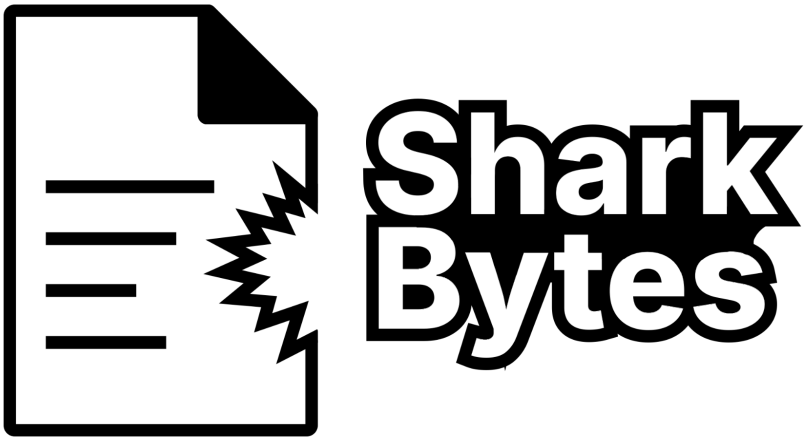
Tech Requirements - 11/26/2025



Clients: David Rogowski, Mara Dzul, Pilar Rinker

Mentor: Scott LaRocca

Team: Jered Angous, Daniel Arden, Alexander King, Anthony Narvaez

Table of Contents

0.0: Introduction

Over 40 million people depend on the Colorado River and its surrounding ecosystems for food, water, and livelihoods, underscoring the importance of this vital resource. Conservation efforts are essential to monitoring and preserving the wellbeing of these resources amid growing pressures from climate change, prolonged drought, and overuse. Now, more than ever, the data collected by organizations such as the U.S. Geological Survey, Arizona Game and Fish Department, Grand Canyon Monitoring and Research Center, and U.S. Fish and Wildlife Service are critical in identifying environmental stressors, guiding resource allocation, and informing effective conservation strategies.

However, the current data collection workflow used by these organizations relies on an outdated and poorly maintained application known as SHOALS. While still essential to their operations, the use of SHOALS requires dealing with an unintuitive user interface, a total lack of documentation, and minimal maintenance. These issues have led to frustrations in the field, increased time spent troubleshooting, and a higher risk of data entry errors, all of which undermine the efficiency and effectiveness of conservation efforts.

Our team aims to address these technical challenges by developing a modern, intuitive, and maintainable data entry system that is flexible enough to cover the needs of many field biologists. This document intends to establish the core requirements for this system. It includes the functional and performance requirements, risk assessment, and success criteria that will contribute not only to the success of the project, but to the ongoing success of these conservation efforts as well.

1.0: Problem Statement:

In order to effectively understand the current limitations of SHOALS, it is helpful to understand the workflow used by biologists when collecting data in the field. This will provide valuable context that highlights how SHOALS is currently used and how our new app can help improve the data collection process.

Although the workflow might be slightly different depending on the specific organization and type of fishing, a typical survey trip progresses as follows:

1) SHOALS control files are adjusted depending on the specific river mile the team will be working on. This is usually done before the trip officially starts, and includes changing the list of species the team is expecting to encounter.
2) Once in the field, the team will connect their Bluetooth PIT tag scanner and catch fish depending on their respective fishing method. The fish will then be processed and scanned for any previous PIT tags.
3) If a PIT tag is detected, the data from the last catch of that fish will be displayed on the SHOALS app.
4) The fish will then be processed (e.g. identified to species, measured, weighed, sexed, etc). This new data is entered into the program for storage. It is important to note that when measuring fish, the new data is not only stored in the machine's local memory, but also on up to 3 external flash drives. The fish are then released, and steps 2-4 will be repeated.

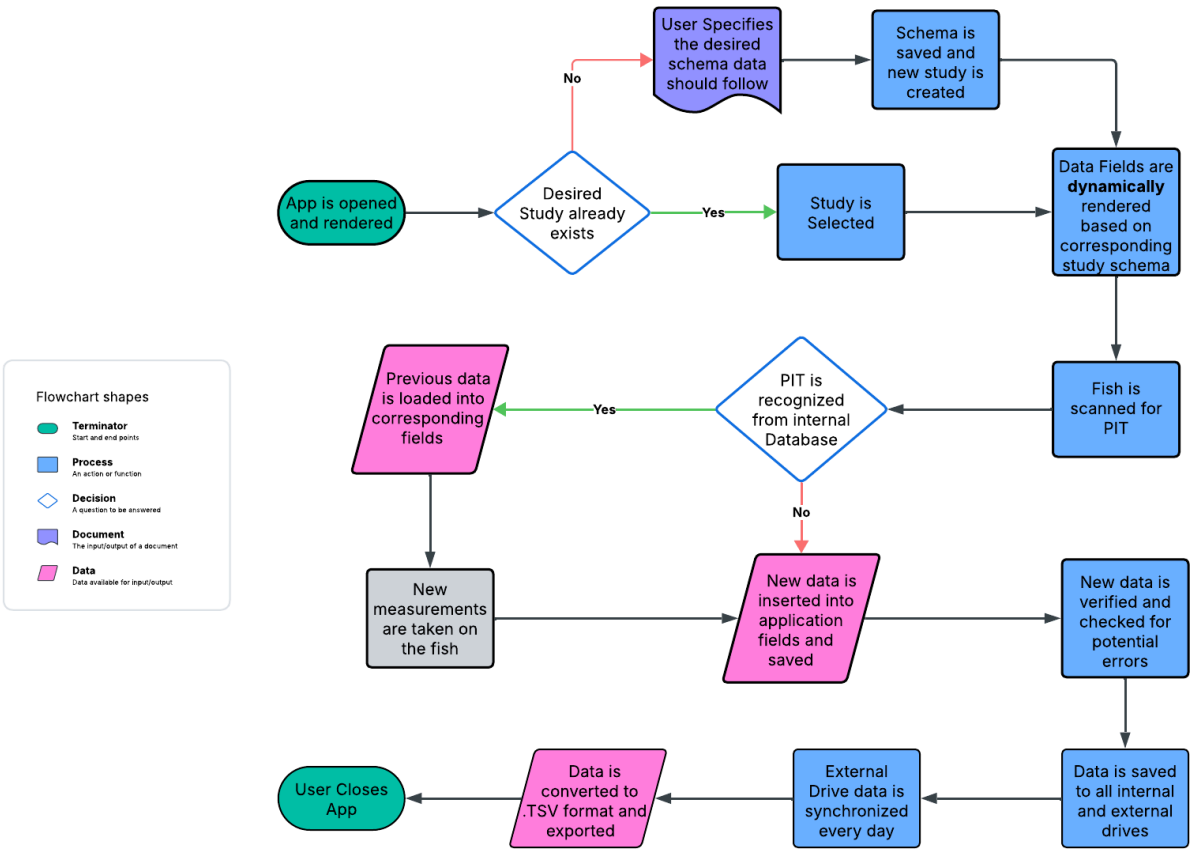
5) At the end of the day, the researchers will synchronize the data on all external drives. Synchronization is done using a feature built into SHOALS and ensures that the whole team has their own copies of the same data. This also increases redundancy, which is crucial when working in areas where data loss is likely.
6) At the end of the trip, the data will be synchronized one last time, and then exported to a staging database where it is processed and cleaned up before placing it in the main database.

