Database Management System for Monitoring the Effect of Climate on Wildlife Interactions
Progress Report 2

NCSU College of Veterinary Medicine
Dr. Suzanne Kennedy-Stoskopf
Dr. Lauren Charles-Smith

CSC 492 Team 12:

Bo Chulindra
Chris King
Scott Gentry

North Carolina State University
Department of Computer Science
Senior Design Center

November 7, 2011

I. Summary of Current Progress

New features added since last progress report:

- View/manipulate ArcGIS Online maps within the experiment page
- Overlay feature layers (which include data points). Each visualized data point provides a popup with a description of the point of data.
- View saved experiments.
- Calculate interaction data
- Automated web UI testing via Selenium.
- Automated server request/post tests.
- Locally hosted ArcGIS Server

II. Resources Needed

The following is a list of the hardware and software needed for the project. The list has been separated into hardware and software the project already has, and those that are needed.

Resources the project already has:

- Dedicated Server
- ArcGIS 10 license
- Pivotal Tracker
- GitHub
- Linux OS
- Python
- Hudson

Resources the project needs:

- Backup Storage for the database
- ArcGIS Server

III. Requirements

The following are a list of our requirements, broken up into functional and non-functional, and then broken up into components and the requirements for each of those components.

Functional Requirements:

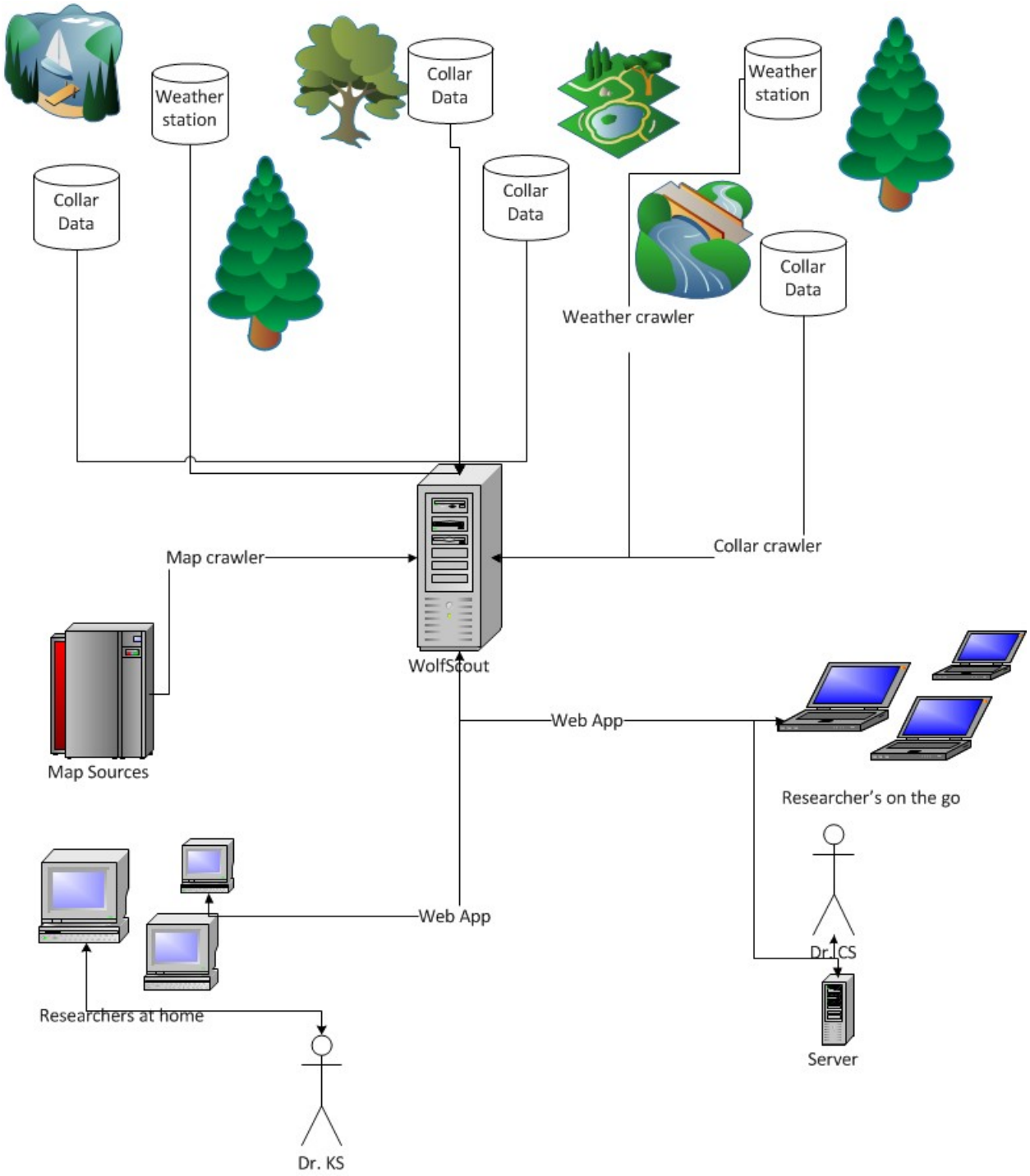
- Graphical UI
    - May be a web app or stand-alone GUI
        - Will depend on if a web app can provide the same functionality
    - Retrieves/store data from/to the database:
        - Collar data (temperature, time, location)
        - Climate data
        - Maps
    - Integrates with ArcGIS
        - Retrieves maps from ArcGIS
        - Provides functionality of ArcGIS Desktop
            - Overlay one or more feature layers onto a map
            - Display one or more feature layers without a map
            - Display multiple features layers at once
            - Zoom in/out
    - Authenticates the user
    - Calculates interaction data
- Database
    - Meets OpenGIS standards
    - Links data according to their time and locations
- Server
    - Reads the following data from a file:
        - Collar data
        - Weather station data
    - Stores data in and serves data from the database
    - Calculates the following:
        - Interaction data
        - Home-range

