The team's Product Owner is responsible for managing the product's goals and steps to reach these goals. They are in charge of ensuring that the product backlog reflects the requirements set forth by the client and that the requirements are met, resulting in the final product. The Scrum Master works in conjunction with the Product Owner to break up the tasks set in the product backlog are scheduled and completed in a sprint. On top of that, the scrum master is responsible for assisting those in the development team if questions arise, and to help solve problems if need be.

## Design

BirdSpotter is being designed to be an all-in-one solution for the analysis and visualization of seabird data. Ultimately, the goal is for the design to be a user-friendly and accessible solution for anyone who may require access to this data out of interest or for research purposes. BirdSpotter is being developed with some attention to the security of the users

and with an aim to maintain the integrity of the habitats as well, based on user access levels.

## Design Constraints

There are a number of design constraints that guide the design and user experience of the BirdSpotter interface. The primary design constraint to the interface is that it must exist in a web form. Aside from the web requirement, the system must also be responsive, secure, and easy to use to the extent that is specified in section 2: Functional Requirements and section 3: Non-Functional Requirements of the SRS document in the Appendix.

### Responsiveness

For almost all functions of the BirdSpotter app, the requirements for responsiveness are for the web page to respond within 5 seconds about 90% of the time for most of the views and actions that a user may request. There are also requirements for how quickly datasets must be uploaded to or downloaded from the BirdSpotter app. All datasets must be displayed within 5 seconds 85% of the time upon a request, and upload/download must be completable within 10 seconds 90% of the time.

### Accessibility

Users may have accessibility constraints such as color blindness or visual impairments which could make usage of the BirdSpotter app more difficult. As such we need to have options that will facilitate their use of the app by people who may otherwise have difficulty with a default view, to the extent that is reasonable in our development. We have decided to work within the WCAG standard, which covers not only accessibility for people with disabilities, but also general web usage and accessibility specifications, such as maintaining the status of pages that a user has previously been to. The specifications outlined are quite extensive, but we will be working to have compatibility as much as we can.

### Ease of Use

As the ease of use is a subset of web accessibility, we will primarily be using the standards laid out in WCAG for the ease of use. Within the guidelines are specifications such as clear labeling and navigation between pages, and readability requirements for the page as well (outlined in section 3.1 of the standard).

### Security

BirdSpotter is designed to be a public-facing application, and while a low-fidelity version of the data is displayed publicly through the BirdSpotter web interface. A further level of access to detailed data, data export, as well as the ability to import data is restricted to

registered users. Due to the access requirements of the client, the BirdSpotter application will not be using any external identity service: user accounts will be created and managed by an admin. BirdSpotter will use session validation to ensure that users accessing the protected parts of the web interface have a valid, authenticated session. User passwords will be stored in the database hashed and salted, and the database itself will not be exposed to the public internet: it will only be accessible by the BirdSpotter application. BirdSpotter will also use the tooling built into Django to avoid SQL injection vulnerabilities by using parameterized queries instead of raw SQL. The website will also require HTTPS, to ensure that communications between the client and server are encrypted and can be validated to be authentic. HTTPS will be achieved using Lets Encrypt, which allows for free short-lived HTTPS certificates via ACME challenge based verification.

## Design Approach

BirdSpotter is being developed using a Model-View-Controller(MVC) design pattern. As the name suggests, it consists of three parts, the model, the view, and the controller. The view is what the user sees, it is presented by the system as a means for the user to interact with the system. The main views for BirdSpotter include the map view, data view, bias view, and import view. The map view overlays data points on a map to display the location of each bird along with other metadata and also an image of the bird, which is all displayed when the user hovers over a data point. The data view displays visualizations for a geographic location based on changes over time and percentages based on the total population for the specific area. The bias view will be used to display information that is pertinent to researchers interested in the machine learning algorithm and will display information about each dataset if it was formed from the machine learning program. Lastly, the import view, which will handle file import and export allowing users to import data into the system and will allow users to name their datasets and designate the visibility of the data set, whether it be visible to all, a certain few specific people, a role, or to just the owner.

The controller modifies and presents the views, along with controlling data flow and executing other system processes. The controller is responsible for the vast majority of the system's logic and actions and all system information passes through the controller at some point. Lastly, there is the model, which is responsible for the storing and presentation of data to and from the controller. The model can be seen as the middle man between the application itself and the database that the application uses and allows for more programmatic means of interacting with the database once the layer is properly set up.



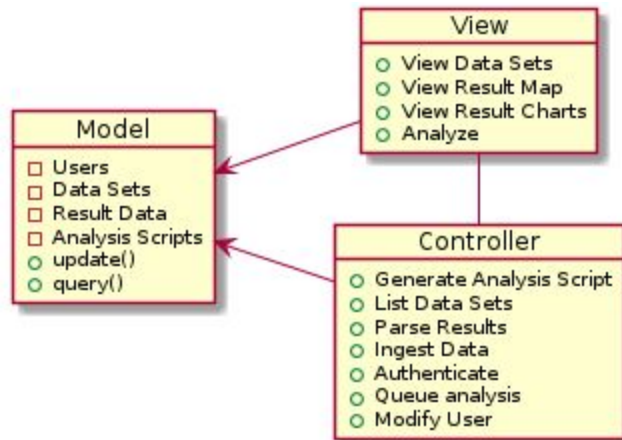


Figure 1. Logical System Architecture

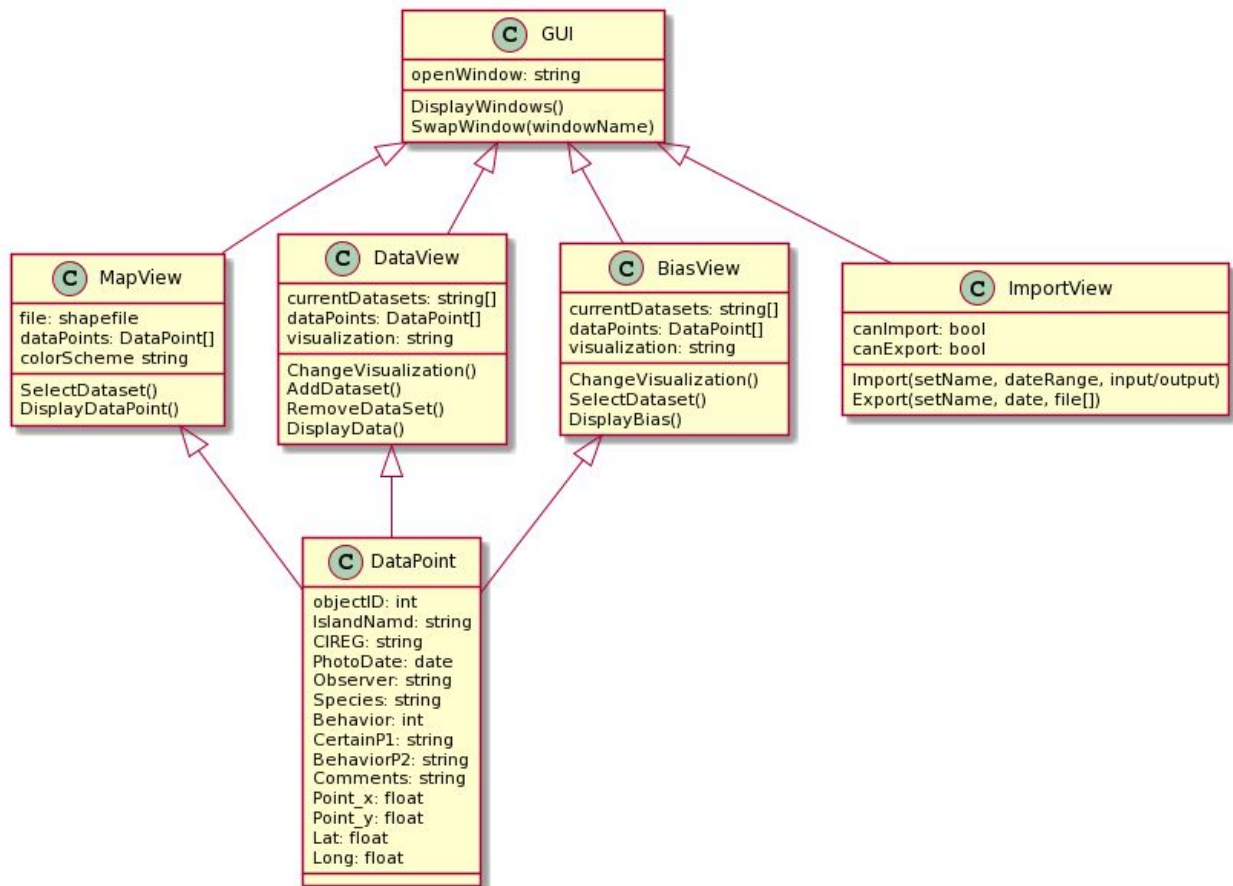


Figure 2. View Decomposition Diagram

The system itself will consist of 4 main elements. The application itself will be the centerpiece of the system and control all other elements of the system. This will be done with a web server which will be created using Django. This application will have access to

data storage, which in this case is a PostgreSQL database located on the same virtual machine run by the ACG. Along with user data, datasets, and other application data, the database will also contain paths on the same machine to the file storage which will be used to store image files exported from the machine learning project, along with the raw GeoTIFF files that the machine learning project takes as input. This leads to the final part, the machine learning project, which is being developed externally. This is hosted on the ACG's compute cluster.

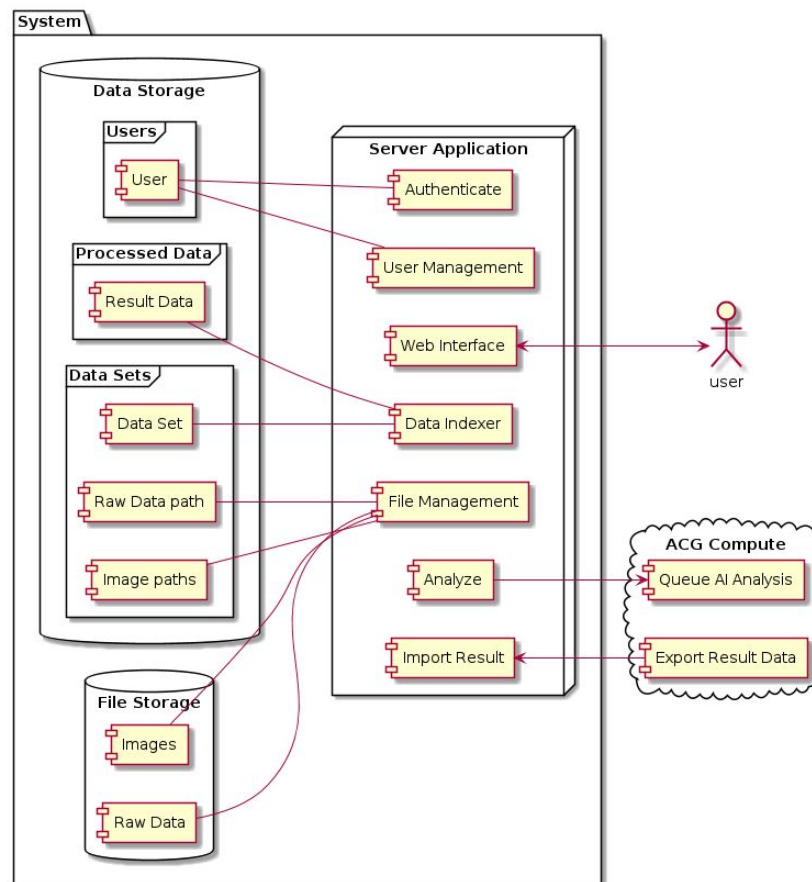


Figure 3. Technological System Architecture.

In regards to the database, there are 3 main models, and each main model also has a smaller model associated with it. Of the 3 main models, there is the user model, which stores all information pertaining to a user in the system, including user information as permissions. Permissions are handled using a role model associated with different roles such as Administrator, Scientist, Citizen Scientist, etc. Next is the dataset model, which contains information about a collection of data that was obtained from observations formed either from manual or automated categorization. This categorization is stored in the Shapefile model which mimics the provided shapefile fields, in addition, there is also a link to the Image model for each entry in the table that will contain the file path or URL for the image depending on the final implementation. The RawData model is similar to Image in

that it contains the file path or URL for a GeoTIFF which was the originating source of the information and is stored for further processing or download by authorized users.

