Field Day

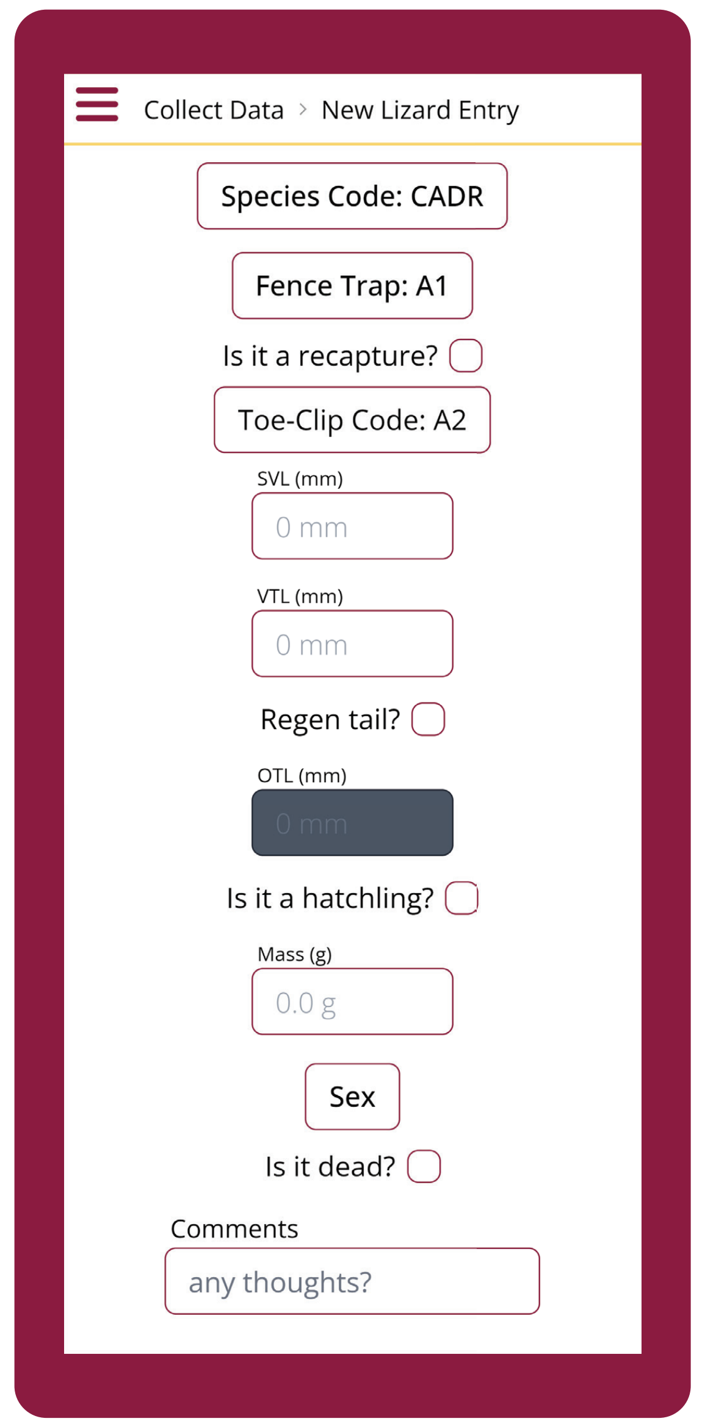
Wildlife Data Collection & Management Tool

Software Engineering Capstone 2022-2023

Isaiah Lathem, Ian Skelskey, Jack Norman, Zachary Jacobson, and Dennis Grassl

Mobile App

The mobile app is a progressive web app (PWA) that has been designed to support the research activities of biology and ecology students as they study wildlife in field settings. Specifically, the app facilitates the collection of population data for various species using capture mark-recapture techniques. To address the challenge of conducting research in remote areas without access to wifi or cellular data, the app caches the database locally when installed on the device. This allows for continued data collection even when offline. During offline data collection, the cache is updated, and once an internet connection is established, the cache synchronizes with the database. This innovative approach to mobile app design supports the research efforts of students in the field, enabling them to collect and analyze data more efficiently and effectively.