Nonfunctional Requirements:

- Database
    - Can handle 2 million GPS data points per month
    - PostgreSQL 8.4
    - PostgreSQL-8.4-PostGIS
- Server
    - Can handle 50-60 users at once
    - If fails, can be brought back up within a day.
    - Ubuntu 11.04+

    - Python 2.7.x
    - Supporting Packages:
        - Django==1.3
        - Fabric==1.2.0
        - PIL==1.1.7
        - South==0.7.3
        - coverage==3.5.1b1
        - django-nose==0.2
        - gunicorn==0.13.1
        - logilab-astng==0.22.0
        - logilab-common==0.56.1
        - nose==1.1.2
        - nosexcover==1.0.7
        - paramiko==1.7.7.1
        - psycopg2==2.4.1
        - pycrypto==2.3
        - pylint==0.24.0
        - unittest2==0.5.1
        - wsgiref==0.1.2
- Documentation
    - Non-trivial code is documented
    - Administrative server management is documented

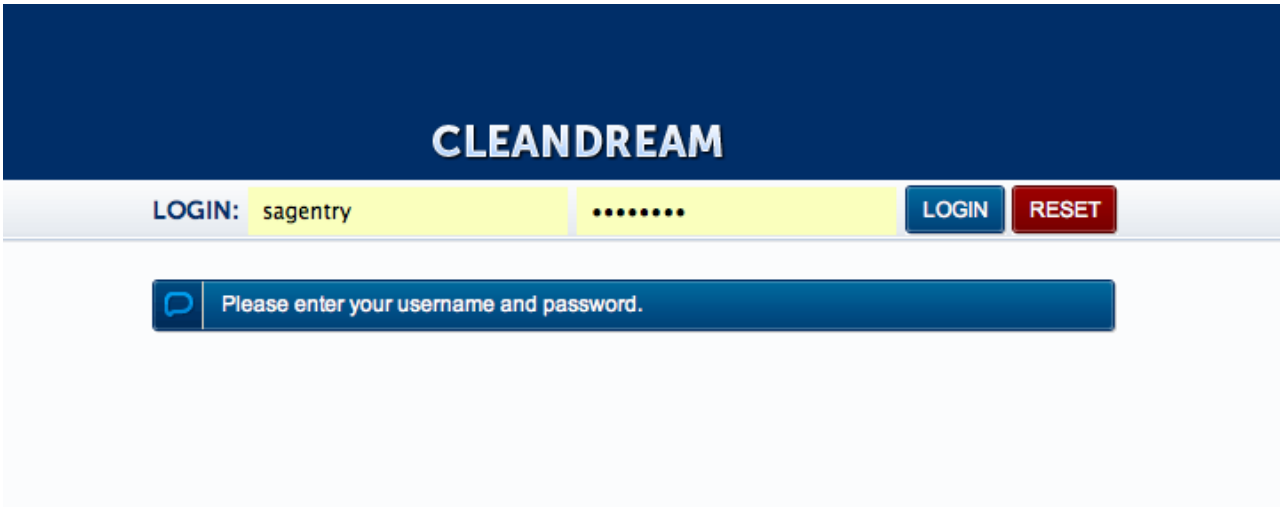
VIII. Design


Weather station
Collar Data
Collar Data
Collar Data
Weather station
Collar Data
Weather crawler
Map crawler
Collar crawler
WolfScout
Map Sources
Web App
Researcher's on the go
Web App
Researchers at home
Dr. CS
Server
Dr. KS

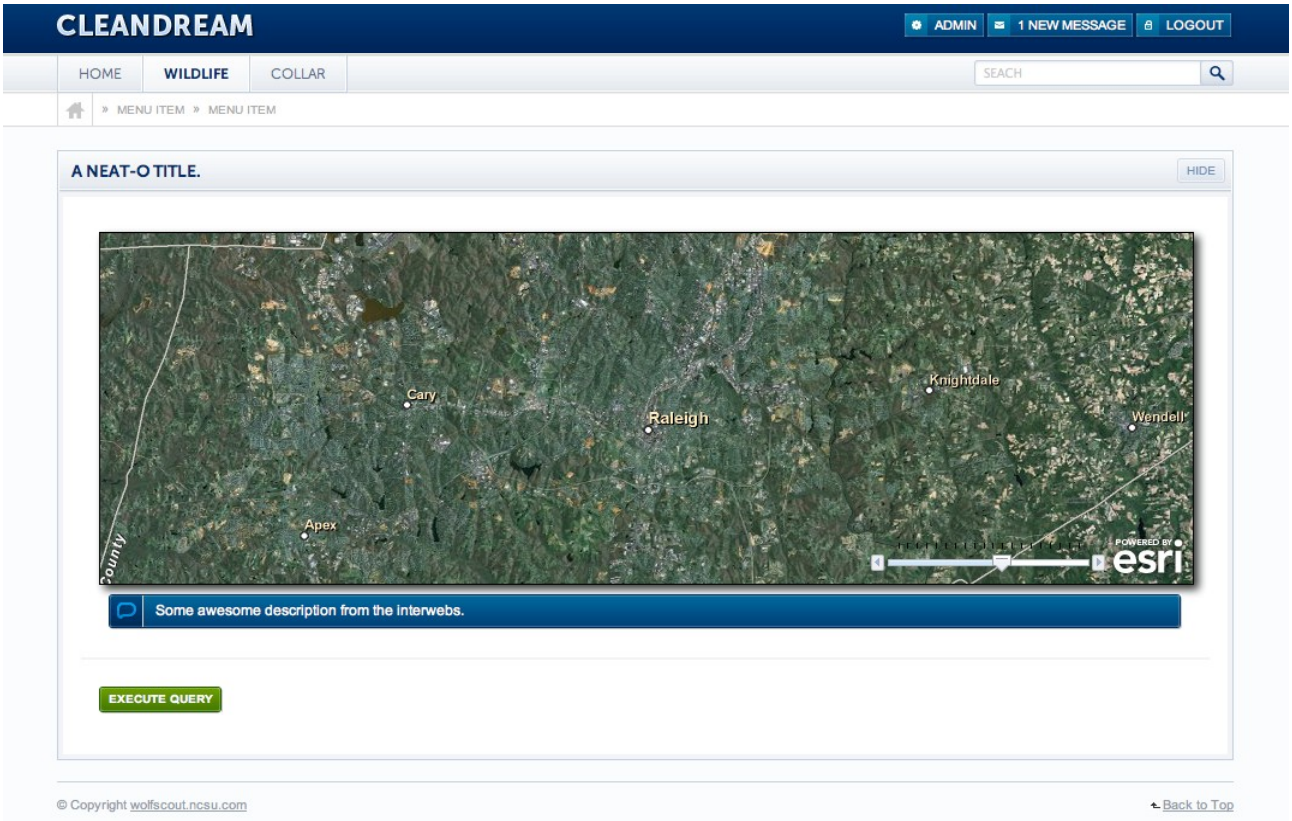
Above is a (overly simple) diagram of our overall architecture. The server, in the center and named WolfScout, grabs all the collar data, weather station data, and maps with crawlers. Users can then access that data via a web application.

