Supplementary Material | Instruction manual for **catRlog**

Supplement to:

catRlog: A photo-identification project management system based in R

Keen EM, Wren J, O'Mahony É, Wray J.

Mammalian Biology

Corresponding author:
eric.k@marecotel.org

North Coast Cetacean Society
Marine Ecology & Telemetry Research
Sewanee: The University of the South

Setup | Setup the **catRlog** directory on your computer

1. Install R.
2. Install R Studio. Open it and make sure it runs properly.
3. **Recommended**: Install a .csv editing program (e.g., SublimeText) to use instead of Excel.
4. Download **catRlog** as a zipped folder from GitHub.
5. Unzip the folder.
6. Open **catRlog / z R / function-load-packages.R** in R Studio.
7. Run all the code in this R file to ensure that all necessary packages are able to load correctly & without issue.

(This next step is optional but highly recommended, and all instructions are given as if you have completed this next step; if not completed you will have to open up all R files from **catRlog / z R**)

8. In the **catRlog / z R folder**, for each enumerated R file, from **0.1 stage events.R** to **6 analysis dashboard.R**, create an alias (on Mac) or a shortcut (on Windows) and move it to the **catRlog** folder of the same number (e.g., **3.0 matching session.R alias** gets moved to the **catRlog / 3 matches** folder).

**** **DO NOT MOVE R files.** Only aliases/shortcuts thereof. ****

9. **catRlog** is ready for use! Test its functionality using the example data (pg. 30).

10a. If you are starting a new photo-ID project, go to page 2.

10b. If you already have a historical catalog and an encounter database that you are curating, go to page 3.

Getting started in catRlog | New users, new project

For new users with a new project (i.e., a group with one or more photo collections that have not yet been matched to one another), follow these steps:

1. Setup the catRlog directory on your computer (pg. 1).

2. Familiarize yourself with the **catRlog** system using the example data (pg. 30).

3. Bring in a single photo collection according to the instructions on pg. 5.

4. Use this collection to add the first IDs to what will become your historical catalog. (see pg. 11; bottom half of page).

5. Once the historical catalog has been started, finish processing this first photo collection (if only part of it was used to build the historical catalog) by stepping through Stages 1 – 5 (pg. 5 – 10 & 12 – 17).

6. Bring in subsequent photo collections and process them until your historical catalog is up to date.

7. Stages 5 and 6 can then be carried out as needed.

Getting started in catRlog | New users, pre-existing project

For users with pre-existing projects, follow these steps:

1. Setup the catRlog directory on your computer (pg. 1).
2. Familiarize yourself with the **catRlog** system using the example data (pg. 30).
3. Format data according to catRlog requirements (see Tables 1, 2, and 3 in the main text).
4. Begin by formatting your historical catalog correctly (see pg. 18 and Table 1 in main text).
5. If you have event tables (pg. 8), bring them into the **catRlog** directory according to directions and formatting requirements on pg. 8.
6. You may then bring in new photo collections and begin the processing sequence (starting on page. 5).

Overview of workflow | Recommended process for using **catRlog**

Stage 0 photos
(Download images from cameras into a separate folder for raw field photos.)

Best if completed after each day of field work

Copy photos into **catRlog / 0 photos.**
Remove extraneous images, crop images, and edit for matching. pg. 3

Stage 1 events
Prepare event table based on these photos, and place in **catRlog / 1 events.** pgs. 4 – 6
Fill in event tables with field observations (group membership, latitude, longitude, etc.)

Stage 2 scores
Score photos for distinctiveness and quality. pgs. 7 - 8

Carry out during the analysis season

[Setup historical catalog here, if you have not yet done so.] p. 9

Stage 3 matches
3.0 Match a photo collection to the historical catalog. pgs. 10 - 11
3.1 Reconcile match sessions from multiple analysts. pgs. 12 - 13
3.2 Register new arrivals in the historical catalog. pg. 14
3.3 Assign ID codes to events table(s) based on photo-identifications. pg.

Stage 4 catalog
Peruse digital flipbook version of historical catalog. pg. 16
Update catalog photos with better IDs from new photo collections. pg. 17

Stage 5 print
Produce print-ready PDF catalogues of historical catalog. pgs. 18 - 19

Stage 6 analysis
Compile datasets to address specific research questions. pgs. 20 - 23

Carry out after several sampling periods are processed

catRlog directory structure

- **0 photos**
  - **photos** — **Period Platform** `.jpg's` `.png's` `.bmp's`
  - **staged events** `.csv's`
  - 0.1 stage events `.R alias`
- **1 events**
  - Period Platform `.csv's`
  - **backups** `.csv's`
- **2 scores**
  - **score sessions** `.csv's`
  - 2 score photos `.R alias`
- **3 matches**
  - **match sessions** `.csv's`
  - **reviewed matches** `.csv's`
  - 3.0 matching session `.R alias`
  - 3.1 review matches `.R alias`
  - 3.2 register new IDs `.R alias`
  - 3.3 update events `.R alias`
- **4 catalog**
  - **catalog** `.jpg's` `.png's` `.bmp's`
  - catalog key `.csv`
  - **old keys**
  - **replaced photos**
  - 4.1 digital flipbook `.R alias`
  - 4.2 update catalog photos `.R alias`
  - fix history `.txt`
- **5 print**
  - **front matter**
  - **ID pages** `.pdf's`
  - 5 printing press `.Rmd alias`
  - settings `.csv`
- **6 analysis**
  - 6 analysis dashboard `.R alias`
  - **datasets** `.csv` `.rds`
    - events
    - groups
    - ILV
    - capture-history
    - dyads
    - association-matrix
- **z R**
  - 00 source functions `.R`
  - … `.R's`

Stage 0: Photos | Bring photographs into the **catRlog** system

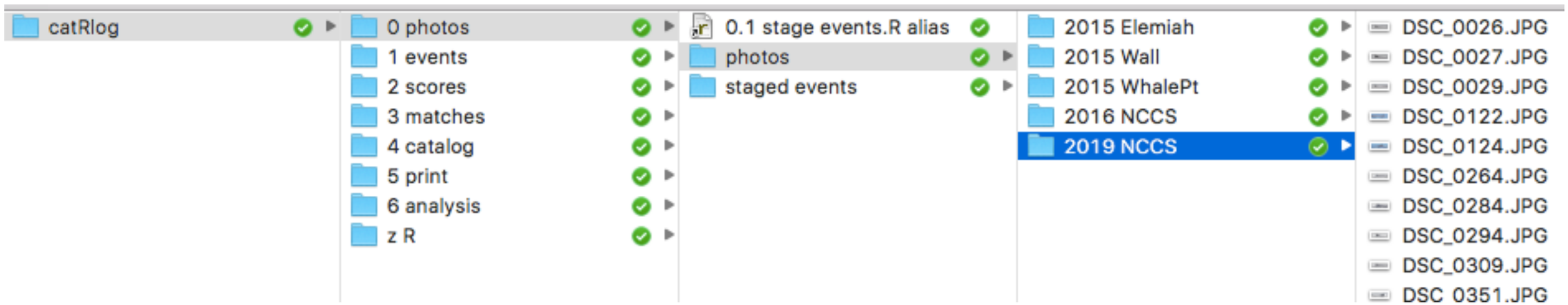
- Place your photo collections within **catRlog > 0 photos > photos**
- Each photo collection goes in a subfolder, named with the format **< Sampling Period > < space > < Platform >**
  - Example: “2019 NCCS”
- Image files can have any filename structure you wish.

Recommendation:
Rather than bringing in all photos collected in the field, copy only the photos useful for photo-identification into the catRlog system.

- Copy all photos into the catRlog system, then delete photos until you have the single best image of each unique feature of each individual within each unique group encounter.
- Then crop each image to the animal (catRlog works best with an aspect ratio of 3:1 width:height).
- Then adjust brightness and contrast as needed to facilitate matching.

(These measures ensure that there is a backup version of photos that remain unedited and uncropped.)

Example of directory layout:



Stage 0: Photos | Create an events table for a photo collection**, part 1**

In the next stage, 1: **events**, you will need a data table that includes a row for each photo-identification image within its corresponding photo collection. Often, the most efficient way to create this data table is to begin with a list of photos in the collection then manually add notes based upon field observations.

To automatically generate the base data table for a photo collection, follow these steps:

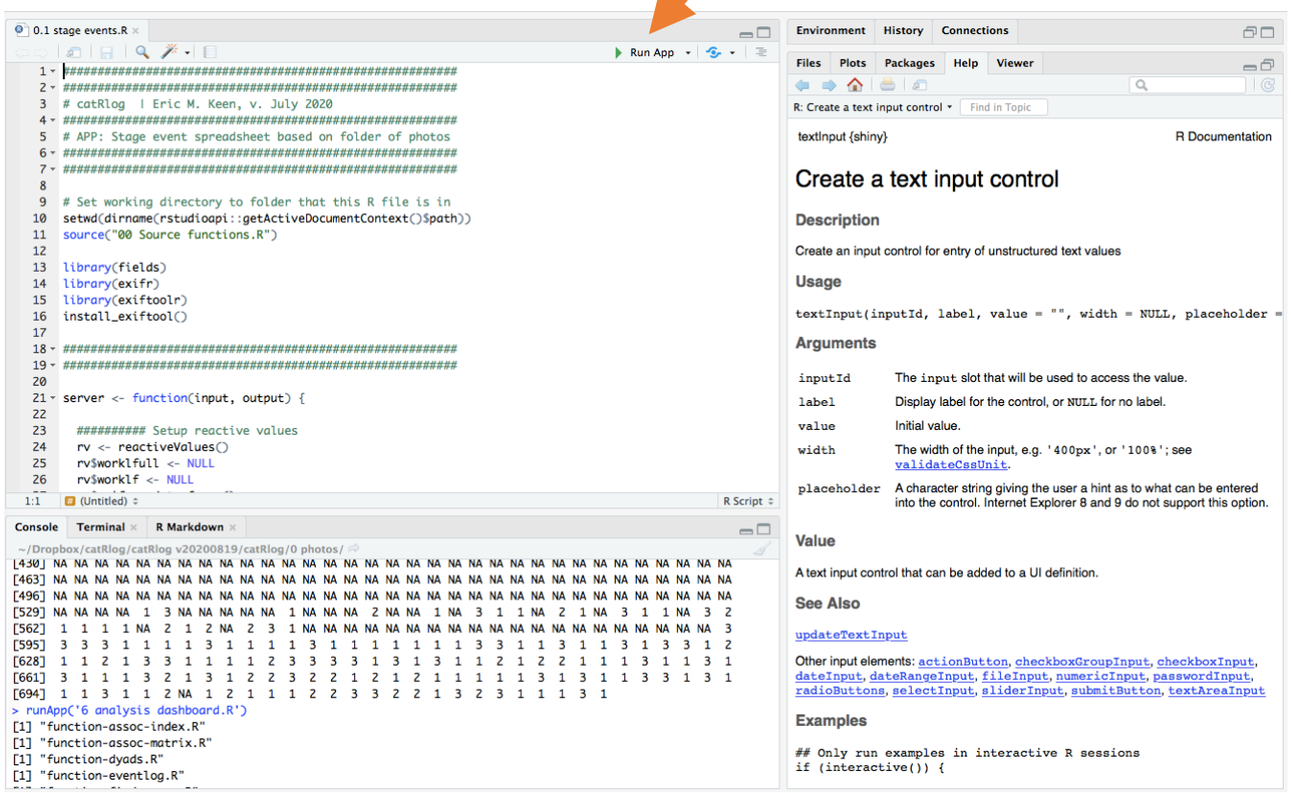
1. Go to **catRlog/0 photos/**.

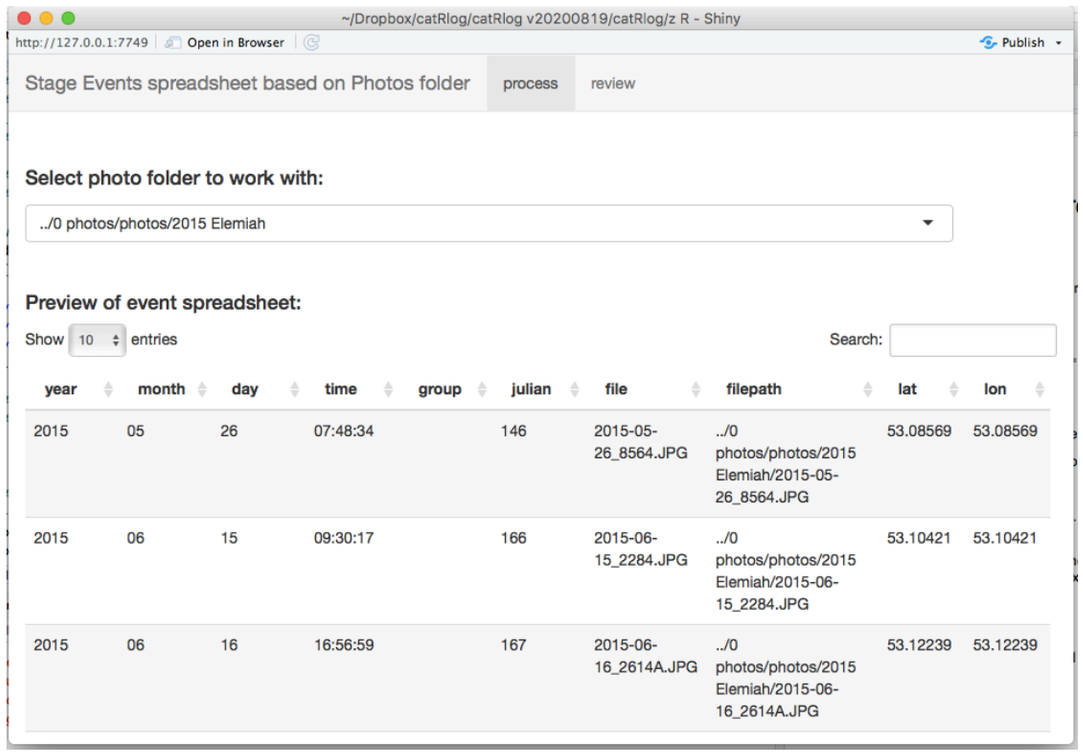
2. Place your photo collection within the subfolder 'photos' *(see previous page of this manual).*