SHOALS, along with the above mentioned process, has supported data collection operations for years and continues to be a critical component of the workflow. However, due to a lack of maintenance and documentation of the software, inefficiencies discovered while using the program remain. Below are some of the main pain points that researchers have experienced in their use of SHOALS:

- Inaccurate “measurement inconsistency” alerts lead to underestimating specimen proportions
- Updating the PIT tag recap file and uploading it to shoals is unintuitive and takes time to figure out
- It is difficult to create a new field layout for different organizations/types of fishing, leading to the current set of 6 executable versions of SHOALS
- Total lack of documentation hinders maintenance and improvement efforts
- The language SHOALS is programmed, Visual Basic, and the retirement of the main developer over 6 years ago raises obsolescence concerns

These limitations demonstrate a clear need for a modern, flexible, and reliable new system to replace SHOALS and better support long term research efforts. In order to address these areas of concern, our team proposes the following solution.

2.0: Solution Vision



Our team plans to address the issues with SHOALS by creating a more flexible and modern data entry system. This new application will provide a maintainable and well documented replacement to the aging SHOALS program and allow biologists and researchers to continue providing critical monitoring data efficiently and effectively.

Specifically, we aim to implement:

- A customizable set of fields so users can adapt it to their use case.
- A robust backup system that saves to local storage and multiple USB drives upon every saved entry.
- Bluetooth compatibility with PIT Tag Readers.
- Easily interpretable graphical data summaries on collected data.

Our system intends to provide a drop-in replacement to SHOALS rather than entirely changing the way researchers conduct their work. As such, it will operate in a very similar fashion to its predecessor. Our system will interface with the same hardware and use the same data that is being collected by our sponsors on their research and monitoring trips.

Collected data will be securely stored in a local database and backed up to multiple external drives using SQLite. At the end of the survey trip, users will also have the ability to easily export the raw data to the main database. Also similar to SHOALS, the new program will allow for reliable connections with Bluetooth PIT scanners via Bluetooth allowing for continued efficient data collection.

The goal of the project is to provide a system that preserves the data and protects it from corruption; therefore no major data transformations or calculations will affect the data itself. However, general statistics on the data including average weights, lengths, and other quantities, will be displayed in-app as graphs and easily interpretable charts. The application will also provide methods for easy format conversions from the

database-specific files used internally to the tab-separated value format currently used by research teams.

While other methods might have yielded marginally faster response times or smaller storage requirements, using SQLite and database files allows for unmatched data security and redundancy. Our team has elected to incur these slight penalties in exchange for significantly improving the reliability of our app especially in environments where data loss is prevalent. We are confident that we can provide a solution that is considered enjoyable to use, easy to learn, and trustworthy by our sponsors, and hope its use can be extended into other conservation agencies in the future.

3.0: Project Requirements

Having considered the high-level vision for this project, we now consider the specific requirements that comprise the new application and how these impact both the researchers using our program and the development of the application itself. These requirements have been divided in order of priority. Specifically, the four categories follow the MoSCoW framework and include requirements that "Must," "Should," "Could," and "Won't" be completed throughout this project.

3.1: Functional Requirements

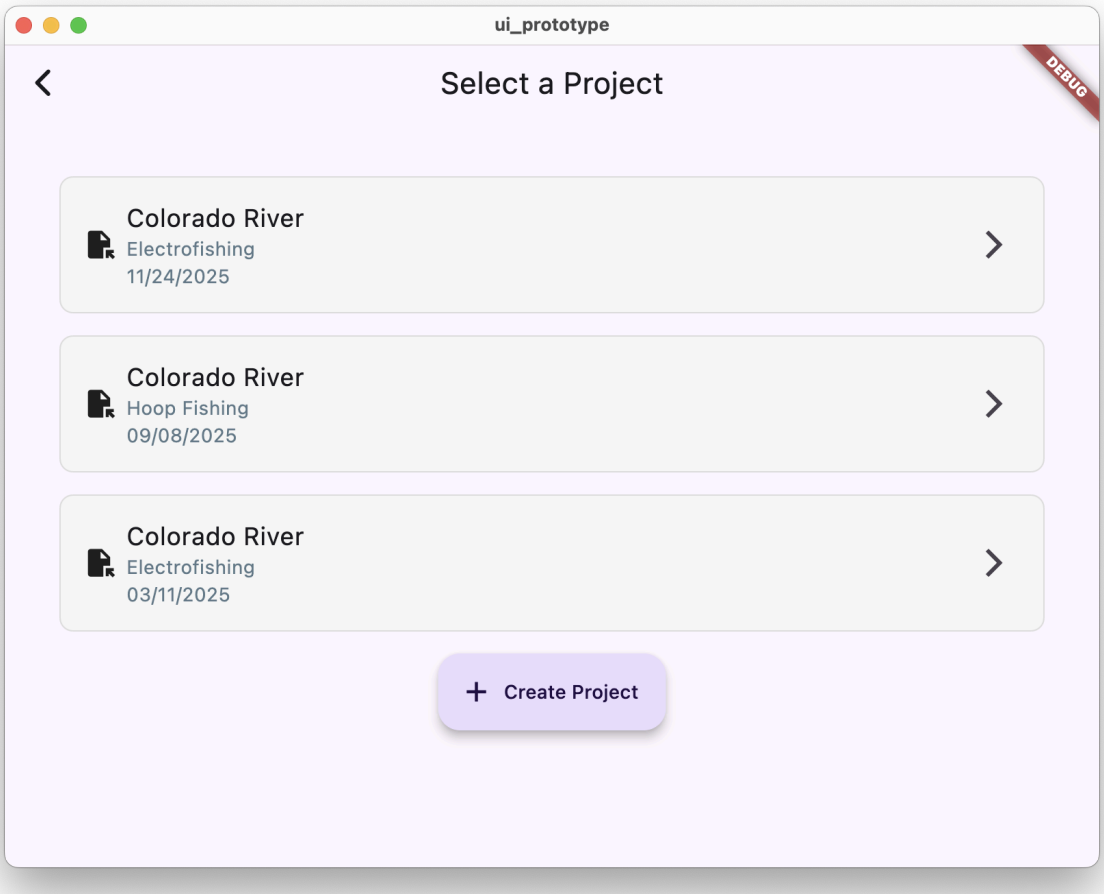
3.1.1: Interface to Load Different Study Designs

