MyVivarium: A cloud-based lab animal colony management application with near-realtime ambient sensing

Robinson Vidva[a], Mir Abbas Raza[b], Jaswant Prabhakaran[c], Ayesha Sheikh[b], Alaina Sharp[b], Hayden Ott[b], Amelia Moore[b], Christopher Fleisher[b], Pothitos M. Pitychoutis[b], Tam V. Nguyen[c], and Aaron Sathyanesan[b,d] *

[a] *Digirobi Solutions, Bengaluru, Karnataka, India*

[b] *Department of Biology, College of Arts & Sciences, University of Dayton, Dayton, OH, USA*

[c] *Department of Computer Science, College of Arts & Sciences, University of Dayton, Dayton, OH, USA*

[d] *Department of Electrical and Computer Engineering, School of Engineering, University of Dayton, Dayton, OH, USA*

*Corresponding author:

Aaron Sathyanesan, PhD

Contact email: asathyanesan1@udayton.edu

Address: 300 College Park, Dayton, OH, USA

ABSTRACT

Research-animal colony management is a critical determinant of scientific productivity in labs conducting preclinical research. However, labs generally use *ad hoc* paper-based or spreadsheet-based methods to manage animal colonies. Current software-based solutions are limited based on cost, ease of implementation and deployment, and lack of remote access. In addition, most current solutions lack integration of realtime monitoring of ambient variables that affect colony wellbeing and breeding efficiency. Keeping these functionalities in mind, we built *MyVivarium* – a cloud-based animal colony database management web application that can be easily deployed and sustained for a cost comparable to starting and maintaining a lab website. For mouse colony management, MyVivarium allows for the tracking of individual mice or litters within holding-cage and breeding-cage records by multiple users at designated levels of access. Physical identities of cages are mapped onto the database using printable cage-card templates with cage-specific QR codes, enabling quick record updating using mobile devices. Tasks can be easily assigned with reminders for experiments or cage maintenance. Finally, MyVivarium integrates near-realtime internet-of-things (IoT)-based ambient sensing using low-cost open-source hardware to track humidity, temperature, vivarium-worker activity, and room illuminance. Taken together, MyVivarium is a novel open-source cloud-based application template that can serve as a low-cost, simple, and efficient solution for digital management of research animal colonies.

KEYWORDS: vivarium, colony management, mouse colony, mouse facility, open-source technology, IoT sensors, biological database.

INTRODUCTION

Animal colony management is a routine task performed by many laboratories that use model systems to answer biological research questions. Many labs that use rats or mice as a model system often are responsible for animal husbandry and well-being of the animal subjects under their purview. This can result in the day-to-day management of hundreds of animal cages per lab. While technological solutions have been widely adopted in other areas of routine lab tasks ranging from scientific data management to chemical inventory, animal colony management has remained fairly insulated from adoption of information management systems. Although large mouse repositories often invest in software solutions to increase efficiency in mouse colony management, barriers remain in the average academic lab choosing a dedicated software-based solution (Maier et al., 2015).

With the steadily increasing *per diem* costs of maintaining rodent research colonies ("Per Diem Animal Housing Rates | Boston University Office of Research," 2016), efficient handling of resources is an increasing concern. This is especially true for new and early career principal investigators (PIs) who typically transition from independent bench research to a management-heavy role. Typical start-up budgets for new PIs prioritize big-ticket equipment purchases, personnel costs (Dolgin, 2018), and may even include statistical software costs, however, research management tools and applications are not usually given much consideration, with a dedicated laboratory information management system (LIMS) often being designated a luxury add-on. Given the significantly high prices of commercially available LIMS, this lack of consideration may seem justified. Further, it is often surmised that animal colony management can be accomplished *ad hoc*. While large-scale survey data of mouse researchers is lacking, anecdotal evidence from the collective research experiences of the authors of this manuscript indicate that most solutions for mouse colony management generally take the form of a spreadsheet using *Microsoft Excel* or *Google Sheets* which are regularly updated in a semi-

collaborative manner. During a typical colony inspection, a researcher would visit the vivarium, which is often separated in location from the main area of the lab, carrying along a paper copy of the colony cage list. Following routine updates to the paper list, this information is then soft-copy transcribed at a later time since directly updating spreadsheet entries in the vivarium while monitoring mouse cages may not be as portable-device-friendly. Information directly related to a particular strain, such as genotype visualized on a gel image, is usually maintained separately in a lab notebook. While many labs have *de facto* adapted this general form of mouse colony data management, this method is cumbersome and has multiple points of failure due to its reliance on disparate tools and the lack of an integrated framework.