3. Open '**0.1 stage events.R**' in R Studio

4. Click 'Run App'.

Click 'Run App' to launch this app





5. After clicking 'Run App', the app window appears.

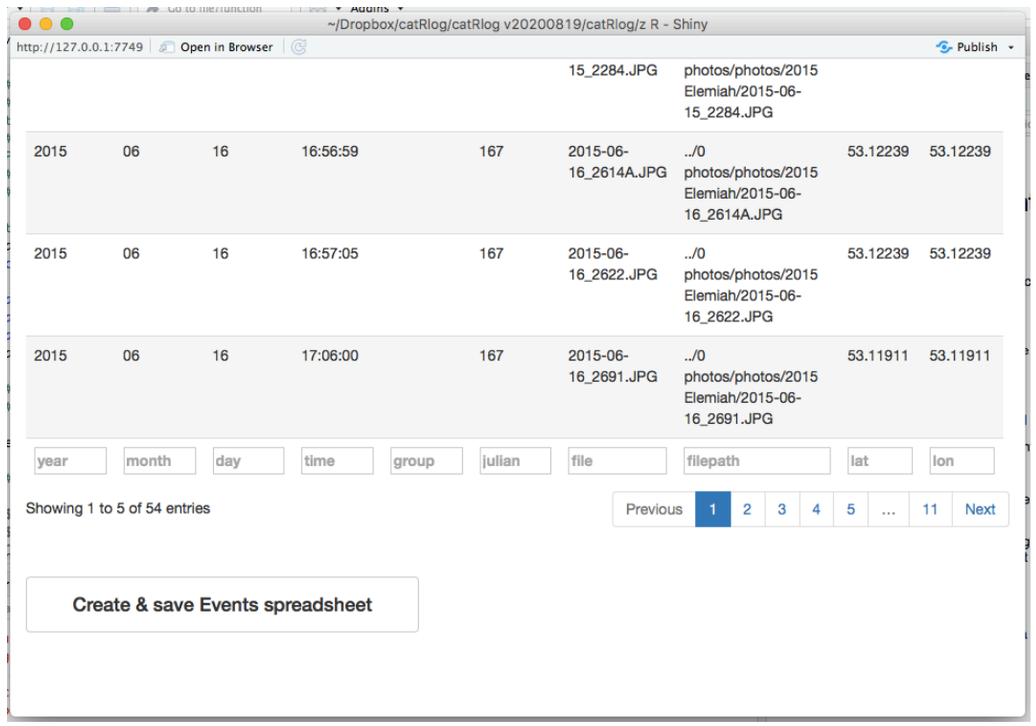
6. Select the photo collection for which to produce an events table.

7. Below you see a preview of what this data table will look like.

Stage 0: Photos | Create an events table for a photo collection**, part 2**

8. If you wish to review the image file metadata being used to produce this events data table, check out the '**review**' tab.

9. Back on the 'process' tab, scroll down and click **"Create & Save Events Spreadsheet".**



10. The event table is saved as a spreadsheet (.csv) to the folder **catRlog > 0 photos > staged events.**

11. Note that its name exactly matches that of its corresponding photo collection.

12. Move this spreadsheet file into **catRlog > 1 events.**

Note: This step exists to prevent the act of accidentally overwriting an events spreadsheet that you have already manually augmented with field observations.

13. Now open the spreadsheet and fill in the 'group' column to specify the group that each image corresponds to.

Use a unique group identifier (numeric) for each date. (i.e., group number can reset each new day of effort).

Example of an events spreadsheet produced automatically using the 'stage events' app. Note the 'group' column that has been staged for the user.

year	month	day	time	group	julian	file	filepath	lat	lon
2016	06	07	14:51:59		159	2016-06-07_5560.JPG	../0 photos/photos/2016 NCCS/2016-06-07_5560.JPG	53.0537638888889	-129.261383333333
2016	06	07	14:52:52		159	2016-06-07_5571.JPG	../0 photos/photos/2016 NCCS/2016-06-07_5571.JPG	53.043925	-129.271230555556
2016	06	07	14:53:17		159	2016-06-07_5585.JPG	../0 photos/photos/2016 NCCS/2016-06-07_5585.JPG	53.043925	-129.271230555556
2016	06	07	14:53:35		159	2016-06-07_5593.JPG	../0 photos/photos/2016 NCCS/2016-06-07_5593.JPG	53.043925	-129.271230555556
2016	06	07	15:11:13		159	2016-06-07_5875.JPG	../0 photos/photos/2016 NCCS/2016-06-07_5875.JPG	53.0437694444444	-129.280594444444
2016	06	07	15:14:32		159	2016-06-07_5911.JPG	../0 photos/photos/2016 NCCS/2016-06-07_5911.JPG	53.0407	-129.280294444444
2016	06	07	15:25:00		159	2016-06-07_5997.JPG	../0 photos/photos/2016 NCCS/2016-06-07_5997.JPG	53.0423222222222	-129.281911111111
2016	06	09	07:06:35		161	2016-06-09_6173.JPG	../0 photos/photos/2016 NCCS/2016-06-09_6173.JPG	53.0985194444444	-129.197377777778

Note: This app uses an R package 'exiftool' to extract metadata from the files in your photo collection. If the camera stored latitude and longitude with each photo, these fields will be automatically filled in within the event data table.

Stage 1: Events | Complete preparation of events table for a photo collection

Every events table in **catRlog > 1 events** corresponds to a photo collection in **catRlog > 0 photos > photos.**

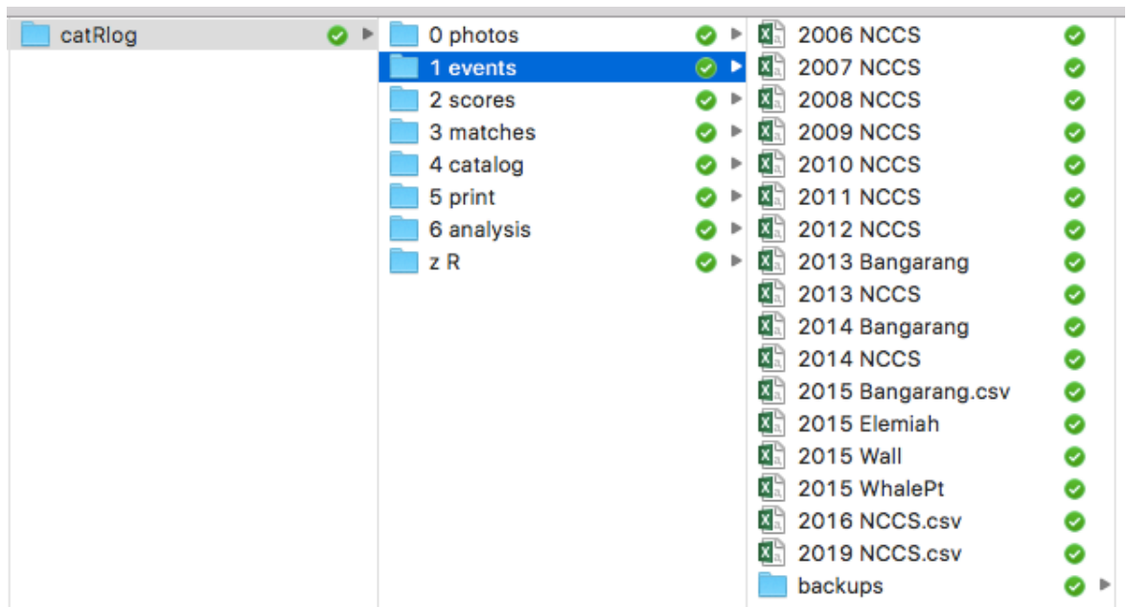
The filename of each events table must match the name of its corresponding photo collection.

Each row in the events table corresponds to a single image file within the photo collection.

Note: an events table can be generated automatically based upon the photo collection. See previous pages.

Note: you are free to have additional rows in this events table if you wish to document a sighting that was not photo-identified. The '**file**' field for such rows should be '**NA**'.

Example of directory layout:



Required columns (in any order)**:**

- '**file**' (image filename in the corresponding photos collection; the filetype extension is not required)
- '**year**' (format YYYY),
- '**month**' (values 1-12),
- '**day**' (1-31),
- '**group**' (a unique numeric identifier for each date of field effort).

Optional columns recognized by catRlog:

- '**id**' (providing the historical catalog ID; if not provided, this column will be created and filled in during the matching process)
- '**lat**' (Latitude, in decimal degrees; negative values indicate Southern Hemisphere),
- '**lon**' (Longitude, in decimal degrees; negative values indicate 0 W – 180 W).

Note: Other columns with unique names are fine and are ignored.

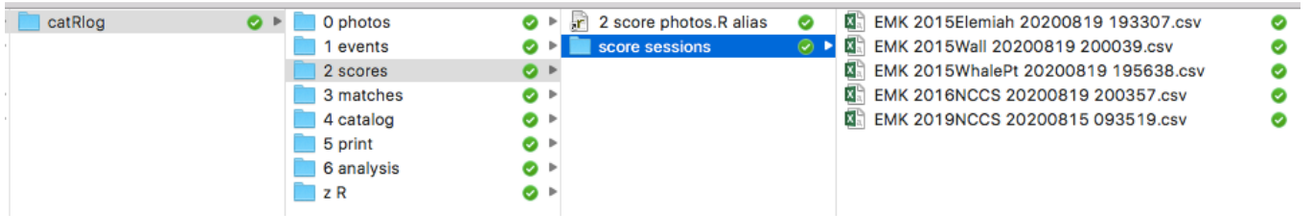
Example of the minimum required information in an events table.

file	year	month	day	group
0812_02Squally_7DE_47927.JPG	2014	8	12	1
0812_02Squally_7DE_48270.JPG	2014	8	12	2
0812_02Squally_7DE_48299.JPG	2014	8	12	2
0812_02Squally_7DE_48233.JPG	2014	8	12	2
0812_02Squally_7DE_48003.JPG	2014	8	12	3
0812_02Squally_7DE_47930.JPG	2014	8	12	3
0812_02Squally_7DE_48208.JPG	2014	8	12	4
0812_02Squally_7DW_47710.JPG	2014	8	12	4

Stage 2: Scores | Rate each photo by quality & distinctiveness, **part 1**

Use the **scoring app** to efficiently rate each photo in a collection according to standard metrics.

Example of directory layout: image scores are saved in 'session' files. :



1. Go to **catRlog/2 scores/**.

2. Open '**2 score photos.R**' in R Studio

3. Click 'Run App'.

4. The first tab asks that you enter analyst initials. Doing so unlocks the option to select a 'scoring session' file where your work will be saved. You can 'Start a new session' by providing a 'Session Description' (**tip**: I usually use the name of the photo collection).

5. Select the photo collection you wish to score.

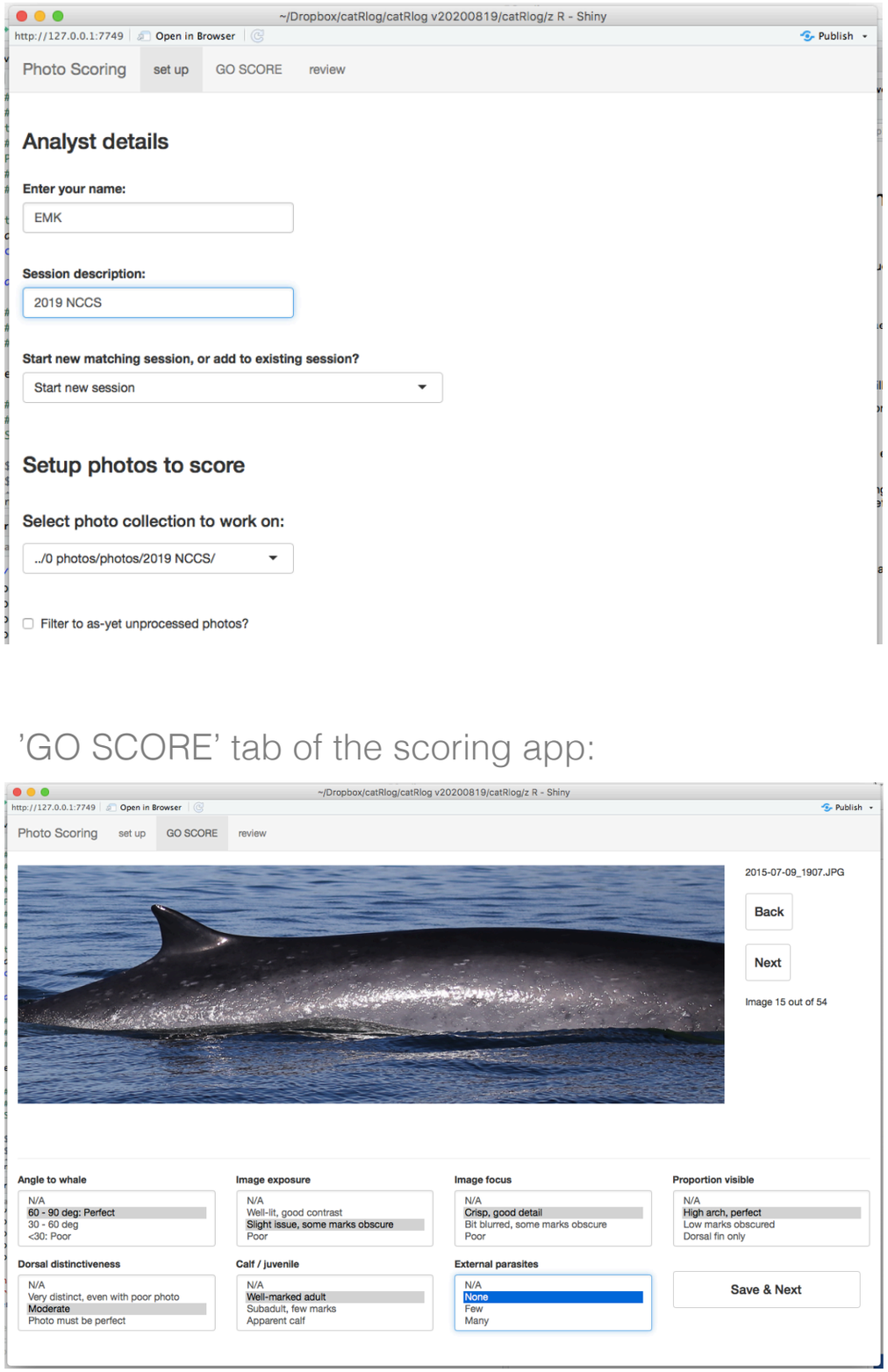
Note: you have the option to filter photos down to those that have not yet been scored according to the scoring session file you have selected. This is handy if you are scoring a collection across multiple work sessions.