Must

User story

As a field researcher, I want the software to be able to load different study designs from a graphical interface.

Screenshot



Description

The application should have an interface to open past studies, sorted by how recently they were last opened. It is important to note that the word “study” refers to the specific research or conservation agency using the application, as well as the specific data field layouts they work with. Additionally, there should be a Create Study button which allows the user to create a new study layout from a template, or create a new template from scratch.

Sub-functions:

- **Chronological list of last-opened trips**
    - Allows commonly used study designs to be easily accessible by displaying the last few study designs used on the application.
- **May contain type of study, date of study, location of study, and other relevant details**
    - Saved trip information allows researchers to quickly select the desired trip type without having to manually input all of the data.

3.1.2: Highly Customizable Fields

Must

User story

As a project administrator, I want to add, remove, and rename data fields and dropdown options so that I can adapt the application to the needs of different projects without modifying the code.

Screenshot

{} fish_project.json > ...

```json
[
    {
        "table_name": "specimen",
        "field_name": "weight",
        "label": "Weight",
        "tooltip": "Weight in grams of current specimen.",
        "column": 1,
        "row": 3
    },
    {
        "table_name": "specimen",
        "field_name": "species",
        "column": 2,
        "row": 1
    }
]
```

fish_project.sql

```sql
CREATE TABLE Specimen (
    PitID INTEGER PRIMARY KEY,
    Species TEXT,
    Weight REAL,
    sex TEXT,
    check(sex IN ('M', 'F', 'U'))
);
```

Description

The project administrator will create SQL schemas for the variety of fields they need in the application. Additionally, the user can create a JSON file format for managing application-level metadata, such as where the field should be placed, tooltips, labels, et cetera. By using JSON for this, the metadata level is flexible enough to adapt to new functionality without requiring SQL migrations or breaking compatibility with older or future versions of the application.

Sub-functions:

- **Database query to get list of fields**
    - This allows administrators or researchers to define new study layouts quickly and efficiently without having to deal with separate versions of the application. They simply need to define the desired fields in an SQL schema.

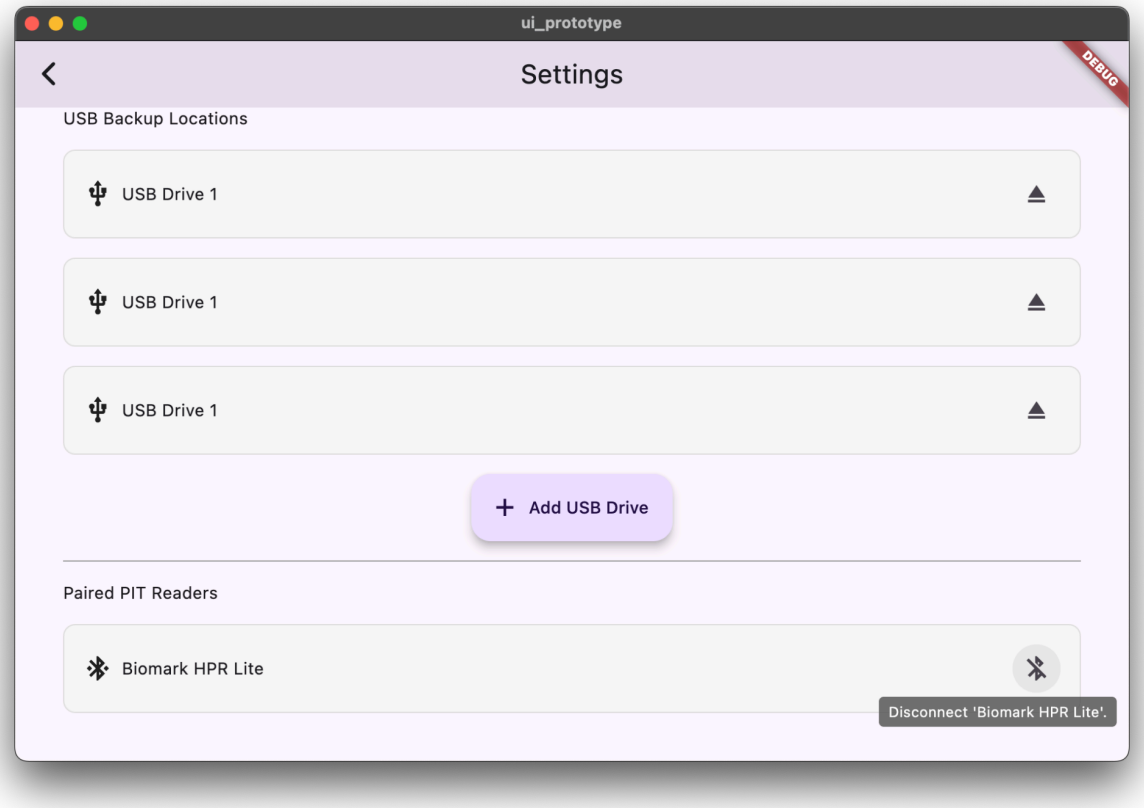
3.1.3: Compatibility with PIT Tag Readers

Must

User story

As a field researcher, I want the software to pair with PIT tag readers to automatically import scanned identification data without requiring manual entry.