Rodent strains used in preclinical research as model systems of disease have genetic or chromosomal modifications that often make them more sensitive to changes in ambient temperature, relative humidity, and light (Chesler et al., 2002; Jaric et al., 2022; Kudo et al., 2019; Richter et al., 2011). In addition, human presence or foot traffic may also negatively affect genetically or chromosomally modified mouse strains. Consequently, alterations in these vivarium ambient factors can result in reduced colony productivity and breeding efficiency as well as general colony wellbeing. In-house vivarium staff usually perform routine checks on these ambient factors – however, since these checks are only performed once or twice a day, alterations in ambient factors that occur between inspection windows or outside normal working hours may be missed. Sudden unexplained losses in colony breeding performance can cause significant delays in experimental timelines and decreased return-on-investment. The effects of lab animal colony related disruption can potentially be disproportionately worse for early-career PIs compared to mid-career or senior PIs since earlier success in terms of research output for new PIs may be predictive of successful grant proposals during this stage. Commercial solutions to monitor ambient factors in rodent colonies do exist, however, these are significantly

cost-prohibitive. A cost-effective colony-management solution that can also track alterations in ambient environmental factors would thus be of significant benefit.

The advent of the modern web has brought with it ever-improving tools for rapid and remote management of scientific data (Hall and Tiropanis, 2012; Wood et al., 2014). Leveraging the decreasing cost and increasing ease of modern web development, early career researchers have become conversant with designing websites for sharing their research programs (Mishra, 2020). We reasoned that if the barrier to entry for using a rodent colony management system were reduced to the level of launching and maintaining a laboratory website, this would increase the adoption of such a system.

Recently, a few open source database applications for rodent colony management solutions have been introduced (Pawar and Batra, 2018; Renschen et al., 2020). However, these systems were primarily implemented to run on local servers and networks, limiting their access especially in situations of contingency (Chambliss and Tolan, 2020; Gao et al., 2021). While these solutions can be moved to the internet, it is not trivial to do this securely and seamlessly. For someone such as a bench-biologist with limited knowledge of web development, this is especially difficult to carry out. Further, to our knowledge, no open-source solution for lab animal colony management has integrated ambient sensing. In this manuscript, we describe the design of *MyVivarium* – an open-source application framework for lab animal colony management with cloud-based access as the primary starting point. We built MyVivarium using a PHP-based technology stack running on a managed cloud-hosted virtual private server. Importantly, we prioritized cost by using an inexpensive server option (~$6 USD per month for 1GB RAM, 25GB SSD, 1 TB Transfer, 1 Core Processor), however, application performance was not affected with overall server CPU usage being relatively low. MyVivarium enables all the necessary standard operating procedures for vivarium maintenance including cage and mouse record maintenance with the ability to generate QR-code-enhanced cage cards for easy mobile access

and tracking. In order to efficiently track experiments linked to mice in specific cages, tasks and reminder can be set up with designated researchers assigned to these tasks. Finally, we describe the assembly and integration of automated low-cost Raspberry-Pi-IoT ambient sensing monitoring sensors which continuously stream temperature, humidity, illuminance, and vivarium room activity data to MyVivarium in near-realtime. Fluctuations in ambient environmental factors based on user-defined thresholds and time-windows trigger automated alerts that are sent via a chat message to designated personnel. MyVivarium is thus a complete open-source solution for cost-efficient cloud-based lab animal colony management suitable for academic research labs as well as vivarium management staff.

RESULTS

We built MyVivarium using modern web technologies for both server-side and client-side operations (**Figure 1**). The backend uses *PHP* for server-side scripting and *MySQL* for database management. The front end utilizes *HTML*, *CSS*, and *JavaScript*, including the *Bootstrap* framework for responsive design and user interface components. *Composer* is employed for dependency management, incorporating libraries such as *PHPMailer* for email functionalities and *Dotenv* for environment variable management.

Key features of MyVivarium include user registration and profile management, lab and cage management, real-time ambient sensing with internet-of-things (IoT) sensors, notes management, tasks and reminders, email notifications, and print cage card functionality (**Figure 2**). The application supports secure session handling, cross-site request forgery (CSRF) protection, and role-based access controls. The user management functionality includes features for registration, login, profile management, email verification, and password reset via email. Lab and cage management modules enable users to add, view, edit, and delete breeding

134 and holding cages. Each module provides detailed information about the cages, including
135 associated litter and mouse data. Additionally, users can manage strains by adding, editing, and
136 deleting strain records. We have shared all necessary code and a how-to guide in a public
137 repository (https://github.com/myvivarium/MyVivarium) for users to get their specific MyVivarium
138 web applications set up according to their particular needs and requirements. We have also
139 included a link to a demo website in the repository to showcase the different functionalities
140 within MyVivarium (https://demo.myvivarium.online/).