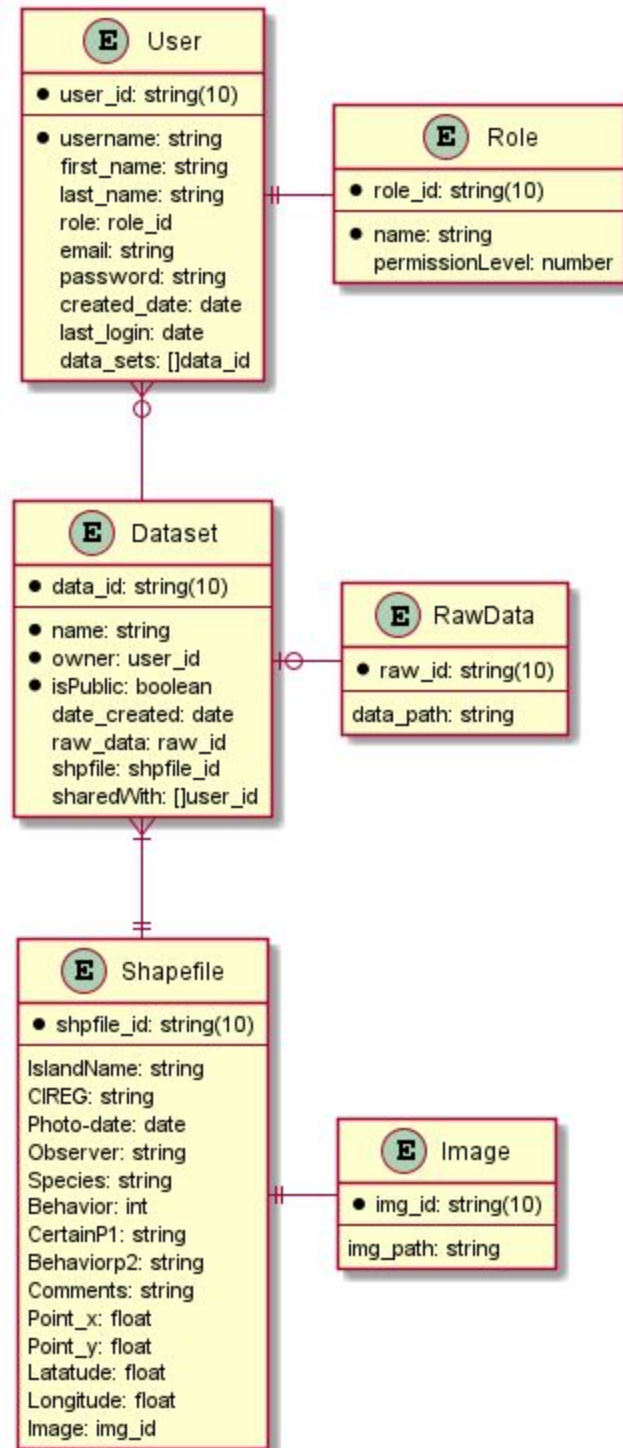


Figure 4. Database Description

## Design Problems

While there were a number of design problems that were addressed in the earlier stages of development, there are a few problems that remain open issues at this point in BirdSpotter's development process. The largest outstanding issue is the method of queuing processing jobs with the ACG's compute service, as the interface for this is still somewhat undefined, as the AI is still in development, and so the scripting to run jobs on the ACG's cluster has not been created yet, and the method of queuing these run scripts is still unclear. As the AI development progresses, the script and its parameters will need to be acquired from the client. The method of running these scripts will need to be determined via communication with both the AI team and the ACG.

## Design Details

Specifics of the design for the BirdSpotter app are outlined in greater detail in the UIDD document. The UIDD provides a detailed description of the design used in the user interface as well as a walkthrough of the interface and its modules. Furthermore, details are provided for the data items within the system as well as the formats of non visual data produced by the application.

## User Interface

The user interface has been designed with the user flow in mind to better the user experience. Each node labelled “\_view” represents a screen within the application.

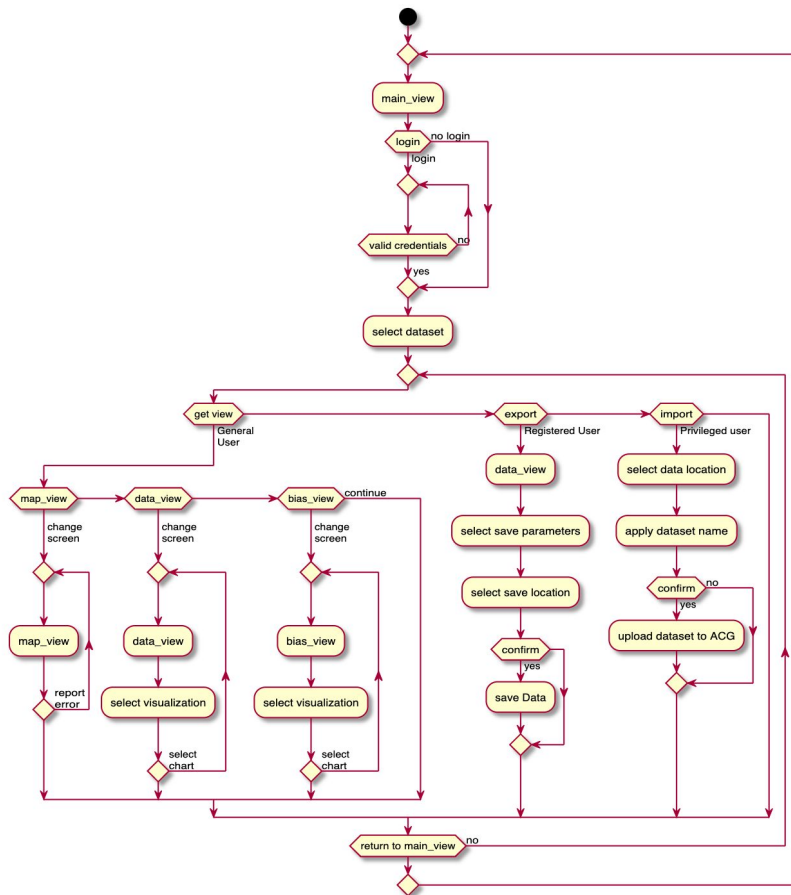


Figure 5: Navigation View Diagram



# Equipment

The equipment is the resources that the team is using to best implement a solution to the provided problem. Python was specified as the desired language for the application as such that leads to the use of libraries to implement features quickly and efficiently. Below are a select few that form the backbone of the application along with the current solution for hosting the application and its related services.

## Advanced Computing Group(ACG)

The Advanced Computing Group is an organization at the University of Maine that provides computing infrastructure not only to the University, but also to research for the state of Maine. This project will be hosted on a virtual machine hosted on ACG servers, along with the possibility of a file vault service for the larger files. The machine learning

## Libraries

We will be using several python libraries to facilitate the development of the BirdSpotter app, which are Django, Plotly, and GeoPandas. We are also using all of the packages that are required to run all of these docs, which can be found on the pages for any of these libraries.

Django - We are using Django as our web framework and to get the website working. It contains functions for various types of HTTP responses as well as having templates with which we can format the BirdSpotter app.

GeoPandas - GeoPandas is the library that we will be using to handle all of the shapefiles for BirdSpotter. It allows for easy opening and modification of shapefiles, which should make the management of the seabird data significantly easier.

Plotly - Plotly can work directly with GeoPandas to display the data that is stored in shapefiles on a map, or to simply parse the data and display it in a simpler graph to show raw numbers.

## Tests

The test plan is designed to ensure quality at multiple levels by testing the low-level functionality of the code, the interaction of the different modules. It also is designed to prevent regressions, and ensure that the product meets the given requirements. There are different types of tests for each of these needs: unit tests, integration tests, regression tests, and acceptance tests.

## Unit Test Plan

The unit test plan is intended to ensure code quality and correctness. Unit tests will be developed alongside the system code and should aim to test each part of the code independently. While some unit tests will verify pieces of system requirements directly, it is necessary to write lower level unit tests than those that are required for system requirements verification, as it is essential to verify the expected functionality of these lower level modules, to avoid bugs down the line.

Further, whenever a new pull request is opened for the master version branch or the active development branch, the continuous integration pipeline will run all unit tests for all modules where changes have been made. This ensures that the changes do not cause any conflicts with existing logic at a unit test level while keeping the overhead to a minimum. The full set of unit tests will be run at the end of each sprint as part of the regression testing plan, in order to catch any regressions before the sprint is completed, ensuring that technical debt due to coding errors are caught before the development team moves on to the next sprint.

## Integration Test Plan

Integration tests are essential to ensure that the different units of the system are functioning together properly. These tests do not verify system-level functional requirements, but some of these tests are based on the system-level requirements, in order to verify that the business logic tested by the unit tests works together correctly as a system.

Integration tests will be written and updated as system-level features are completed, and the full suite of integration tests will be run at the end of each sprint. These tests are run less frequently and written at the end of feature development because they are much more complex to write, run, and debug. Developing and using these tests more frequently would encourage them to be less thorough, and add additional overhead.

## Acceptance Test Plan

The acceptance testing plan is two-fold: user acceptance testing and operational acceptance testing. User acceptance testing ensures that the client is satisfied that the system conforms to the requirements, while the operational acceptance test confirms the system conforms to the non-functional requirements.

User acceptance testing primarily will take place near the end of development, with a completed version of the application. The users will use the application and answer a series of questions relating to each of the functional requirements, as well as the usability of the application in general. Here they will be assessing whether the application meets each of the functional requirements as a whole, as well as whether they are satisfied with the application's usability.



Operational testing will be performed near the end of development, before user acceptance testing. Operational testing will verify the non-functional requirements, via stability, performance, and security tests. These security tests will be fairly basic but will include fuzzing and static code analysis. Fuzzing involves inputting random data into all of the application inputs.

## Analysis

This section refers to the current state of the BirdSpotter program, as well as an overview of the plans for future development. BirdSpotter is being designed to facilitate research pertaining to seabird population data for anyone who may have a need for access to this data. Visualization will be done on both a map view and with standard graphical data displays. We have also decided on several levels of access for the sake of restricting data that could potentially negatively affect seabird habitats. Further notes on the design process of the BirdSpotter app are outlined in the Design section near the beginning of the doc.

## Visualization

The Visualization for the BirdSpotter app is in progress and has the beginnings of most of the proposed views as well as an outline for a navigation bar on every page of the app. On the main page, there is a placeholder option for login, not yet implemented fully, or implemented on the other pages in the current state of the app. The main screen will allow for dataset selection for viewing and, if you are a Registered User, Exporting. Figures pertaining to the design of the visual aspects of BirdSpotter are displayed in the Appendix - UIDD.

## Accounts

Accounts are a necessity to deal with varying permissions that may not be desired for all users. The functions that are to be restricted behind account type are Import/Export data. Export is restricted to Registered Users, as it is possible that someone could look at the available data and particularly the Lat/Long data to disturb the birds that have been logged by a dataset. Additionally, the Lat/Long data should be unavailable or slightly masked to Public Users for the same reason. Import data is restricted to Privileged Users, one step above Registered Users, because the type of data is difficult to collect, and we do not want data that does not relate to nesting birds to mix in with the BirdSpotter app.

## Progress

The BirdSpotter app has some progress in the Main View, Map View, Import View, and login section, as mentioned in the Visualization section, as well as a placeholder in the Data View. We have decided on the libraries that are going to be used to develop the app,

notably Django, Plotly, GeoPandas, as well as all of the prerequisite libraries to them. There are still some issues to be resolved, such as getting access to UMaine's AGC, Deciding definitively on licensing, and setting up an account management framework. Details on each of these sections are elaborated on in the Design section.

## Conclusions

The BirdSpotter app is targeted towards biologists in the field of nesting birds, specifically sea birds. The application works in tandem with an AI that assists users in identifying species of birds from a collection of images gathered via drone or airplane. It is designed with an emphasis on data presentation and the organization of group data collection.

## References

Slatin, J., White, J., Slatin, J., Caldwell, B., Guarino, L., Vanderheiden, G., & Chisholm, W. (2018, June 5). Web Content Accessibility Guidelines (WCAG) 2.1. Retrieved December 16, 2020, from <https://www.w3.org/TR/WCAG21/>

## Acknowledgments

The development of the BirdSpotter has only been possible with the guidance and assistance of several great groups. We would like to thank Dr. Cynthia Loftin, Dr. Roy Turner, and all of the others working with them for granting us the opportunity to work on this project in the first place. We would also like to thank professor Yoo for reaching out to this year's clients and helping to provide all of us with valuable project development and management experience. Finally, we have to thank the University of Maine and the Advanced Computing group for providing us with resources with which we can facilitate the development and deployment of the BirdSpotter project.

# Appendix

# COS 397: Computer Science Capstone I

---

## **System Requirements Specification Document**

(Adapted from Susan Mitchell and Michael Grasso)



## **BirdSpotter**

### **System Requirements Specification**

Developed for: Dr. Cynthia Loftin

Developed by: Penobscot Development Group

## **Table of Contents**

	<u>Page</u>
<b>1. Introduction</b>	<b>4</b>
<b>1.1 Purpose of This Document</b>	
<b>1.2 References</b>	
<b>1.3 Purpose of the Product</b>	
<b>1.4 Product Scope</b>	
<b>2. Functional Requirements</b>	<b>6</b>
<b>3. Non-Functional Requirements</b>	<b>17</b>
<b>4. User Interface</b>	<b>20</b>
<b>5. Deliverables</b>	<b>21</b>
<b>6. Open Issues</b>	<b>21</b>
<b>Appendix A – Agreement Between Customer and Contractor</b>	<b>23</b>
<b>Appendix B – Team Review Sign-off</b>	<b>24</b>
<b>Appendix C – Document Contributions</b>	<b>25</b>

## 1. Introduction

### 1.1 Purpose of This Document

This SRS document provides information on the requirements and specifications associated with the BirdSpotter application. The goal of this document is to give an overview of the various functionalities and components of the application. Additionally, a synopsis of the artifacts delivered to the customer is provided. This document is intended for developers and the client.