Screenshot



Description

The application should use the operating system's Bluetooth protocol to pair to Bluetooth PIT tag readers. The program should then be able to link this to a field in the

main data collection interface, so when a new entry is scanned, the PIT tag ID is automatically populated into the field.

Sub-functions:

- **Pair to Bluetooth PIT Tag Reader**
    - Requires the use of the operating system's built in bluetooth communication tools, but simplifies the development process.
- **Retrieve scanned tag information from reader and use to populate ID field**
    - This function manages the detection of a PIT tag in a fish and uses the detected PIT identification number to query the database for that fish's information.
    - It also uses the field information to identify and populate corresponding fields with the data associated with the PIT tag.

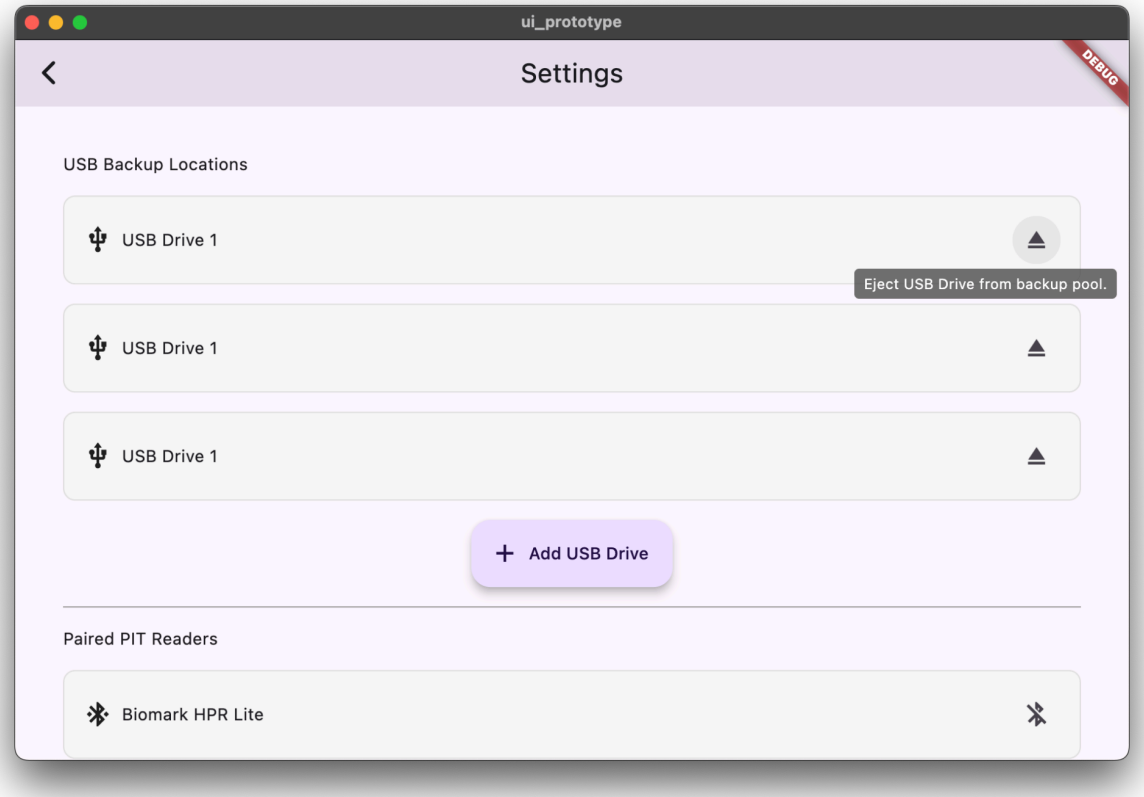
3.1.4: Automatically back up to external storage

Must

User story

As a field user, I want the software to automatically back up entries to both local and external storage devices, so if the data entry device fails, the data is still safe.

Screenshot



Description

Due to the high cost of research trips, and therefore the high value of the data being collected, it is important to have systems in place to ensure data integrity. One of the main features to provide this is with automatic backups. Flash drives can be designated as external storage locations that are automatically backed up to when new data is entered. While it is highly recommended to use these backup systems, the app will still work without external storage locations. The data would simply be stored locally on the data entry device's storage.

Sub-functions:

- **Database file management**
    - Handles the updating of the data files in the USBs and in the internal laptop storage. It involves the detection or creation of the .sql or .db files and their corresponding data directories.

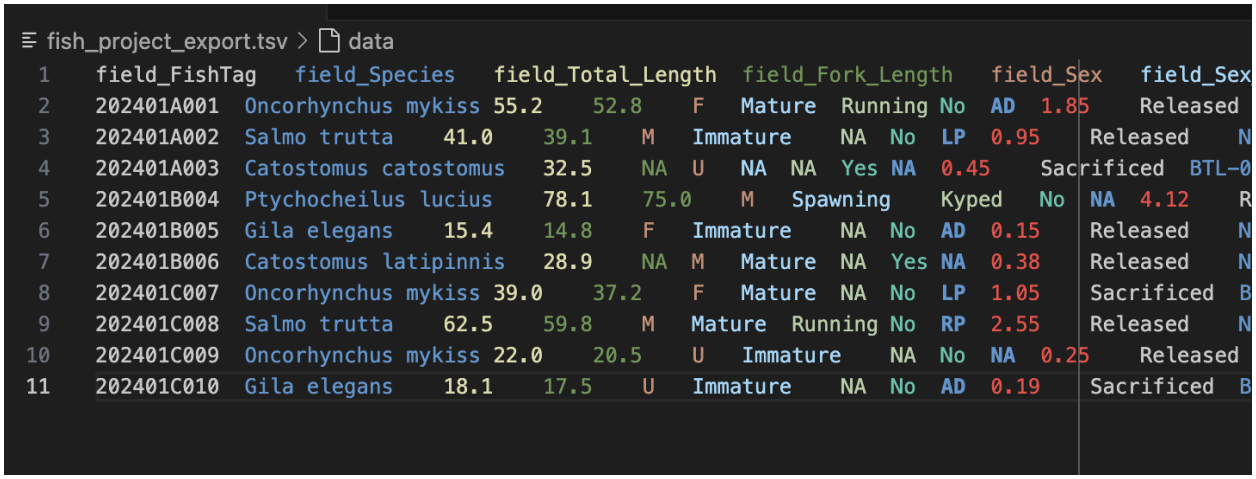
3.1.5: Export to .TSV format

Must

User story

As a data manager, I want the software to provide the option to export collected data to a .TSV file in order to easily integrate it into the main database.