141 Cage, mouse, and litter management forms the core of MyVivarium (**Supplemental Figure 1**).
142 We designed the database architecture using active feedback from vivarium workers over the
143 course of development of MyVivarium. The MyVivarium database includes the `cages` table
144 which forms a core part of the architecture and contains essential information about each cage,
145 including the cage ID, the name of the principal investigator, the number of mice, and any
146 specific remarks linked to a cage (**Supplemental Figure 1**). Linked to this core table is the
147 `users` table, which holds necessary information about users, including their role within the lab,
148 position or designation, and MyVivarium access approval status. This ensures that each cage
149 can be associated with specific users, facilitating proper access and management of
150 responsibilities. The `holding` table stores essential information about individual mice in holding
151 cages, including their strain, date of birth, sex, and parent cage. This table ensures that all
152 necessary details about each holding cage are recorded systematically. The `mouse` table stores
153 detailed information about individual mice within the holding cages. The database architecture
154 for breeding cages in MyVivarium includes the `breeding` table which contains information
155 about each breeding pair, including their source cage IDs, type of cross, male and female IDs,
156 and their dates of birth. Linked to this table is the `litters` table which records details about
157 offspring, such as the number of pups per litter, number of surviving pups, sex, and any other
158 relevant remarks. This setup enables monitoring and analysis of breeding outcomes per cage.

The `strains` table contains information about various strains of mice including strain IDs, names, and other identifiers, enabling tracking of different genetic lines. In order to link documentation related to the institutional animal care and use committee (IACUC) to specific strains, we also included an `iacuc` table. This enables better integration of IACUC-related tasks to help with the preparation of compliance and animal usage reports. Additional tables, including `files` and `notes`, support the documentation and communication needs within MyVivarium. The `files` table stores metadata about uploaded documents, while the `notes` table enables users to record observations and updates related to specific cages or mice. The `tasks` table manages tasks and assignments, tracks their status, and enables increased efficiency for time-sensitive vivarium tasks in the range of days-to-months. For more day-to-day tasks such as cage-changes and regular maintenance, we included a `maintenance` table to enable quick recording of maintenance logs. Finally, the schema also includes the `cage_iacuc` and `cage_users` tables to establish relationships between cages, IACUC protocols, and users. Together, these tables create a robust framework for vivarium workers to manage and track details of holding cages and the individual mice within them as well as breeding cages and the breeders and litters within them.

From a user-experience perspective, once approved users are logged in to their lab's MyVivarium, holding and breeding cages are accessible in the web application via the "Dashboards" menu (**Supplemental videos 1-4** [desktop]**, 5-8** [smartphone]). Within the holding cage dashboard, new holding cages can be added using the "+" icon. The cage and mouse characteristics can be added based on the features mentioned previously under the database architecture. Individual cages can be viewed ("eye" icon), edited ("pencil/writing" icon) (**Supplemental video 9**), or deleted ("trash can" icon) (**Supplemental video 10**). Under the view menu, cage and mouse details can be viewed, along with a helpful "Sticky note" feature to note any special remarks regarding the cage or mice. Under the edit menu, individual cage or

184 mouse characteristics or features can be edited and saved. We also included separate notes
185 specific to individual mice which can be helpful when mice are earmarked for experiments or to
186 note any odd behaviors or health issues for individual mice that may need follow-up. The
187 “Maintenance logs” feature captures quick day-to-day entry logs for cage changes and other
188 routine maintenance tasks (**Supplemental videos 11-12**). This feature can be used for multiple
189 cages on the same page. An upload feature can be used to upload any files pertinent to the
190 cage or individual mice such as genotyping gel image data (**Supplemental videos 13-14**). The
191 breeding cage menu also has similar buttons within it with distinct editability for features that are
192 different from holding cages such as litter and breeder mouse details (**Supplemental videos**
193 **15-16**). For both holding and breeding mouse cages, we have included a feature that
194 conveniently generates cage cards on ready-to-print templates (**Supplemental Figure 2a,**
195 **Supplemental videos 17-18**). Each cage card template has a QR code that links it to its
196 MyVivarium record (**Supplemental video 19**). Our print feature allows cage cards to be readily
197 printed using a standard laser or deskjet printer on commercially available cardstock with 3" x 5"
198 perforations (4 per page) that can then easily fit in standard cage card holders (**Supplemental**
199 **Figure 2b**).

