A low-cost solution for documenting, tracking, and verifying cage-level animal husbandry tasks using wireless QR scanners and cloud-based spreadsheets

Elizabeth A. Hobson

Department of Biological Sciences, University of Cincinnati

elizabeth.hobson@uc.edu

ORCID: 0000-0003-1523-6967

All animal activities are approved by the University of Cincinnati IACUC protocol 21-02-23-01

Abstract

Animal care is a critical component underlying successful behavioral and cognition experiments. Technological solutions for documentation and verification of care can aid in monitoring that activities are completed according to standard operating procedures and ensure that no individuals are overlooked. Here, I summarize a low-cost, flexible, and easy to use system that I developed to document and monitor care of animals for our research group. The system enables real-time and remote-enabled verification that critical daily tasks have been completed for every cage and helps us monitor our longer-term tasks to make sure that our care team is adhering to our set schedule. The main materials and components needed to implement this system are QR codes, a thermal laminator, a QR scanner, a computer to manage data input, and a database into which the data are scanned and summarized. There are six steps to setting up our system: (1) purchase a QR scanner, (2) generate and print QR codes, (3) set up the central hub for data input, (4) input data from QR scans, (5) filter and collate the raw data, and (6) summarize filtered data to verify and track care. Paired with simple scripts in a cloud-based spreadsheet, scanned QR code data can then be easily summarized in real time to provide verification of care. The flexibility of the system allows it to be customized to a large range of species.

Keywords

Animal care, paperless documentation, verification, QR code

Behavioral and cognition experiments are frequently conducted with animals in controlled lab environments. Animals involved in these experiments need to be cared for on a regular schedule and documenting this care is often required by each university's Institutional Animal Care and Use Committee (IACUC) or similar regulatory bodies. Requirements for documentation of care have helped improve transparency, consistency, and the quality of animal care (National Research Council 2011).

High quality care is especially important in behavioral and cognitive experiments, where animals generally need to be healthy, active, and behaving as normally as possible when they participate in these experiments. Most traditional concern around animal welfare has focused on mitigating any negative impacts on animal health and wellbeing. However, there is growing awareness that "positive welfare" (managing animal welfare beyond what is needed for immediate survival and basic mitigation of alleviating suffering) should be the new standard of care (Fife-Cook and Franks 2019). This focus on welfare is particularly relevant to behavioral and cognitive studies because recent work has argued that animal welfare conditions can affect cognitive capacity and cognitive function (Franks 2018). In behavioral or cognition studies, variation in welfare could then be an important factor that affects cognitive performance, and variability in care that affects welfare could then negatively affect our ability to cleanly evaluate cognitive and behaviorally-focused hypotheses. Properly managing the health and welfare of animals in behavioral or cognitive research is something that should be a goal of any group running behavioral or cognitive experiments.

However, the effort involved in documenting details of animal care can be time consuming (Haywood and Greene 2008; Green et al. 2017). Technological solutions can help reduce the burden of documenting animal care activities. For example, a technology-based animal care tracking system can aid in facilitating the documentation of animal care activities but can additionally serve to reduce the time required to document care compliance as well as being a method for validating that care has been received for each animal on the correct schedule. Quick response (QR) codes are widely used in many fields in biology. QR codes can encode urls as quick links to open websites, can link to online data entry forms, or can contain simple text. Pairing QR codes with smartphone/smart device technology has become a popular method to document animal care (e.g., Green et al. 2017), track specimens (e.g., Diazgranados and Funk 2013), or facilitate data entry (e.g., Oteyo and Toili 2020).

Although using technology to more easily document animal care can save time spent reporting basic care, the quality and consistency of care also need to be monitored and tracked. Technological solutions can also help track and verify that care has been provided to each animal on the proper schedule. Verification would help double-check that no animals have been accidentally overlooked and that care (especially tasks that do not occur on a daily basis) are following the intended schedule. A verification system like this would help improve the consistency of care while reducing animal care staff stress about potentially missing critical care. This method would be especially useful when many animals are being cared for across multiple cages, tanks, or enclosures and multiple categories of care are completed for each animal on different time scales (for example when some care needs to be completed every day, but other care needs to be completed once a week).

I developed a new system to document and verify animal care activities. My group recently started working with a new system in the lab and now have about 45 adult Northern bobwhite quail (*Colinus virginianus*) housed in about 20 cages, approved for use in behavioral and cognition experiments by the University of Cincinnati IACUC protocol 21-02-23-01. The population is cared for by several lab members and has tasks that need to be completed on different time intervals. For example, on a daily schedule each cage’s food and water needs to be changed and the health of all the birds needs to be checked, on a weekly schedule we clean the cage trays, and at least once every two weeks we provide enrichment. The need for a method to more easily track all of these activities, verify that everything has been completed according to the schedule, and to manage care across multiple people were the drivers behind the design of our system.