Screenshot



fish_project_export.tsv > data

	field_FishTag	field_Species	field_Total_Length	field_Fork_Length	field_Sex	field_Sex
1	202401A001	Oncorhynchus mykiss	55.2	52.8	F	Mature Running No AD 1.85 Released
2	202401A002	Salmo trutta	41.0	39.1	M	Immature NA No LP 0.95 Released N
3	202401A003	Catostomus catostomus	32.5	NA	U	NA NA Yes NA 0.45 Sacrificed BTL-0
4	202401B004	Ptychocheilus lucius	78.1	75.0	M	Spawning Kyped No NA 4.12 R
5	202401B005	Gila elegans	15.4	14.8	F	Immature NA No AD 0.15 Released N
6	202401B006	Catostomus latipinnis	28.9	NA	M	Mature NA Yes NA 0.38 Released N
7	202401C007	Oncorhynchus mykiss	39.0	37.2	F	Mature NA No LP 1.05 Sacrificed B
8	202401C008	Salmo trutta	62.5	59.8	M	Mature Running No RP 2.55 Released N
9	202401C009	Oncorhynchus mykiss	22.0	20.5	U	Immature NA No NA 0.25 Released
10	202401C010	Gila elegans	18.1	17.5	U	Immature NA No AD 0.19 Sacrificed B

Description

The program needs to be able to export the data into a human-readable text file format so that it can be parsed and inserted into the primary database. Our program must be able to join the tables from the database into one table, and then write this data to the preferred file format (specified by our clients to be .tsv). This will be achieved through the use of an ‘Export’ button that allows researchers to take the data in one drive and export it in the specified .tsv format.

Sub-functions:

- **Program must generate Join Table and execute Select All statements on current database**
    - Ensures all data is synchronized and converts the database file format into an easier to work with .tsv format.
- **Program must reformat and export this data into a .TSV formatted file**
    - Simply queries all of the data in the database files and prints out each line in the database as a set of tab separated values.

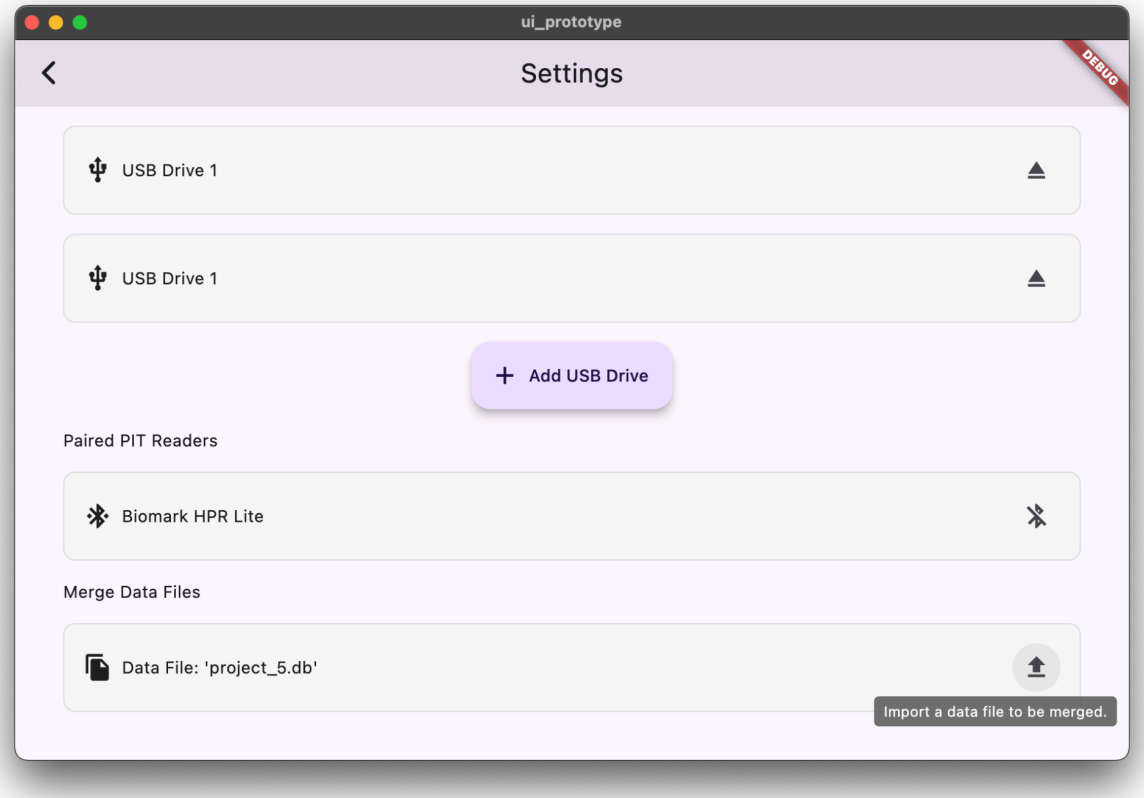
3.1.6: Sync different data files together

Must

User story

As a data manager, I want to merge the different data files' entries together, so that entries recorded on different devices by different team members can be consolidated into a single, up-to-date dataset.

Screenshot



Description

The application will handle synchronization of different data files by creating a new SQLite database file for the merged dataset. This merged database will add an additional field to the existing tables: a unique identifier, such as the record's creation timestamp or a unique device identifier (UUID). Combined with the existing primary key, this will form a composite primary key, ensuring that each record is uniquely identifiable. After a merge, future data entry will continue to write to each device's existing database. If any of the databases being merged contains fields not present in the others, the application will throw an error.