200 Since vivarium work is closely linked to planning and scheduling experiments, we designed a
201 “Tasks” feature to enable further integration of colony management into research projects.
202 Individual tasks can be added using the “Manage tasks” menu under settings (**Supplemental**
203 **videos 20-21**). We used an intuitive design to assign tasks to specific researchers and link
204 these tasks to particular cages, with the option of adding due dates. Email reminders related to
205 tasks can be sent at user-defined frequencies. Once the user is logged in to their lab’s
206 MyVivarium site, tasks appear on the homepage in a form similar to a Kanban board which can
207 significantly enhance efficiency and productivity in project management (Jacobs et al., 2022).

208 Recently, the relationship between ambient housing conditions of rodents used in research
209 studies and potential confounding effects on metabolism and physiology has attracted much
210 attention (Corrigan et al., 2020; Ganeshan and Chawla, 2017; Unger et al., 2023). It is important
211 to monitor ambient environmental factors since fluctuations in these factors can inadvertently
212 alter animal health and behavioral responses, potentially skewing data, which would be of
213 particular relevance to studies in neuroscience, endocrinology, physiology, and metabolism
214 research. We designed and assembled a standalone open-source IoT-based multi-sensor
215 system that is capable of monitoring temperature, humidity, illuminance, and vivarium worker
216 activity, and feed this data directly into MyVivarium. We used inexpensive but accurate Inter-
217 Integrated Circuit (I^2C) sensors to enable the monitoring of ambient factors (**Figure 3a-b**) that
218 can have an effect on colony wellbeing, metabolism, and physiology.

219 Harnessing the ability of I^2C sensors (targets) with unique addresses to communicate with a
220 controller board synchronously, we serially connected a SHT30 (temperature and humidity),
221 VL53L1X (proximity), and a VEML7700 (illuminance) sensor, each equipped with its own inbuilt
222 pull-up resistor, to a Raspberry Pi 4B. We deployed a total of three sensor module sets (two in
223 each of the vivarium rooms assigned to AS, and one in the vivarium room assigned to PP). In all
224 three deployments, we placed the assembled sensor modules in the vivarium rooms in a
225 configuration similar to that shown in **Figure 3c**. We used custom python scripts to record
226 ambient factors from data transmitted to the Pi, including temperature (ºC), relative humidity
227 (%), illuminance (lux), and vivarium worker-to-sensor proximity (cm) which is a proxy for
228 vivarium worker presence or activity. We then used docker containers in IOTstack (SensorsIOT,
229 2024) to log sensor data to an InfluxDB database and visualize sensor datastreams using
230 Grafana. We registered a low-cost domain name and used a Cloudflare tunnel to expose the

local Grafana application to be accessible through this domain. We embedded the Grafana dashboard into MyVivarium using the inline frame element (iframe).

Using our Pi-IoT system we have been able to record near-realtime ambient factor data continuously for months (**Figure 4**). In our Grafana dashboard we marked the maximum ("max") and minimum ("min") temperature (18-26ºC) and relative humidity (30-70%) as recommended by *The Guide*, an authoritative publication issued by the office of laboratory animal welfare (OLAW), which is a part of the National Institutes of Health (NIH) (National Research Council, 2011), and the Jackson Laboratory, a non-profit mouse repository ("Mouse Room Conditions | The Jackson Laboratory," 2023) for easy identification and reporting of alterations in these conditions. In general, temperature stayed within the recommended window (**Figure 4a:** data from four months, **Figure 4b**: data denoted in **Figure 4a** with arrows and grey bars from May 2nd 2024), however, on few occasions across four months, relative humidity had short transient increases above the recommended maximum of 70% (**Figure 4c**). In general, increases in humidity seemed to follow the trend of relative humidity in Dayton, OH (**Figure 4c**, grey plot), with prolonged increases in external environmental humidity preceding transient spikes in vivarium rooms 1 (**Figure 4c**, **blue**) and 2 (**Figure 4c**, **orange**). Our illuminance sensor placements were able to effectively capture lights ON and OFF in rooms 1 and 2 (**Figure 4e-f**). Our proximity sensors were also able to detect activity in both rooms as inverted spikes (**Figure 4g-h**).