### 1.2 References

User Story document, referenced in the requirements section:  
<https://airtable.com/shrHLgtG7XOBeyVpv>

### 1.3 Purpose of the Product

The BirdSpotter application is a bird tracking app that helps manage aerial imagery data used for bird counts, and allows the user to apply AI algorithms to interpret and report species identification and certainty estimates of the identification. The app enables a user to display geographical data on a map or organize statistical data in various forms such as graphs or tables.

### 1.4 Product Scope

The Bird Spotter app facilitates breeding bird counts in aerial imagery by integrating a machine learning process with a centralized platform for data storage and visualization. The goal of the app is to provide a centralized platform with data storing, visualization, and streamlined statistical functionalities. These are the key features of the application and make up the core of the program. The main audience for the application is biologists involved in research related to birds.

### 1.5 User Definitions

There are four levels of users defined by the requirements in this document. Each user is completely defined by the functional requirements below, so the following list is only provided to add more conveniently accessible context. Note that the users are organized in a linear hierarchy, meaning that each level of permission includes the permissions of each level below it. For example, a registered user has all the permissions a public user has, with additional registered user specific permissions.

#### Public User:

A public user represents a user who has not been granted any permissions by the site administrators. They only have the ability to view a low-resolution version of the results of analysis (map view with low resolution zoom & only aggregate datapoints),

with no export abilities for the data or the analysis. Additionally, they can only view data sets that are not marked as private and do not have access to the resultant statistics.

#### Registered User:

A registered user represents a user who has been granted an account. They are required to log into their account before they can access their elevated permissions. A registered user can view private data that they are granted access to. They also have the ability to export and download data sets they can view.

#### Privileged User:

Like a registered user, a privileged user must log in to access their elevated privileges. A privileged user has the additional ability to upload new data sets and edit the metadata of the sets that they own.

#### Administrator:

A site administrator is the most elevated role and should only be granted to internal, trusted users. A site administrator has the ability to view all data sets, private and public, and to edit any data set owned by any user. They also have the ability to edit the user privileges of all other non-administrator users.

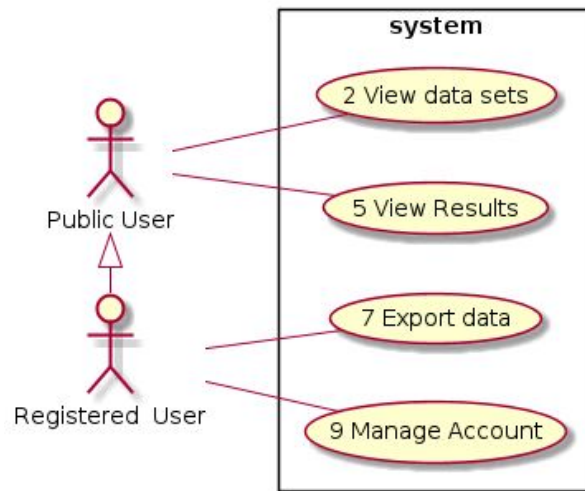


Figure 1. Use Cases: Part One



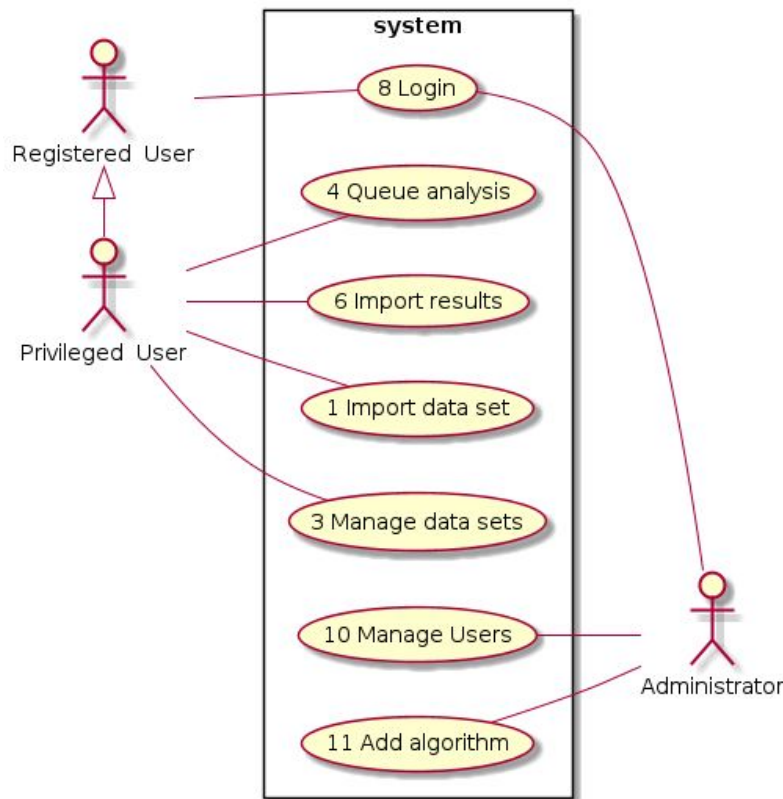


Figure 2. Use Cases: Part two

## 2. Functional Requirements

<b>Number</b>	1.0	
<b>Name</b>	Import data set	
<b>User Story ID(s)</b>	1, 30	
<b>Summary</b>	Upload your own data set for analysis	
<b>Priority</b>	5	
<b>Preconditions</b>	Successful Login	
<b>Postconditions</b>	Data set will be available in the system to analyze	
<b>Primary Actor</b>	Privileged User	
<b>Secondary Actors</b>	Client system	
<b>Trigger</b>	User selects "Import data set" menu option	
<b>Main Scenario</b>	<b>Step</b>	<b>Action</b>
	1	Privileged User selects "Upload new data set"
	2	System displays displays an upload prompt to choose a file
	3	Privileged User selects desired file
<b>Extensions</b>	<b>Step</b>	<b>Branching Action</b>

	4a	Privileged User selects “make this data set visible to all users”: System adds data set to internal data repository
	4b	Privileged User selects “make lower-resolution data open to the public”: System adds data set to public data repository
	4c	Privileged User selects “don’t share this data”: System adds data set to user-specific data repository
<b>Open Issues</b>		

<b>Number</b>	2.0	
<b>Name</b>	View Data	
<b>User Story ID(s)</b>	21	
<b>Summary</b>	View list of data sets available	
<b>Priority</b>	5	
<b>Preconditions</b>	None.	
<b>Postconditions</b>	List of data sets is displayed, the user can perform actions on them	
<b>Primary Actor</b>	Public User, Registered User	
<b>Secondary Actors</b>	None	
<b>Trigger</b>	User loads website	
<b>Main Scenario</b>	<b>Step</b>	<b>Action</b>
	1	System displays list of data sets
<b>Extensions</b>	<b>Step</b>	<b>Branching Action</b>
	2a	Public User : System displays additional metadata for data sets, such as data entered, general location, species of focus
	2a	Registered User : System displays additional metadata for data sets, such as data entered, exact, species of focus, observer(s), data set owner
<b>Open Issues</b>		

<b>Number</b>	2.1	
<b>Name</b>	Sort/Filter Data	
<b>User Story ID(s)</b>	19	
<b>Summary</b>	Sort, filter	
<b>Priority</b>	3	
<b>Preconditions</b>	User is viewing data set list (Use case 2.0)	
<b>Postconditions</b>	None.	
<b>Primary Actor</b>	Registered User	
<b>Secondary Actors</b>	None.	
<b>Trigger</b>	User sorts by a column, selects a specific data set requirement, or enters a search term	
<b>Main Scenario</b>	<b>Step</b>	<b>Action</b>
	1	Registered User sorts by a column, selects a specific data set requirement, or enters a search term

	2	System displays newly-limited data set list matching user requirements or keywords
<b>Open Issues</b>		

<b>Number</b>	3.0	
<b>Name</b>	Manage Data	
<b>User Story ID(s)</b>		
<b>Summary</b>	Modify data set permissions and other meta-data	
<b>Priority</b>	4	
<b>Preconditions</b>	Privileged User is viewing data sets (Use case 2.0)	
<b>Postconditions</b>	Data set metadata is modified	
<b>Primary Actor</b>	Privileged User	
<b>Secondary Actors</b>	None.	
<b>Trigger</b>	Privileged User selects data set to modify	
<b>Main Scenario</b>	<b>Step</b>	<b>Action</b>
	1	Privileged User selects data set to modify
<b>Extensions</b>	<b>Step</b>	<b>Branching Action</b>
	2a	Privileged User is not the owner (uploader) of the selected data set: System displays “permission denied” message. System returns to data set view.
	3a	Privileged User selects “Add meta-data”: Proceed to Use case 3.1
	3b	Privileged User selects “Change visibility”: Proceed to Use case 3.2
<b>Open Issues</b>		

<b>Number</b>	3.1	
<b>Name</b>	Add Metadata	
<b>User Story ID(s)</b>	8	
<b>Summary</b>	Add additional relevant metadata to the data(collection method, notes, etc)	
<b>Priority</b>	4	
<b>Preconditions</b>	Privileged User is viewing data set list (Use case 2.0), User has selected data set (Use case 3.0), User is owner (uploader) of the data set	
<b>Postconditions</b>	Data set metadata is modified	
<b>Primary Actor</b>	Privileged User	
<b>Secondary Actors</b>	None	
<b>Trigger</b>	Privileged User selects “Add meta-data”	
<b>Main Scenario</b>	<b>Step</b>	<b>Action</b>
	1	Privileged User selects “Add meta-data”
	2	System displays current meta-data (if any) in as editable
	3	Privileged User edits meta-data
<b>Extensions</b>	<b>Step</b>	<b>Branching Action</b>
	4a	Privileged User selects “Save”: System overwrites old meta-data with new meta-data

		System closes meta-data and returns to data set view
	4b	Privileged User selects “Cancel”: System closes meta-data and returns to data set view
<b>Open Issues</b>		

<b>Number</b>	3.2	
<b>Name</b>	Manage Visibility of Data	
<b>User Story ID(s)</b>	18	
<b>Summary</b>	Modify the visibility of data sets the owner has uploaded	
<b>Priority</b>	4	
<b>Preconditions</b>	Privileged User is viewing data sets (Use case 2.0), User has selected data set (Use case 3.0), User is owner (uploader) of the data set	
<b>Postconditions</b>	Data set	
<b>Primary Actor</b>	Privileged User	
<b>Secondary Actors</b>	None	
<b>Trigger</b>	Privileged User selects “Change visibility”	
<b>Main Scenario</b>	<b>Step</b>	<b>Action</b>
	1	Privileged User selects “Change visibility”
	2	System displays 3 options (public, internal, only you)
	3	Privileged User selects an option.
	4	System prompts user to confirm
<b>Extensions</b>	<b>Step</b>	<b>Branching Action</b>
	5a	User selects “confirm”: System changes visibility based on user’s selection
	5b	User selects “cancel”: System returns to Data set view (Use case 2.0)
<b>Open Issues</b>		

<b>Number</b>	4.0	
<b>Name</b>	Queue Analysis	
<b>User Story ID(s)</b>	15	
<b>Summary</b>	Choose algorithm and run against previously specified data set	
<b>Priority</b>	5	
<b>Preconditions</b>	Successful Login	
<b>Postconditions</b>	Data set analysis is registered on the ACG Queue	
<b>Primary Actor</b>	Privileged User	
<b>Secondary Actors</b>	ACG	
<b>Trigger</b>	User selects the “Queue Analysis” menu option	
<b>Main Scenario</b>	<b>Step</b>	<b>Action</b>
	1	System displays list of available algorithms
	2	Privileged User selects desired algorithm
	3	System displays displays additional info about chosen algorithm with confirm/cancel buttons
<b>Extensions</b>	<b>Step</b>	<b>Branching Action</b>

	4a	Privileged User selects “confirm”: System queues data set to be analyzed with chosen algorithm
	4b	Privileged User selects “cancel”: Revert to step 1.
<b>Open Issues</b>	2	

<b>Number</b>	5.0	
<b>Name</b>	View Results	
<b>User Story ID(s)</b>	7	
<b>Summary</b>	View visualizations of the results of analysis of data set	
<b>Priority</b>	5	
<b>Preconditions</b>	Successful Login, User has selected data set (Use case 3.0), selected data set has associated analysis data (Use case 6.0)	
<b>Postconditions</b>	None.	
<b>Primary Actor</b>	Public User, Registered User	
<b>Secondary Actors</b>	None.	
<b>Trigger</b>	User selects “View completed analysis” menu option	
<b>Main Scenario</b>	<b>Step</b>	<b>Action</b>
	1	User selects “View completed analysis”
	2	System displays view options
<b>Extensions</b>	<b>Step</b>	<b>Branching Action</b>
	3a	User selects “View Map”: Proceed to Use case 3.2
	3b	User selects “View Charts and Aggregates”: Proceed to Use case 3.3
<b>Open Issues</b>	5	

<b>Number</b>	5.1	
<b>Name</b>	View Charts and Aggregates	
<b>User Story ID(s)</b>	11, 13, 9, 24	
<b>Summary</b>	View the aggregate data (certainty, number of data points, etc)	
<b>Priority</b>	4	
<b>Preconditions</b>	Successful Login	
<b>Postconditions</b>	Chart view is displayed	
<b>Primary Actor</b>	Public User, Registered User	
<b>Secondary Actors</b>	None.	
<b>Trigger</b>	User selects “Charts and Aggregates” menu option	
<b>Main Scenario</b>	<b>Step</b>	<b>Action</b>
	1	User selects “Charts and Aggregates”
<b>Extensions</b>	<b>Step</b>	<b>Branching Action</b>
	2	Registered User:

		System displays overall statistics by nesting type, species and island or specific geolocation, including certainty
<b>Open Issues</b>	5	

<b>Number</b>	5.2	
<b>Name</b>	View Map	
<b>User Story ID(s)</b>	12, 13, 10, 22, 24	
<b>Summary</b>	View geodata on a map, with the ability to re-color and rescale the map	
<b>Priority</b>	5	
<b>Preconditions</b>	Successful Login	
<b>Postconditions</b>	Map view is displayed	
<b>Primary Actor</b>	Public User, Registered User	
<b>Secondary Actors</b>	None.	
<b>Trigger</b>	User selects “Map View” menu option	
<b>Main Scenario</b>	<b>Step</b>	<b>Action</b>
	1	User selects “Map View”
<b>Extensions</b>	<b>Step</b>	<b>Branching Action</b>
	2a	Public User: System displays lower-resolution map (data points represent larger geographical areas) System displays general geographical statistics
	2b	Registered User: System displays detailed map (up to 1 datapoint per bird) System displays fully detailed geographical statistics
	3	System displays menu to re-color and re-scale map
<b>Open Issues</b>	3, 5	

<b>Number</b>	5.3	
<b>Name</b>	View Details by Data Point	
<b>User Story ID(s)</b>	6, 11, 32, 24	
<b>Summary</b>	View statistics, location, source photos, AI categorization and certainty by datapoint	
<b>Priority</b>	4	
<b>Preconditions</b>	User is viewing map (Use case 5.2)	
<b>Postconditions</b>	None.	
<b>Primary Actor</b>	Public User, Registered User	
<b>Secondary Actors</b>	None.	
<b>Trigger</b>	User interacts with datapoint on map	
<b>Main Scenario</b>	<b>Step</b>	<b>Action</b>
	1	User interacts with datapoint
<b>Extensions</b>	<b>Step</b>	<b>Branching Action</b>
	2a	Public User: System displays statistics for the aggregate data point
	2b	Registered User:

		System displays statistics and exact location for that datapoint System displays source image for that datapoint, if available
<b>Open Issues</b>	5	

<b>Number</b>	5.4	
<b>Name</b>	Aggregate data points per-island	
<b>User Story ID(s)</b>	11, 13, 32	
<b>Summary</b>	System displays a datapoint per island (instead of proximity or per-bird)	
<b>Priority</b>	4	
<b>Preconditions</b>	User is viewing map (Use case 5.2)	
<b>Postconditions</b>	Map view is displayed with a single point over the island per set.	
<b>Primary Actor</b>	Registered User	
<b>Secondary Actors</b>	None.	
<b>Trigger</b>	User selects the “Aggregate by Island” toggle option	
<b>Main Scenario</b>	<b>Step</b>	
	1	User selects the “Aggregate by Island”
	2	System displays a single datapoint per-island, sizing datapoints by population
<b>Open Issues</b>		

<b>Number</b>	6.0	
<b>Name</b>	Import Results	
<b>User Story ID(s)</b>	20	
<b>Summary</b>	Results of analysis are associated with a data set	
<b>Priority</b>	5	
<b>Preconditions</b>	Login. User is viewing data sets (Use case 2.0)	
<b>Postconditions</b>	Results are associated with a data set and can be viewed	
<b>Primary Actor</b>	Privileged User	
<b>Secondary Actors</b>	None.	
<b>Trigger</b>	Privileged User selects “Upload data” menu option	
<b>Main Scenario</b>	<b>Step</b>	<b>Action</b>
	1	Privileged User selects data set that their results are associated with
	2	Privileged User selects “Upload data”
	3	System displays displays an upload prompt to choose a file
	4	Privileged User selects desired file
	5	System imports results and associated them with the chosen data set
<b>Open Issues</b>		

<b>Number</b>	7.0	
<b>Name</b>	Export Data	
<b>User Story ID(s)</b>	5, 17, 33	
<b>Summary</b>	Data is exported on to the users computer	

<b>Priority</b>	5	
<b>Preconditions</b>	Successful Login, User is viewing datasets (Use case 2.0)	
<b>Postconditions</b>	Data is exported to the users computer	
<b>Primary Actor</b>	Registered User	
<b>Secondary Actors</b>	ACG	
<b>Trigger</b>	Registered User selects “Export Data”	
<b>Main Scenario</b>	<b>Step</b>	
	1	Registered User selects a data set.
	2	Registered User selects “Export Data”
	3	System displays export options (export geodatabase, statistical data, or analysis script)
<b>Extensions</b>	<b>Step</b>	
	3a	Registered User selects “Export geodatabase”: System exports geodatabase to client machine
	3b	Registered User selects “Export statistical data”: System exports statistical data to client machine
	3c	Registered User selects “Generate and Export analysis script”: System displays list of available algorithms User selects algorithm System exports generated analysis script to client machine
<b>Open Issues</b>		

<b>Number</b>	8.0	
<b>Name</b>	Login	
<b>User Story ID(s)</b>	2	
<b>Summary</b>	Registered User logs in	
<b>Priority</b>	5	
<b>Preconditions</b>		
<b>Postconditions</b>	Registered User will have access to the functions of their account	
<b>Primary Actor</b>	Registered User	
<b>Secondary Actors</b>	ACG	
<b>Trigger</b>	Registered user selects login	
<b>Main Scenario</b>	<b>Step</b>	
	1	Registered User selects “log in”
	2b	Registered User enters username password pair
	2b	Registered User enters invalid username password pair
	3a	Registered User is granted access to their account
	3b	Registered User is not granted access to their account
<b>Open Issues</b>	1	

<b>Number</b>	9.0	
<b>Name</b>	Manage Account	
<b>User Story ID(s)</b>	25, 26	



<b>Summary</b>	Manage Account Settings	
<b>Priority</b>	4	
<b>Preconditions</b>	Successful login	
<b>Postconditions</b>	System updates user account information	
<b>Primary Actor</b>	Registered user	
<b>Secondary Actors</b>	None.	
<b>Trigger</b>	Registered user selects edit information on account page	
<b>Main Scenario</b>	<b>Step</b>	
	1	Registered User selects Account tab
	2	Registered User clicks “Edit info” button
	3	Registered User changes desired information
	4	Registered User select “Done” button
<b>Open Issues</b>	1	

<b>Number</b>	10.0	
<b>Name</b>	Manage Users	
<b>User Story ID(s)</b>	27, 29	
<b>Summary</b>	Modify user settings	
<b>Priority</b>	5	
<b>Preconditions</b>	Successful login	
<b>Postconditions</b>	System displays user management screen	
<b>Primary Actor</b>	Administrator	
<b>Secondary Actors</b>	ACG	
<b>Trigger</b>	Administrator selects “Manage Users”	
<b>Main Scenario</b>	<b>Step</b>	<b>Action</b>
	1	Administrator selects “Manage Users”
	2	Systems displays user list
<b>Extensions</b>	<b>Step</b>	<b>Branching Action</b>
	1a	Administrator clicks “Add user” : Proceed to use case 10.1
	1b	Administrator clicks delete icon for desired user: Proceed to use case 10.2
	1c	Administrator clicks edit icon for desired user: Proceed to use case 10.3
<b>Open Issues</b>	1	

<b>Number</b>	10.1	
<b>Name</b>	Add User	
<b>User Story ID(s)</b>	2, 34	
<b>Summary</b>	Adds a user to the system	
<b>Priority</b>	5	
<b>Preconditions</b>	Administrator is logged in	

<b>Postconditions</b>	A user is added to the system	
<b>Primary Actor</b>	Administrator	
<b>Secondary Actors</b>	ACG	
<b>Trigger</b>	Administrator selects “Add User”	
<b>Main Scenario</b>	<b>Step</b>	<b>Action</b>
	1	Administrator selects “Add User”
	2	A prompt appears asking for an email address
	3	Administrator enters a valid email address
<b>Open Issues</b>	1	

<b>Number</b>	10.2	
<b>Name</b>	Remove User	
<b>User Story ID(s)</b>	2	
<b>Summary</b>	Remove a user from the system	
<b>Priority</b>	5	
<b>Preconditions</b>	Administrator is logged in	
<b>Postconditions</b>	A user is removed from the system.	
<b>Primary Actor</b>	Administrator	
<b>Secondary Actors</b>	ACG	
<b>Trigger</b>	Administrator selects “Remove User”	
<b>Main Scenario</b>	<b>Step</b>	<b>Action</b>
	1	Administrator clicks delete icon for desired user
	2	System displays confirmation dialog
	2a	Administrator clicks “Confirm” button
	2b	Administrator clicks “Cancel” button
<b>Open Issues</b>	1	

<b>Number</b>	10.3	
<b>Name</b>	List Accounts	
<b>User Story ID(s)</b>	27	
<b>Summary</b>	View the accounts logged within the system	
<b>Priority</b>	5	
<b>Preconditions</b>	Successful login	
<b>Postconditions</b>	A list of accounts appears	
<b>Primary Actor</b>	Administrator	
<b>Secondary Actors</b>	ACG	
<b>Trigger</b>	Administrator clicks “List Accounts”	
<b>Main Scenario</b>	<b>Step</b>	<b>Action</b>
	1	Administrator clicks “List Accounts”
	2	System displays a list of accounts
<b>Open Issues</b>		

<b>Number</b>	10.4	
<b>Name</b>	Assign Privileges	
<b>User Story ID(s)</b>	29	
<b>Summary</b>	Assign a user privileged access to the application	
<b>Priority</b>	5	
<b>Preconditions</b>	Successful login	
<b>Postconditions</b>	User is assigned privileged access	
<b>Primary Actor</b>	Administrator	
<b>Secondary Actors</b>	ACG	
<b>Trigger</b>	Administrator selects “Set Access Level”	
<b>Main Scenario</b>	<b>Step</b>	
	1	Administrator selects “Set Access Level”
	2	System displays a prompt with a dropdown containing levels
	3	Administrator selects “Privileged”
	4	Administrator selects “Apply”
	5	User is assigned “Privileged” access
<b>Extensions</b>	<b>Step</b>	
	3a	Administrator selects “Administrator”
	4a	Administrator selects “Apply”
	5a	User is assigned “Administrator” access
<b>Open Issues</b>		

<b>Number</b>	11.0	
<b>Name</b>	Add Analysis Script	
<b>User Story ID(s)</b>	16	
<b>Summary</b>	Upload an analysis script matching a predefined format to be available to users as an analysis method	
<b>Priority</b>	3	
<b>Preconditions</b>	Successful login	
<b>Postconditions</b>	Analysis script is available for use	
<b>Primary Actor</b>	Administrator	
<b>Secondary Actors</b>	ACG	
<b>Trigger</b>	Administrator selects “Upload Analysis Script” option	
<b>Main Scenario</b>	<b>Step</b>	<b>Action</b>
	1	Administrator selects “Upload Analysis Script” option
	2	System displays an upload prompt
	3	Administrator selects Analysis Script file
	4	System adds Analysis Script
<b>Open Issues</b>		

### Testing

1. Import raw data in geotiff format and ensure data is saved into the database

2. Upload numerous raw data files and ensure that they show up in the user's data section
3.
  - 3.1. Change the visibility of a data object from private to public and ensure proper visibility
  - 3.2. Change the visibility of a data object from public to private and ensure proper visibility
  - 3.3. Perform a delete action on the data and ensure raw data and results are removed from the system
4. Select the "Perform analysis" option and ensure that the proper script is generated and sent to the ACG server
5. Ensure that selecting a data object that has already had analysis performed on it displays correctly in both the data overview and map views
6. Upload data using a shape file, and ensure that it is saved correctly as results in a data object
7. For a data object that has already had analysis performed on it, select the "Export results" option and verify that the corresponding shape file is exported
8. Perform a login attempt with a registered account providing the correct credentials and ensure that the user is redirected to the homepage
9.
  - 9.1. Perform edit actions on all user data fields and ensure that the changes are stored upon pressing the "Done" button
  - 9.2. Perform edit actions on all user data fields and ensure that changes are not stored upon pressing the "Edit" button
10.
  - 10.1. Ensure that adding a user from the admin page creates a user in the system and sends the corresponding email address a notification along with a link to a create password page
  - 10.2. Ensure that removing an existing user from the system removes access to the system along with all data associated with that account
  - 10.3. Upon selecting "List Accounts" ensure that all accounts in the system are displayed
  - 10.4. For an account that exists in the system, ensure that changing the privilege for an account is reflected in the system and the user can view data according to their privilege level

### 3. Non-Functional Requirements

<b>NFR #</b>	1
<b>Name</b>	Sign in
<b>Priority</b>	5
<b>Description</b>	A user will reach the home page in less than 2 seconds from pressing the submit button or enter key when the correct credentials are provided 95% of the time

<b>Tests</b>	Sign in using correct credentials and measure response time from the system
--------------	---

<b>NFR #</b>	2
<b>Name</b>	Display available data sets
<b>Priority</b>	3
<b>Description</b>	The system shall display all data sets available to the user within 5 seconds 85% of the time
<b>Tests</b>	Open the UI for selecting a data set and measure the time taken to completely display all available data sets

The project description is well-written and

<b>NFR #</b>	3
<b>Name</b>	Upload data sets
<b>Priority</b>	4
<b>Description</b>	The system shall import a data set provided by the user in less than 10 seconds 80% of the time.
<b>Tests</b>	Import data sets of varying size and measure response time

<b>NFR #</b>	4
<b>Name</b>	Export data sets
<b>Priority</b>	3
<b>Description</b>	The system shall export a specified data set in within 5 seconds 90% of the time
<b>Tests</b>	Export data sets of varying sizes and measure the response time of the system

<b>NFR #</b>	5
<b>Name</b>	Display data overview

<b>Priority</b>	5
<b>Description</b>	The system shall display an overview of that data with graphics in less that 5 seconds of the user switching to the display tab 90% of the time
<b>Tests</b>	Select large data sets and measure time taken to parse data, create, and display charts

<b>NFR #</b>	6
<b>Name</b>	Display map with data points
<b>Priority</b>	5
<b>Description</b>	The system shall display the map overview with bird locations in less than 10 seconds 80% of the time
<b>Tests</b>	Measure time taken to render map and bird locations for varying sized data sets

<b>NFR #</b>	7
<b>Name</b>	Registration request notifications
<b>Priority</b>	3
<b>Description</b>	An admin shall receive an email notification for access requests within 5 minutes of the request being submitted 90% of the time
<b>Tests</b>	Create a user request and measure elapsed time until the notification email is received

<b>NFR #</b>	8
<b>Name</b>	Private data sets shall not be visible to other users
<b>Priority</b>	5
<b>Description</b>	The system shall not show data sets to users that do not have explicit access to such a data set unless that user has administrative privileges.
<b>Tests</b>	Make a private data set under one user and ensure that another non-administrative user cannot see said data set.