Sub-functions:

- **Check if database files share common fields and data types**
    - Handles the detection of discrepancies between database files and selects which entries need to be re-entered into their respective database files.
- **Import records from both databases into new merged database file**
    - This allows all database files on the system (including backups) to be synchronized with the most up-to-date information. Researchers often use this feature at the end of a data collection day to ensure everyone is working with the same data the next day.

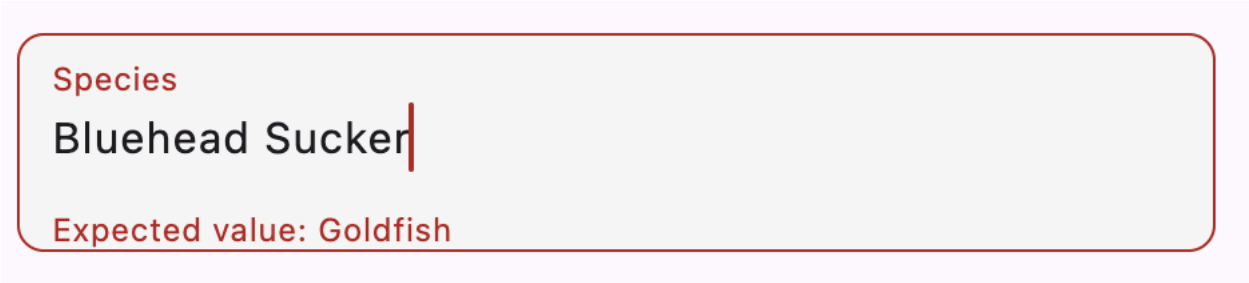
3.1.7: Historical Data Consistency Verification

Must

User story

As a field researcher, I want the software to automatically display the last recorded catch associated with a scanned PIT tag, so I can quickly identify discrepancies in species, weight, and other attributes when entering new data.

Screenshot



Description

When the data is entered into the field, it will query the historical entry database using the unique identifier for a specific fish. This data will then be used to verify the new data being entered. Fields like ‘Species’ or ‘sex’ will throw overridable warnings if new entries mismatch what is in the database. These warnings will be in the form of red highlighting and error error text stating what the expected value was.

Other fields, such as ‘Weight’ and ‘Length’ are expected to change over time. Therefore the verification for such fields is more focused towards ensuring that the

differences aren't impossible or highly unlikely. For example, if a fish is significantly shorter than its last measurement, then a similar warning will also be thrown. This is also the case for weights that are too far below or too far above what is expected.

These warnings allow the user to correct any errors before entering the data which helps maximize the accuracy of the data. However, there are specific instances in which the warnings can be dismissed. For example, smaller fish being eaten by larger fish can produce incorrect-looking results. To handle these edge cases, researchers will have the option to override any warnings that might come up.

Sub-functions:

- **Get entered value from text field**
    - This function handles taking the live input of the user and calling the query function used to identify if the user is entering one of the expected values or not.
- **Return result from database query, and if there's a mismatch, add error text to text field**
    - This function queries the database with every input by the user ensuring it matches one of the expected inputs for that field. This helps reduce misidentification or other inaccuracies.

3.1.8: Noises for Verifying Populated PIT Tag Field

Must

User story

As a user, I want the program to make different sounds based on the specific action taking place in order to work efficiently.

Screenshot ~ N/A

Description

If the user scans a PIT Tag with a different ID than the one currently entered in the field, it will make a sound alerting the user to this discrepancy (the client described this as an “uh-oh!” sound). However, if the user scans a PIT Tag with the same ID, it will make a different sound. This helps the user in not double-tagging the fish. Other alert sounds can be added in the future for other issues such as a historical inconsistency or a duplicate. This dependency on sound has allowed researchers to work significantly faster together and thus is a major requirement for the project.

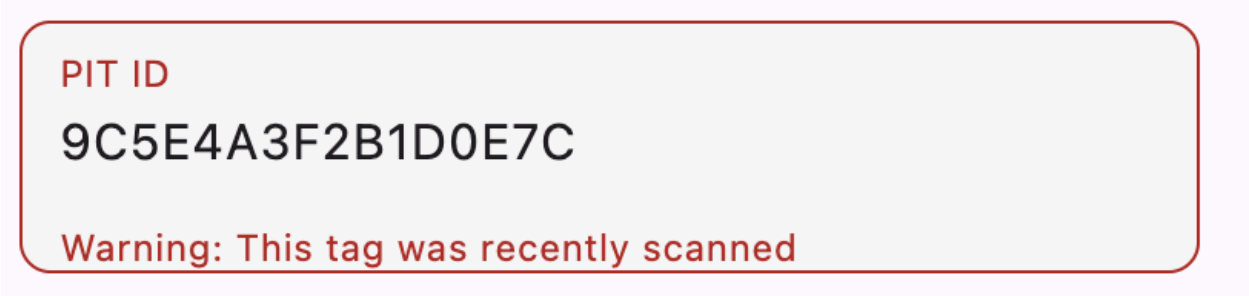
3.1.9: Recent Duplicate Warning

Must

User story

As a user, I want the program to alert me if I scan the PIT Tag of a recently tagged fish, so we don't add unnecessary duplicates to the database.

Screenshot



Description

If the user scans an ID that was recently scanned in an arbitrary amount of time (1 day, 1 week, etc.), the program will alert them via a warning message that the fish was recently scanned, and asking the user to verify they want to enter the data again before saving the new entry. This keeps the data clean and free of redundant entries.

3.1.10: Separate Unique Identifier than PIT Tag for Database

Must

User story

As a user, I don't want the program to use the PIT Tag as a primary key, as some fish do not receive PIT tags, and a PIT tag may be captured multiple times over the years.

Screenshot N/A

Description

In the database schema, each specimen entry will have an autoincrementing ID. The PIT Tag field will be nullable in case the user needs to use another unique identifier such as dye.

3.1.11: Error Checking

Should

User story

As a user, I want the program to alert me when outliers are detected in a dataset, so that I can review and validate the data.

Description

As a user enters data into fields, a check will be done to determine if the info might be incorrect. When data is entered into a field, it will be compared to multiple data collections. If the data entered is outside a range, a warning will alert the users to check, confirm, or change the data.