6. Go to the 'GO SCORE' tab. Each photo in the collection will be displayed.

7. Select quality scores for each field. Once all fields have been changed from NA, the option to click "Save & Next" will appear. This will save your scores to the scoring session.

8. When you are finished scoring photos, simply close the app.

'Setup' tab of the scoring app:



'GO SCORE' tab of the scoring app:

Stage 2: Scores | Rate each photo by quality & distinctiveness, **part 2**

Example of the scoring session file (saved to catRlog > 2 scores > score sessions:

2020-08-15 09:35:48	EMK	/2019 NCCS/	DSC_0026.JPG	../0 photos/photos/2019 NCCS/DSC_0026.JPG	1	3	2	2	2	1	1
2020-08-15 09:40:38	EMK	/2019 NCCS/	DSC_0027.JPG	../0 photos/photos/2019 NCCS/DSC_0027.JPG	1	3	3	3	2	1	1
2020-08-15 09:40:59	EMK	/2019 NCCS/	DSC_0122.JPG	../0 photos/photos/2019 NCCS/DSC_0122.JPG	1	1	1	1	1	1	1
2020-08-15 09:41:41	EMK	/2019 NCCS/	DSC_0122.JPG	../0 photos/photos/2019 NCCS/DSC_0122.JPG	1	1	1	1	1	1	1
2020-08-15 09:41:55	EMK	/2019 NCCS/	DSC_0124.JPG	../0 photos/photos/2019 NCCS/DSC_0124.JPG	1	2	1	1	2	1	1
2020-08-15 09:42:20	EMK	/2019 NCCS/	DSC_0264.JPG	../0 photos/photos/2019 NCCS/DSC_0264.JPG	2	2	2	1	1	1	1
2020-08-15 09:42:35	EMK	/2019 NCCS/	DSC_0284.JPG	../0 photos/photos/2019 NCCS/DSC_0284.JPG	1	2	1	1	1	1	1

Column key:

1 Date/time of scoring decision
2 Analyst initials
3 Photo collection
4 Image filename
5 Full path to image file (referenced from catRlog > z R)
6 Angle to animal (1 = option 1 in app; 2 = option 2 in app; 3 = option 3 in app)
7 Image exposure
8 Image focus
9 Proportion of feature visible
10 Feature distinctiveness
11 Calf / juvenile
12 External parasites

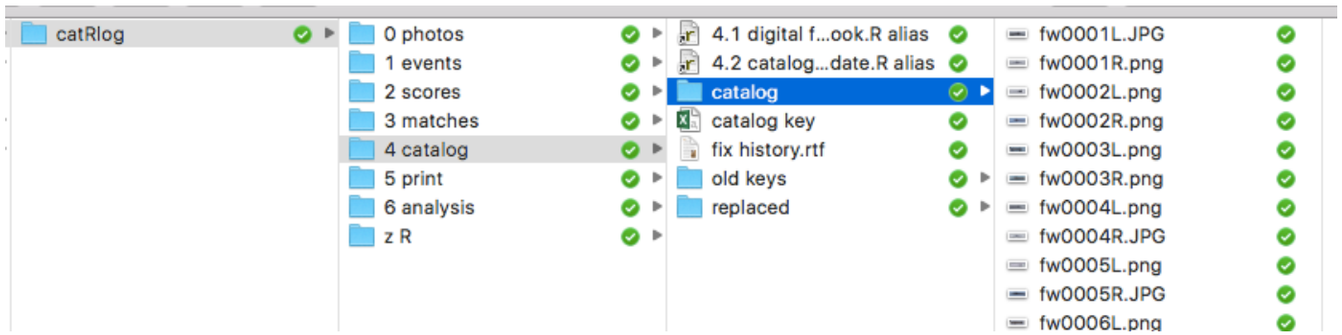
Note: While scoring photos is logically Stage 2 of the project management process, it doesn't *NEED* to happen until you reach Stage 6, in which you are compiling datasets based upon quality scores and distinctiveness. So it is fine to carry out matching before or during the process of scoring images. In fact, scoring can be done in the absence of historical catalog data in 'catRlog/4 catalog/'

Note: To customize quality scores for your specific needs, you need to enter the corresponding R code for the Shiny app in catRlog > z R > 2 score photos.R. Relevant sections of code occur at roughly line 160 in the server, and line 266 in the ui.

Checkpoint:
Already have a historical catalog?

If you are already curating a historical catalog outside of the catRlog system, bring it in:

A. ID photos go in **catRlog / 4 catalog / catalog**



- Each photo must be named as follows:
- "< ID code >< feature code > . < file type extension > "
- The ID code can be any combination of letters and numbers.
- The feature code must be only a single character.
- *Example:* "fw0001L.jpeg", is the left dorsal of individual fw0001.

B. Create **catRlog / 4 catalog / catalog key.csv**
(or use the template provided in **catRlog**)

id	local	injury	nick	hole	calf	mother	distinctiveness	dfo
fw0001	Bering McFinnigan	0	0	0	0	0	3	FWC
fw0002	Pecos	0	0	0	0	0	2	FWC
fw0003	Lag	0	1	0	0	0	1	FWC
fw0004		0	0	1	0	0	2	FWC
fw0005	Drifter Lonesome	0	1	0	0	0	1	FWC
fw0006		0	0	0	0	0	3	FWC
fw0007	Haskell	0	0	1	0	0	1	FWC
fw0008	Tunix	0	1	0	0	0	1	FWC
fw0009	Caribou MARVELOUS	0	1	0	0	0	1	FWC
fw0010	Garbanzo Von Trapp	1	1	0	0	0	1	FWC

- In this spreadsheet, each row is an individual.
- Required fields:
- **id =** ID code, corresponding to an image in the **catalog** folder. (do not include the feature code)
- **local =** Local nickname / identifier (leave blank if not used)

For the following fields, use 0 if not relevant:
- **injury =** 0 (absent) or 1 (present)
- **nick =** Are there nicks or small injuries on the dorsal fin?
- **hole =** Are there holes in the dorsal fin?
- **calf =** was this individual first seen as a calf?
- **mother =** Is this individual a known mother?

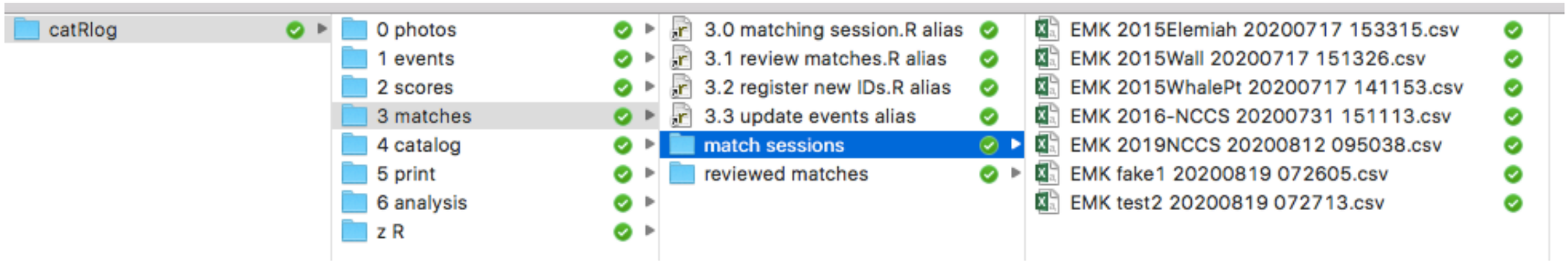
If you are starting a new project that does not yet have a historical catalog yet, follow these steps:

Take the first photo collection you wish to bring in and use it (or a part of it) to establish a historical catalog.

1. Find some easily identifiable individuals within the photo collection, copy them into **catRlog / 4 catalog / catlog**, and rename them according to **A** above.
2. Add rows in **catRlog / 4 catalog / catalog key.csv** that correspond to these new IDs, according to **B** above. Fill out trait / mark fields.
3. After a few entries are made in the **catalog** folder and **catalog key.csv,** the apps within catRlog should be functional.
4. Treat your first photo collection as you would any other one (see pg. 4) and go through the matching process (you will have a lot of new whales!), review those matches, then use the "Register new whales" app (see pg. 15) to update the historical catalog.

Stage 3: Matches | Carry out a matching session, **part 1**

Example of directory layout: Match decisions are saved in 'match session' files. :



1. Make sure you have placed your photo collection within the subfolder 'photos', if you have not yet done so *(see previous page of this manual).*

2. Go to **catRlog/3 matches/**.

3. Open '**3.0 matching session.R**' in R Studio

4. Click 'Run App'.

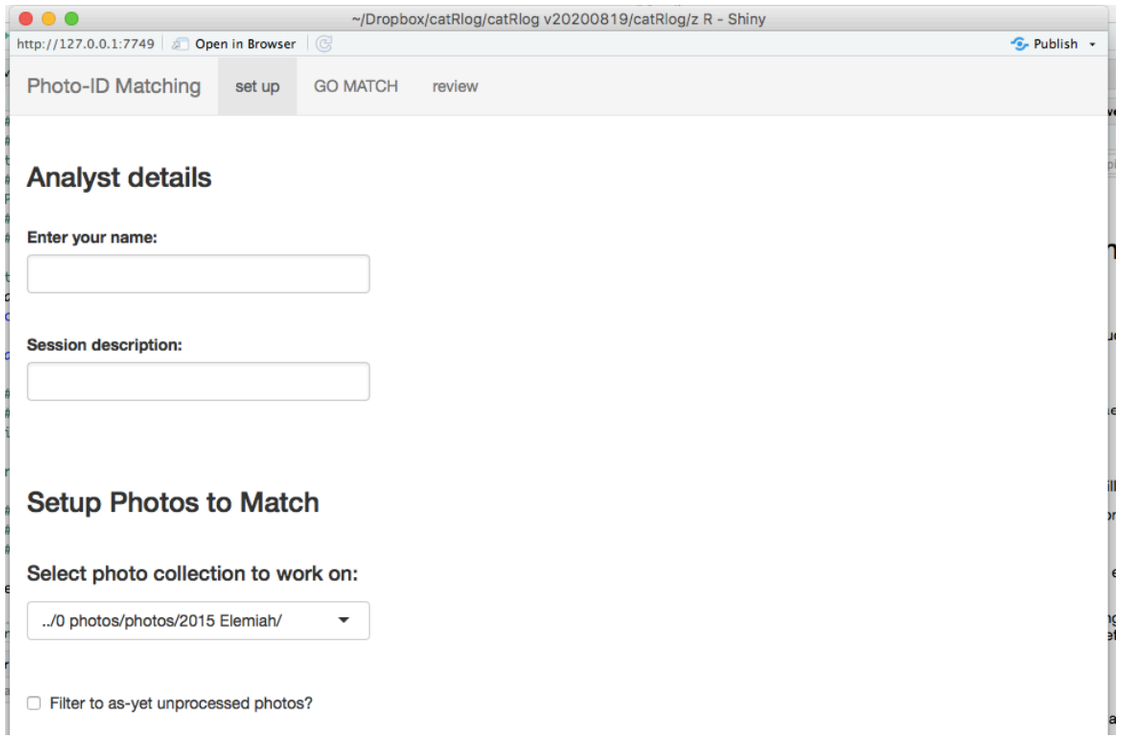
5. The first tab asks that you enter analyst initials. Doing so unlocks the option to select a 'matching session' file where your work will be saved. You can 'Start a new session' by providing a 'Session Description' (**tip**: I usually use the name of the photo collection I am working on).

6. Select the photo collection you wish to match.

Note: you have the option to filter photos down to those that have not yet been matched according to the matching session file you have selected. This is handy if you are matching a collection across multiple work sessions.

7. Go to the 'GO MATCH' tab. Each photo in the collection will be displayed.

'Setup' tab of the matching app:



Stage 3: Matches | Carry out a matching session, **part 2**

The top image is from the photo collection. This is the photo-ID you are trying to find in the Reference (historical) catalog.

The bottom image is from the Reference (historical) catalog (**catRlog > 4 catalog > catalog**). Information about this image is being pulled from 'catalog key.csv' in **catRlog > 4 catalog**

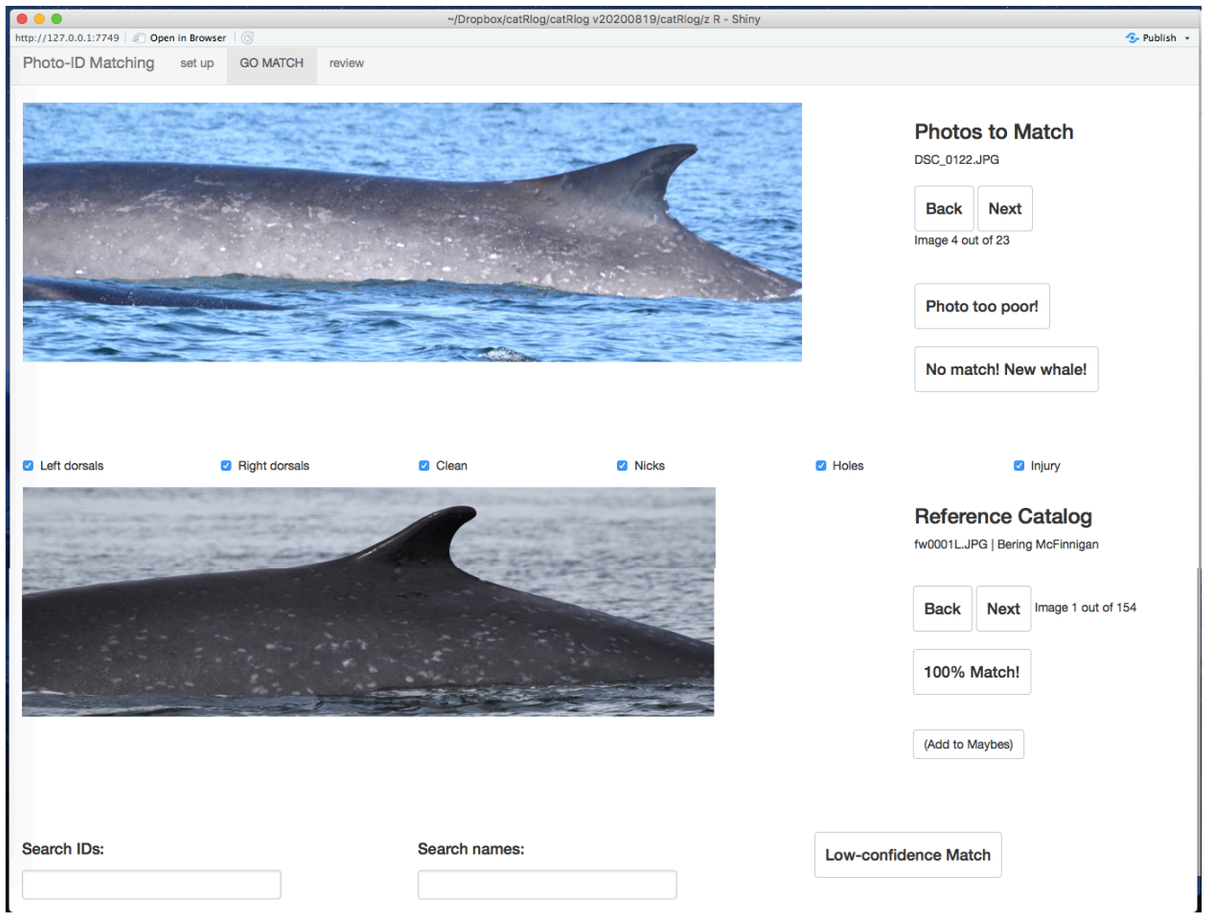
Recommended workflow:
For each photo to match, flip through the reference catalog to look for a match.