<b>NFR #</b>	9
<b>Name</b>	Non-registered actors shall not have access to the system
<b>Priority</b>	5
<b>Description</b>	An actor that does not have a registered account shall not be able to view any page other than the sign in page
<b>Tests</b>	Ensure that all endpoints of the system are unable to be accessed using a non-authenticated user.

<b>NFR #</b>	10
<b>Name</b>	Non-registered actors shall not have access to the system
<b>Priority</b>	5
<b>Description</b>	An actor that does not have a registered account shall not be able to view any page other than the sign in page
<b>Tests</b>	Ensure that all endpoints of the system are unable to be accessed using a non-authenticated user.

<b>NFR #</b>	11
<b>Name</b>	Sensitive information zoom behavior
<b>Priority</b>	3
<b>Description</b>	Sensitive information shall not be displayed in any greater detail than a 2 sq. mile aggregation
<b>Tests</b>	Attempt to zoom in on sensitive information with a non-authorized user and check that their view is constrained to a minimum scale of 2 sq. miles.

#### 4. User Interface

See “User Interface Design Document for *BirdSpotter*.”

## 5. Deliverables

Item	Date	Format
Systems Requirement Specification	10/28/20	PDF
System Design Document	11/10/20	PDF
User Interface Design Document	11/24/20	PDF
User Manual	Spring 2021	PDF/Github
Administrator Manual	Spring 2021	PDF
Source Code	Always available	Github
Prototype	12/2/20	Web link
Executable program	12/2/20	Github
Additional Software	Spring 2021	Github

## 6. Open Issues

1. The type of sign in or account management has not yet been decided, though we have a few levels of access listed in with the user stories document referenced in the 'external resources' section
2. Whether the use of the machine learning can be automatically applied to given data or must be done manually, and related to this is should any user or only privileged users have access to the compute time of the ACG. Currently requirements are written for both options, with the manual method being the higher priority.
3. We have not fully decided on the scope of the data visualisation abilities, and we may later choose to extend our data model and visualization to include a time dimension.
4. The license that the code should be made under has not yet been decided, and should be discussed in the future. Likely open source would be a good choice due to the nature of the application but confirmation would be needed.
5. The library to be used: initially we had planned on using arcGIS because it was previously used for the project, but it appears to come with some licensing costs. We are currently working this out as developers.



6. How the run script works. If it is something that is just changing a couple user specified variables then it can be done automatically by the system, otherwise the user imports a run script that they created and that is used.

## Appendix A – Agreement Between Customer and Contractor

By signing on the provided line below, the client acknowledges and agrees that the deliverables described by the above terms in sections 1, 2, and 3 are satisfactory. The client also accepts the delivery dates for each item described in section 5 and all other clauses described in this document.

Client signature:

Cynthia S. Loftin date: 11/2/2020

Client comments:

Type text here

The project description and functional requirements are well-written and appear complete.

There are some issues that will be resolved as we work through the project with the team. We look forward to working with the team!

By signing on the provided lines below, all Penobscot Development Group members acknowledge and agree to meet all requirements for deliverables described in sections 1, 2, and 3. Additionally, signing indicates that the Penobscot Development Team agrees to deliver all items by their respective dates described in section 5 and to accept all other clauses described in this document.



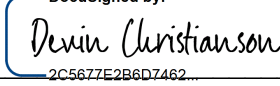
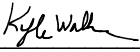

Penobscot Development Team signatures:

<u>Alexander</u>	date: <u>October 28th 2020</u>
<u>Frank Morris</u>	date: <u>October 28th 2020</u>
DocuSigned by: <u>Devin Christianson</u> 2C5677E2B6D7462...	date: <u>October 28th, 2020</u>
<u>Kyle Wall</u>	date: <u>October 28th, 2020</u>
<u>Michael Stein</u>	date: <u>October 28th, 2020</u>

Changes to requirements or dates may be required as the development cycle progresses. All new drafts of this document must be reviewed, agreed upon, and signed by both the Penobscot Development Group and the client. Upon the signing of a new draft, the previous draft becomes void and the new draft supersedes any obligations set by the previous draft.

**Appendix B – Team Review Sign-off**

By signing below, all Penobscot Development Group members acknowledge that they have reviewed all requirements for deliverables described in sections 1, 2, and 3. Additionally, signing indicates that each team member has reviewed the delivery dates described in section 5.

 _____	date: <u>October 28th 2020</u>
 _____	date: <u>October 28th 2020</u>
<div><div>DocuSigned by:</div> 2C5677E2B6D7462...</div> _____	date: <u>October 28th, 2020</u>
 _____	date: <u>October 28th, 2020</u>
 _____	date: <u>October 28th, 2020</u>

## **Appendix C – Document Contributions**

The below table describes the Penobscot Development Group members contribution to the document for internal purposes.

10% - Jacob Morin - Section 1.5, Appendices A, B, C  
40% - Devin Christianson - Section 2, 3, 5  
10% - Alexandre Feren - Section 1.2, Section 2, Section 6  
20% - Kyle Walker - Section 1, Section 2  
20% - Nick Kania - Section 2, Section 3, Section 5

# COS 397: Computer Science Capstone I

---

## **System Design Document**

(Adapted from Susan Mitchell)



BirdSpotter  
System Design Document

**Table of Contents**

	<u>Page</u>
<b>1. Introduction</b>	
<b>1.1 Purpose of This Document</b>	3
<b>1.2 References</b>	3
<b>2. System Architecture</b>	
<b>2.1 Architectural Design</b>	3
<b>2.2 Decomposition Description</b>	4
<b>3. Persistent Data Design</b>	
<b>5.1 Database Descriptions</b>	7
<b>5.2 File Descriptions</b>	8
<b>4. Requirements Matrix</b>	9
<b>Appendix A – Agreement Between Customer and Contractor</b>	12
<b>Appendix B – Peer Review Sign-off</b>	13
<b>Appendix C – Document Contributions</b>	14

## 1. Introduction

### 1.1 Purpose of This Document

The purpose of this document is to define system design of this application. This information is provided to assist the developers in the creation and maintenance of the architecture of the system. The document lays out the system architecture, information organization, and information storage architecture.

### 1.2 References

This document references:

- System Requirement Specification document:  
<https://drive.google.com/file/d/1gdiXsWQ8f2nXvK7SPesUauaKTBTPlzuF/view?usp=sharing>
- User Interface Design Document: To be completed next

## 2. System Architecture

### 2.1 Architectural Design

The logical system architecture follows the “Model View Controller” design pattern, where the model is the stateful part of the system, the view describes what users see, and the controller describes system inputs, outputs, and system manipulation. This design pattern as applied to this system can be seen in Figure 1.

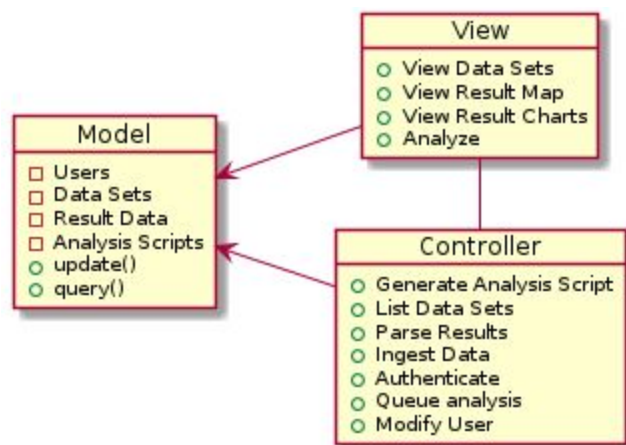


Figure 1. Logical System ArchitectureM

As is shown in Figure 1, the system has three logical components. The first component is the view, which contains the views a user sees such as the list of data sets or the data on a map. The second is the controller, which handles inputs and outputs between the components and the user, and the model, which contains the state of the system, such as the user list, data sets, result data, and analysis scripts.

The technology architecture shows the technological layout of the system, at a software/hardware level. As can be seen in Figure 2, the system has 3 major technological components: the data storage (a relational database), the file storage, the web server, and the ACG computation system.

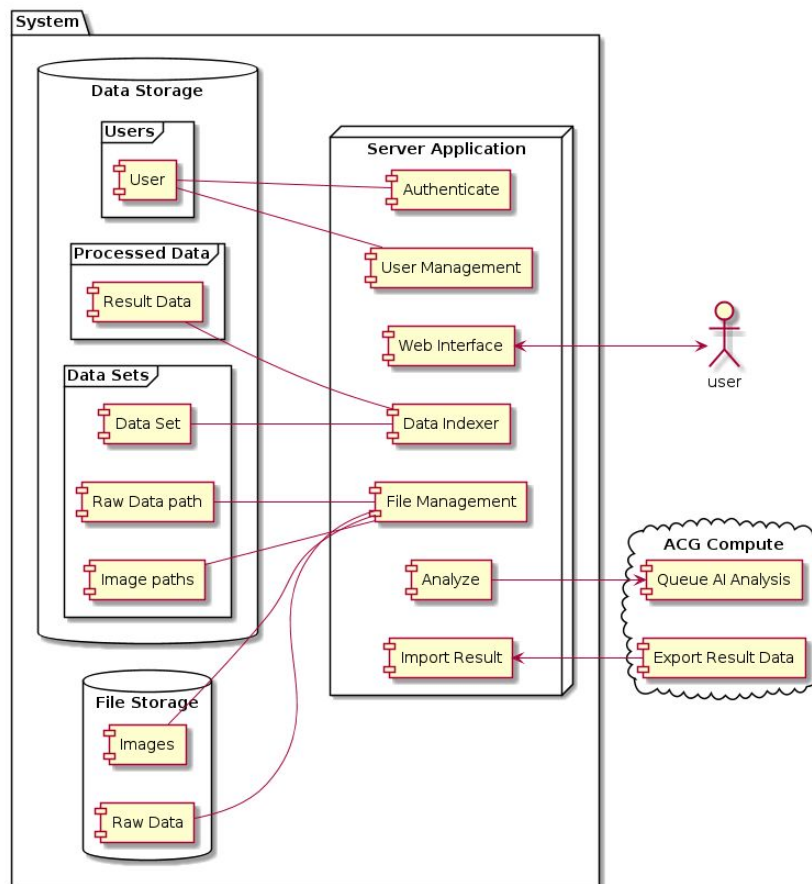


Figure 2. Technological System Architecture

As Figure 2 shows, most of the system model is contained within the storage, while the web server application is the controller, and serves the view to the client via a web interface. Analysis is also sent to an external actor, the ACG, for computation.

## 2.2 Decomposition Description

**Model:** This section is the structure of the application, and consists primarily of the users, input and output datasets, and analysis scripts, which are external to the scope of the COS397 development, but will need to be accessed. The users are split into four separate levels of access: Public, which can only view the visualizations, Registered, which can also download the data, Privileged, which can also upload data and analysis of the data, and finally Administrator, which can also modify other users' permissions as well as adding algorithms for data processing to the backend. Datasets will effectively be the input to the machine learning portion of this project; a compilation of UAV collected imagery. Result Datasets will be a combination of spreadsheets and shapefiles, similar to what is contained in the "Demariscove" folder of the COS397 Seabird



Project folder in google drive. Analysis Scripts will be the machine learning scripts provided by the machine learning portion of the research. Finally update() and query() will update and request data from the Datasets and Result datasets.