Collect Data › New Lizard Entry

Species Code: CADR

Fence Trap: A1

Is it a recapture? ☐

Toe-Clip Code: A2

SVL (mm): 0 mm

VTL (mm): 0 mm

Regen tail? ☐

OTL (mm): 0 mm

Is it a hatchling? ☐

Mass (g): 0.0 g

Sex

Is it dead? ☐

Comments: any thoughts?

Shown above is the user-friendly data collection screen for a new lizard entry, designed to streamline the capture mark-recapture process and support research in the field. Input is validated before submission to minimize human error.

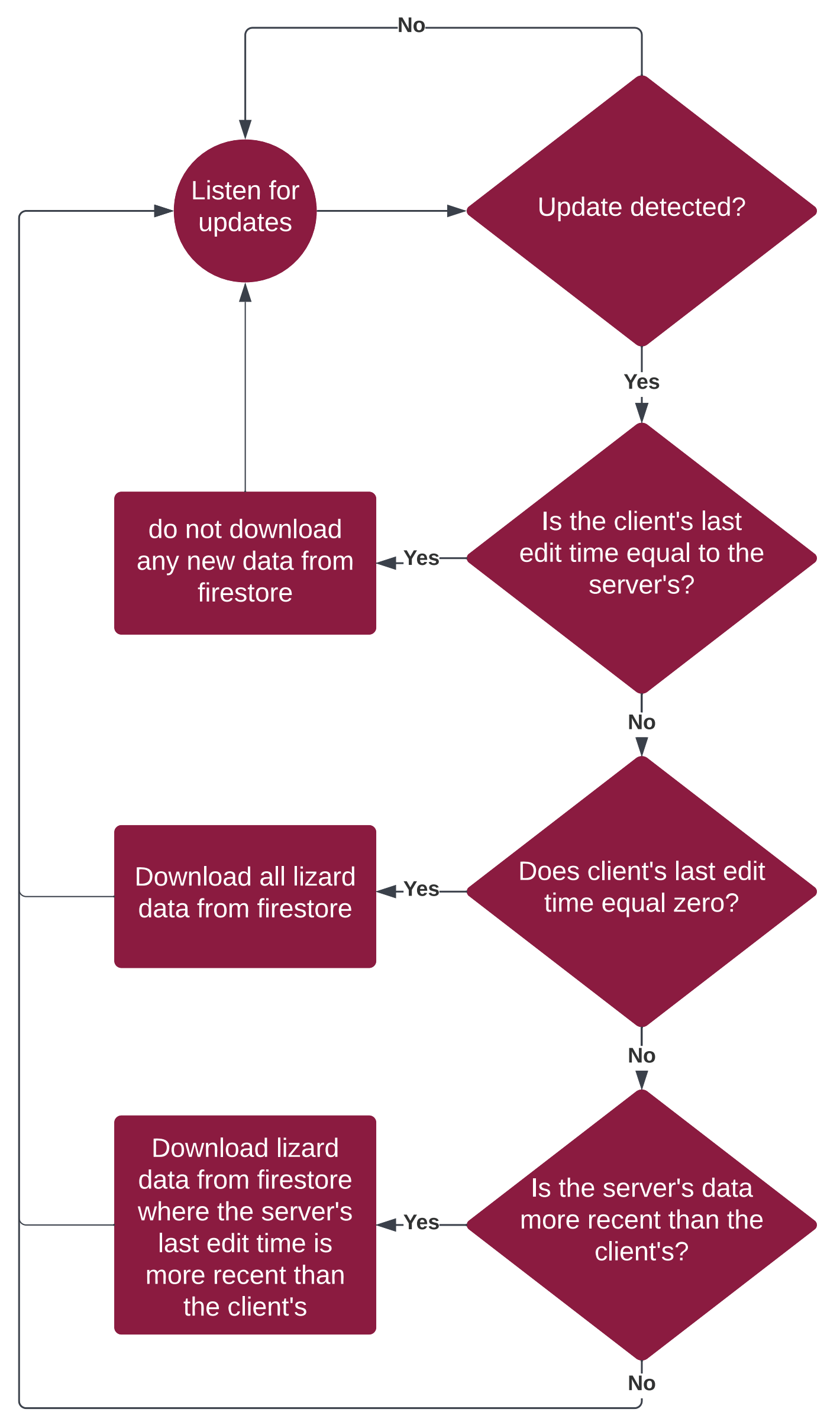
Field Testing

Ian Skelskey (left) had the opportunity to field test the data collection tool in collaboration with biology students and researchers including Dr. Bateman (center).

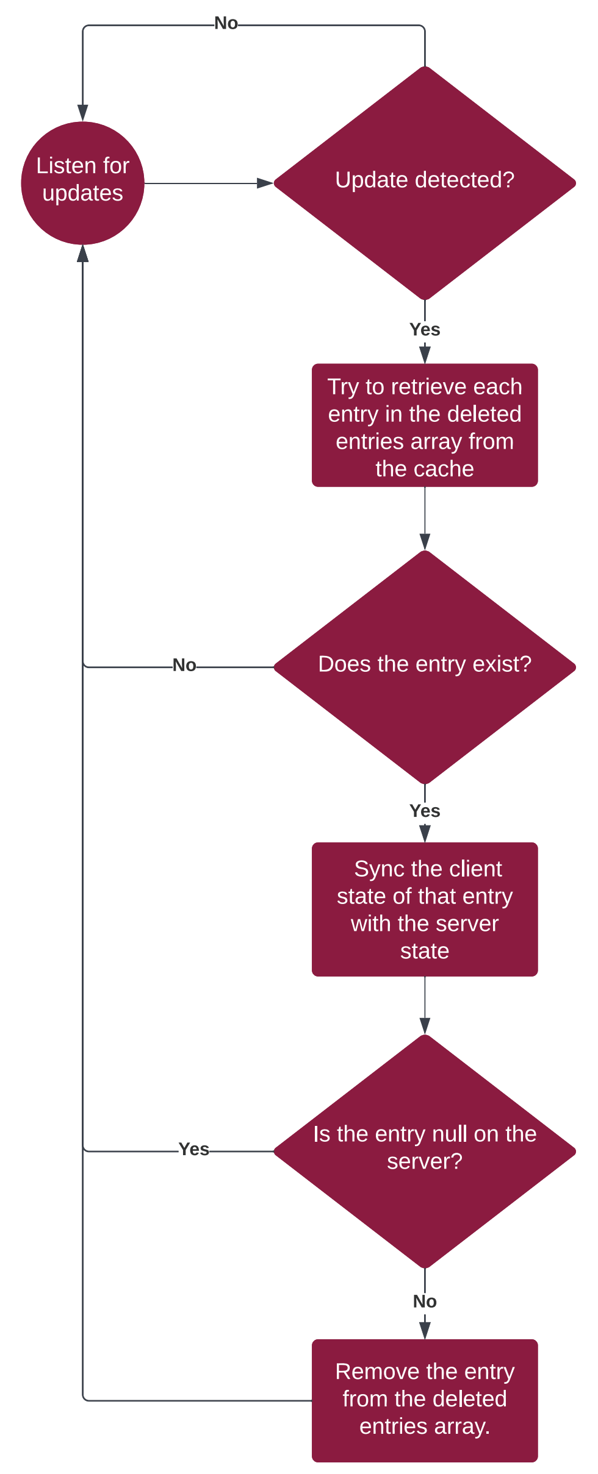
Syncing with the Database

Our team has implemented a Firestore document system to exclusively store metadata pertaining to modifications made to the database. The application integrates listeners to detect any updates to these metadata documents, which then triggers synchronization logic. This methodology is intentionally designed to curtail database operations and reduce project costs, in line with our objectives. The two flowcharts presented below outline some of the steps involved in this synchronization process. Whenever a new or updated entry is detected, the client's last edit time is compared to the server's to determine the most recent version. Alternatively, when a deleted entry is detected, its status is verified and updated, if necessary.