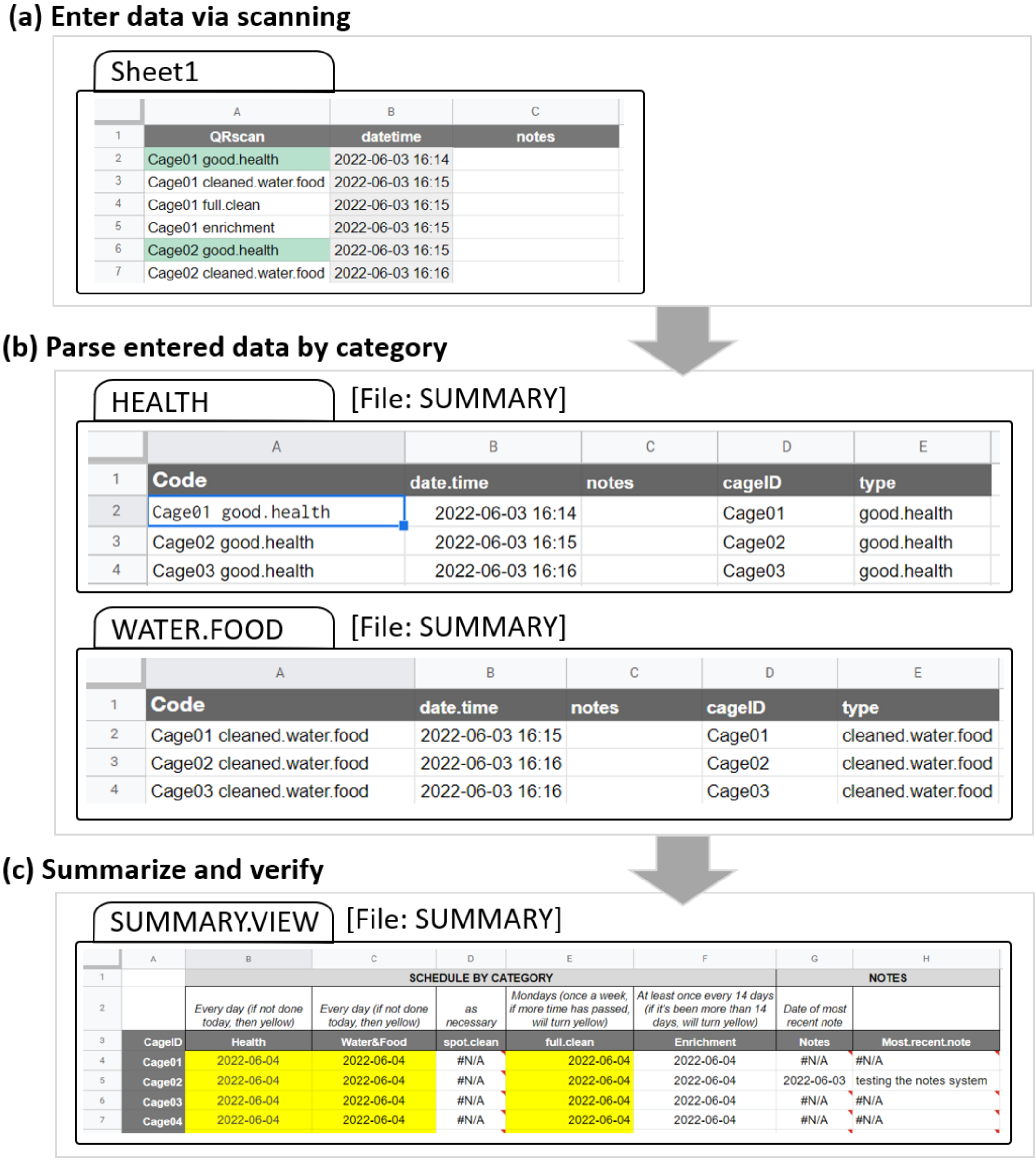
My system is a low-cost, flexible, and easy to use method that facilitates documentation of our animal husbandry activities, enables real-time and remote-enabled verification that critical daily tasks have been completed for every cage, and helps us monitor our longer-term tasks to make sure that our care team is adhering to our set schedule.

Here, I summarize the system, which we have been using and validating in the lab since summer 2021. The main materials and components needed to implement this system are QR codes, a thermal laminator, a QR scanner, a computer to manage data input, and a database into which the data are scanned and summarized. QR tags are generated for each activity that we want to track for each cage and then two Google Sheet workbooks are used to input and summarize the data. The system requires access to the internet (to access Google Sheets) but does not require wireless internet access to use. A formatted and publicly-accessible example of the both spreadsheets, containing all scripts used to automatically add the date and time, filter, collate, summarize, and conditionally format the data, and the summary table is available at https://bit.ly/3yycw1b. This file can be copied as a template and then modified for each lab's individual requirements as necessary.

System overview

There are six main steps to setting up the system, which are described below: (1) purchase a QR scanner, (2) generate and print QR codes, (3) set up the central hub for data input, (4) input data from QR scans (Fig. 1a), (5) filter and collate the raw data (Fig. 1b), and (6) summarize filtered data to verify and track care (Fig. 1c).

Figure 1. Google spreadsheet into which QR tags are scanned and data are summarized, showing the three types of sheets necessary for the system. Panel (a) shows the format of the data input sheet. Panel (b) shows how data from the spreadsheet in panel a are filtered by category (here, into separate sheets for health checks and food/water replacements). Panel (c) shows the daily summary sheet, updated by date of last scan per cage and per category as well as the date of the most recent note and the contents of the most recent note per cage.



(a) Enter data via scanning

Sheet1

	A	B	C
1	QRscan	datetime	notes
2	Cage01 good.health	2022-06-03 16:14	
3	Cage01 cleaned.water.food	2022-06-03 16:15	
4	Cage01 full.clean	2022-06-03 16:15	
5	Cage01 enrichment	2022-06-03 16:15	
6	Cage02 good.health	2022-06-03 16:15	
7	Cage02 cleaned.water.food	2022-06-03 16:16	

(b) Parse entered data by category

HEALTH [File: SUMMARY]

	A	B	C	D	E
1	Code	date.time	notes	cageID	type
2	Cage01 good.health	2022-06-03 16:14		Cage01	good.health
3	Cage02 good.health	2022-06-03 16:15		Cage02	good.health
4	Cage03 good.health	2022-06-03 16:16		Cage03	good.health

WATER.FOOD [File: SUMMARY]

	A	B	C	D	E
1	Code	date.time	notes	cageID	type
2	Cage01 cleaned.water.food	2022-06-03 16:15		Cage01	cleaned.water.food
3	Cage02 cleaned.water.food	2022-06-03 16:16		Cage02	cleaned.water.food
4	Cage03 cleaned.water.food	2022-06-03 16:16		Cage03	cleaned.water.food