- You may filter the reference catalog according to features and marks. (See the checkboxes halfway down the screen.) These checkboxes correspond to columns in 'catalog key.csv'.
- If you know of an individual in the catalog that might be the match, you may search for it based on ID code and/or local name (you may need to scroll down to see this feature).
- If the query is not easy to identify and several reference catalog IDs may qualify as a match, use the 'Add to maybes' button to track candidate matches. After going through the full catalog, you can then click the checkbox to 'Filter to maybes' (this will only appear after the button 'Add to Maybes' has been clicked once) and look more closely at the candidates you selected.

- Make a decision:
  - If you found a match and are 100% confident, click the "**100% Match**" button.
  - If you may have found a match but are not certain, click the "**Low-Confidence Match"** button. (**Note:** You may need to scroll down to see this option)
  - If the photo is simply too poor to find a match with confidence, click the '**Photo too poor!'** button.
  - If the photo is of an individual that is definitely not in the reference catalog, click '**No match! New whale!'.**

  - Clicking any of these buttons will save your match decision and automatically take you to the next photo to be matched.

'GO MATCH' tab of the matching app:



8. If you wish to review the match decisions you are saving, check out the '**review**' tab.

9. Once you are done matching all the photos in this collection, simply close the app.

Stage 3: Matches | Review and reconcile matching sessions, **part 1**

In most applications, multiple analysts will attempt matching a photo collection to the historical catalog. These redundant decisions must then be reconciled in order to arrive at a final decision for each photo-identification. This step will produce a single, finalized match record

Note: This step is necessary regardless of how many analysts have worked on a photo collection, even if only a single analyst was used.

1. Go to **catRlog/3 matches/**.

2. Open '**3.1 review matches.R**' in R Studio

3. Click 'Run App'.

4. Under the first tab, 'setup', select the match sessions that you want to reconcile. You can select several.

5. Once you select these match sessions, the option to "Import Match Sessions" will appear. Click that button.

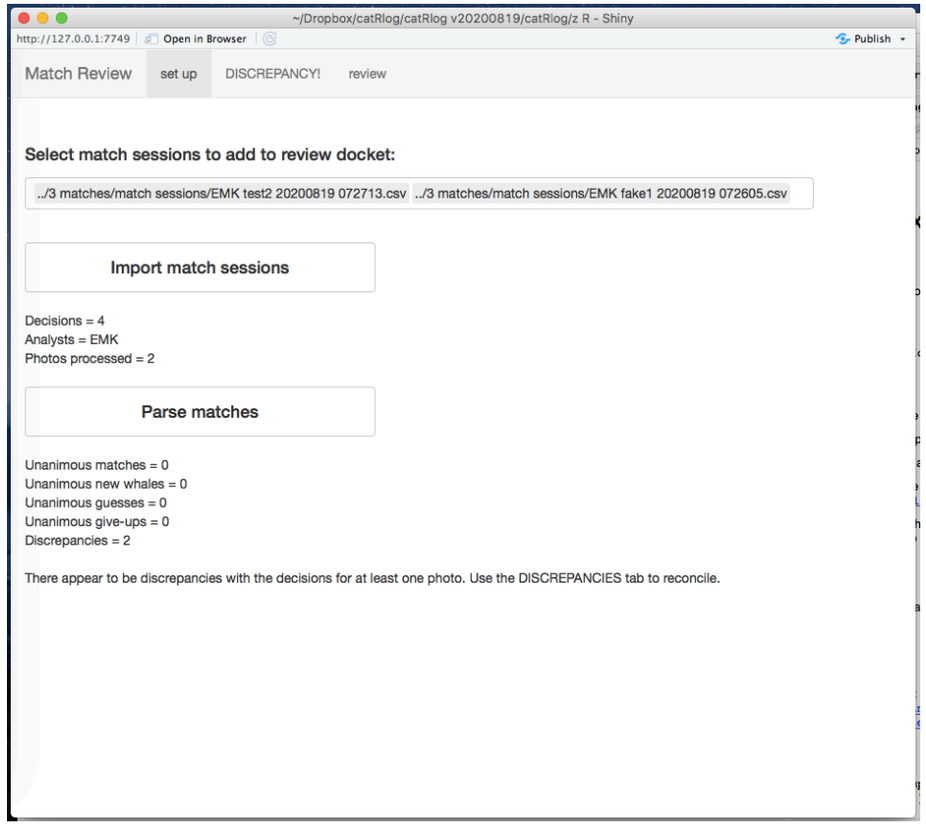
6. After clicking, a new button will appear to "Parse matches" into sets of unanimous decisions and 'discrepancies' (i.e., inconsistent match decisions amongst analysts). Click it.

7a. If there are any discrepancies, you will be alerted as such. *(see screenshot to top-right for an example.)* In this case, go to the "DISCREPANCY" tab *(see next page).*

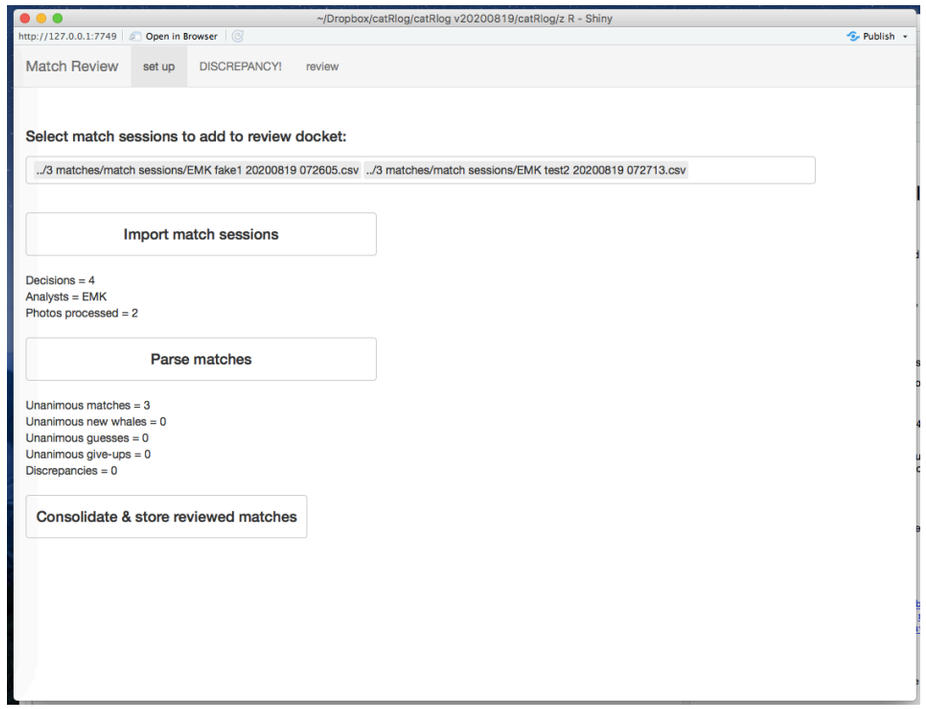
8. If there are no discrepancies, a button will appear to 'Consolidate & store reviewed matches' (*see screenshot to bottom-right*). Click it. This will produce a .csv file in **catRlog/3 matches/reviewed matches** with the filename "Reviewed Matches <YYYMMDD HHMMSS> .csv".

9. Once done, simply close the app.

'Setup' tab of the Review Matches app, when discrepancies have been found between match sessions:



. . . and without discrepancies.



Stage 3: Matches | Review and reconcile matching sessions, **part 2**

Dealing with discrepancies:

7b. Go to the “DISCREPANCY” tab.

This page is divided into four sections:

Top section displays the photo connected to the first disputed match.

Second section displays the potential matches within the reference catalog. (If the decision is that the photo is too poor to match or that this is a new individual, this space will be blank.

Third section displays the match decisions in table format.

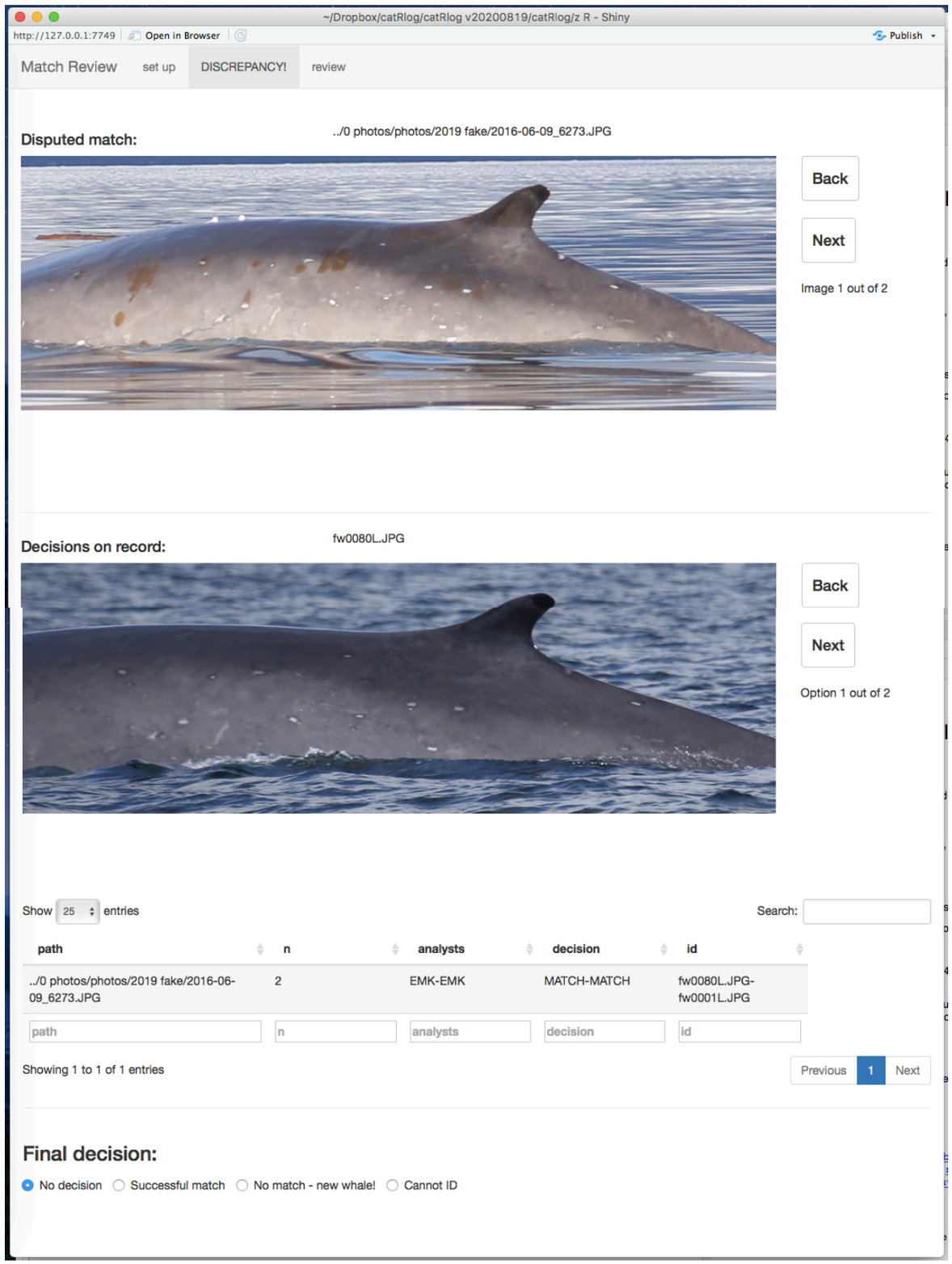
Fourth section is where you make your final decision about this match. If you decide that one of these matches is correct (‘Successful match’ option), you will be able to choose which reference catalog ID is the correct one.

Once you click an option other than “No decision”, the option will appear to ‘Save Final Decision’.

Do this for every discrepancy until there are no more outstanding.

You may then go back to the ‘setup’ tab and find that there is now an button option to ‘Consolidate & store reviewed matches’ (see previous page).

“DISCREPANCY” tab in the Review Matches app.



Stage 3: Matches | Register new IDs in the historical catalog

Before the photos you have matched can be assigned a corresponding ID code, all of those ID codes must exist.

However, the matching process is likely to uncover individuals who are not yet in the historical catalog.