New or Updated Data


Listen for updates
Update detected?
No
Yes
Is the client's last edit time equal to the server's?
Yes
do not download any new data from firestore
No
Does client's last edit time equal zero?
Yes
Download all lizard data from firestore
No
Is the server's data more recent than the client's?
Yes
Download lizard data from firestore where the server's last edit time is more recent than the client's
No

Deleted Data


Listen for updates
Update detected?
No
Yes
Try to retrieve each entry in the deleted entries array from the cache
Does the entry exist?
No
Yes
Sync the client state of that entry with the server state
Is the entry null on the server?
Yes
No
Remove the entry from the deleted entries array.

Logo designed by Andy Bridges in 2012

Problem

Field researchers encounter several challenges when collecting and analyzing wildlife population data, including human error during data collection, limited connectivity in remote areas, and high costs of maintaining a secure cloud database. These issues can compromise the accuracy of results and hinder scientific research.

Solution

To address these issues, the Field Day project employs innovative app design and cross-disciplinary collaboration to streamline data collection and analysis. The team focused on reducing deployment costs and enhancing user experience by migrating from Amazon Web Services to Firebase, which offers a free tier and is more cost-effective. The team also improved the Mobile App and Web App to better handle limited connectivity and enhance authentication security using 2-factor Google authentication. These changes have resulted in a more efficient, secure, and cost-effective user experience.

Success Metrics

Cost:

- Switched backend to Firebase: Firebase offers a free tier and is much cheaper than AWS, reducing costs.

Security:

- Enhanced authentication security: Authentication is more secure because it piggybacks off of the ASU 2-factor Google authentication, which is more secure than the previously distributed simple username/password combination.

Efficiency:

- Improved PWA offline functionality: PWA now downloads the entire database only once after the user logs in for offline use and downloads new/edited data incrementally as the data is added via introduction of metadata tracking.
- Enhanced WebUI performance: WebUI now only queries what it needs depending on what the user is doing, instead of downloading all data from the server and storing it locally for each instance.

Development Stack

Front End:

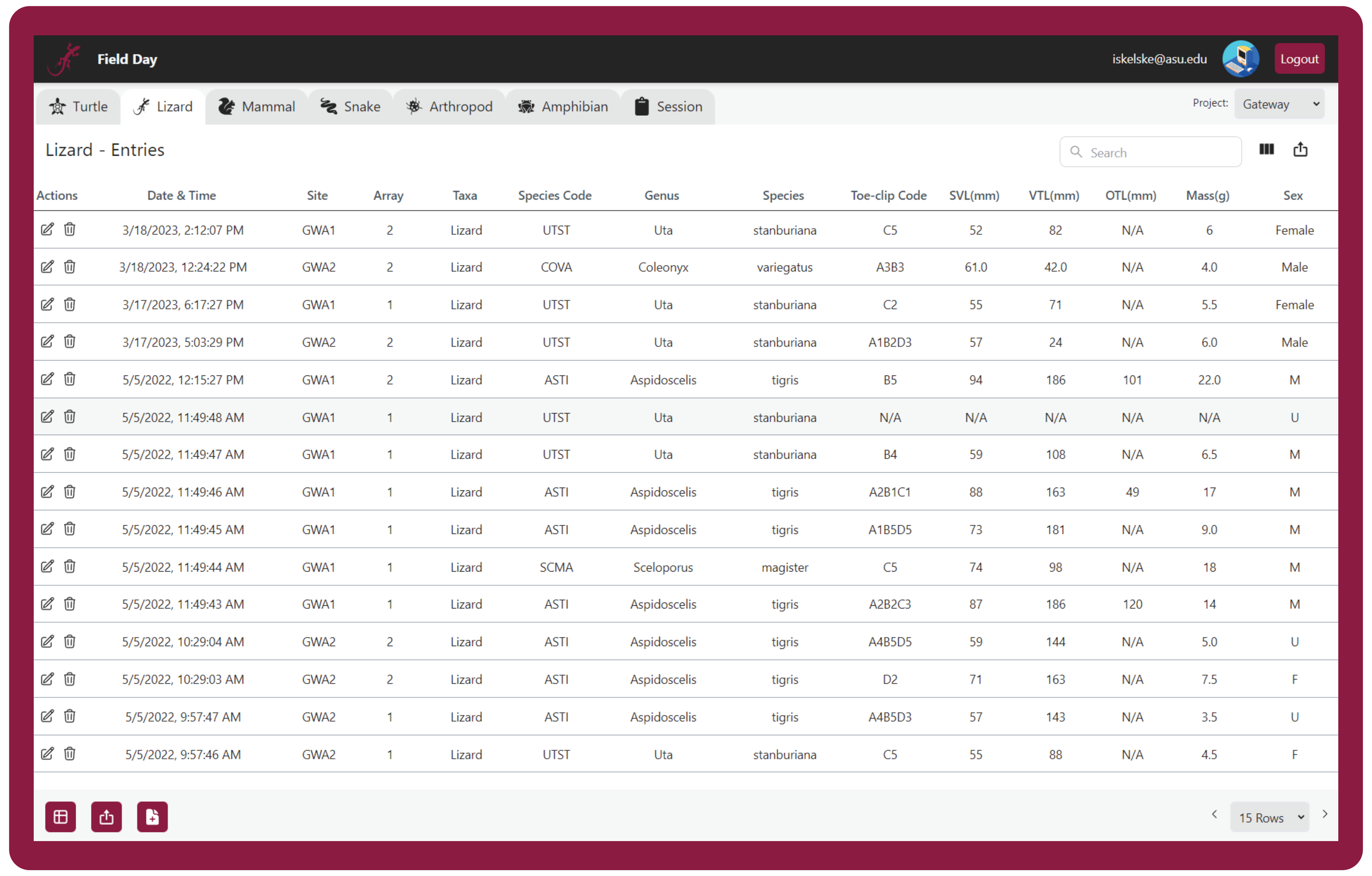
Our application's front end was developed using modern technologies and frameworks, including React, Vite, and Tailwind CSS. React provided us with a modular and component-based structure, allowing us to efficiently build the user interface. Vite, a fast build tool, enabled us to optimize our build times and improve our development workflow. Tailwind CSS, a utility-first CSS framework, made it easy to create a clean and responsive user interface.

Back End:

Our application's back end was deployed using Google Firebase and utilized Firestore as our database management system. Firebase allowed us to easily manage our backend services and reduced our overall deployment time. Firestore was used to store all of our data, providing us with a scalable and reliable solution for managing data. Additionally, we utilized Create-React-App to scaffold our project, streamlining our development process and allowing us to focus on building features.

Web App

The web app is a tool designed for use on larger screens, such as laptops, desktop computers, and tables, that enables users to view, manage, and export the data collected in the field using the PWA. Authentication security is enhanced with the use of Google authentication. To optimize performance and reduce costs, data is loaded dynamically from the database based on user activity. Users can efficiently view, search, edit, and sort the collected data in a table view, as well as export it to CSV format. Furthermore, the app allows for the management of answer sets, which are templates for data input that specify the permissible data types for a given field. This feature streamlines data entry and ensures data consistency, enabling researchers to more effectively analyze and interpret their findings.