(c) Summarize and verify

SUMMARY.VIEW [File: SUMMARY]

	A	B	C	D	E	F	G	H
1		SCHEDULE BY CATEGORY					NOTES	
2		*Every day (if not done today, then yellow)*	*Every day (if not done today, then yellow)*	*as necessary*	*Mondays (once a week, if more time has passed, will turn yellow)*	*At least once every 14 days (if it's been more than 14 days, will turn yellow)*	*Date of most recent note*	
3	CageID	Health	Water&Food	spot.clean	full.clean	Enrichment	Notes	Most.recent.note
4	Cage01	2022-06-04	2022-06-04	#N/A	2022-06-04	2022-06-04	#N/A	#N/A
5	Cage02	2022-06-04	2022-06-04	#N/A	2022-06-04	2022-06-04	2022-06-03	testing the notes system
6	Cage03	2022-06-04	2022-06-04	#N/A	2022-06-04	2022-06-04	#N/A	#N/A
7	Cage04	2022-06-04	2022-06-04	#N/A	2022-06-04	2022-06-04	#N/A	#N/A

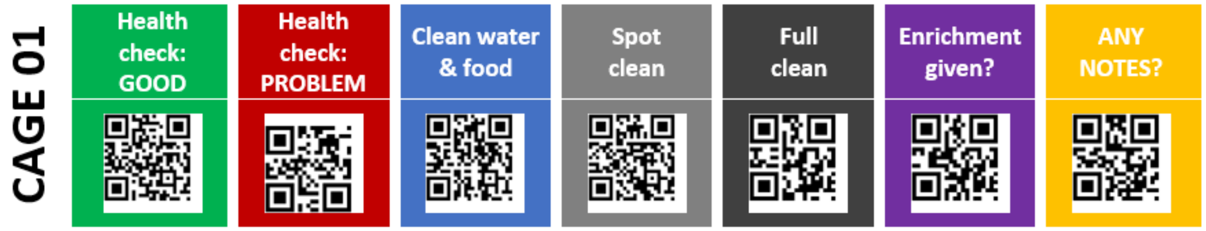
Step 1: Purchase QR scanner device

Data is input into the system by scanning QR codes into a spreadsheet using a handheld QR scanner. These scanners are wireless and transmit over Bluetooth to a USB receiver plugged into a computer which acts as a central hub (see Step 3). Wireless QR scanners are available for between 30-100$USD. I used a Tera 2D Barcode Scanner (Model D5100) which cost about 45$USD at time of purchase in 2021. For QR scanning capability, look for a “2D” barcode scanner (“1D” scanners can only read traditional barcodes, not QR codes). These scanners are usually rechargeable with a USB cable; the more expensive models often come with a charging cradle that functions as a stand. The battery life is quite good on some of these models; in our lab, we charge our scanner usually only once every 2-3 weeks. Different models vary in their range – in our case, our model is able to transmit to a central computer from an adjacent room, despite concrete walls and two closed doors between the scanner and the receiver (each QR scanner model will have range specifications which should be checked to pick a model that will work well in the desired scanning environment).

Step 2: Generate and print QR codes

Custom QR codes can be freely generated using the R package “qrcode” (Teh and Onkelinx 2021). QR codes can also be generated using several free websites for those not comfortable with R. In our system, each code contains the cage ID and the category or task that is being scanned, separated by a space (for example, “Cage01 good.health”). See Box 1 for the R script I used to generate tags for 20 cages, each with seven different categories of tasks or activities that we scan and track. In my lab, we use the categories “good.health” and “PROBLEM” to enter data on health checks, “cleaned.water.food”, “spot.cleaned”, and “full.clean” to track different types of cleaning activities, “enrichment” to record when birds are given enrichment items like an alfalfa feeder or a dust bath, and “NOTE” to add a row to the database into which we can enter any extra notes by hand after scanning (see Box 1 for suggestions on how to customize the QR code text).

Figure 2. Example cage QR tag with QR codes for six animal care categories and one for notes.



After the image files for each QR code have been generated, they can be compiled into a table with one row of codes per cage. I used Microsoft Word to compile the tables: I found that this approach, while somewhat time-consuming, provided the easiest method for color and layout customization. In Word, I added a colored background to each QR code that corresponded to each code's category to help more quickly differentiate categories and reduce the chances of scanning the wrong code (see Fig. 2). We generally have good success scanning QR codes that are sized at 0.75 inches.

Once printed, these tags are cut out, with each cage's tags in a single strip and laminated for long-term durability. I used an inexpensive thermal laminator to protect the tags. Tag strips were laminated with 5 mil laminator pouches. Laminators are widely available for 20-60$USD (e.g., Apache AL13 Thermal Laminator). An optional paper trimmer greatly facilitates this step (e.g., Worklion Paper Cutter with Security Blade). I used zip ties to secure the tags to the front of each cage. To avoid mis-scanning adjacent codes, we notched out a paper card with a 1-inch opening and use it to cover all QR codes except the desired one when scanning.