## View:

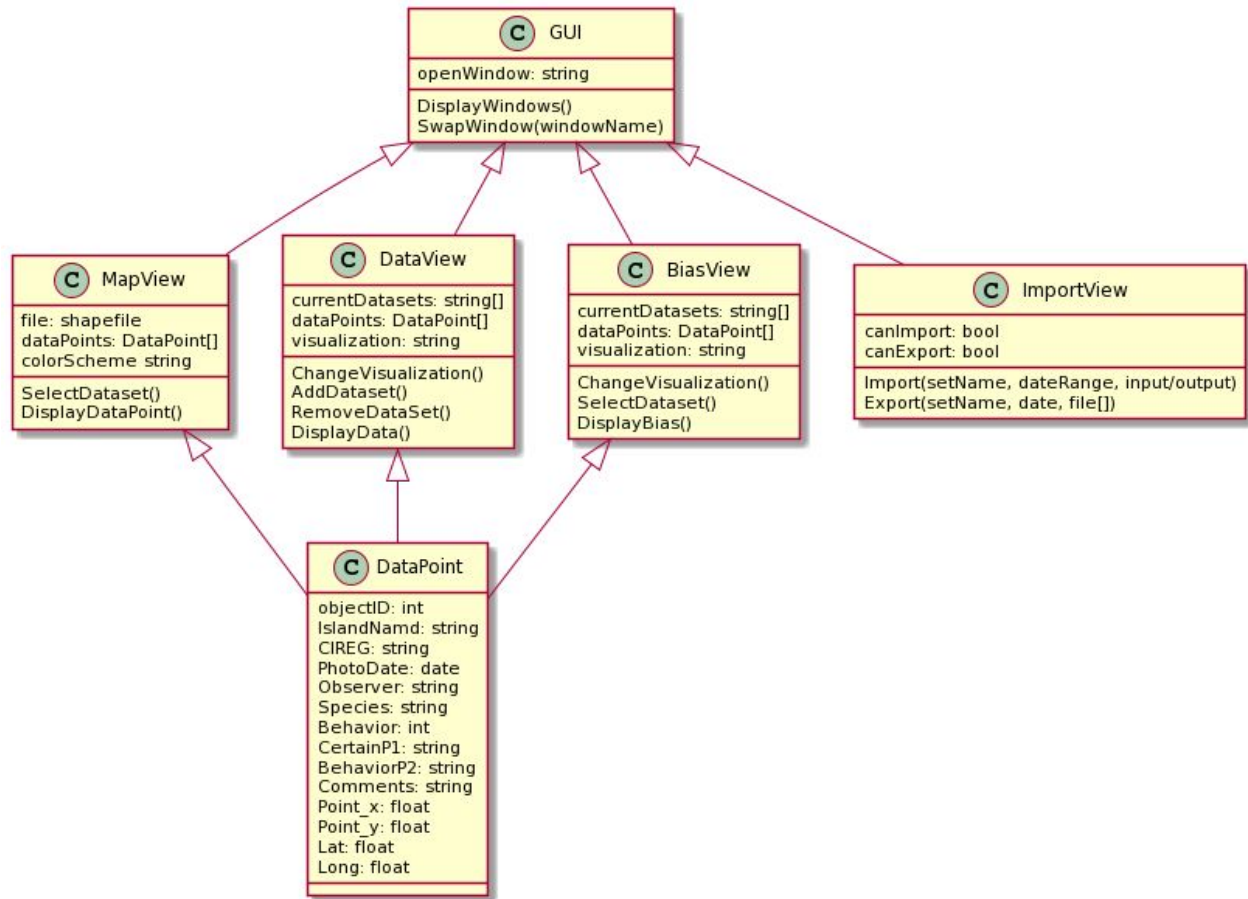


Figure 3: View Decomposition Diagram

This section is effectively the GUI of the application, and there is a breakdown in the above diagram: “View Decomposition Diagram” as well. It will be split into 4 views, and allow for various forms of data visualization. The first of these is the Map View, where given a shapefile and related spreadsheet file, will display the data points on a map determined by the shapefile. Based on the user type and Species, the map will be able to display data with appropriate levels of precision ie) less precision for a more sensitive species. This view will also have the option for the user to select the dataset(s) that they would like to display on the map. The next view is the Data View, where there will be a more standard data breakdown such as pie charts and frequency of bird types in the given region/dataset(s). The user will also be allowed to select data by attributes like Species, Behavior, and Observer. Next is the Bias View, which will work similarly to the data view, but with a different focus. Here, the focus is on the certainty of the output from the output dataset. As with the data view, the user will be able to select data by Species, Behavior, etc. Data and bias view fall under the ‘view result charts’ portion of section 2.1. Finally, there is the Import View, which will vary the most by user permissions, and is only available to someone with the permissions of a Registered User or higher. From here, a Registered User will be able to download a dataset from those that are available, and a Privileged User will be able to import data of their

own, and potentially upload corresponding analysis if it fits the output of the machine learning that is being developed. This view is also related to “ingest data” in the Controller section.

**Controller:** The controller will be the logic behind the application and interfacing between the ACG and machine learning that is being developed. Any user can get the system to list the available datasets, and use that to view the dataset using the application. Additionally, public users will be able to parse the results that are being displayed in various ways as is mentioned in the view section. Login/Authentication will be handled through the controller, and will ensure that correct permissions apply to the user that is being authenticated. Once authenticated, Registered Users can download data that is stored using the ACG for their own needs. On top of all previous abilities, Privileged Users may also Queue the analysis of a dataset of their own on the ACG servers. Finally, at the highest level of permissions, Administrators may modify users, changing their permissions and account type. Administrators are also the only users that will be allowed to add new analysis scripts to be used for analysis of the uploaded data from Privileged Users and above.

### 3. Persistent Data Design

#### 3.1 Database Descriptions

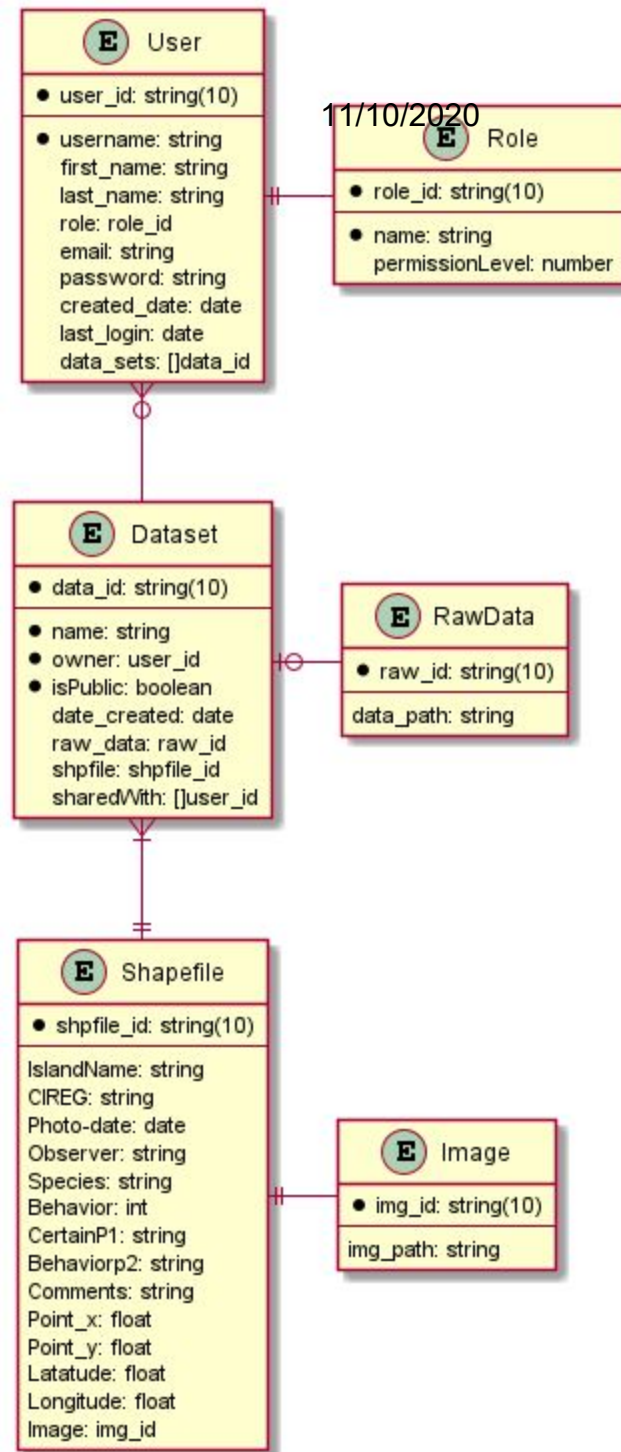


Figure 4. Database Description

User

This is any user within the system and will contain all user information including profile information and permissions within the system.

### Role

A role that a user can have (Admin, user, public scientist, etc.) and the associated permissions level.

### Dataset

A collection for a raw data item (geotiff) and the resulting Shapefiles that are returned from the external ACG compute server, or Shapefiles that are imported by a user. Dataset manages the visibility of a data set and provides information about who can view that data.

### Shapefile

An abstraction of the shapefile format used for the application. Information from a shapefile coincides with each field in Shapefile. Shapefile also added an Image for each record which shows the bird that is present at each data point.

### Image

Contains the url to obtain the image from the file server

### Raw Dat

Contains the url to obtain the raw data (geotiff) from the file server

## 3.2 File Descriptions

### Shapefile:

Shapefiles (.SHP) contain information that represents geocoded locations on a map. Each location is represented by a line in the SHP file table and all lines must contain the fields outlined below in order.

Field Name	Data Type	Description
OBJECTID	int	Unique ID for this location
IslandName	string	Name of the island the location is on
CIREG	string	Registration number for the island the location lies on
PhotoDate	date	Date of the photo of this location
Observer	string	Name of the observer who identified the bird at this location
Species	string	The species of the bird identified at this location
Behavior	int	Boolean representing whether or not the bird is nesting in this location

CertainP1	string	Certainty of identification. Y is certain, NB is unknown behaviour, and NS is unknown species.
BehaviorP2	string	Legacy field for second pass of behavior classification
Comments	string	Observer comments on the identification
POINT_X	float	X value of this location in relation to the raster image
POINT_Y	float	Y value of this location in relation to the raster image
Lat	float	Latitude value for this location
Long	float	Longitude value for this location

Example Shapefiles are located on the COS 397 Seabird Project Shared Google Drive

GEOTIFF files are image files which allow for storage of geological features as well.

Example GEOTIFF files are located on the COS 397 Seabird Project Shared Google Drive

#### 4. Requirements Matrix

#	Name	Description	Test Case ID	Status
1.0	Import data set	Upload your own data set for analysis	TC1.0	TC1.0-No Run
2.0	View Data	View list of data sets available	TC2.0	TC2.0-No Run
2.1	Sort/Filter Data	Sort, filter	TC2.0	TC2.0-No Run
3.0	Manage Data	Modify data set permissions and other meta-data	TC3.1 TC3.2 TC3.3	TC3.1-No Run TC3.2-No Run TC3.3-No Run
3.1	Add Metadata	Add additional relevant metadata to the data (collection method, notes, etc)		
3.2	Manage Visibility	Modify the visibility of data	TC3.1	TC3.1-No Run

	<b>of Data</b>	<b>sets the owner has uploaded</b>	<b>TC3.2</b>	<b>TC3.2-No Run</b>
<b>4.0</b>	<b>Queue Analysis</b>	<b>Choose algorithm and run against previously specified data set</b>	<b>TC4.0</b>	<b>TC4.0-No Run</b>
<b>5.0</b>	<b>View Results</b>	<b>View visualizations of the results of analysis of data set</b>	<b>TC5.0</b>	<b>TC5.0-No Run</b>
<b>5.1</b>	<b>View Charts and Aggregates</b>	<b>View the aggregate data (creainty, number of data points, etc)</b>	<b>TC5.0</b>	<b>TC5.0-No Run</b>
<b>5.2</b>	<b>View Map</b>	<b>View geodata on a map, with the ability to re-color and rescale the map</b>	<b>TC5.0</b>	<b>TC5.0-No Run</b>
<b>5.3</b>	<b>View Details by Data Point</b>	<b>View statistics, location, source photos, AI categorization and certainty by data point</b>	<b>TC5.0</b>	<b>TC5.0-No Run</b>
<b>5.4</b>	<b>Aggregate data points pere-island</b>	<b>System displays a data point per island (instead of proximity or per-bird)</b>	<b>TC5.0</b>	<b>TC5.0-No Run</b>
<b>6.0</b>	<b>Import Results</b>	<b>Results of analysis are associated with a data set</b>	<b>TC6.0</b>	<b>TC6.0-No Run</b>
<b>7.0</b>	<b>Export Data</b>	<b>Data is exported on to the users computer</b>	<b>TC7.0</b>	<b>TC7.0-No Run</b>
<b>8.0</b>	<b>Login</b>	<b>Registered User logs in</b>	<b>TC8.0</b>	<b>TC8.0-No Run</b>
<b>9.0</b>	<b>Manage Account</b>	<b>Manage Account Settings</b>	<b>TC9.1 TC9.2</b>	<b>TC9.1-No Run TC9.2-No Run</b>
<b>10.0</b>	<b>Manage Users</b>	<b>Modify user settings</b>	<b>TC10.1 TC10.2 TC10.3 TC10.4</b>	<b>TC10.1-No Run TC10.2-No Run TC10.3-No Run TC10.4-No Run</b>
<b>10.1</b>	<b>Add User</b>	<b>Adds a user to the system</b>	<b>TC10.1</b>	<b>TC10.1-No Run</b>
<b>10.2</b>	<b>Remove User</b>	<b>Remove a user from the system</b>	<b>TC10.2</b>	<b>TC10.2-No Run</b>
<b>10.3</b>	<b>List Accounts</b>	<b>View the accounts logged within the system</b>	<b>TC10.3</b>	<b>TC10.3-No Run</b>
<b>10.4</b>	<b>Assign Privileges</b>	<b>Assign a user privileged access to the application</b>	<b>TC10.4</b>	<b>TC10.4-No Run</b>

<b>11.0</b>	<b>Add Analysis Script</b>	<b>Upload an analysis script matching a predefined format to be available to users as an analysis method</b>		
-------------	----------------------------	--	--	--

## Appendix A – Agreement Between Customer and Contractor

By signing on the provided line below, the client acknowledges and agrees that the deliverables described by the above terms in sections 1, 2, 3, and 4 are satisfactory, as well as all other clauses described in this document.