To register new IDs in the historical catalog, do the following:

1. Go to **catRlog/3 matches/**.
2. Open '**3.2 register new IDs.R**' in R Studio
3. Click 'Run App'.

4. Select the Reviewed Matches you want to look in for new individuals.

5. Click 'Import Reviewed Matches'

6. If there are any new whales to register, go to the "REGISTRATION!" tab.
    - The first new whale will be displayed.
    - Above it is a small note indicating what the current final ID code is in **catRlog/4 catalog/catalog key.csv**

8. Select the feature displayed in this image.

9. Type in the new ID code you will assign this individual.

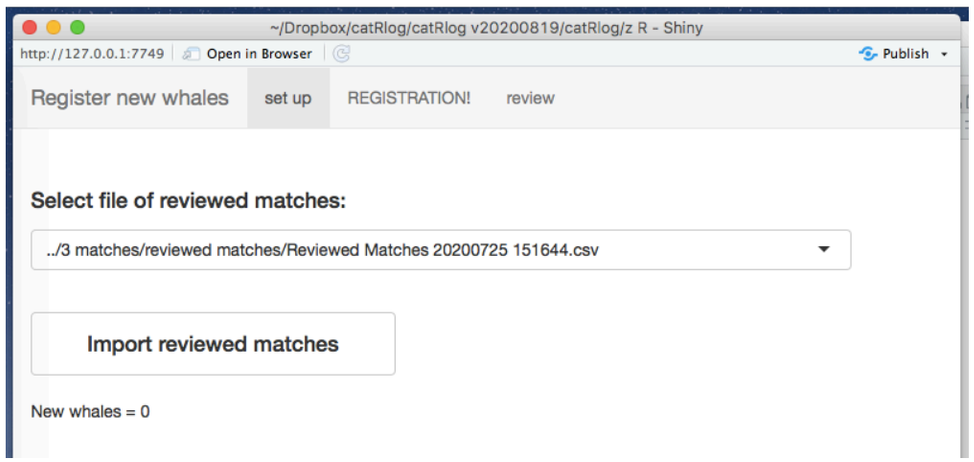
10. If you wish, provide a local identifier / nickname in the 'New Name' input

11. Once you type in a new ID, the button to "Assign ID (and add to catalog, if needed)." (The parenthetical refers to the fact that you will likely have multiple photos of a single new arrival.) Click it.

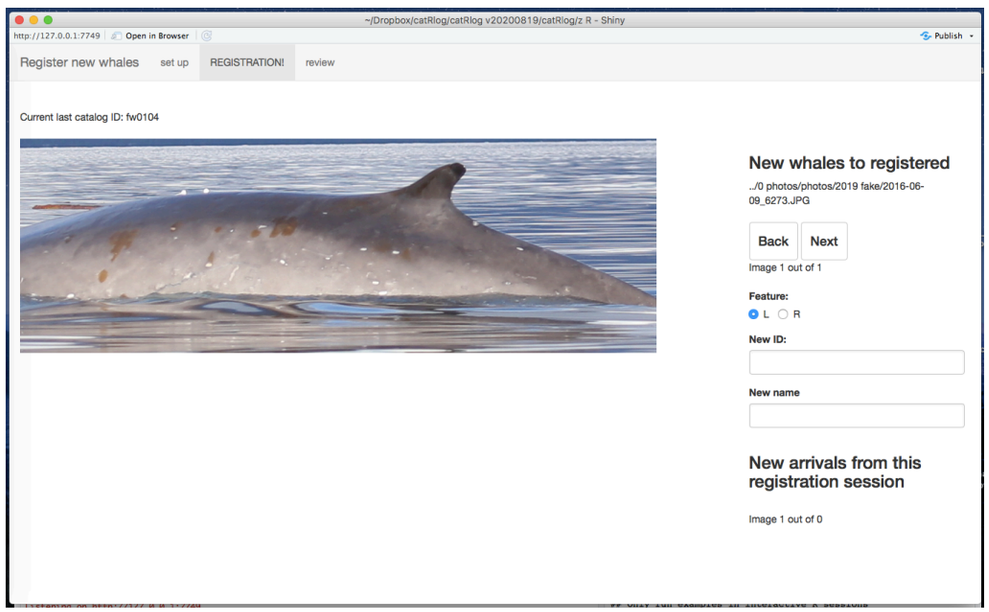
When you do, this image will be copied to **catRlog/4 catalog/catalog** and renamed according to its new ID and the feature you selected. In addition, a new line for this new ID code will be written in **catalog key.csv**

Note: Before changes are made, a backup version of the key will be save to the subfolder **old keys**.

"setup" tab in the Register New IDs app.



"REGISTRATION" tab in the Register New IDs app.



12. This individual will then be added to the "New Arrivals" section at the bottom of the page. If you have subsequent images of this new whale, you will be able to use these images as a reference for typing in the correct ID code.

13. Once you have registered all new whales, simply close the app.

14. Finally, open **catalog key.csv** and manually enter feature scores (0 or 1) to each category. This step is essential; the new whale will not show up in matching / digital flipbook apps until these details are added.

Stage 3: Matches | Assign match results to event tables

Once all matches have an ID code, including new arrivals, you can update event tables to include these match results.

To assign match results to event tables, do the following:

1. Go to **catRlog/3 matches/**.

2. Open '**3.3 update events.R**' in R Studio

3. Click 'Run App'.

4. Select the Reviewed Matches you want to use.

5. Select the event table you want to update with the IDs in that match file.

6. Click "Assign matches to these event files"

This action will cause the following to occur:

a. A backup version of the selected event file will be saved in **catRlog / 1 events / backups.**

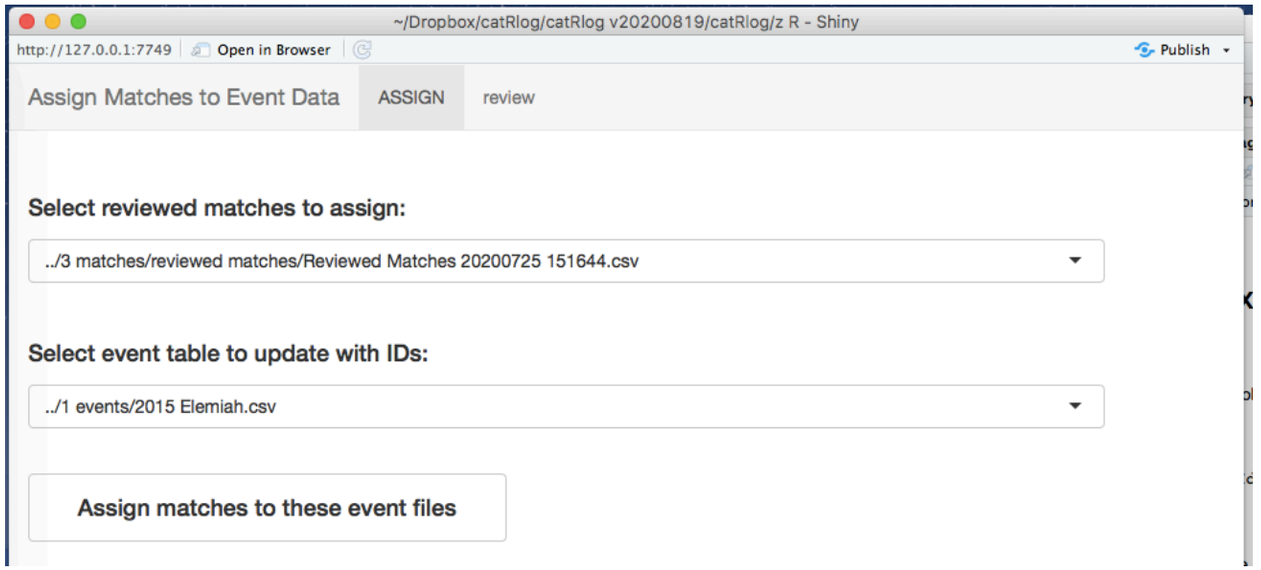
b. In the original version of the event file, a column will be added and named '**id'**.

Note: If a column named 'id' already exists, that column will be renamed 'id.old' so that it is not overwritten.

c. The match decision will be placed in this column according to the image filename listed in column '**file**'. Any match decision of "Photo too Poor" will appear as NA. Any event row whose image filename is not included in the match file will have 'NA' placed in its '**id**' column.

7. Close the app.

"ASSIGN" tab in the Assign Matches to Event Data" app.

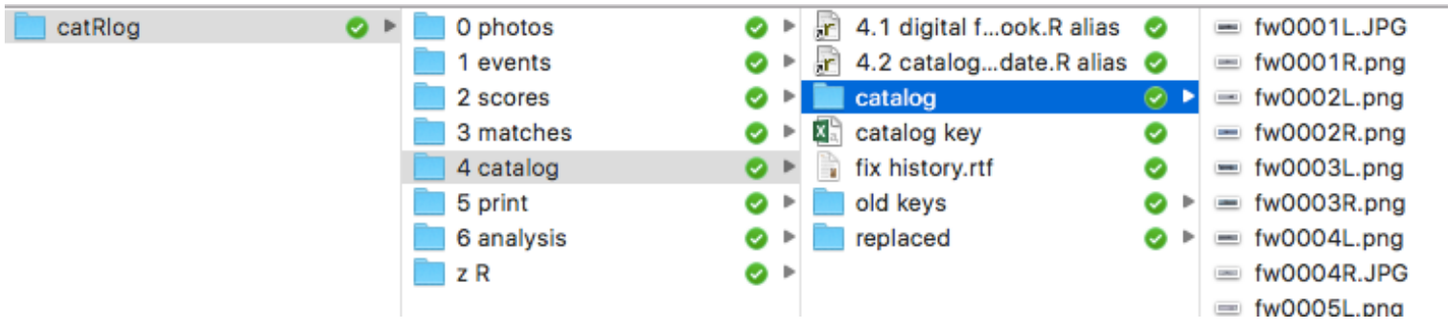


Stage 4: Catalog | Structure of the historical catalog and its key

In the catRlog system, the historical catalog is located within **catRlog / 4 catalog /.**

The historical catalog has two components:
(1) A folder of ID photos (subfolder **catalog**)
(2) A spreadsheet "key" (**catalog key.csv**)

Example of catalog folder:



Example of catalog key.csv:

id	local	injury	nick	hole	calf	mother	distinctiveness	dfo
fw0001	Bering McFinnigan	0	0	0	0	0	3	FW(
fw0002	Pecos	0	0	0	0	0	2	FW(
fw0003	Lag	0	1	0	0	0	1	FW(
fw0004		0	0	1	0	0	2	FW(
fw0005	Drifter Lonesome	0	1	0	0	0	1	FW(
fw0006		0	0	0	0	0	3	FW(
fw0007	Haskell	0	0	1	0	0	1	FW(
fw0008	Tunix	0	1	0	0	0	1	FW(
fw0009	Caribou MARVELOUS	0	1	0	0	0	1	FW(
fw0010	Garbanzo Von Trapp	1	1	0	0	0	1	FW(

ID photos in the catalog folder must be named < ID code > < feature > . < file type >

Example: 'fw0001L.JPG' is the left dorsal image of fw0001.

The ID code can be any length and any combination of letters and numbers.

The feature code can only be one character in length.

Each photo in the **catalog** folder must correspond to a row in the **catalog key**.

The catalog key has a single row for each individual in the catalog.

Required columns within the catalog key are: id, local, injury, nick, hole, calf, mother (see Table 1 in main text for details).

Additional fields are fine – they will be ignored by **catRlog**.

Each time an individual is added to the historical catalog, the **catalog** folder gets a new image and the **catalog key** gets a new row.

Stage 4: Catalog | Peruse historical catalog as a digital flipbook

A quick and easy way to explore your historical catalog is to use an interactive app referred to as a "Digital Flipbook."

1. Go to **catRlog/4 catalog/**.

2. Open '**4.1 digital flipbook.R**' in RStudio

3. Click 'Run App'.

The 'Review' tab allows you to explore **catRlog / 4 catalog / catalog key.csv**

In the 'Flipbook' tab, flip through the images contained in **catRlog / 4 catalog / catalog.**

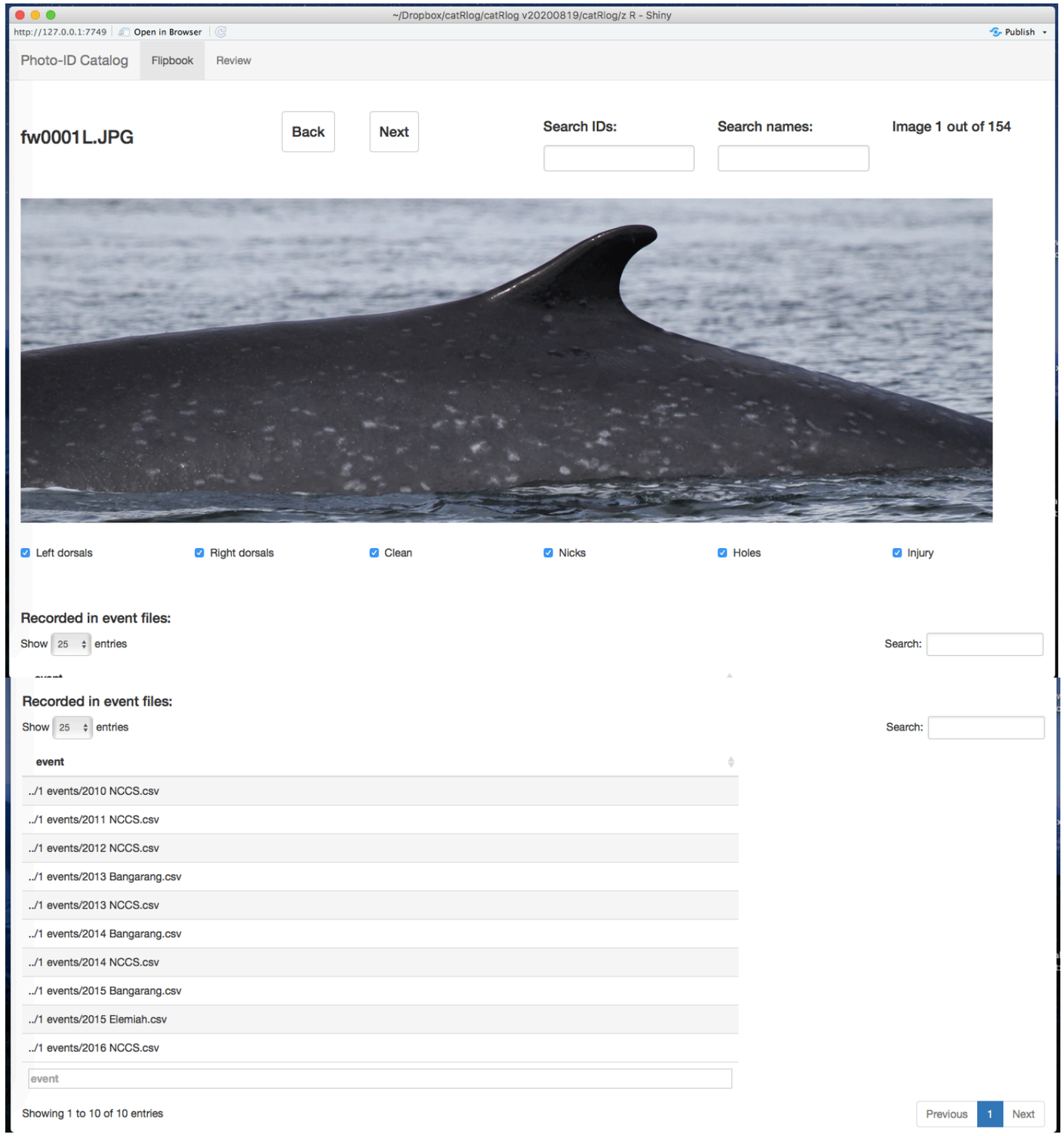
Search the catalog based on ID code (column 'id' in **catalog key.csv**) or local name (column 'local'). Partial matches are used.