Box 1. R script to generate custom QR tags

```
# load packages
library(tidyverse)
library(qrcode)
library(png)

# create vector of cage IDs – customize cage numbers here
cages <- str_pad(seq(1:20), 2, pad = "0") #this function forces single digit numbers to have a leading 0

# create vector of categories for tags
checks <- c("good.health", "PROBLEM", "cleaned.water.food", "spot.cleaned", "full.clean",
"enrichment", "NOTE")

# create dataframe with all cage IDs and all categories of tag checks – customize category names here
cages.checks <- expand.grid(cages, checks)
colnames(cages.checks) <- c("CageID", "Check.Type")
cages.checks <- cages.checks %>% arrange(CageID, Check.Type)

# create full text labels for QR codes (use space as separator between cageID and task category)
cages.checks$Label <- paste("Cage", cages.checks$CageID, cages.checks$Check.Type, sep=" ")

# loop to create png images for each QR tag
for(i in 1:nrow(cages.checks)){
  #specify path & file name for each png image
  mypath <- file.path("qrs.spot_full.clean", paste(cages.checks$Label[i], ".png", sep = ""))
  #create png using qrcode_gen from package
  png(file=mypath)
  qrcode::qrcode_gen(cages.checks$Label[i])
  dev.off()
}
```

Step 3: Set up central hub for data input

Data from the QR scanner are transmitted to a central computer via a USB-connected Bluetooth receiver. Many kinds of computers can be used as the central hub: at a minimum, the computer needs a USB port, an internet connection, and the ability to access Google Sheets. In my lab we use a basic desktop computer on which we do other kinds of data entry. Any computer that can connect to the internet (either via wired connection or via Wi-Fi to access Google Sheets) and which has a USB port (for the QR scanner’s Bluetooth receiver) can be used as the hub. For example, small single-board computers like the Raspberry Pi could be a very low-cost option.

Step 4: Input data from QR scans

Data are scanned into a Google Sheet for data input (Fig. 1a). This sheet contains three columns: Column A (“QRscan”) contains the QR scan data, Column B (“datetime”) contains date/time stamps showing when each tag was scanned, and Column C contains optional notes that can be added by hand after scanning. To enter the scan data, the data input sheet must be open on the computer and the cursor must be in the next empty cell in the first column. To facilitate data entry, I added a Google Apps Script to automatically navigate to the next empty cell when the Google Sheet is opened (see script in Box 2). To begin scanning, activate the wireless QR scanner and start scanning tags. Each scan will add the text associated with each QR tag to a new row in the QRscan column. Each scan is automatically timestamped with the time at which each scan was completed via a Google Apps Script (Box 2). Column C (“notes”) provides an area to add any notes by hand after scanning. The input sheet provided does not require any modifications to use prior to data entry, even if the cage numbers and/or the care categories have been customized and differ from the example code.

Box 2. Google Apps Script to automatically add date/time and move cursor to next empty row. Information on how to add date/time is from a tutorial by Dan Nguyen, http://blog.danwin.com/how-to-automatically-timestamp-a-new-row-in-google-sheets-using-apps-script/. From Google Sheets, select “Extensions” then “Apps Script”, paste the code below into the script editor, and save it as a Code.gs file named “datetime auto adder**”.**

```
// Automatically add date/time stamp
var SHEET_NAME = 'Sheet1';
var DATETIME_HEADER = 'datetime';

function getDatetimeCol(){
  var headers = SpreadsheetApp.getActiveSpreadsheet().getSheetByName(SHEET_NAME).getDataRange().getValues().shift();
  var colindex = headers.indexOf(DATETIME_HEADER);
  return colindex+1;
}

function onEdit(e) {
  var ss = SpreadsheetApp.getActiveSheet();
  var cell = ss.getActiveCell();
  var datecell = ss.getRange(cell.getRowIndex(), getDatetimeCol());
  if (ss.getName() == SHEET_NAME && cell.getColumn() == 1 && !cell.isBlank() && datecell.isBlank()) {
    datecell.setValue(new Date()).setNumberFormat("yyyy-MM-dd hh:mm");
  }
};


// Automatically teleport to last row for easier data entry
function onOpen(e) {
  var spreadsheet = e.source;
  var sheet = spreadsheet.getActiveSheet();
  var lastRow = spreadsheet.getLastRow();
  if (sheet.getMaxRows() == lastRow) {
    sheet.appendRow([""]);
  }
  lastRow = lastRow + 1;
  var range = sheet.getRange("A" + lastRow + ":A" + lastRow);
  sheet.setActiveRange(range);
}
```

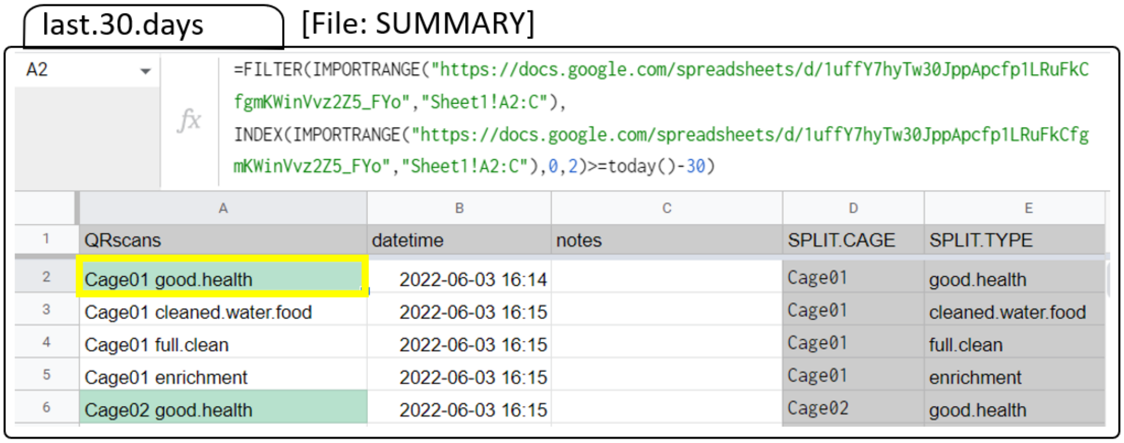
Step 5: Filter and collate raw data

The second Google Sheet workbook contains several sheets that serve as intermediaries between the raw input data and the summary sheet. These intermediary sheets should not be edited during use of the system and are just used to facilitate connecting the input sheet to the summary sheet. The first sheet ("last.30.days") filters the raw data (from the input sheet) to include only scans made during the past 30 days (see code in Box 3, this sheet requires no modification to work in the system even if care categories and/or cage IDs have been customized and differ from the example provided).

