**Birdon theBrain:**Booty for Birders

Core Functionality

* **Signup**
* **Signin**
* **Signout**
* **Data Sourced from Users**
  + Sighting Lists:
    - Backyard
    - Individual Outing
    - Annual
    - Lifetime
  + List Functions:
    - Create
    - Read
    - Update
    - Delete
  + List Content for Each Bird Sighting:
    - common name of bird
    - county
    - state
    - date
    - comments (optional)
    - image (maybe)
  + Searches: Users can search other users' data, using filters like the following:
    - Name of user
    - County of user
    - State of user
    - Name of bird
    - Dates of sightings
* **Data Sourced from eBird API: Cornell's database can allow users to search for info like the following**
  + Most recent sighting of specific bird in given geographical region (city, county, state, country)
  + Locations of birding sites most visited by area birders
  + List of "interesting" birds sighted in given geographical area
  + List of sighting "hotspots" nearby
  + Bird Behaviors
  + Average Weight
  + Conservation Status (e.g., endangered)
* **Links to Major Birding Websites**
  + Merlin
  + eBird

Technologies:

* Flask
* Python
* Jinja
* SQL Database
* API Calls To: Internal and External Databases
* HTML
* CSS

Additional Functionality as Time Permits:

* Tokenization: Day 6
* Blog
* Connect to free image hosting site to allow users to upload profile pics and pics of birds they've sighted
* Bird Fact of the Day
* Bird Pic of the day
* Bird Game (maybe hangman with only bird names
* Users can click on bird name as they add to their list to get more info on that bird (still looking for API)

**Execution Plan:** Do main functions once, then go back and add iterations.

Keep it simple!

Test often!

1. Signin: (should be functional, but needs testing)
2. Register: ditto
3. Signout: ditto
4. User/Registration/Model/Form:
   1. Add fields to Class Registration From in authorms.py
   2. Update Registration form adding in new fields
5. Bird Model:
   1. ~~figure out how to add User id to bird model~~
   2. **Add columns:**
      1. Backyard [make this a checkbox]
      2. Annual (can tie to date column to keep annual lists separate)
      3. Lifetime
      4. Outing Label
6. Bird Form: see authforms for model
7. List Entry - CREATE BIRD SIGHTING:
   1. Create Database Model: (which should be bird model already created)
   2. Create Get route to render list entry page.
   3. Create Form Template:
      1. Should be able to have one simple page for all bird entries
      2. (Perhaps include checkbox now/later for backyard list)
      3. Would you like to associate these with a specific outing/trip? If so enter name here.
   4. Create post route:
      1. Get data from form template
      2. Input data into database
      3. Return success message to user.
      4. **User\_id is not yet being added to database**
      5. Check if in annual and lifetime lists and add if not.
      6. Redirect to list create page.
8. SEARCH INTERNAL / Read: (Start with only one type of search; get it working; get rest of basic functionality working, like by common name.)
   1. Create search by state. Need to search functions for all of the following still:
      1. user,
      2. common bird name,
      3. date,
      4. state,
      5. county
      6. backyard
      7. annual
      8. lifetime
   2. Create a Get Request Route to find data in database.
   3. Return data to Template. Have done for single object in a list (i.e. row), need to create loop in list\_search\_results.html to handle populating table with multiple rows.
9. DELETE BIRD FROM SIGHTING LIST:
10. UPDATE/ALTER ROW IN BIRD LIST:
11. **SEARCH/READ – Ebird:**
12. **HomePage:** 
    1. Cat fact
    2. Cat image

Upgrades, if time:

1. Create Sighting Form:
   1. Use calendar popup to standardize date and input properly (‘YEARMODY’) string
   2. Checkbox for Backyard list
   3. Dropdown county list.
2. Other?