Here are some screenshots of the prototype UI.

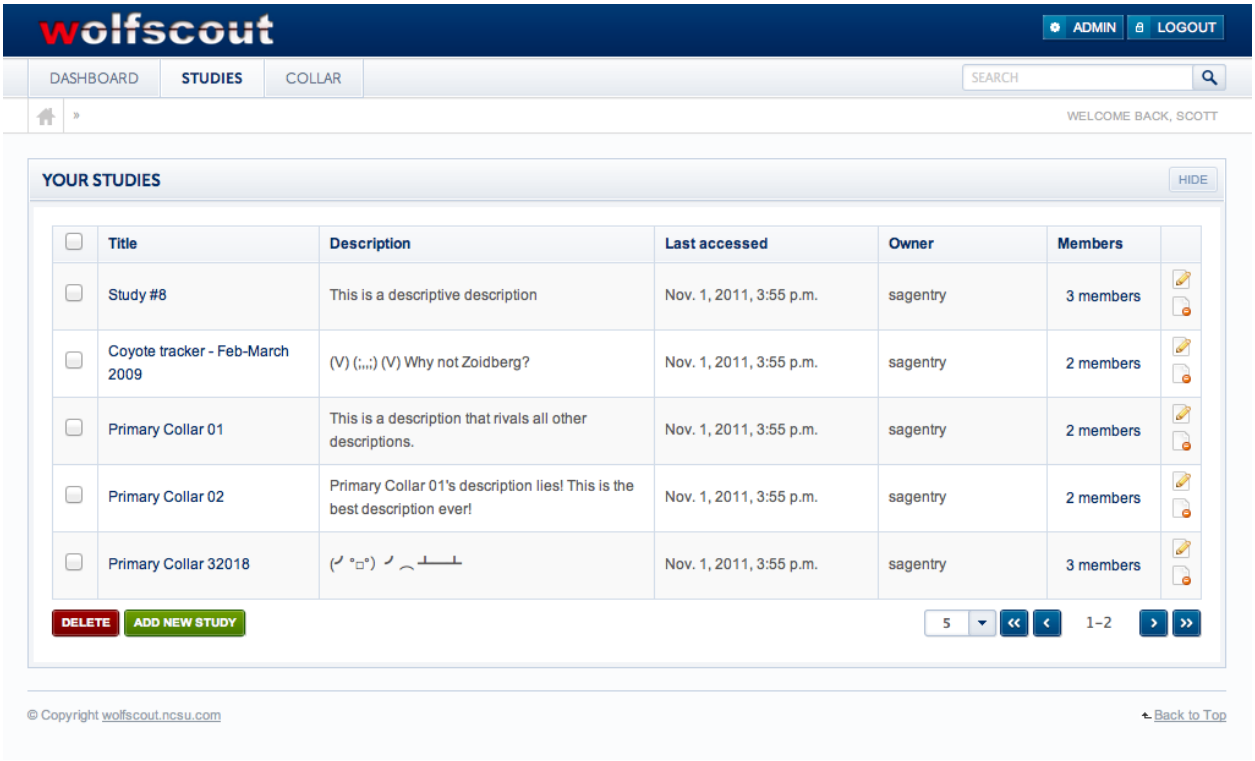
Login Page