Box 3. Google sheet code to automatically filter raw data from input sheet.

(a) Filter to show just the most recent scans: The code below is entered into cell A2 of the "last.30.days" sheet (highlighted below in yellow). The criteria specified in IMPORTRANGE shows how to import data within the last 30 days.

```
=FILTER(IMPORTRANGE("https://docs.google.com/spreadsheets/d/1uffY7hyTw30JppApcfp1LRuFk
CfgmKWinVvz2Z5_FYo","Sheet1!A2:C"),
INDEX(IMPORTRANGE("https://docs.google.com/spreadsheets/d/1uffY7hyTw30JppApcfp1LRuFkCf
gmKWinVvz2Z5_FYo","Sheet1!A2:C"),0,2)>=today()-30)
```



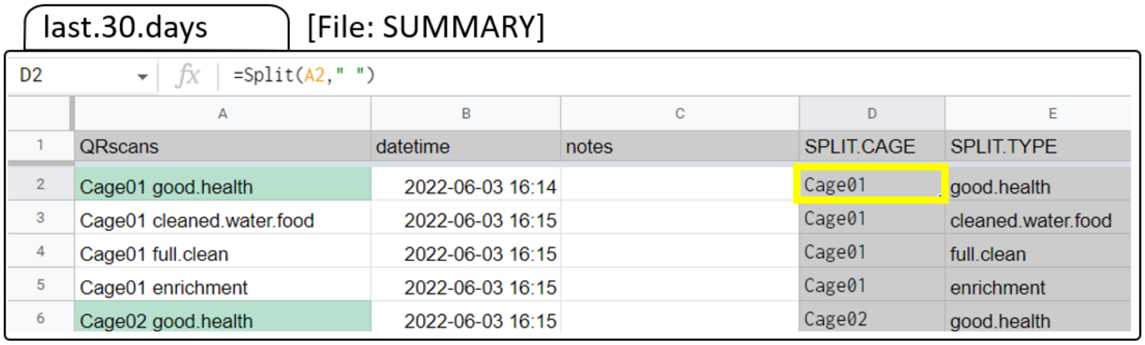
[File: SUMMARY]

A2

```
=FILTER(IMPORTRANGE("https://docs.google.com/spreadsheets/d/1uffY7hyTw30JppApcfp1LRuFkC
fgmKWinVvz2Z5_FYo","Sheet1!A2:C"),
INDEX(IMPORTRANGE("https://docs.google.com/spreadsheets/d/1uffY7hyTw30JppApcfp1LRuFkCfg
mKWinVvz2Z5_FYo","Sheet1!A2:C"),0,2)>=today()-30)
```

	A	B	C	D	E
1	QRscans	datetime	notes	SPLIT.CAGE	SPLIT.TYPE
2	Cage01 good.health	2022-06-03 16:14		Cage01	good.health
3	Cage01 cleaned.water.food	2022-06-03 16:15		Cage01	cleaned.water.food
4	Cage01 full.clean	2022-06-03 16:15		Cage01	full.clean
5	Cage01 enrichment	2022-06-03 16:15		Cage01	enrichment
6	Cage02 good.health	2022-06-03 16:15		Cage02	good.health

(b) Split cage ID and categories from Column A into two separate columns: Column D (highlighted below in yellow) in the "last.30.days" sheet contains code to automatically split the information in Column A to separate the cage numbers and the categories of tasks (which then appear separately in Columns D and E). Add the split code ("=Split(A2," ")") into all cells in column D by selecting cell D2 and dragging it to the bottom of the sheet. Google Sheets will automatically update the split code to apply to each row (for row 3, the split will be applied to cell A3, etc.).



[File: SUMMARY]

D2

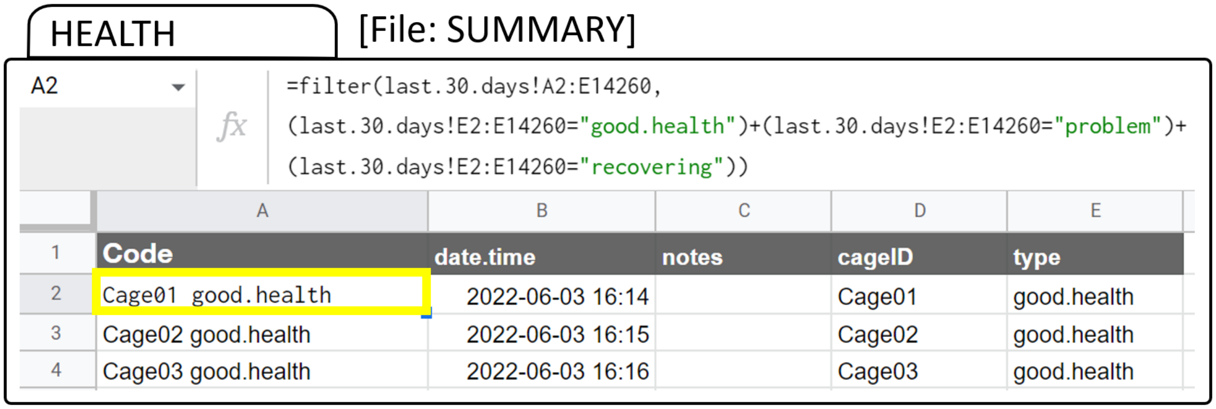
```
=Split(A2," ")
```

	A	B	C	D	E
1	QRscans	datetime	notes	SPLIT.CAGE	SPLIT.TYPE
2	Cage01 good.health	2022-06-03 16:14		Cage01	good.health
3	Cage01 cleaned.water.food	2022-06-03 16:15		Cage01	cleaned.water.food
4	Cage01 full.clean	2022-06-03 16:15		Cage01	full.clean
5	Cage01 enrichment	2022-06-03 16:15		Cage01	enrichment
6	Cage02 good.health	2022-06-03 16:15		Cage02	good.health

This filtered dataset is then split into categories, with one sheet per type of care that is being tracked. Each category of QR codes that are tracked should be filtered into separate sheets. In our system, we use 6 intermediary sheets: a health check, logs of water/food changes, spot cleaning, full cleaning, and enrichment, and a sheet for extra notes. Figure 1b shows how data from the spreadsheet in Figure 1a are filtered by category (here, showing the “HEALTH” and “WATER.FOOD” categories). Box 4 shows the code needed to filter by category.