Comparisons will be determined by the field, for example, weight, which takes from the previous tag history. If the weight entered is above a user-specified value, or if the weight is lower than the previous history, a flag will be created.

Sub-functions:

- **Calculate outliers**

A function to calculate if the current field entry data is an outlier, based on the previous catch history of the same species. A sample of N size will be determined at a later date to have the best accuracy and performance. The outlier will be determined with one or more methods, such as a Z-score or Interquartile Range (IQR).

3.1.12: Data Summarization

Should

User story

As a user, I want to see a graphical summary of the data I entered, so that I can quickly identify trends, patterns, and outliers in the dataset.

Description

After all data has been collected and stored, the user can request to generate a comprehensive graphical and textual summary of the dataset. The goal of this feature is to help users quickly interpret their data and gain meaningful insights through clear visual representations and descriptive statistics.

The summary will include both visual and textual elements to present the data in a user friendly and informative way. Visualizations may include bar charts, pie charts, and scatter plots to illustrate relationships and distributions within the data. For example, bar charts will display the number of fish caught by species, allowing users to easily compare catch frequencies. Scatter plots will show correlations between fish length and weight, helping users identify potential trends, clusters, or outliers in their data.

In addition to visuals, textual summaries will provide concise written descriptions and key statistics, such as the total number of fish recorded, the number of species represented, and summary metrics like average, minimum, and maximum lengths and weights. These textual summaries will complement the visuals by offering precise numerical context to support data interpretation.

Sub-functions:

- **Data Aggregation**

The application will perform SQL select statements on the current project's database to get relevant numerical data. This data will then be stored in a Dart data structure, likely a map or two arrays.

- **Graph Visualization**

The aggregated data in the Dart data structure will be fed into a Flutter graphing library, such as fl_chart, and rendered to the display as a graph. The primary focus will be scatter plots, but different types of graphs are an option.

3.1.13: Keyboard Shortcuts

Should

User story

As a user, I want common keyboard shortcuts (e.g., save, tab, and undo) so I can complete tasks more efficiently.

Description

The system could support a set of common and intuitive keyboard shortcuts to improve user efficiency and streamline interactions within the application. These shortcuts will allow users to perform frequent actions quickly without relying solely on mouse input. Examples include standard commands such as Ctrl+S to save progress, Tab to move between input fields, and Ctrl+Z to undo the most recent action.

Additional shortcuts may be implemented for navigation, data entry, and editing, such as Ctrl+C / Ctrl+V for copying and pasting, and Enter to confirm selections. These keyboard interactions will be consistent with established platform conventions to ensure familiarity and ease of use.

3.1.14: GUI-Based Study Creation

Should

User story

As a user, I want a convenient graphical user interface to easily create new studies without having to manually edit control files.

Description

The current workflow for SHOALS involves manually editing custom control files to change what the expected values for certain fields are as well as hard coding new studies into the app. This is cumbersome and difficult to learn for new members of the team. Therefore, the GUI-Based method for creating studies would allow for a more intuitive and easy-to-learn method for study creation.

While desirable, the issue we face with the implementation of this feature is time. Creating a secure and consistent method for translating UI values into a compatible SQL Schema requires several layers of verification and data management. It is unlikely there will be enough time to implement this feature without sacrificing other core features. Therefore, our team has opted to prioritize the main functionality of the new application before attempting to implement this commodity.

3.1.15: Mobile App Support

Could

User story

As a device manager, I want to have the ability to run the application on mobile devices so we can add affordable mobile devices to our fleet.

Description

As we are using Flutter, a multi-platform application framework, it theoretically should be possible to port our application to mobile devices easily. However, we will be using multiple database files, which may create issues for mobile platforms. Additionally, Bluetooth support varies per-platform, and may not work with the older Bluetooth protocol used by PIT Scanners

3.1.16: NFC Tag Support

Won't

User story

As a user, I wish to add NFC Tag support to the application, so we can utilize affordable NFC readers instead of separate PIT Tag Readers.

Description

NFC Tags are not commonly used for wildlife tracking, so this functionality is not necessary to implement.

3.2: Performance Requirements

3.2.1 Scanner to UI Latency

The current workflow involves several crew members. Teams will often have one person scanning and processing fish while another enters data, allowing them to scan more fish in less time. Therefore, it is important that any information from previous catches be displayed in a reasonably fast time. That is, the system should detect the scan, identify the PIT identification number, look up the specific entry in the database, and display this to the user in under 300ms. This will allow researchers to continue working efficiently with the new program.

3.2.2 Guaranteed Data Integrity

Biological survey trips can cost several tens of thousands of dollars per trip, making the data that is collected throughout the trip equally valuable. It is imperative that any data that is submitted to the new program be protected against corruption and data loss. While data loss can happen due to loss of equipment or damage to external drives, the application itself should lose exactly 0 entries.

3.3: Environmental Project Requirements

3.3.1: Hardware

Description

Our application must functionally run on our data entry devices, consisting of a fleet of different laptops with a variety of specifications. These laptops usually range from government-issued devices to refurbished or salvaged laptops.

3.3.2: Operating System Cross-Compatibility

Description

Our clients need the application to work on Windows operating systems version 10 and up. However, even though most laptops being used run Windows, our clients plan to begin using Linux in the near future. Therefore, our application should run equally well on Windows, and on Linux, so we can use a wider variety of machines for our fleet of data entry devices. It is important to note that while our application will compile on MacOS, bluetooth support will not be implemented for these as that operating system does not support Serial Port Profiles, making communication with the scanner significantly more difficult. This, combined with the fact that Macs are hardly used by the research teams, pushed us to focus on the functionality for the Windows and Linux operating systems.

3.3.3: Offline Support

Description

Our application needs to work without the internet, as in the field I may not have access to a reliable internet connection. Internet connection also raises concerns with government agencies as these would have to be thoroughly checked by the IT department to ensure sensitive government information cannot be leaked.

3.3.4: Installation

Description

The clients want to be able to run the software as a standalone executable without needing to install other software so that they can run it on any computer, even offline. This makes transferring the app between computers easier, especially since internet access is highly regulated for government devices.