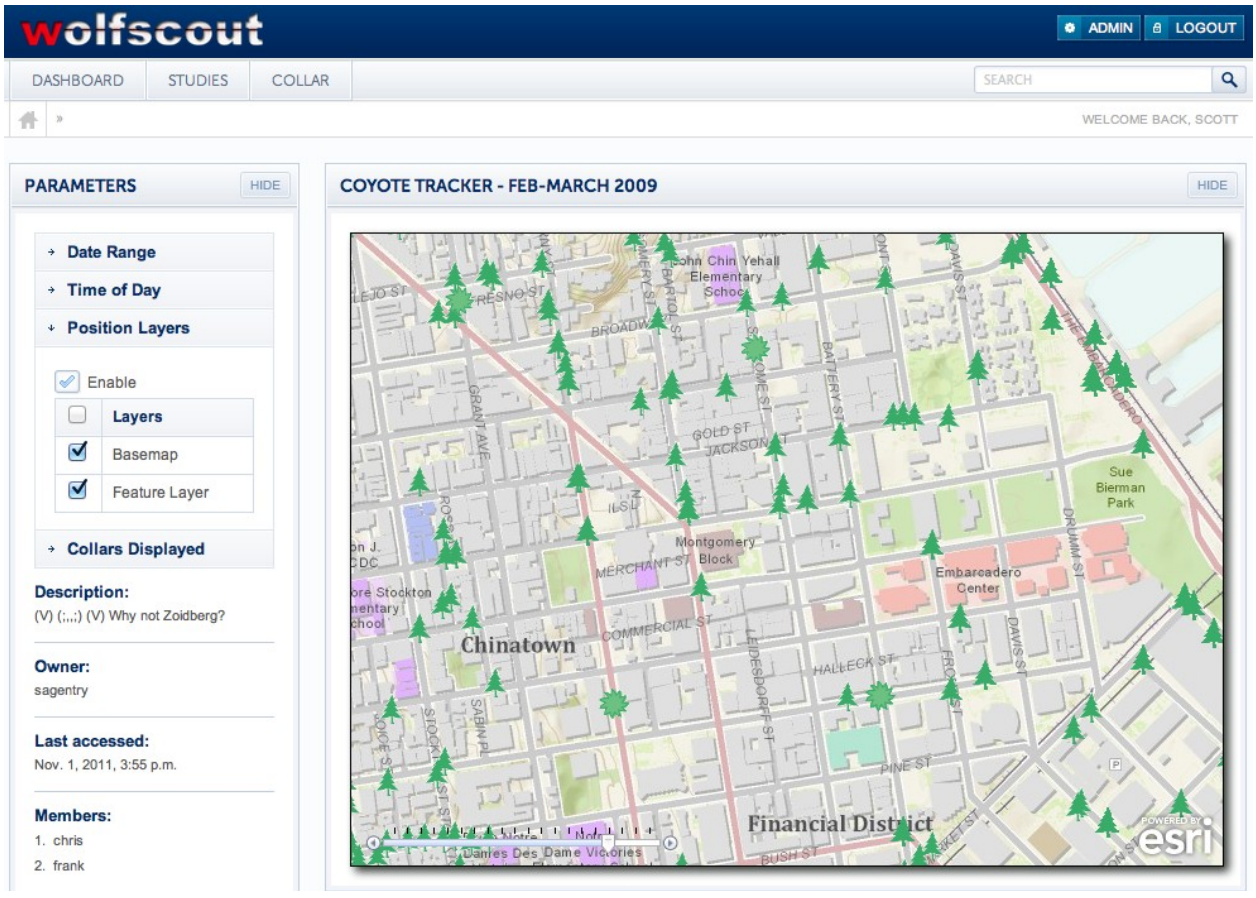
An early look at the UI allowing users to interact with maps