Box 4. Google sheet code to automatically filter recent data by category. This is an example showing how “health” data are filtered. We want to have the “good health”, “problem”, and “recovering” scans all compiled in this sheet so that we can double-check that health was assessed every day. In cell A2 of the “HEALTH” intermediary sheet (highlighted below in yellow), the following code is entered, which will filter and import all relevant data from the recent data, which is already filtered into the sheet “last.30.days”. The category-only data are in column E of the sheet “last.30.days”.

```
=filter(last.30.days!A2:E14260, (last.30.days!E2:E14260="good.health") +
(last.30.days!E2:E14260="problem") + (last.30.days!E2:E14260="recovering"))
```



```
=filter(last.30.days!A2:E14260,
(last.30.days!E2:E14260="good.health")+(last.30.days!E2:E14260="problem")+
(last.30.days!E2:E14260="recovering"))
```

	A	B	C	D	E
1	Code	date.time	notes	cageID	type
2	Cage01 good.health	2022-06-03 16:14		Cage01	good.health
3	Cage02 good.health	2022-06-03 16:15		Cage02	good.health
4	Cage03 good.health	2022-06-03 16:16		Cage03	good.health

For labs that want different animal care categories, these sheets by care type will require modifications: to customize, generate QR codes containing the new categories of care that are desired, change the name of the sheet to match each new category of care (i.e. for a new category called “dose” that replaces “good.health”, change “HEALTH” to “DOSE” in Box 4), then change the filter code in cell A2 to pull the correct data from the input data sheet (i.e. change “good.health” to “dose” and delete the code mentioning the other health types).

Step 6: Summarize filtered data to verify and track care

The final type of sheet is the summary and verification sheet, shown in Figure 1c. This sheet uses the cage IDs to summarize the date of the last scan by cage and category (see code in Box 6). We use this summary sheet as a general overview of care and as a way to easily double-check and verify that care is being provided to each cage on the correct schedule, using several methods to track care. The date of the most-recent QR scan for each cage by each category is shown in the summary table (Box 5a), along with the contents of the most-recent note per cage (Box 5b).

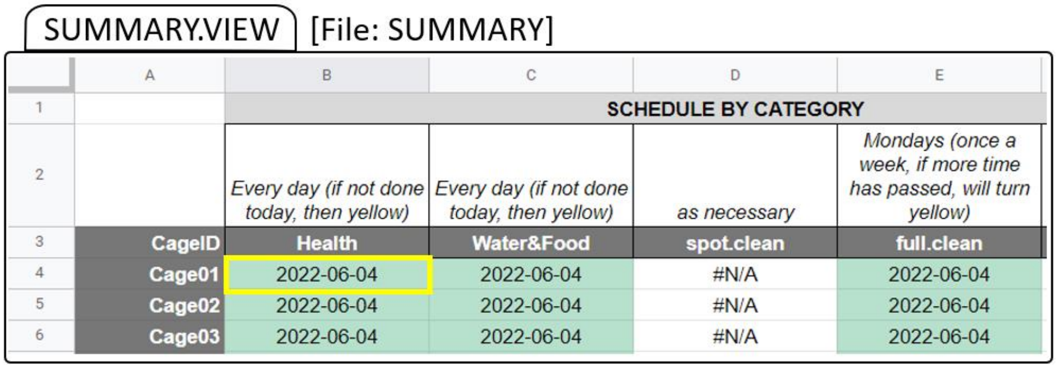
Box 5. Google sheet code to return the date of the most recent scan by cage and category and note information.

(a) Show dates of most recent scans by category: This is an example showing how "health" data are filtered. In cell B4 (highlighted below in yellow), the following code is entered:

```
=large(filter(HEALTH!$B$2:$B,HEALTH!$D$2:$D=A4),1)
```

This code filters out the most recent date of health scans for the cage name in cell A4 (here, Cage01). Cell B5 would then have the same code, but reference the cage name in cell A5. Code for all other categories are set up similarly, for example C4 contains data about water and food changes:

```
=large(filter(WATER.FOOD!$B$2:$B,WATER.FOOD!$D$2:$D=A4),1)
```

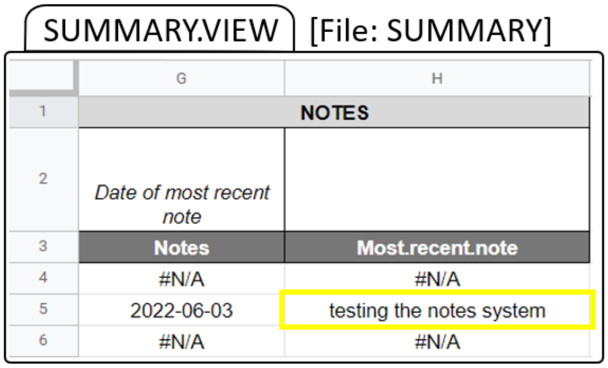


SUMMARY.VIEW [File: SUMMARY]