Client signature:

\_\_\_\_\_ date: \_\_\_\_\_

Client comments:

By signing on the provided lines below, all Penobscot Development Group members acknowledge and agree to meet all system design deliverables described in sections 1, 2, 3, and 4, as well as all other clauses described in this document.

Penobscot Development Team signatures:

 \_\_\_\_\_ date: 11/10/2020

 \_\_\_\_\_ date: 11/10/20

 \_\_\_\_\_ date: 11/10/2

 \_\_\_\_\_ date: 11/10/2020

\_\_\_\_\_ date: \_\_\_\_\_

Changes to this document may be required as the development cycle progresses. All new drafts of this document must be reviewed, agreed upon, and signed by both the Penobscot Development Group and the client. Upon the signing of a new draft, the previous draft becomes void and the new draft supersedes any obligations set by the previous draft.



## Appendix B – Team Review Sign-off

By signing below, all Penobscot Development Group members acknowledge that they have reviewed all system design deliverables described in sections 1, 2, 3, and 4.

Paul Morris date: 11/10/2020

Dennis Christiansen date: 11/10/2020

Melchor Alonzo date: 11/10/2

Kyle Walker date: 11/10/2020

\_\_\_\_\_ date: \_\_\_\_\_

## **Appendix C – Document Contributions**

The below table describes the Penobscot Development Group members contribution to the document for internal purposes.

20% - Jacob Morin - Section 3.2, Appendices A,B,C  
20% - Devin Christianson - Section 1, Section 2.1, Appendices A,B,C  
20% - Alexandre Feren - Section 2.2  
20% - Kyle Walker - Section 4  
20% - Nick Kania - Section 3.1

# COS 397: Computer Science Capstone I

---

## **User Interface Design Document**

(Adapted from Susan Mitchell and Karuna Joshi)



BirdSpotter  
User Interface Design Document

**Table of Contents**

	<u>Page</u>
1. Introduction	
1.1 Purpose of This Document	3
1.2 References	3
2. User Interface Standards	3
3. User Interface Walkthrough	5
4. Data Validation	9
5. Report Formats	10
Appendix A - Agreement Between Customer and Contractor	12
Appendix B – Peer Review Sign-off	13
Appendix C – Document Contributions	14

## 1. Introduction

BirdSpotter is a graphical interface for integrating and viewing machine learning and field survey data for rapid population estimation of colonial nesting birds. The creation of this interface will allow officials to quickly and effectively draw conclusions based on data provided from human and machine learning observations of bird species, activity, and location.

### 1.1 Purpose of This Document

This document aims to provide a detailed description of the design used in the user interface as well as a walkthrough of the interface and its modules. Furthermore, details are provided for the data items within the system as well as the formats of non visual data produced by the application.

### 1.2 References

This document references:

- System Requirement Specification document:  
<https://drive.google.com/file/d/1gdiXsWQ8f2nXvK7SPesUauaKTBTPlzuF/view?usp=sharing>
- System Design Document:  
[https://docs.google.com/document/d/1f183yU4CEzHg-fvArPLX7b3H\\_0B\\_rnroG33YCjInfZI/edit?usp=sharing](https://docs.google.com/document/d/1f183yU4CEzHg-fvArPLX7b3H_0B_rnroG33YCjInfZI/edit?usp=sharing)

## 2. User Interface Standards

The User Interface Standards dictate what the various components of the BirdSpotter app will look like on the basic level. Here is where the format of the buttons, screen, and menus will be laid out. Additionally, the color scheme of the app will be outlined here formally, for the sake of clarity and consistency both internally and externally.

### Screen Layout:

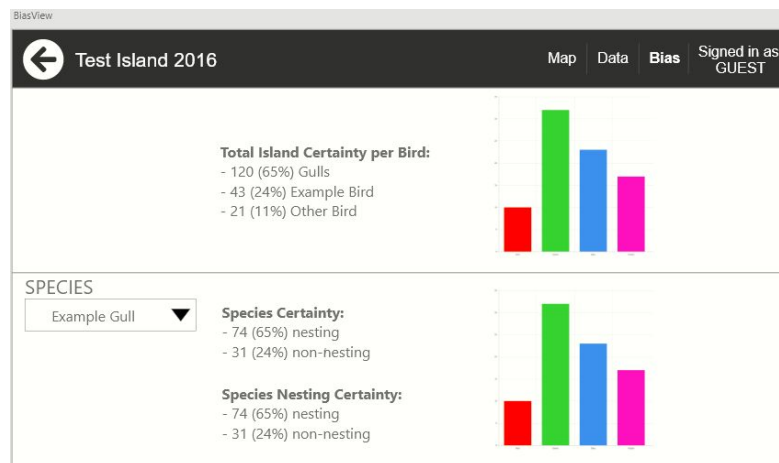


Figure 1: Bias View Template

Each view will be broken up into 2 sections: the view switching section, which is the buttons at the top of the page, and which will link to map\_view, data\_view, bias\_view, and one to log out for every user. There will also be a button to log out of an account for registered users and above (those who are logged in), and when not in main\_view, there will be a back button that links back to main\_view, as well as showing which dataset is currently being viewed. This is the button displayed in the top left of the template. When the button is not available to a given user, the link will be disabled, and visually differentiated from the others by making the text brighter.

The second section: the graphics section, will be the majority of the screen, and is where the relevant information to each view will be displayed. In map\_view, this will be the map and the data points on the map. On the data view, it would be the charts displaying the statistics for that particular dataset. This will be located below the view switching section.

These sections can be seen in Figure 1: Bias View Template. It is the view that will be shown when looking at the certainty and potential bias of the current dataset.

### Color Scheme:

Currently, the color scheme for the app is mostly a fairly neutral off-white color for the background of the app, and a dark grey for the buttons and navigation. The background color and button colors will remain consistent among the sections of the various views. All text displayed on the background will be black, and all text displayed on the buttons will be white.

The only exception to the coloring scheme of the buttons is that the logout button will have red text to help steer the user away from accidentally logging out.

### Navigation:

Navigation was mentioned briefly in the screen layout section, but it will be done by clicking on the relevant button, such as the “Data” button, which will bring the user to the Data View. Clicking on the back arrow will lead to the MainView, where another dataset may be selected, or a registered user could choose to export the data.

### 3. User Interface Walkthrough

This section will go over the layout of the User interface of the app and direct the reader as to how the app is laid out and what each view does. Effectively, this section is just combining all of the formalized components from section 2 into more relevant and easily digestible sections that outline the User Interface and relations for the BirdSpotter app. The relations and flow are covered in Figure 2 and its description, and the views themselves will be discussed further down the section with the relevant mockups.

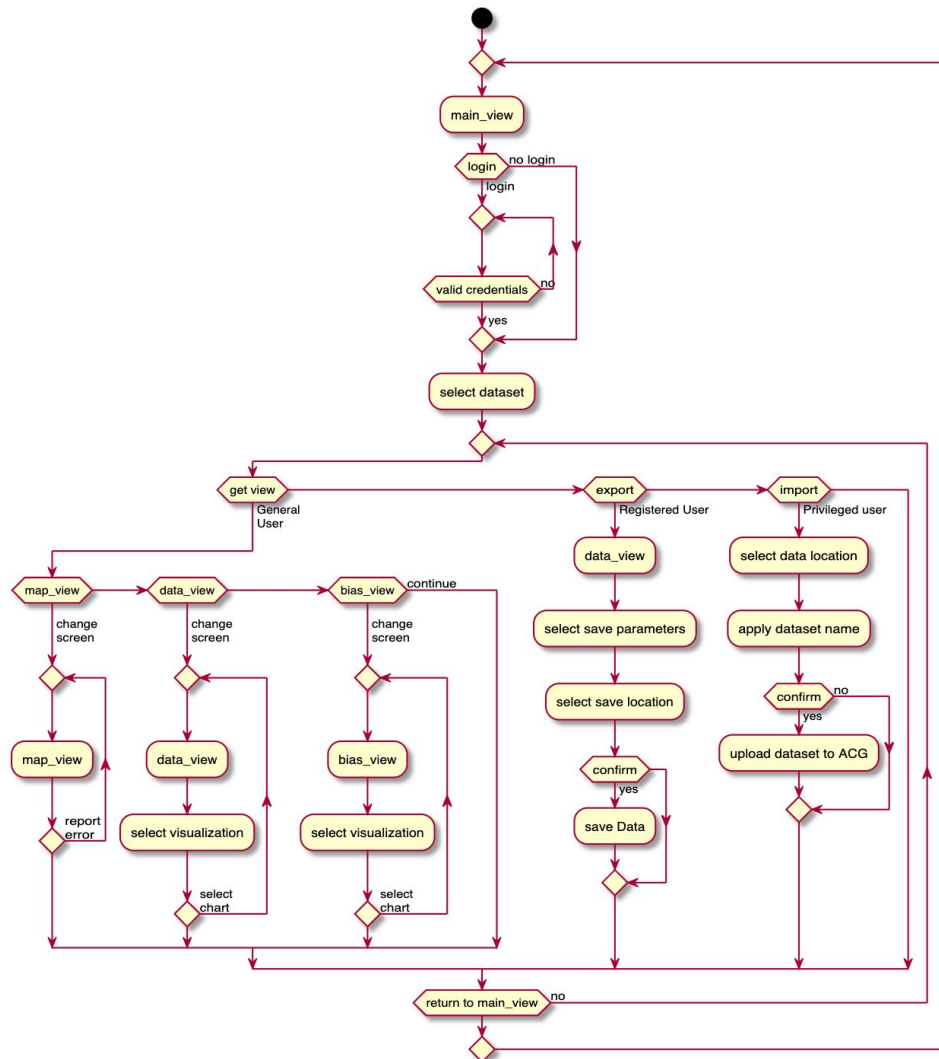


Figure 2: Navigation Diagram

The navigation diagram (shown above) shows the sequence of navigation that a user can take when using the BirdSpotter app. All rounded bubbles ending with “\_view” are representative of one of the screens that will be available to users, these views are **main\_view**, **map\_view**, **data\_view**, and **bias\_view**. Additionally, from the main screen, registered users, upon selecting a dataset, can select export, and privileged users can select import. These will be

buttons that are available to users from within the main view. On the map view, “report error” refers to where a user may want to report mistakes in classification of a datapoint on the map. On the Data View and Bias view, “select visualization” refers to opening a menu to change the visualization, and “select chart” refers to the actual selection of the visualization/chart they would like to see.

It is also worth noting that there will always be an option to sign in/log out on each screen, but it would clutter the chart so it has been omitted for clarity.

Name	Lat-Long	Date Uploaded	Date Of Imagery	Actions
+ Test Island 2016	43.884969,-69.375191	11/16/2020	5/31/2016	<a href="#">View</a>
+ Test Island 2017	44.884983,-59.378891	11/16/2020	5/31/2016	<a href="#">View</a>

Figure 3: Dataset View/Main View

The main view is the initial landing screen, as well as where a user can select datasets to work with, login to their account, and select which screen they would like to navigate to next. Only Registered Users and Privileged users will have access to the Import and Export tabs that are displayed on this screen to avoid potential issues with public users disturbing habitats based on downloaded geological data and so that only people who are more likely to care about the base data will have easy access to it.

Import, available to Privileged Users and above allows users to upload their own GeoTIFF files to the ACG to analyze.

Analyze allows users to select which analysis algorithm they would like to use on the dataset they are looking at. The algorithms to select from will be the ones provided by the machine learning team. Once a user has selected the algorithm they would like to use, they will be directed to a form to fill in the additional user parameters that the algorithm requires. These the requirements for these parameters will be specified by the designers of the AI algorithm.



The “View” button in the table of datasets is a drop down, and will present the user with an option to select any of the views regarding datasets: Map View, Data View, Bias Vies, and Export View. The Export View is only available to Registered Users and above.

Finally, there is the tab labeled with “Signed in as GUEST”, which is the login/logout tab. This is available on every screen, so it will not be mentioned on further screen diagrams. This is where users will login/logout, and will also allow a user to change their passwords and other settings if they are a Registered User or above.