Filter the catalog according to photo feature and/or individual marks (refers to columns in **catalog key.csv**)

Find which event tables contain sightings of this individual.

Close the app when done.

Main screen of the "Digital Flipbook" app.



Stage 4: Catalog | Update photos in the historical catalog

After processing a new photo collection, it is possible that you will have photos of individuals that . . .
1) Are higher quality than the ID photo in the historical catalog,
2) Contain more recent marks that are not depicted in the historical catalog, and/or
3) Depict a feature that is not yet documented within the historical catalog.

To replace photos or add photos of new features to the historical catalog, do the following:

1. Go to **catRlog/4 catalog/**.
2. Open '**4.2 catalog update.R**' in R Studio
3. Click 'Run App'.

The top image will show an individual in the reference (historical) catalog.

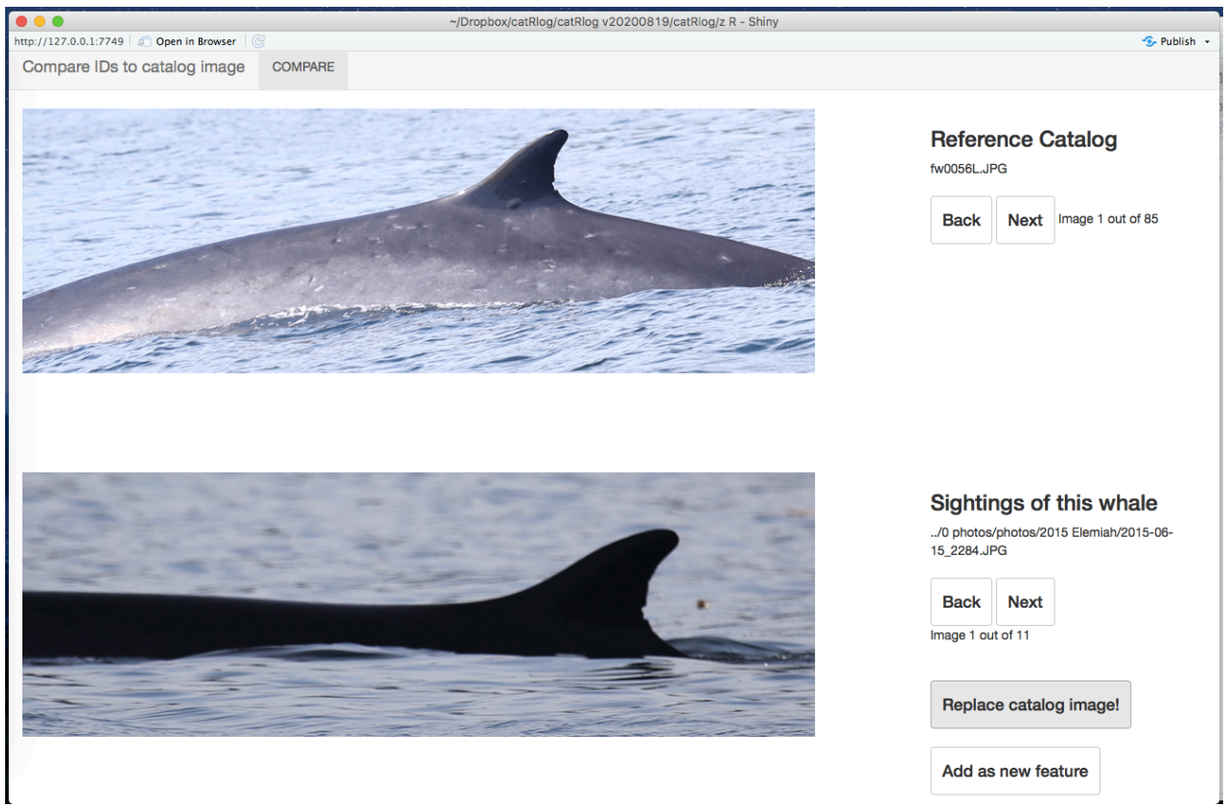
The bottom images will be all instances of this individual found among the photo collections currently present within the **catRlog** system.

Note: the reference catalog will be filtered only to those individuals that have been seen in the photo collections.

Step through the sightings of this individual.

If there is a better photo of the feature depicted in the historical catalog, click on "**Replace catalog image!".**
This will replace the current image in **catRlog / 4 catalog / catalog** (which will be sent to **catRlog / 4 catalog / replaced**, in case you make a mistake) with the new photo.

App for updating photos in the historical catalog:



If there is a photo with a feature not yet documented in the historical catalog, click "Add new feature".
This will copy the new photo to **catRlog / 4 catalog / catalog** with the filename "<ID>X.<filetype>". *Example*: "fw0043X.JPG".
Navigate to this folder in Finder and manually replace "X" with the correct feature code.

Once you have done this for all individuals in the reference catalog, close the app.

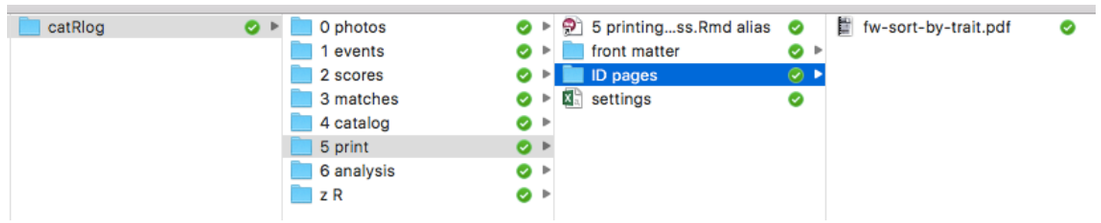
Note: If you ever have to manually correct the historical catalog in any way (e.g., finding an accidental duplicate within the catalog), use the '**fix history.txt**' text file to document the changes you make.

Stage 5: Print | Create a print-ready PDF of your historical catalog, **part 1**

To 'publish' your historical catalog as a print-ready PDF, do the following:

1. Go to **catRlog/5 print/**.
2. First open up the **settings.txt** file.

Example of directory layout:



Settings.txt file

setting	value	default	detail
Use defaults only	FALSE	TRUE	Options: TRUE or FALSE; Useful in case you can't get specific settings to work
Start index	1	all	Set to 'all' for including all; useful if wishing to test out printing press quickly or produce PDF in chunks
End Index	9	all	
Image compression (to control file size)	0.3	0.3	.1=full compression and lowest file size; 1=no compression and largest file size
Features to include	L	L-R	Max. 2 features; Separate features by hyphen
How to sort IDs	id	id	Choose the column in the catalog key to use to sort.
Include times seen	TRUE	TRUE	Options: TRUE or FALSE; if this catalog is to be distributed to the public you may not want too many details to be included.
Include years seen	TRUE	TRUE	Options: TRUE or FALSE; if this catalog is to be distributed to the public you may not want too many details to be included.
Include platforms from which whales are seen	TRUE	TRUE	Options: TRUE or FALSE; if this catalog is to be distributed to the public you may not want too many details to be included.

These settings will determine how the PDF is created. The default settings are staged for a two-feature catalog in which the abbreviations are L (i.e., left dorsal)s and R (i.e., right dorsals). However, this can be changed by opting out of the default settings.

The most important setting is "**Features to Include".** You can specify one or two features and provide any feature code you wish – it just has to correspond to the feature code used in the image filenames within **catRlog / 4 catalog / catalog.** See further details within the **detail** column of **settings.txt.**

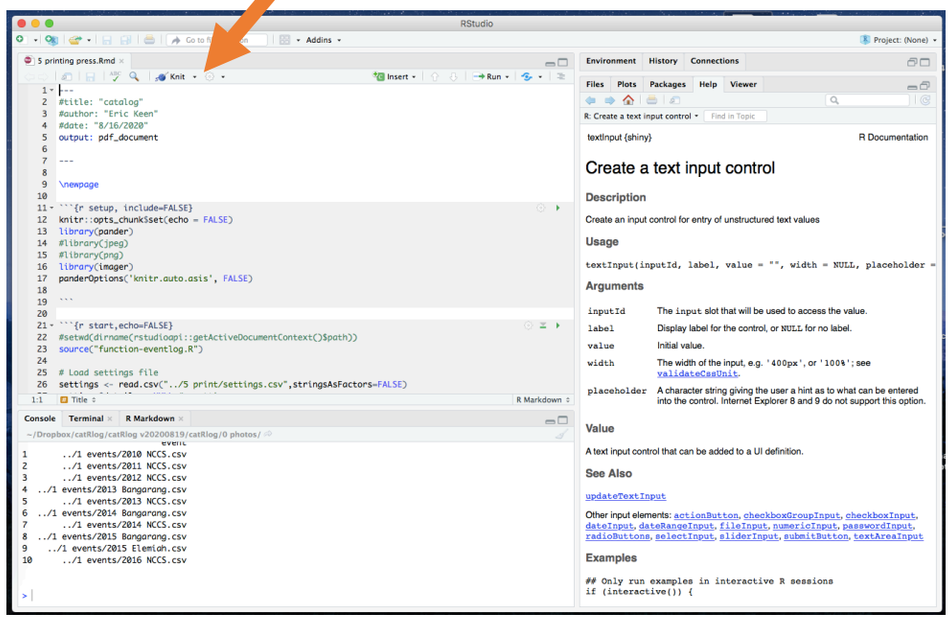
Tip: if your catalog is large, I suggest building first with only a few IDs, e.g., 9, to make sure that PDF production works at the basic level.

3. Once you have customized your settings and saved **settings.txt,** close the file.

4. Open up **5 printing press.Rmd** in R Studio.

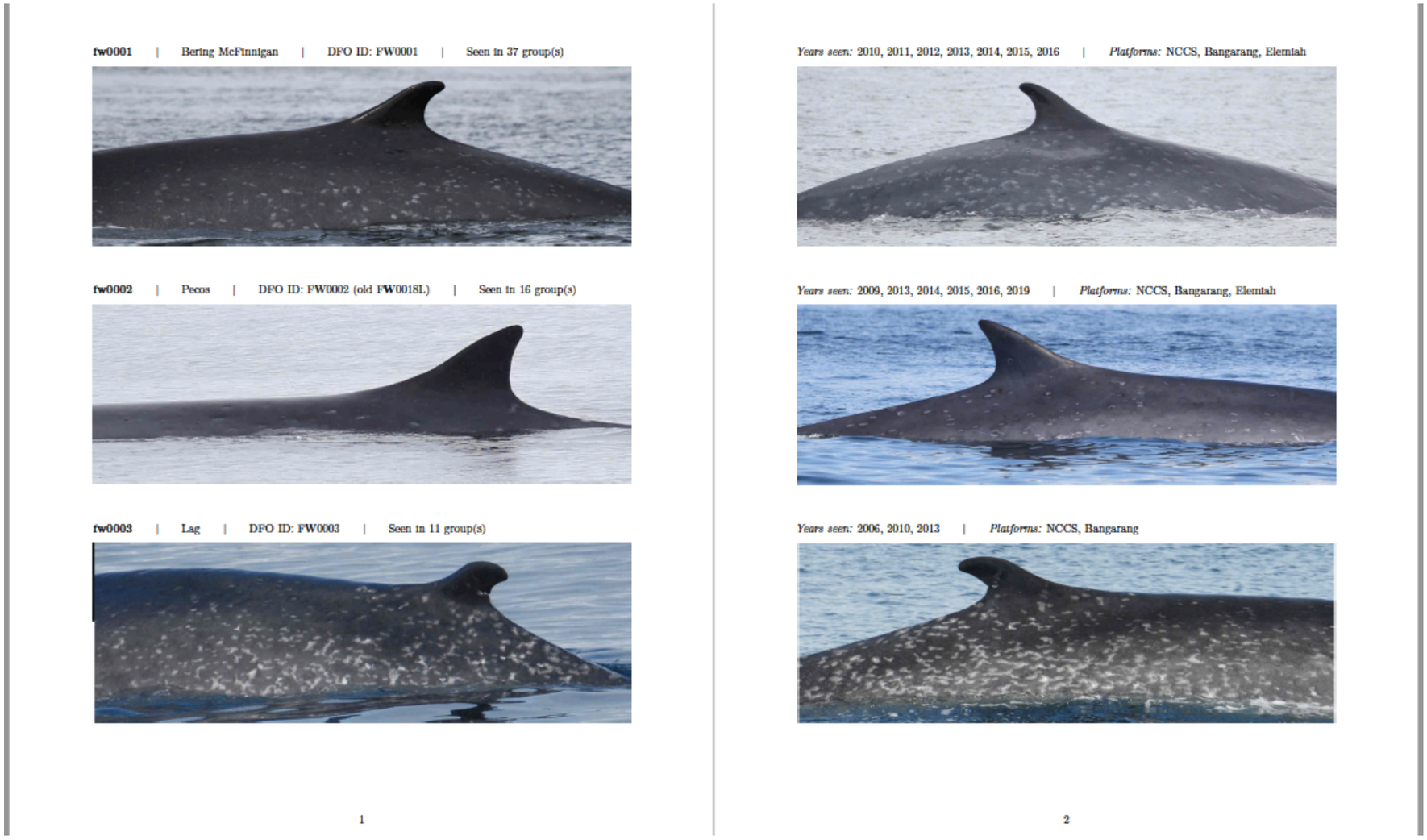
5. Click "Knit" (*see screenshot to the right)*

Click the "Knit" button to produce the PDF.