	A	B	C	D	E
1		SCHEDULE BY CATEGORY			
2		*Every day (if not done today, then yellow)*	*Every day (if not done today, then yellow)*	*as necessary*	*Mondays (once a week, if more time has passed, will turn yellow)*
3	CageID	Health	Water&Food	spot.clean	full.clean
4	Cage01	2022-06-04	2022-06-04	#N/A	2022-06-04
5	Cage02	2022-06-04	2022-06-04	#N/A	2022-06-04
6	Cage03	2022-06-04	2022-06-04	#N/A	2022-06-04

(b) Show the contents of the most-recent note: Column G identifies the date of the most-recent note (if any) and then adds the note to Column H pertaining to the proper cage name (here, from cell A5, highlighted in yellow). The following code parses the notes:

```
=filter(NOTES!$C$2:$C, NOTES!$D$2:$D=A5,
NOTES!$B$2:$B=G5)
```



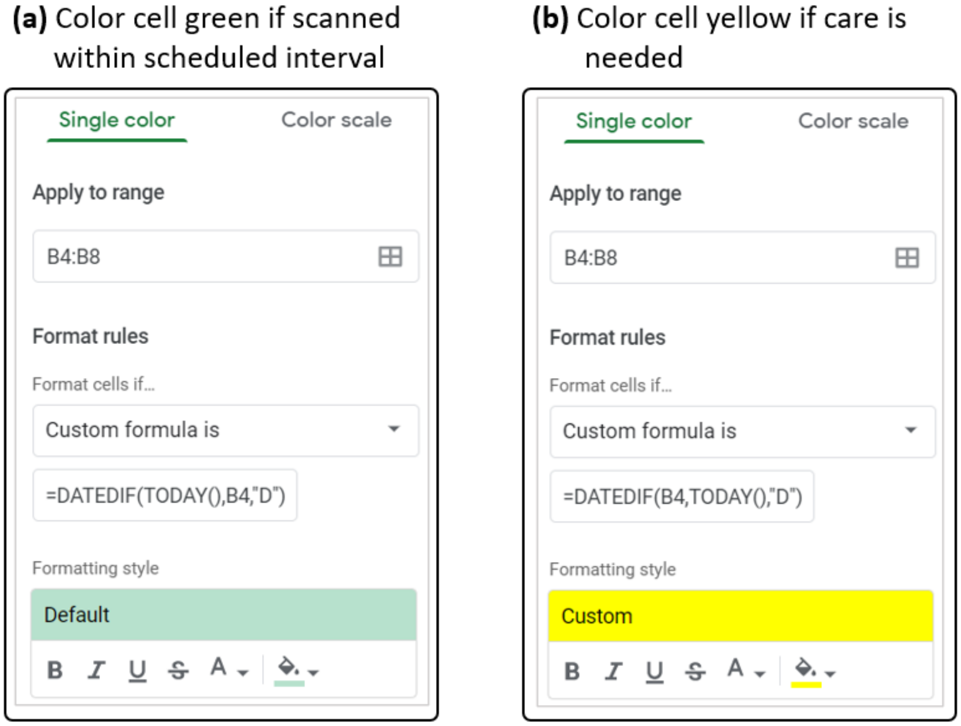
SUMMARY.VIEW [File: SUMMARY]

	G	H
1	NOTES	
2	*Date of most recent note*	
3	Notes	Most.recent.note
4	#N/A	#N/A
5	2022-06-03	testing the notes system
6	#N/A	#N/A

As an additional indicator of whether tasks have been completed on schedule, we also use conditional formatting rules to control and automatically update the color of the cells based on the time since the last scan and the interval at which each category of care needs to be completed (Fig. 1c). To set or change the conditional formatting rules, select “Format” then “Conditional formatting” in the Google Sheet to view the rule menu. In “Apply to range” specify the cells which the rule applies to (here, conditional formatting is applied consistently across all cages within each care category). In “Format rule”, choose “Custom formula is”, then add the conditional formatting desired and choose the response color to use when the rule is satisfied. For an example of our two rules for color-coding to identify whether a task has or has not been completed each day, see Box 6. In our system, we set each cage’s cell to green when each care category has been completed within the time limit set by the conditional formatting (Box 6a) while tasks which still need to be completed are set to yellow (Box 6b). For example, Box 5a shows that all the health checks and changes of food and water have been completed “today” for all cages and are shaded green; if a cage has not been checked or had these changed on the current date, the cell color turns yellow (see Figure 1c).

This summary table provides a quick and easy way to verify that care has been completed for each cage. Any cage that was accidentally missed can be very quickly and easily identified and care can be completed. This system also greatly aids in monitoring the schedule for tasks that are completed on a different cycle.

Box 6. Setting cell-level conditional formatting. This is an example showing how "health" summary data are conditionally formatted to update the color of the cells based on when each cage was last scanned. The example below are two conditional formatting rules applied to cell B4 in the SUMMARY.VIEW sheet. Similar code is used for all cages in this category, with other categories calling the corresponding other intermediary sheets. (a) if Cage01 has been scanned today, update the cell color to green (code= "=DATEDIF(TODAY(),B4,"D")=0"); (b) if Cage 01 has not been scanned today and needs care, update the cell color to yellow: (code= "=DATEDIF(B4,TODAY(),"D")>0").