Using our system, we were able to monitor changes in ambient temperature, humidity, and illuminance (**Supplemental videos 22-23**). In response to increased humidity that was observed for a specific day (**Figure 5a**), we reported this to all users of our MyVivarium as a general message using "Sticky Notes" (**Figure 5b**). While this is an easy way to notify users of any abnormal events, it requires users to actively monitor the dashboard values. Our illuminance sensor values on March 12th 2024 indicated that the lights in room 1 did not come

256 ON as usual (**Figure 4e, asterisk**) due to an erroneous manual override of the automated light
257 cycle. This error had not been identified and rectified till later that day. In order to reduce the
258 likelihood of such events that could significantly alter the mood and wellbeing of mice, we set up
259 an automated alerting system. We used the alerting system within Grafana to define thresholds
260 for illuminance sensor values. We set two separate alert rules to fire corresponding to abnormal
261 events within distinct phases of the light cycle i.e. lights ON > 15 minutes during the 6 PM – 6
262 AM window and lights OFF > 15 minutes during the 6 AM – 6 PM window. After confirming that
263 the alert was working within Grafana, we created a dedicated Google space to receive alerts via
264 an integrated webhook. The webhook application program interface (API) is then used by the
265 Grafana alerting system to send a predefined chat message with a link to the dashboard directly
266 into the Google space when the alert fires ("Build a Google Chat app as a webhook | Google
267 for Developers," 2023). We tested our alerting system by blocking light entry to the illuminance
268 sensor using black unreflective paper and were able to successfully receive alerts to the google
269 space (**Figure *5*c-d, Supplemental video 24**). We have provided all code
270 (https://github.com/myvivarium/RPi-IoT) and documentation (https://github.com/myvivarium/RPi-
271 IoT/wiki) needed to set up our RPi-IoT system. An example snapshot of our Grafana interactive
272 dashboard can be found here:
273 https://snapshots.raintank.io/dashboard/snapshot/BS9oMWCz8rpT2H3xoGVoHyDHSobyJrrW

274 For admin-related tasks, we used a tier-based approach to control management access to
275 MyVivarium. Typically, when setting up MyVivarium for the first time, a PI or lab
276 director/manager would designate themselves as an administrator (admin-tier, full access) and
277 delegate one or a few key personnel as admins (admin-tier, full access) (**Supplemental video**
278 **25**). This enables distributive administration to increase efficiency of approving authorized
279 users, determining access levels, and deleting unused or old user profiles. In addition to the
280 admin-tier which has full access privileges, we have included a standard-tier user type with

limited access to modifying and editing cage, mouse, and litter records. This enables greater control over which vivarium users have record-modifying privileges in order to reduce the likelihood of erroneous record modification. Admins can also perform strain management wherein new strains can be added to MyVivarium (**Supplemental videos 26-27**). This is advantageous for multiple reasons including standardization of mouse strain designation in cage and mouse records to prevent non-standard strain nomenclature by users, and easy linking of mouse strains to repositories such as The Jackson Laboratory to enable streamlining of mouse purchases or – in the case of custom generated lines – to enable sharing mouse lines with collaborators. We have also included and would like to emphasize the use of the *Research Resource Identifier* (RRID) for mouse strains to enable precise and standardized record-keeping ("MMRRC - Introduction," n.d.). Once strains have been added, users can easily search for these strains in the search bar when adding and filling out a cage record. Standardizing mouse strains in this manner helps researchers in both day-to-day research activities and in the sharing of results with the broader community of practice, especially for cases when a single mouse model can be represented by multiple strains in mouse repositories. In practice, since mouse strains and mouse models are often referred to by common names and not their lengthy official names or strain numbers listed in repositories, we have included common names to link particular strains to popular usage. As an example, in our mouse colony we house *Ts65Dn* mice which are an established Down syndrome mouse model (Haydar and Reeves, 2012). Ts65Dn mice are generally available as two separate strains through Jackson Laboratory – strain #005252 ("005252 - Ts(17(16))65Dn Strain Details," n.d.; "RRID | Resource Report (RRID: IMSR_JAX:005252)," n.d.) and strain #001924 ("001924 - Ts65Dn Strain Details," n.d.; "RRID | Resource Report (RRID: IMSR_JAX:001924)," n.d.). These strains have official names B6EiC3Sn.BLiA-Ts(17^{16})65Dn/DnJ and B6EiC3Sn a/A-Ts(17^{16})65Dn/J respectively. "Ts65Dn" is the common name of both these strains. Using the combination of the RRIDs, official, and common names in MyVivarium helps disambiguate these strains in colony management.