Stage 5: Print | Create a print-ready PDF of your historical catalog, **part 1**

Example of a 2-page spread from the printed catalog:



Compiling your PDF may take several minutes.
You can track progress under the “R Markdown” console tab in R Studio.

The PDF will be saved to **catRlog / zR / 5-printing-press.pdf**

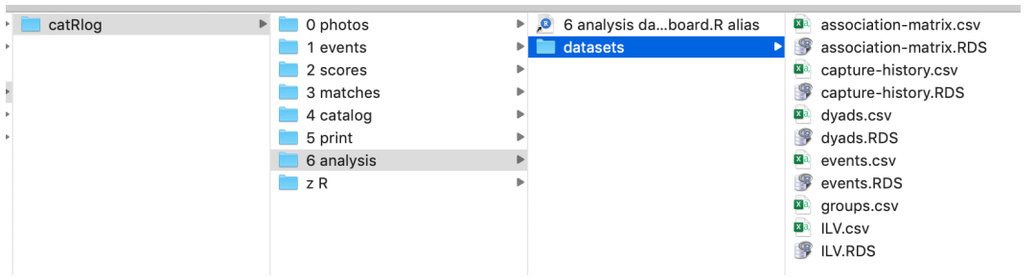
Move this PDF file into **catRlog / 5 print / ID pages** and rename as you wish.

Inside **catRlog / 5 print**, there is also a folder for front matter material (e.g., introduction, cover pages, acknowledgements, etc.). You can produce these in separate software, save them as a PDF, then use a PDF editor to merge the front matter together with the ID pages file of your choice.

Note: The printing stage makes use of a separate kind of R Studio product: the **Rmarkdown** language and the package ‘**knitr**’, among others.

Stage 6: Analysis | Compile datasets for use in various analyses

Example of directory layout:



When it is time to use your photo-identification database to address a research question, compile your data into the datasets you need using the following app:

1. Go to **catRlog/6 dashboard/**.
2. Open '**6 analysis dashboard.R**' in R Studio
3. Click 'Run App'.

(See screenshot to the right for the layout of this app.)

4. Select the datasets you wish to compile. *(See examples of these data products on next page.)*

Note: All of these datasets hinge upon the event tables in **catRlog / 1 events**

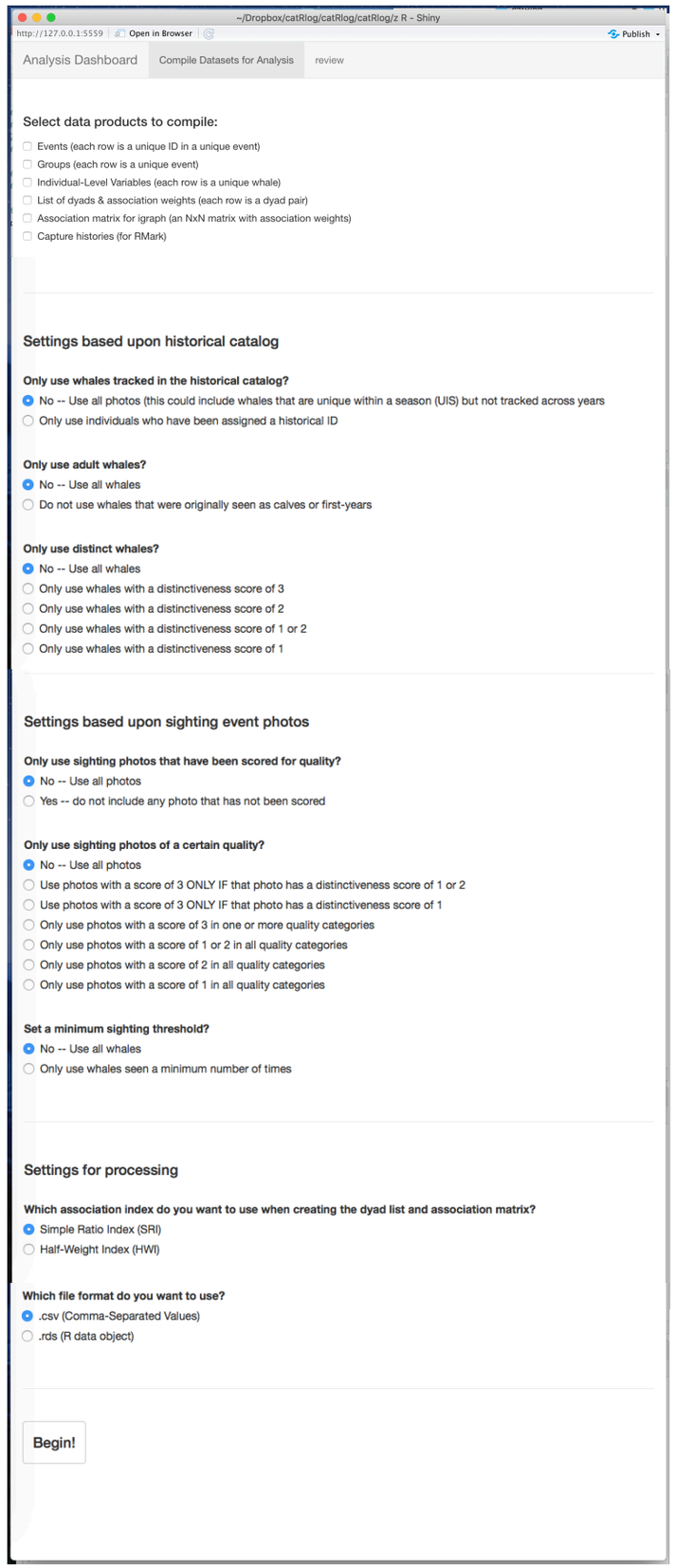
5. If you wish to control which events are included based on the quality of photographs, the distinctiveness of individuals, the reproductive status of individuals, or a minimum sighting threshold, specify those settings.

Note: Several of these settings require photo quality scores (see **Stage 2**). This app will summarize these scores as follows: the feature distinctiveness score will appear in a column named 'distinct'. The image quality scores will be summarized into a single number (3 = one or more categories was a 3; 12 = all scores were either 1 or 2; 2 = all scores were 2; 1 = all scores were 1.)

6. Specify settings regarding the kind of association weights to calculate (if applicable), and the file type of the data output.

7. Click "**Begin**!".
Track progress on the R Studio console.

Once the process is complete, you will be able to review the datasets under the 'review' tab, or within **catRlog / 6 analysis / datasets**



Stage 6: Analysis | Dataset examples, **part 1**

events.csv

A compilation of all events tables (each row is a photo-identification event; i.e., there can be multiple rows for a single group encounter)

New columns:
groupid = unique identifier for each group (YYYYMMDDGG);
doy = the calendar day of year;
yfrac = the date as a fraction of year;
path = path to photo collection;
distinct and **score** (see note on previous page.)
(**lat** and **lon** are included even if they did not exist in the original event tables. Missing values are assigned **NA**).

sitid	id	year	month	day	doy	yfrac	time	group	platform	lat	lon	file	path	distinct	score
2006073001	fw0033	2006	07	30	211	2006.57808219178	1205	01	NCCS	53.045	-129.2125	FW_2006-07-30_NCCS_SAshdownIs_D-9788	../0 photos/photos/2006 NCCS/	NA	NA
2006073001	fw0031	2006	07	30	211	2006.57808219178	1205	01	NCCS	53.045	-129.2125	FW_2006-07-30_NCCS_SAshdownIs_D-9798	../0 photos/photos/2006 NCCS/	NA	NA
2006073001	fw0033	2006	07	30	211	2006.57808219178	1205	01	NCCS	53.045	-129.2125	FW_2006-07-30_NCCS_SAshdownIs_D-9802	../0 photos/photos/2006 NCCS/	NA	NA
2006080101	fw0009	2006	08	01	213	2006.58356164384	1800	01	NCCS	53.01881667	-129.2328833	FW_2006-08-01_NCCS_CampaniaSnd_D-9917	../0 photos/photos/2006 NCCS/	NA	NA
2006080101	fw0010	2006	08	01	213	2006.58356164384	1800	01	NCCS	53.01881667	-129.2328833	FW_2006-08-01_NCCS_CampaniaSnd_D-9965	../0 photos/photos/2006 NCCS/	NA	NA

ILV.csv

A summary of the encounter history for each individual (ILV = Individual-Level Variables; each row is a unique ID)

Columns **id, local, mother,** and **calf** are copied from **catRlog / 4 catalog / catalog key.csv**
n = number of encounters
ny = number of years encountered.
doy.min and **doy.max** are the first and last days of year, respectively, that this individual has been encountered.
years, doys, and groups are blank-space-separated lists of the years, days of year, and groups in which this individual has been encountered. These lists make it efficient and easy to compile dyadic associations and other data.

id	local	mother	calf	n	ny	doy.min	doy.max	years	doys	groups
fw0001	Bering McFinnigan	0	0	28	7	159	273	2010 2011 2012 2013 2014 2015 2016	213 222 200 255 262 179 195 200 22	2010080101 2010081001 2011071901 2011091201 2012091801 2013062801 2013071419 2013071924 2013081212 20
fw0002	Pecos	0	0	12	6	183	270	2009 2013 2014 2015 2016 2019	209 183 184 240 225 252 230 270 19	2009072802 2013070202 2013070301 2013082801 2014081301 2014090901 2015081803 2015092701 2016071201 20
fw0003	Lag	0	0	8	3	183	238	2006 2010 2013	226 213 217 222 227 238 195 183	2006081401 2010080101 2010080501 2010081001 2010081501 2010082601 2013071412 2013070202
fw0004		0	0	2	2	193	270	2011 2015	193 270	2011071201 2015092701
fw0005	Drifter Lonesome	0	0	13	5	180	270	2011 2013 2014 2015 2019	245 253 180 240 241 207 239 194 21	2011090201 2011091001 2013062909 2013082810 2013082907 2013072601 2014082701 2015071305 2015080510 20
fw0006		0	0	1	1	159	159	2011	159	2011060801
fw0007	Haskell	0	0	11	5	180	262	2007 2011 2012 2013 2016	190 200 236 262 180 198 205 238 23	2007070901 2011071901 2011082401 2012091804 2013062908 2013071703 2013072402 2013082601 2013082603 20
fw0008	Tunix	0	0	37	9	161	272	2008 2009 2010 2012 2013 2014 2015 2016 2019	206 209 183 213 231 263 200 177 18	2008072401 2009072802 2010070201 2010080101 2012081801 2012091902 2013071924 2013062601 2013070204 20
fw0009	Caribou MARVELOUS	0	0	1	1	213	213	2006	213	2006080101
fw0010	Garbanzo Von Trapp	0	0	1	1	213	213	2006	213	2006080101

Stage 6: Analysis | Dataset examples, **part 2**

groups.csv

A summary of each group encounter (each row is a unique encounter)

New columns:
groupid = unique identifier for each group (YYYYMMDDGG) – unique to the entire event table.
py = Platform year descriptor
platform = Platform
year, month, day
doy = the calendar day of year;
yfrac = the date as a fraction of year;
Time
group = group identifier from day of field effort
size = minimum group size (according to number of photo-identifications in encounter)
(**lat** and **lon** are included even if they did not exist in the original event tables. Missing values are assigned **NA**).
id = blank-space-separated list of the individuals identified within this group.

groupid	py	platform	year	month	day	doy	yfrac	time	group	size	lat	lon	id
2006073001	2006 NCCS	NCCS	2006	07	30	211	2006.578	1205	01	2	53.045	-129.2125	fw0033 fw0031
2006080101	2006 NCCS	NCCS	2006	08	01	213	2006.584	1800	01	3	53.01881667	-129.2328833	fw0009 fw0010 UIS
2006081401	2006 NCCS	NCCS	2006	08	14	226	2006.619	1045	01	1	53.01686667	-129.2027167	fw0003
2007070901	2007 NCCS	NCCS	2007	07	09	190	2007.521	1005	01	1	53.09048333	-129.1028833	fw0007
2007082701	2007 NCCS	NCCS	2007	08	27	239	2007.655	1420	01	1	53.09666667	-129.1916667	fw0028

capture-histories.csv

Capture history summary of each individual, formatted for use in Rmark.