(a) Color cell green if scanned within scheduled interval

Single color | Color scale

Apply to range

B4:B8

Format rules

Format cells if…

Custom formula is

=DATEDIF(TODAY(),B4,"D")

Formatting style

Default

(b) Color cell yellow if care is needed

Single color | Color scale

Apply to range

B4:B8

Format rules

Format cells if…

Custom formula is

=DATEDIF(B4,TODAY(),"D")

Formatting style

Custom

Practical use and general troubleshooting suggestions

We have been validating and tweaking this scanning system in the lab since July 2021. On a day-to-day basis, we use this system as a two-step verification of care. The care team completes all care activities for the day, then scans all the cages to document the type of care provided. This cage-by-cage scanning allows us to easily proof that care is complete: for example, before scanning each tag, we do a final check that the health of all individuals is good, and that water and food have been cleaned and replaced. If everything looks good, those QR codes for that particular cage are scanned before moving on to the next cage. Any accidentally missed water or food containers can be identified and rectified at this stage.

At the end of scanning, we check the summary sheet to make sure that all cage cells across all categories of care are green (indicating care has been provided on that day), which provides an additional verification that care to all cages has been completed, and no cages have been accidentally overlooked. Any cage with a cell that is not colored green is re-checked and re-scanned. In our experience, this scanning and verification system has helped us rectify missed care at least once per week and has made our care team aware that missed care is a common issue to be aware of and careful to address. In our lab, we have noticed spikes in aggression around the time that food or water changes have been accidentally delayed; this extra aggression can necessitate temporary separation of groups or permanent reassignment to new groups, which both cause critical disruptions to our social and cognitive experiments. Avoiding these disruptions by maintaining and ensuring consistent animal care is of high priority.

Because the system does not require Wi-Fi to function, it can be used in a wide range of settings. As long as a computer is available for data input and that computer can connect to the internet to access Google Sheets, the system as described here will work. Some scenarios may require greater security for animal care records. To improve security, a new Google account can be created for the care records and two-factor authentication can be enabled to better control access. For higher security, a local system could be used to track care. Similar code could be implemented in a program like Microsoft Excel (but this would require a redesign of the specific system described here to adapt the code for use outside Google Sheets).

During system validation, my team has identified several common errors in system use that can cause issues. I have tweaked the system to account for several of these areas of fragility. Here, I summarize some common user errors and provide solutions.

1) **Running out of new rows in the input sheet:** To address running out of new rows, just go to the bottom of the input sheet and add the desired number of new rows.

2) **Mismatch between cage IDs or category codes in the QR tags and the spreadsheet criteria:** To ensure cage ID or category codes match, check the QR code generation procedures to make sure that the codes generated match the desired cage ID and category formats and check the filtering criteria in the intermediary and summary sheets.

3) **Accidentally scanning QR codes into a sheet other than the input sheet:** To avoid erroneously scanning QR codes into a sheet other than the data input sheet, put a warning label on the scanner handset that the input sheet needs to be open and the cursor needs to be in the next empty cell in Column A. If codes are scanned into the wrong area, the user can use the "undo" function to reset cells to the proper contents and cages can be re-scanned with the data input sheet.

4) **Missing scans:** To address scans that are not properly received by the central computer, the QR codes just need to be re-scanned. In our experience, missed scans are relatively rare, but most commonly occur for the very first scan of a session.

Conclusions

Our animal care monitoring system is low cost and flexible way to document and verify animal care. Paired with simple scripts in a cloud-based spreadsheet, the scanned QR code data can be summarized to provide real-time verification of care that can be checked by any member of the team with access to the Google Sheet and an internet connection. Our team has found this system to be very reassuring both to people completing care as well as supervisors monitoring care. As care is being completed, the summary sheet can be checked while workers are still in the animal care room to double-check that care has been provided to all cages and quickly address any deficits. Remotely, other team members can access the spreadsheets and easily verify whether care has been provided each day. This system is highly customizable and can be adapted for a wide range of species receiving many different types of care. In addition to tracking care, the system can also be used to track experimental procedures, behavioral experiments or cognitive tests, or to log general behavioral observations. Any research group caring for or monitoring many individuals across separate cages, tanks, or enclosures, and especially when multiple categories of care or monitoring are required and when activities are completed on a non-daily schedule, would benefit from a system like this one.

Acknowledgments

Thanks to Sanjay Prasher and Alexis Williams, who helped design an early prototype of the system described here, to Sanjay Prasher, Annemarie van der Marel, Claire O'Connell, Chelsea Carminito, and Xavier Francis, who helped test, validate, and refine the system, and to comments from two anonymous reviewers which helped improve the manuscript. During preparation of this work, EAH was supported by NSF IOS 2015932.

References

Diazgranados M, Funk VA. 2013. Utility of QR codes in biological collections. PhytoKeys. 25(25):21. doi:10.3897/PHYTOKEYS.25.5175.

Fife-Cook I, Franks B. 2019. Positive Welfare for Fishes: Rationale and Areas for Future Study. Fishes 2019, Vol 4, Page 31. 4(2):31. doi:10.3390/FISHES4020031.

Franks B. 2018. Cognition as a cause, consequence, and component of welfare. In: Mench JA, editor. Advances in Agricultural Animal Welfare: Science and Practice. Woodhead Publishing. p. 3–24.

Green T, Smith T, Hodges R, Fry WM. 2017. A simple and inexpensive way to document simple husbandry in animal care facilities using QR code scanning. Laboratory Animals. 51(6):656–659. doi:10.1177/0023677217718004.

Haywood JR, Greene M. 2008. Avoiding an Overzealous Approach: A Perspective on Regulatory Burden. ILAR Journal. 49(4):426–434. doi:10.1093/ILAR.49.4.426.

National Research Council. 2011. Guide for the Care and Use of Laboratory Animals, 8th Edition. Washington D.C.

Oteyo IN, Toili MEM. 2020. Improving Specimen Labelling and Data Collection in Bio-science Research using Mobile and Web Applications. Open Computer Science. 10(1):1–16. doi:10.1515/COMP-2020-0002/MACHINEREADABLECITATION/RIS.

Teh V, Onkelinx T. 2021. qrcode: Generate QRcodes with R. Version 0.1.4.