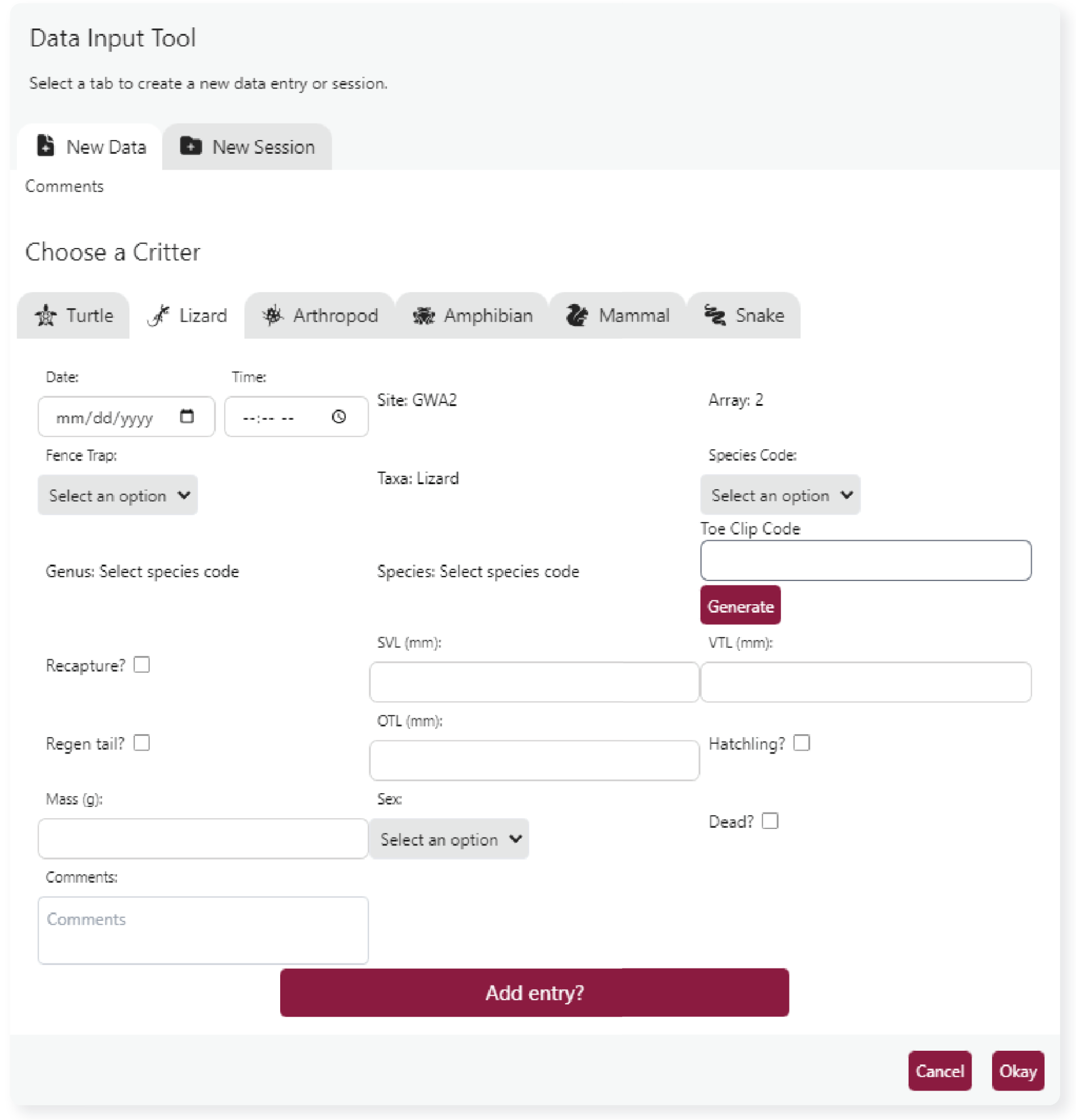


Lizard - Entries

Actions	Date & Time	Site	Array	Taxa	Species Code	Genus	Species	Toe-clip Code	SVL(mm)	VTL(mm)	OTL(mm)	Mass(g)	Sex
	3/18/2023, 2:12:07 PM	GWA1	2	Lizard	UTST	Uta	stanburiana	C5	52	82	N/A	6	Female
	3/18/2023, 12:24:22 PM	GWA2	2	Lizard	COVA	Coleonyx	variegatus	A3B3	61.0	42.0	N/A	4.0	Male
	3/17/2023, 6:17:27 PM	GWA1	1	Lizard	UTST	Uta	stanburiana	C2	55	71	N/A	5.5	Female
	3/17/2023, 5:03:29 PM	GWA2	2	Lizard	UTST	Uta	stanburiana	A1B2D3	57	24	N/A	6.0	Male
	5/5/2022, 12:15:27 PM	GWA1	2	Lizard	ASTI	Aspidoscelis	tigris	B5	94	186	101	22.0	M
	5/5/2022, 11:49:48 AM	GWA1	1	Lizard	UTST	Uta	stanburiana	N/A	N/A	N/A	N/A	N/A	U
	5/5/2022, 11:49:47 AM	GWA1	1	Lizard	UTST	Uta	stanburiana	B4	59	108	N/A	6.5	M
	5/5/2022, 11:49:46 AM	GWA1	1	Lizard	ASTI	Aspidoscelis	tigris	A2B1C1	88	163	49	17	M
	5/5/2022, 11:49:45 AM	GWA1	1	Lizard	ASTI	Aspidoscelis	tigris	A1B5D5	73	181	N/A	9.0	M
	5/5/2022, 11:49:44 AM	GWA1	1	Lizard	SCMA	Sceloporus	magister	C5	74	98	N/A	18	M
	5/5/2022, 11:49:43 AM	GWA1	1	Lizard	ASTI	Aspidoscelis	tigris	A2B2C3	87	186	120	14	M
	5/5/2022, 10:29:04 AM	GWA2	2	Lizard	ASTI	Aspidoscelis	tigris	A4B5D5	59	144	N/A	5.0	U
	5/5/2022, 10:29:03 AM	GWA2	2	Lizard	ASTI	Aspidoscelis	tigris	D2	71	163	N/A	7.5	F
	5/5/2022, 9:57:47 AM	GWA2	1	Lizard	ASTI	Aspidoscelis	tigris	A4B5D3	57	143	N/A	3.5	U