A list of experiments/studies the user has saved



An early look at data being overlaid on a map



IX. Implementation:

Code:

- apps/crawler/gpscollar
    - **collar.py**: Reads in collar data from a text file, parses it, and inserts into database
    - **models.py:** Provides a model for collar data. Models allow easy manipulation of database data from the web UI.
    - **views.py**: Pulls collar data from the database and outputs it in HTML
    - **tests.py:** Automated tests for collar.py
- apps/study
    - **views.py**: Pulls studies from the database and outputs it in HTML
- apps/study/migrations
    - **all**: Encapsulates database model changes into migration files. This eases migrations for developers on the team that need the changes. It also allows for reversing changes.
- tests/server
    - **tests.py**: Automated tests for POST and GET requests to the server
- tests/selenium
    - **allSeleniumTests.py:** Automated Web UI tests in Selenium.
    - **loginPage.py:** Provides support for the login page for Selenium testing
- **README.md**: Provides instructions for new developers to set up their environment. Can also be viewed by visiting https://github.com/NCSU-VSR/wolfscout and scrolling down until README.md is displayed
- **.gitignore:** File for indicating which files should not be tracked by version control
- **fabfile.py**: Provides methods to handle common source control actions such as commit, running tests, and updating local code
- **requirements.txt:** Specifies the required packages for the development environment.
- **urls.py:** Specifies which view is served for any particular URL request.

Development environment:

- GitHub
    - Source code is stored on GitHub at *github.com/NCSU-VSR/wolfscout*
- Server
    - Deployed on Senior Design lab machine
- Database
    - On Senior Design lab machine

X. Test Plan/Cases

Automated tests are written for collar.py. Our test plan is to have 80+% code coverage, which means that at least 80% of our code will have an automated test that steps through that code. All tests are run and must pass before any code can be committed or merged in the repository. We use a tool called Fabric to help enforce/facilitate this.

In addition to the automated tests, we have the following test scripts which will be manually run through periodically during development. We have not set a schedule for testing yet because the there is still not enough workable code to merit it.

Note that the test scripts below are not complete. As the system gains more function, more tests will be added.

Test ID	**Description**	**Expected Results**	**Actual Results**
login	Preconditions: You are not logged in to the UI. The server is up and running. 1. Open the UI home page.	You are shown the login page.	
login_validUser	Preconditions: You are not logged in to the UI. The server is up and running. User ‘user’ and password ‘password’ is registered in the system. 1. Open UI home page 2. Enter username ‘user’ and password ‘password’. 3. Submit.	You are logged in.	
login_invalidUser	Preconditions: You are not logged in to the UI. The server is up and running. User ‘bad’ is not registered in the system. 1. Open the UI home page. 2. Enter username ‘bad’ and password ‘bad’. 3. Submit.	You are not logged in. You are still shown the login page. You are shown a message that explains that login failed.	

XI. Task Plan

The following is our task plan for Iteration 1 (ends 9/28) and 2 (ends 10/12). In addition to these task items, we have user stories in Pivotal Tracker. If you don't have access to our project in Pivotal Tracker and would like it, please contact a member of our team and we will see about getting you access.

Item	Owner(s)	Due Date	Status
Iteration (11/7 – 11/21)	All	11/21	
* **Interactions**			
- animal to animal			
- animal to resource			
- supporting summary info			
* **Link to weather info**			
- change amount of info displayed for any point			
* **All maps up and running**			
- ability to turn on/off layers			
* **Use of ArcGIS in:**			
- in Internet			
- in stand alone			
* **Ability to enter new/edit data for each study**			
- ability to upload excel sheet with data on all animals and demographics on animals			
* **Ability to export data**			
- raw data			
- derived data			
- in a delimited text file			
- in an excel sheet			
- in a pdf			
* **Employ home range algorithms with UI**			
* **Tabs in GUI**			
- notes			
- studies (add/edit/download data)			
- map (view/manipulate/download)			
- animal interactions (calculate/download)			
- resource utilization (calculate/download)			
- home range (calculate/view/download)			