307 Finally, in order to enable integration of IACUC-related tasks and compliance such as assigning
308 mouse strains to approved IACUC protocols and providing IACUC protocol access to users, we
309 included a functionality where PIs or designated supervisors with admin access can perform
310 these tasks in a streamlined way. PIs can easily assign mouse strains and upload IACUC
311 protocols to their MyVivarium web applications (**Supplemental video 28**). We also included a
312 "Export CSV" functionality to export all database tables in CSV format for download
313 (**Supplemental video 29**), which can be helpful to monitor usage of mice and enable greater
314 compliance with the 3Rs (Graham and Prescott, 2015) as well as for data analysis regarding
315 mouse breeding efficiency and correlation to ambient environmental factors.

316 **DISCUSSION**

317 Mouse colony management is an integral part of biomedical and preclinical research (Elliott et
318 al., 2018). While technology-driven solutions have increasingly been adopted in other areas of
319 laboratory data management such as that seen with the increasing popularity of electronic
320 laboratory notebooks (Higgins et al., 2022), animal colony management has lagged behind.
321 While there may be multiple reasons why technological solutions may not be applied to certain
322 laboratory tasks, two major reasons are cost and difficulty of implementation. We designed
323 MyVivarium as a low-cost low-barrier solution to starting and maintaining a cloud-based
324 research animal colony management application which is cross-device compatible. In addition,
325 in our public repository we have provided detailed documentation for getting a MyVivarium web
326 application up-and-running in a plug-and-play manner with minimal coding requirements. Our
327 open-source RPi-IoT system can be used to stream near-realtime ambient environmental data
328 from the vivarium rooms into a MyVivarium web application. In comparison to other tools (**Table**

329 **1**), MyVivarium is a significant step forward in enabling efficient animal colony management that
330 is also open-source.

331 Our initial production deployment of MyVivarium for the mouse colony designated to AS was in
332 October 2023, with a preliminary pilot version launched a few months prior in August 2023. We
333 launched a second MyVivarium web application on the same server for the mouse colony
334 designated to PP in December 2023. Currently, the two MyVivarium web app databases have a
335 total of 82 cage records across two different labs with a combined total of 254 mice belonging to
336 12 transgenic and three background lines. Additionally, we have 15 users in total with 6 users
337 having admin access. We have had seamless access to MyVivarium without any interruptions
338 since production deployment.

339 We employed the widely used and open-source LAMP stack (Linux as web server; Apache as
340 database server; MySQL as database; PHP as programming language) ("What is a LAMP
341 Stack?," 2024) to build MyVivarium. Even though the LAMP stack traces its origins to the early
342 days of the modern web (Kunze, 1998), its flexibility and open-source nature has made it a
343 foundational set of web development tools still popular today. While it has been found that the
344 LAMP stack is slower in terms of response time in comparison to an alternative stack such as
345 MEAN (MongoDB as database program; Express.js as web application framework back end;
346 Angular as client-side application framework; Node.js as application runtime) in an "address
347 book" system, it has also been suggested that LAMP is superior to MEAN in handling relational
348 data which is found in an animal colony management scenario (Karanjit, 2016). While we have
349 used widely used web components that have been relatively early origins, concerns about
350 outdated forms of technology and digital objects are not entirely unfounded (Meur and Tarocco,
351 2019; Rothenberg, 1999) since many open-source software could potentially face issues due to
352 a lack of active development. This is evident in the case of colony management software since
353 to our knowledge, of the four significant publicly available open-source colony management

354 tools including MausDB (Maier et al., 2008), JCMS (Donnelly et al., 2010), RodentSQL (Pawar
355 and Batra, 2018), and TopoDB (Renschen et al., 2020), two are effectively unusable due to
356 unavailability of critical installation files (MausDB), and potential incompatibility and
357 obsolescence issues (JCMS) (Sundberg and Simons, 2020). Additionally, while the available
358 MausDB public repository has detailed documentation regarding installation, JCMS lacks any
359 installation documentation with no active forks in development by other users. Given these
360 realities, the fact remains that while not guaranteeing active development, releasing software as
361 open-source remains an effective strategy in combating digital obsolescence (Nano, 2023).