	5/5/2022, 9:57:46 AM	GWA2	1	Lizard	UTST	Uta	stanburiana	C5	55	88	N/A	4.5	F

Displayed above is the table viewer feature of the web application, enabling the efficient manipulation and removal of data entries, arranged and categorized by project and taxa. Displayed data is organized in small batches, the size of which can be adjusted with the dropdown located at the bottom right, optimizing the user experience by minimizing database read operations.

Data Input



Data Input Tool

Select a tab to create a new data entry or session.

Comments

Choose a Critter

Date: mm/dd/yyyy

Time: --:-- --

Site: GWA2

Array: 2

Fence Trap: Select an option

Taxa: Lizard

Species Code: Select an option

Toe Clip Code

Genus: Select species code

Species: Select species code

Recapture? ☐

SVL (mm):

VTL (mm):

Regen tail? ☐

OTL (mm):

Hatchling? ☐

Mass (g):

Sex: Select an option

Dead? ☐

Comments: Comments

Presented above is the data input modal, which overlays the table viewer and facilitates the entry of new session or animal data. The input fields are validated in a manner similar to that of the Progressive Web Application (PWA).

The photograph above depicts a specimen of the common side-blotched lizard (uta stansburiana) in the vicinity of the Gateway project.

Ira A. Fulton Schools of Engineering
Arizona State University