New column:
ch = A digit for each sampling period (in the default case, years; this can be modified within the R code). 0= not observed; 1= observed.

ch	id	local	mother	calf	n	ny	doy.min	doy.max	years
000011111110	fw0001	Bering McFinnigan	0	0	28	7	159	273	2010 2011 2012 2013 2014 2015 2016
000100011111	fw0002	Pecos	0	0	12	6	183	270	2009 2013 2014 2015 2016 2019
100010010000	fw0003	Lag	0	0	8	3	183	238	2006 2010 2013

Stage 6: Analysis | Dataset examples, **part 3**

dyads.csv

Metrics for all possible dyadic associations.
This spreadsheet can be extremely long if your catalog is large.

yrA, yrB = Number of years individuals A and B have been seen
yrX = Number of years A and B have been encountered together in the same groups.
nA, nB = Number of encounters of A and B
X = Number of encounters in which A and B have been found together.
yA = Number of encounters with A in which B was absent.
yB = Number of encounters with B in which A was absent.
yabx = nA + nB – X.
ynull = Number of encounters that included neither A nor B.
hwi = Half-weight index of association
sri = Simple-ratio index of association

dyad	A	B	yrA	yrB	yrX	nA	nB	X	ya	yb	yabx	ynull	hwi	sri
fw0001 - fw0001	fw0001	fw0001	7	7	7	28	28	28	0	0	28	246	1	1
fw0002 - fw0001	fw0002	fw0001	6	7	4	12	28	1	11	27	39	235	0.05	0.0256410256410256
fw0003 - fw0001	fw0003	fw0001	3	7	2	8	28	2	6	26	34	240	0.111111111111111	0.0588235294117647
fw0004 - fw0001	fw0004	fw0001	2	7	2	2	28	1	1	27	29	245	0.0666666666666667	0.0344827586206897
fw0005 - fw0001	fw0005	fw0001	5	7	4	13	28	2	11	26	39	235	0.0975609756097561	0.0512820512820513
fw0006 - fw0001	fw0006	fw0001	1	7	1	1	28	0	1	28	29	245	0	0
fw0007 - fw0001	fw0007	fw0001	5	7	4	11	28	1	10	27	38	236	0.0512820512820513	0.0263157894736842
fw0008 - fw0001	fw0008	fw0001	9	7	6	37	28	10	27	18	55	219	0.307692307692308	0.181818181818182

association-matrix.csv

An N x N matrix in which the columns list out the individuals, the rows list out the same individuals in the same order, and the data in cells provide the association weights. In this example, the diagonal values (representing self-association weights) have not been corrected to 0.

	fw0001	fw0002	fw0003	fw0004	fw0005	fw0006	fw0007	fw0008	fw0009
fw0001	1	0.0256410256410256	0.0588235294117647	0.0344827586206897	0.0512820512820513	0	0.0263157894736842	0.181818181818182	0
fw0002	0.0256410256410256	1	0.0526315789473684	0.0769230769230769	0.0869565217391304	0	0	0.0208333333333333	0
fw0003	0.0588235294117647	0.0526315789473684	1	0	0	0	0	0.0227272727272727	0
fw0004	0.0344827586206897	0.0769230769230769	0	1	0.0714285714285714	0	0	0	0
fw0005	0.0512820512820513	0.0869565217391304	0	0.0714285714285714	1	0	0	0.0416666666666667	0
fw0006	0	0	0	0	0	1	0	0	0
fw0007	0.0263157894736842	0	0	0	0	0	1	0	0
fw0008	0.181818181818182	0.0208333333333333	0.0227272727272727	0	0.0416666666666667	0	0	1	0
fw0009	0	0	0	0	0	0	0	0	1

Manually editing catRlog code, part 1

Modifying catRlog's underlying R code requires knowledge of the R language as well as the mechanics of Shiny and (for the printing press) Rmarkdown. There are several excellent tutorials available online for both of these features of Rstudio. Acquaint yourself with those features before attempting to modify the code.

Before modifying the code, save a stable version of the R code as a backup.

Below we provide an example of a common modification that users will likely wish to make:
Altering the marks / traits used in the digital flipbook app (pg. 19) to filter the historical catalog. A similar filter function exists within the matching app (pg. 13). In this example, let's say that you wish to add the option to filter to individuals with evidence of tooth rakes from predation attempts by killer whales.

1. Add a column in **catRlog / 4 catalog / catalog key.csv.** Name the column 'rakes'. For each individual, add a 0 (rakes absent) or 1 (rakes present) to this column. (The digital flipbook makes it easy to flip through each ID).

2. Open the R file for the digital flipbook:
   catRlog / z R / 4.1 digital flipbook.R

   This R code has two sections: a 'server' function (the behind-the-scenes workings of the app) and a 'ui' function (the design of the user interface for the app).

(Continue on next page.)

Beginning of the R code for the 'digital flipbook' app.

4.1 digital flipbook.R*

```
1  ##############################################################
2  ##############################################################
3  # catRlog  | Eric M. Keen, v. July 2020
4  ##############################################################
5  # APP: Digital Catalog Flipbook
6  ##############################################################
7  ##############################################################
8
9  # Set working directory to folder that this R file is in
10 setwd(dirname(rstudioapi::getActiveDocumentContext()$path))
11 source("00 Source functions.R")
12
13 #library(shiny)
14
15 ##############################################################
16 ##############################################################
17 ##############################################################
18
19 server <- function(input, output) {
20
21   ############################################################
22   # Setup reactive values
```

Manually editing catRlog code, part 2

2. In the 'server' function, find the section of code that sets up filter parameters:

3. On line 49, add the following code to the dataframe:
…`injury=0`**`,rakes=0`**`)`

4. After line 71, add a new line:
`dfilter$rakes[i] <- as.character(key$rakes[keymatch])`

5. Scroll down to lines 131 – 137. Add to this expression:
`. . . Input$injury` **`+ input$rakes`**

6. Scroll down to lines 157 – 164. *Above* the line that begins with "`if(input$clean) . . .` ", add this line:
`if(input$rakes){rakes <- which(dfilter$rakes==1)`

7. Then modify the line that begins with "`adds <- . . .` " :
`. . . holes, injury,` **`rakes,`** `clean)))`

8. In the 'ui' function, find the line that reads:
`column(3,checkboxInput("injury","Injury",value=TRUE))),`

9. Replace the 3 in that line with a 1.

10. Then, just *above* that line, add the following code:
`column(1,checkboxInput("rakes","Tooth rakes",value=TRUE))`

11. You should now be able to open up this app and search the catalog according to this feature.

```
48  # Setup filter parameters
49  dfilter <- data.frame(lf=lf,id=NA,local=NA,ld=0,rd=0,fl=0,nicks=0,holes=0,injury=0) ; dfilter
50
51  ls <- c() ; i=1
52  for(i in 1:nrow(dfilter)){
53    lfi <- as.character(dfilter$lf[i]) ; lfi
54
55    # LDs
56    splits <- strsplit(lfi,"")[[1]] ; splits
57    dotchar <- which(splits==".")[3] ; dotchar
58    dfilter$id[i] <- substr(lfi,1,(dotchar-2))
59    dorsal <- substr(lfi,(dotchar-1),(dotchar-1)) ; dorsal
60    if(dorsal=="L"){dfilter$ld[i] <- 1}
61    if(dorsal=="R"){dfilter$rd[i] <- 1}
62    if(dorsal=="F"){dfilter$fl[i] <- 1}
63
64    # Key attributes
65    keymatch <- which(as.character(key$id)==gsub(rv$refdir,"",dfilter$id[i])) ; keymatch
66    if(length(keymatch)>0){
67      dfilter$nicks[i] <- as.character(key$nick[keymatch])
68      dfilter$holes[i] <- as.character(key$hole[keymatch])
69      dfilter$injury[i] <- as.character(key$injury[keymatch])
70      dfilter$local[i] <- as.character(locals[keymatch])
71    }
72  }
```

```
130 ##########################################################
131 # Filter reference catalog
132
133 observe({
134   (input$search=="trigger") + (input$searchname=="trigger") + input$LD + input$RD
135   + input$FL + input$nicks + input$holes + input$injury
136
```

```
157 nicks <- holes <- injury <- clean <- c()
158 if(input$nicks){nicks <- which(dfilter$nicks==1)} #; nicks
159 if(input$holes){holes <- which(dfilter$holes==1)} #; holes
160 if(input$injury){injury <- which(dfilter$injury==1)} #; injury
161 if(input$clean){clean <- which(dfilter$clean==0)} #; clean
162 adds <- sort(unique(c(nicks,holes,injury,clean)))
```

```
201 fluidRow(column(2,checkboxInput("LD","Left dorsals",value=TRUE)),
202        column(2,checkboxInput("RD","Right dorsals",value=TRUE)),
203        column(2,checkboxInput("FL","Fluke",value=TRUE)),
204        column(1,checkboxInput("clean","Clean",value=TRUE)),
205        column(1,checkboxInput("nicks","Nicks",value=TRUE)),
206        column(1,checkboxInput("holes","Holes",value=TRUE)),
207        column(3,checkboxInput("injury","Injury",value=TRUE))),
208 br(),br(),
```

Troubleshooting & Updating

If something is not working, the first step is to ensure that all data are formatted exactly according to the requirements detailed in this manual and in Tables 1, 2, and 3 of the main text. Also, ensure that only a single app R file is open and try again to 'Run App.'

If no inconsistencies are found and corrected, try carrying out the same stage of analysis with the original version of catRlog and the example data that come with it.

If you are still finding issues, try running catRlog with your data on a separate machine to see if other operating systems and software versions work.

Still having problems? Contact the code author Eric Keen:
eric.k@marecotel.org
He will investigate the issue at the earliest opportunity then add the issue to the list of revisions to incorporate for the next version release.

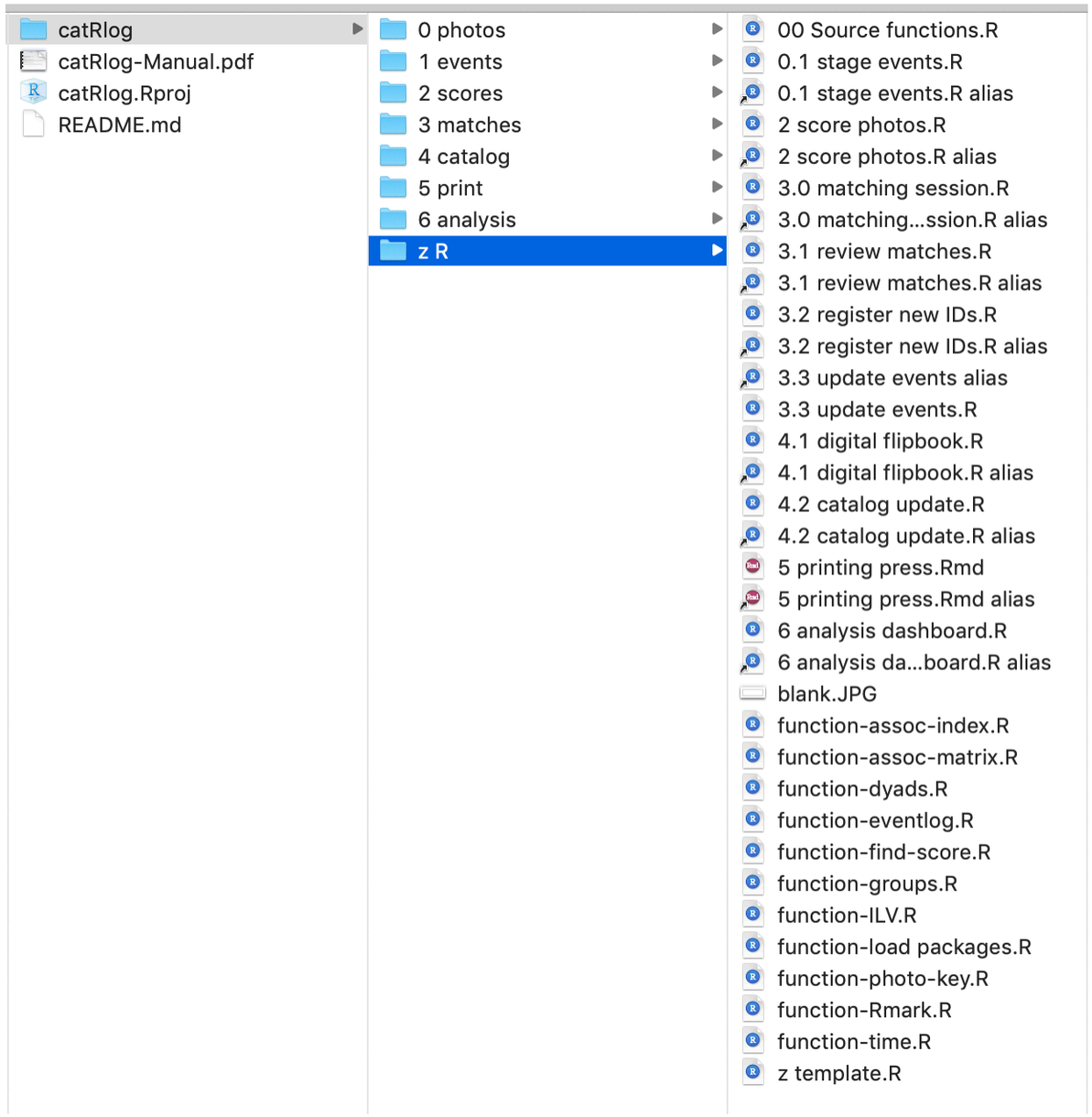
If you require an urgent response, seek out a colleague familiar with R to troubleshoot the issue.

Within the **catRlog** directory, all R code is located with a single folder: **catRlog / z R.** For ease of use, aliases to these R files (i.e., shortcuts) are used to refer to these code files from relevant locations (e.g., in the scoring folder, '2 scores', there is an alias for the scoring app .R file).

This setup makes it simple to update **catRlog** with new versions of its R code, without the risk of disrupting reference paths within the code:

To update:
1. Download updated R files & R aliases from Github (see main text),
2. Replace them within the **catRlog / z R** folder,
3. Then drag-and-drop the alias files to the relevant folders within the **catRlog** directory.

R directory within the **catRlog system**



Explore the catRlog system with example data provided within the directory

The example data that comes with the catRlog directory allows users to start off with a small historical catalog (only four individuals) and two photo collections (containing 5 and 3 image files, respectively) that need to be scored and matched to the historical catalog.

Once you setup **catRlog** on your computer (see page 1), we recommend exploring the system's functionality in the following order:

1. Open the digital flipbook (**catRlog / 4 catalog / 4.1 digital flipbook .R alias**) and try out the navigation, search and filter options. (pgs. 18 – 19)

2. Open the catalog key (**catRlog / 4 catalog / catalog key.csv**) to see how the structure of this spreadsheet relates to the catalog folder (**catRlog / 4 catalog / catalog**) and the information you see displayed in the digital flipbook. (pg. 18)

3. Then try to create a PDF of the historical catalog using the 'printing press' app (**catRlog / 5 print / printing press.Rmd alias**). (pg. 21 – 22). Feel free to customize the printing using the **settings.txt** file.

4. Now begin the sequence of processing a photo collection (start with the collection '2016 Bangarang'). Go to **catRlog / 0 photos** to see this photo collection. Note the aspect ratio and formatting of the images.

5. This example dataset comes with event tables already created for this photo collection. Check them out in **catRlog / 1 events**.

6. Since events tables have already been created, move to Stage 2: scoring **(catRlog / 2 scores / score photos.R alias)** on the photo collection '2016 Bangarang' (pg. 9 – 10).

7. From this point on, follow the instruction manual until you see all of the **catRlog** system's functions and features (pgs. 9 – 26).