362 With an eye towards making MyVivarium as accessible as possible, especially for labs in under-
363 resourced contexts and communities, we prioritized cost-savings in both web components as
364 well as the hardware and software components needed for the RPi-IoT system. Our initial
365 server costs were significantly reduced due to discounts from our hosting provider ($0 for 6
366 months). Similar to our setup, if a “founder lab” sets up a server using a cloud hosting provider,
367 this single server can then used to launch multiple MyVivarium applications for multiple other
368 labs that want to use MyVivarium. This would be an additional cost-saving measure, reducing
369 the server cost further depending on the number of labs that join the server. Importantly, we
370 designed MyVivarium such that each lab would have their own secure instance without their
371 data being accessed by the lab that initially set up the server. When comparing costs of similar
372 systems to monitor IoT in vivarium-related contexts (**Table 2**), we found that our RPi-IoT system
373 was significantly cheaper than the next highest priced system – the MR1 sensor (Pallidus
374 [@PallidusSensing], 2024). It must be mentioned however, that the comparison between our
375 system and the MR1 sensor is not strictly one-to-one since the MR1 is placed within a cage and
376 hence captures ambient data within a cage and captures mouse activity instead of vivarium
377 worker activity. Although our system can readily be modified to this end (the presence sensor in
378 our system is similar in function to what is likely a PIR sensor in the MR1), this is not exactly our

379 aim at this time. In addition, the free version of the cloud interface provided by Pallidus Sensing
380 LLC only allows users 30 prior days of retrieval ("Pallidus Free Cloud," n.d.). Our dashboard
381 design using the Grafana OS version allows for unlimited retrieval of data.

382 While we consider MyVivarium to be a significant advance in comprehensive open-source
383 colony management, we would like to note some limitations of this tool. First, as is the case with
384 all web applications that are hosted on the cloud, MyVivarium web applications can be
385 potentially subject to malicious action by unapproved actors. Cloud security in a MyVivarium
386 web application that is enhanced by Cloudflare, while significantly reducing the chances of
387 attack, does not completely eliminate all likelihood of malicious entry. As best practice related to
388 any web-related application, we provide three suggestions including the prioritization of
389 database security by admins, obtaining regular backups of the database stored in a secure
390 location, and strong password management by all users. While we built MyVivarium as being
391 primarily remote-accessible, users with primary concerns over security can also choose to host
392 a MyVivarium web application running on a local server. Second, a MyVivarium cloud-hosted
393 web application is also subject to internet outages which are becoming more common according
394 to recent analysis by Cloudflare (Belson, 2024). Third, we have not included any version
395 tracking of cage records triggered by user-defined changes. Since MyVivarium is under active
396 development, we plan to incorporate this feature in a future release. Finally, the total number of
397 cage records that we have included in our specific MyVivarium applications is lower compared
398 to other tested solutions (Renschen et al., 2020) emphasizing the need for further testing and
399 performance monitoring as cage records increase. However, given the current seamless
400 performance of MyVivarium in our hands, we do not anticipate significant performance deficits
401 for other labs with hundreds of cages.

402 Taken together, MyVivarium is a low-cost, effective, and rapidly deployable cloud-based
403 solution for research mouse colony management with extensive documentation on installation

and customization. We envision MyVivarium to be of significant benefit especially to new independent investigators who would like to make their colony management more efficient without expending much time or effort by themselves or their personnel. Future development includes the incorporation of MyVivarium into a larger ecosystem of lab management which would track purchasing and budgets, chemical inventory, as well as integration into an easy-to-deploy open-source cloud-based electronic laboratory notebook currently in development. Finally, we also plan to develop and incorporate tools for big data analysis into MyVivarium to enable researchers to identify critical factors that affect colony breeding efficiency, health, and well-being.

METHODS

Web-server and database building strategy

We followed a structured approach to build the web server and database for MyVivarium, ensuring the system is robust and user-friendly. The strategy encompasses installing and configuring the web server, integrating PHP for dynamic content generation, and setting up a reliable database system. We used PHP for server-side scripting, leveraging its ease of integration with web servers and widespread use in web development. The web server utilized is Apache, known for its robustness and extensive documentation, ensuring reliable performance. Composer, a dependency manager for PHP, is employed to manage libraries such as PHPMailer for email functionality and Dotenv for environment variable management. Configuration settings, including database credentials and SMTP settings, are stored in a `.env` file, which the Dotenv library loads into the application for secure and flexible configuration. MySQL is the database management system used, and it was chosen for its reliability and performance. The database schema manages various aspects of vivarium operations, including

user management, breeding cages, holding cages, tasks, notes, and more. Tables are created with appropriate relationships and constraints to ensure data integrity.