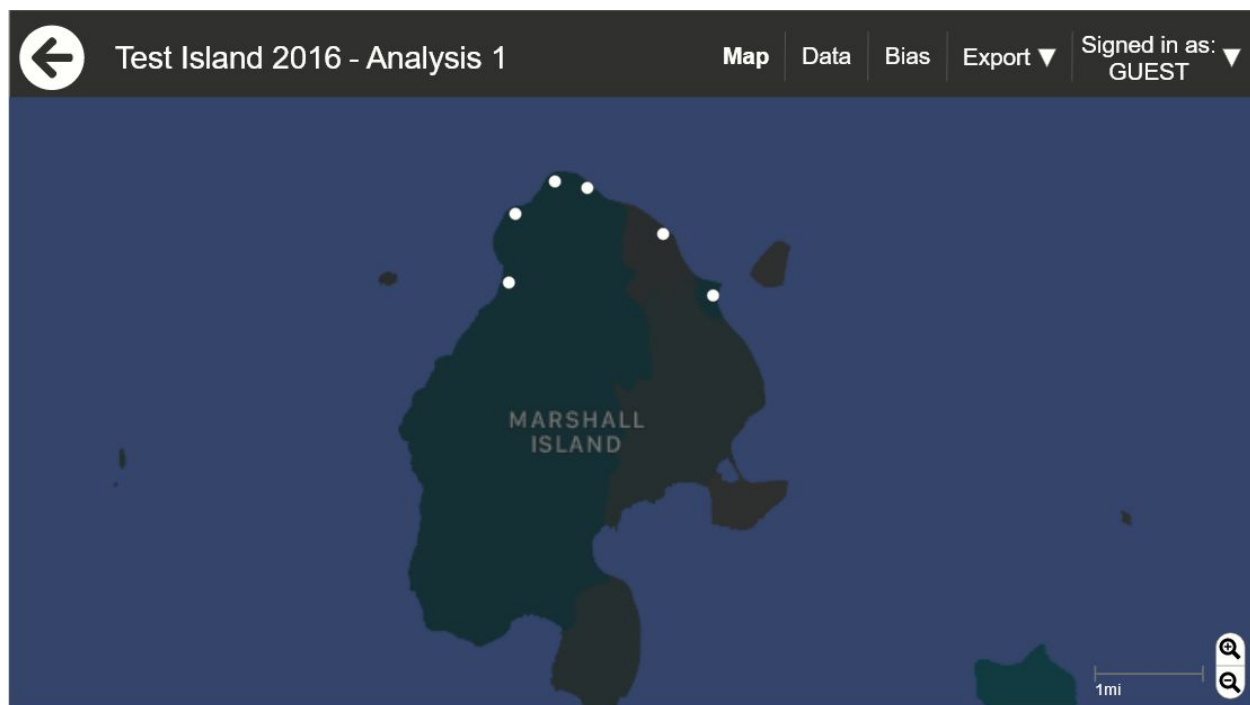


Figure 4: Map View

The map view is where the user will be able to view points in the dataset they have selected in the context of a map. This is also the default tab that is opened when a user selects “View” from the Main View. Users will be able to zoom in and out of the map, allowing for either a more granular view of the data points, or a more *birds-eye* view with a more generalized view of data points. However, if the user is not registered, their ability to zoom in will be restricted so they will only be able to view the more generalized data and not exact data points. Not displayed here is the mouseover information that users will get when hovering over data points, which will display the data and metadata of the selected point. This data will include species, nesting type, the observer, the certainty for that datapoint, and if the user is a Registered User or above, it will also give precise latitude and longitude for the datapoint.

The scale is provided in the bottom right, measured in miles. The + and - magnifying glasses will zoom in and out of the map, respectively when the user clicks on them. Finally, in the top left, the island name or CIREG will be displayed as well as the observer for the analysis, which is listed to the right of the island name. The island name and observer will be visible in this location for every view.

Finally, the export option just to the left of the login/logout button will allow a user to export the current dataset in all of the forms that are available to download. It will be a dropdown

of various download options, and a Registered User will be able to download the relevant spreadsheets, shapefiles, and GeoTIFFs upon the request on this page, and can select the location that they would like to save the file to as well.

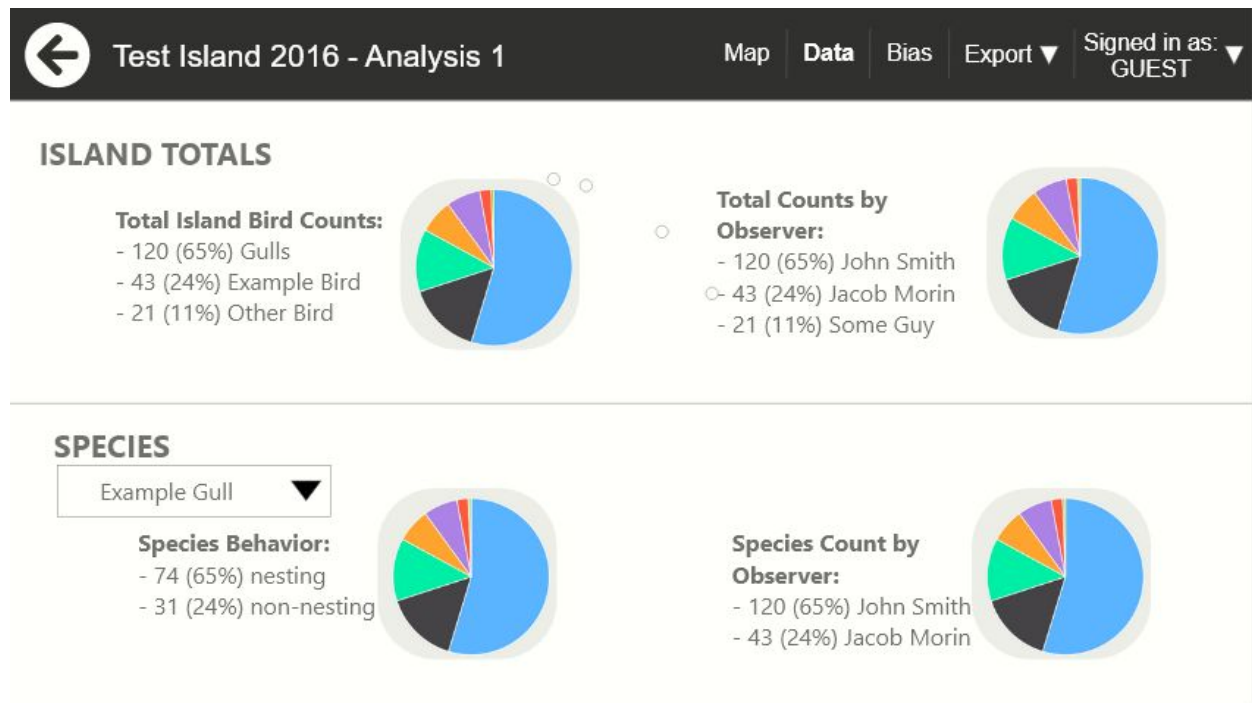


Figure 4: Data View

The data view is where the general statistics of the selected dataset will be displayed in a more traditional visualization method, which the user will be able to select from the available charts. A user may change which data they would like displayed under what is currently displayed as the “Species” dropdown. The data visualisations displayed in this figure are only placeholders. They will be updated in the future as more user experience data is gathered.

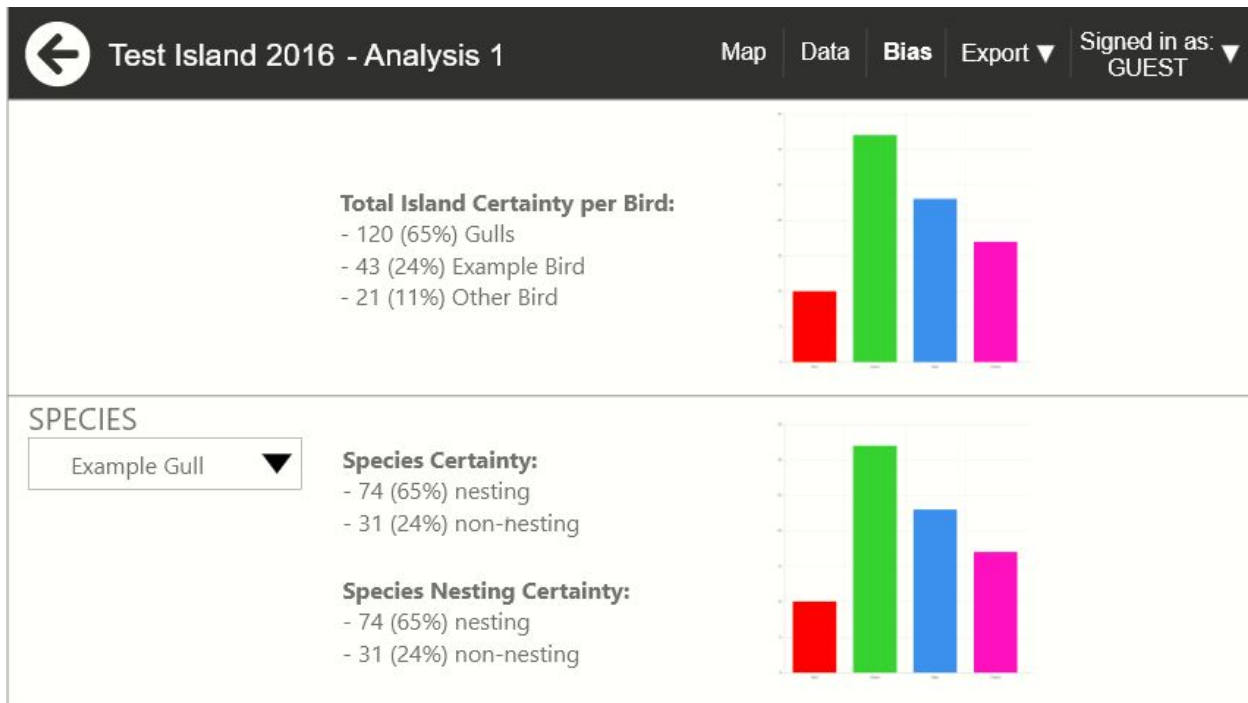


Figure 5: Bias View

The bias view is where the data related to the certainty of observations will be displayed, similar to the data view. In the same way as in the Data View, a user can select how they would like to change the visualization under the “Species” dropdown. The data visualisations displayed in this figure are only placeholders. They will be updated in the future as more user experience data is gathered.

#### 4. Data Validation

Data is anything that can be supplied to the system by a user. This includes both files imported into the system, and forms that the system provided for the user to fill out. The system will accept data supplied in the specified formats, as long as those files are within the limits also specified below. Forms are displayed to allow the user to input information, along with allowing for the filtering of data and changing of location.

##### File Inputs:

Data Item	Data Type	Limit(s)	Format(s)
Analysis: Shapefile	binary	2GB	.zip, .shp
Analysis: Mask images	binary	16GB	.jpg, .png
GeoTIFF	binary	32GB	.tif

**Main View inputs:**

Screen Name	Data Type	Limit(s)	Format(s)
Login	String	Username: 1-30 characters Password: 8-30 characters	form
Enter Metadata	String	Metadata: 0-500 characters	form
Search	Int, String	Latitude/Longitude: $\pm 90 / \pm 180$ Dataset name: 1-20 characters Datum: [NAD27, NAD83, WGS84]	form

**5. Report Formats**

Reports are data outputs to the user that are not viewed through the applications interface, but are instead viewed using external interfaces. In the case of Birdspotter, all of the file exports are reports. These exports include the raw data stored within the application, and processed aggregates of the data, as well as additional AI metadata.

Name	Description	File Type
Raw data export	Raw data set of aerial photography	GeoTIFF
Raw Analysis results export	Raw analysis output, in the form of a shapefile of all identified birds	Shapefile
AI Analysis	User-readable AI results, including estimated certainty, accuracy, and mask images	csv
Results Analysis	User-readable representation of all aggregate data displayed by the application, and the data points behind that data	csv

## Appendix A – Agreement Between Customer and Contractor

By signing on the provided line below, the client acknowledges and agrees that the deliverables described by the above terms in sections 1, 2, 3, 4 and 5 are satisfactory, as well as all other clauses described in this document.

Client signature:

Cynthia S. Loftin

date: 11/24/2020

Client comments:

By signing on the provided lines below, all Penobscot Development Group members acknowledge and agree to meet all system design deliverables described in sections 1, 2, 3, 4, and 5, as well as all other clauses described in this document.

Penobscot Development Team signatures:

Jack Morris

date: 11/24/2020

Devin Christianson

date: 11/24/2020

Kyle Walker

date: 11/24/2020

Michael Stein

date: 11/24/2020


Alexander

date: 11/24/2020

Changes to this document may be required as the development cycle progresses. All new drafts of this document must be reviewed, agreed upon, and signed by both the Penobscot Development Group and the client. Upon the signing of a new draft, the previous draft becomes void and the new draft supersedes any obligations set by the previous draft.

## Appendix B – Team Review Sign-off

By signing below, all Penobscot Development Group members acknowledge that they have reviewed all system design deliverables described in sections 1, 2, 3, 4, and 5.

 date: 11/24/2020

Devin Christianson date: 11/24/2020

 date: 11/24/2020

 date: 11/24/2020

 date: 11/24/2020

## **Appendix C – Document Contributions**

The below table describes the Penobscot Development Group members contribution to the document for internal purposes.

30% - Jacob Morin - Section 2  
15% - Devin Christianson - Section 5  
25% - Alexandre Feren - Section 3, Section 2  
10% - Kyle Walker - Section 4  
15% - Nick Kania - Section 1, multiple section intros