3.3.5: Licensing Requirements

Description

Our team has opted to go with the Copyleft license for software development. This allows us to keep our software, as well as derivative works of our software, open source. This aspect of our project was specifically requested by our sponsors.

3.3.6: Open Source

Description

We will be publishing our application under an open source license on GitHub. This means our application will be transparent about how it works, as anyone can analyze the code, and it has the potential to be maintained by a larger community.

4.0: Potential Risks

As with every major project, there are risks involved in both the development and deployment of this new software that must be taken into consideration in order to deliver a robust and effective software tool.

As for development, the main concerns involve the risks of memory leaks and corrupted data. Unexpected events like the disconnection of an external drive, wear on the drives, or even catastrophic damage to the data collection tools pose a threat to the integrity of the data. To mitigate this, our application will use the built-in security features of SQLite, such as transactional logging and atomicity, to ensure data integrity persists regardless of any unforeseen events.

There are larger-picture risks associated with the effectiveness of our application. As mentioned above, data loss can pose a significant threat to the success of a fish monitoring or research trip. This is a primary concern especially considering these trips can cost many tens of thousands of taxpayer-dollars. Similarly, incorrect or missing data can lead to misrepresentations of the environment in and around the Colorado River. Such misrepresentations have the potential to lead to less-than-ideal changes to conservation and management efforts that have a direct impact on species of concern and people that depend on the Colorado River.

Our team is implementing measures to minimize the risk of data loss and corruption. The application will use SQLite which has transactional guarantees, file handling and software practices which will prevent partial or corrupted writes thus making every data entry operation completely safe. Additionally, the application will write to

multiple external drives, so if something happens to the client's data entry device, the data collected will not be lost. We are committed to providing a solution that biologists and researchers can trust.

5.0: Project Plan

Fall Semester

Planning:

The initial phase of the project is about planning which takes place from week 3 to week 9. In this phase our team met with our clients to get an idea of the client's main problem, project goals, and project vision. The next step of the planning phase was to figure out what technologies would work best for the application and create a project plan to stay on track with development and deployment.

Requirements Analysis:

The next phase of the project is the requirement analysis which takes place from week 9 to week 14. In this phase our team will create a tech requirements document that will include both functional and non-functional requirements then it will be given to our clients to be approved.

Design:

The next phase of this project for the fall semester ranging from week 11 to 16 is the design phase. In this phase the main priority was creating prototypes of different must, should, could, and won't functionalities such as UI wireframes and database schemas to validate the design choices with the clients.

Prototyping:

The final phase of this project for the fall semester that will take place from week 13 to 16 will be the prototyping phase. During this phase our team will begin building early working versions of the application based on the design that was chosen. These prototypes will be used to validate performance, usability, and technical feasibility so our team will be prepared for the development phase.

Spring Semester

Development :

In the first 8 weeks of the spring semester the capstone team will focus on the development phase. In this phase our team will fully focus on the creation of the new and improved application.

Testing:

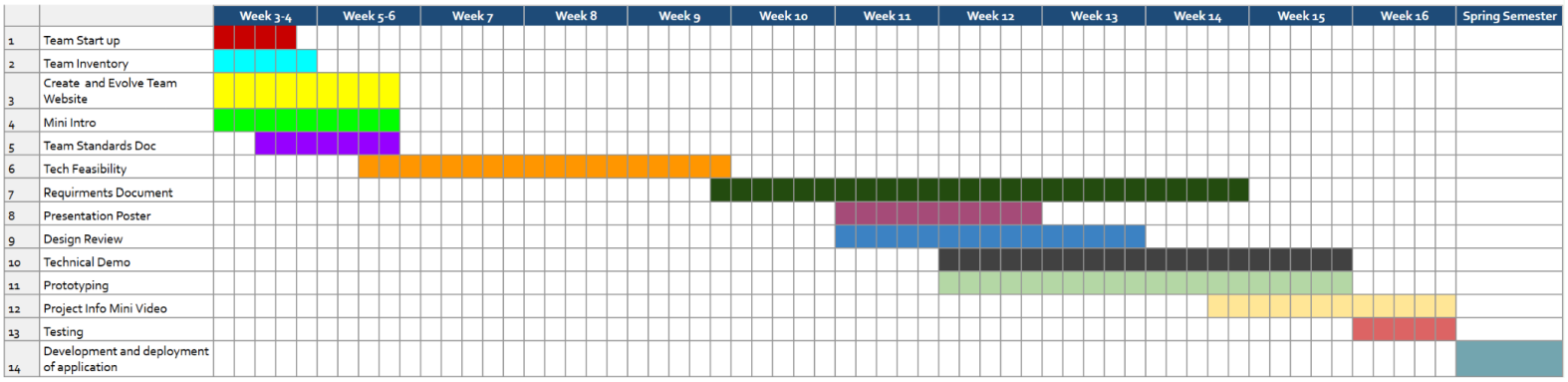
From week 8 to week 10 the team will work on the testing phase. This phase will involve fully testing our application and debugging it so that it is ready for the deployment phase.

Deployment:

By week 10 of the Spring semester our team will proceed with the deployment phase. In this phase we will present our application to our client so it can be taken to the field for deployment.

Maintenance:

In the final 6 weeks of the spring semester our team will focus on the maintenance phase. In this phase we will fix any new errors or issues our clients might encounter with our application.



6.0: Conclusion

The Colorado River is a crucial resource for the American Southwest, supplying water to millions of people. Understanding the health of the ecosystem is therefore essential to advising management decisions that may impact the Colorado River. We are building a data entry app to aid field researchers in collecting vital information for understanding the composition of the Colorado River's various ecosystems. This requirements document serves as an agreement between the capstone team and the clients, defining the functional and non-functional needs of the system. It defines what the team will deliver to the client and serves as the foundation for the application's design, implementation, and testing. Through this work, our team has made significant progress towards the next stages, which involve prototyping and validation with our clients. We are confident that with these requirements now in place, this application will help researchers effectively monitor the health and biodiversity of this ecosystem, helping to inform management decisions that will maintain the health of the Colorado River for years to come.

Glossaries & Appendices

PIT tag: Passive Integrated Transponder

Shoals: A Data entry program currently used by agencies and private contractors

Study: Refers to the specific research/conservation organization and method of fishing