Web server setup

We selected Apache as our web server because of its widespread usage and strong support in the industry. Apache offers a stable and secure platform that can effectively meet the needs of MyVivarium. The setup process entails installing Apache and configuring it to host the MyVivarium application. It involves setting up virtual hosts to manage multiple web applications on a single server and configuring the server to utilize PHP for processing dynamic web pages.

PHP integration

PHP is integrated into the Apache web server to enable the server to process PHP scripts. PHP is a widely used scripting language, especially suited for web development. It allows us to create dynamic content and seamlessly interact with the MySQL database. Composer, a dependency manager for PHP, handles external libraries and dependencies, ensuring the application has all the necessary tools and libraries to function correctly.

Database configuration

We chose MySQL as the database management system because of its reliability and efficiency in handling large amounts of data. The database schema stores and manages different data types needed for MyVivarium, including user information, breeding and holding cage data, tasks, and notes.

Environment configuration

The configuration settings for the database and other essential components are stored in a `.env` file. This file includes environment-specific settings, such as database credentials and

SMTP settings for email functionality. The application utilizes the Dotenv library to load these settings, providing a secure and flexible method for managing configuration data.

Security and compliance

We placed security as a high priority in the development of MyVivarium. Our system is configured to ensure secure data transmission and storage. Passwords are hashed before being stored in the database, and sensitive information is protected using state-of-the-art encryption. Our system is designed to comply with industry standards and regulations, ensuring user data is handled with the utmost care.

Deployment

To deploy the application, we set up the web server and database on a server running Ubuntu, a widespread and stable Linux distribution known for its security. The process involves installing Apache, PHP, and MySQL, configuring the server to host the MyVivarium application, and ensuring all services run smoothly. After setup, we tested the application to ensure it was fully functional and ready for use. A detailed guide on installing a LAMP (Linux, Apache, MySQL, PHP) stack on Ubuntu can help new users set up their own version of MyVivarium (Yang et al., 2024). We used Cloudflare as the DNS provider, adding an additional layer of security to our MyVivarium application. To implement this, we moved our domain from our previous registrar to Cloudflare by updating the domain's nameservers to Cloudflare's nameservers. This process involves logging into the domain registrar's account, locating the DNS settings, and replacing the existing nameservers with those provided by Cloudflare. After the domain was successfully transferred, we subscribed to Cloudflare's free plan, which includes essential security features such as DDoS protection and basic firewall rules. These measures protect the application from malicious attacks and ensure high availability. We configured Cloudflare to filter incoming traffic, block potential threats, and optimize content delivery for improved performance and security.

473 This setup reasonably ensures that a MyVivarium web application remains resilient against
474 cyber threats while maintaining seamless access for legitimate users. Note that this
475 configuration needs to be done separately and is not part of the code in the GitHub repository.
476 Finally, a complete list of all the files in MyVivarium have been included in our GitHub repository
477 (https://github.com/myvivarium/MyVivarium).

478 **Cage-card print functionality and QR-code accessibility**

479 In order to physically link cage-records to MyVivarium, we incorporated a cage-card print
480 functionality. We printed cage-cards directly from a browser and onto cardstock with 3" x 5"
481 perforations (4 per page) (Amazon.com, WA; ASIN: B09BZST43T) using a standard LaserJet
482 printer. We incorporated QR-codes into the cage-card design in order to increase efficiency in
483 managing records using mobile devices. We have tested QR-code accessibility using android
484 tablets, as well as android-based and apple-based smartphones. In all our testing and use, QR-
485 code access has been seamless.

486 **RPi-IoT design and assembly**

487 *Hardware components*

488 We used a Raspberry-Pi based design for our IoT system. All sensors used were I^2C based
489 which made the use of multiple sensors with different addresses straightforward. For each of
490 our sensor modules, we used a Raspberry Pi model 4B with 2 GB RAM (Adafruit Industries
491 LLC, NY) with a 128 GB RAM running the 64-bit Raspberry Pi Bookworm OS
492 (https://www.raspberrypi.com/software/). The Pi was cooled inside a case using both active
493 cooling and passive cooling elements (Amazon.com, WA; B0963HHSWK). We used three I^2C
494 sensors including the SHT30 Temperature and Humidity Sensor (Adafruit LLC, NY), Adafruit
495 VEML7700 Lux Sensor - I2C Light Sensor (Adafruit LLC, NY), and the Adafruit VL53L1X Time
496 of Flight Distance Sensor for 30 to 4000mm (Adafruit LLC, NY). Connectors and wires were

purchased from Adafruit. A detailed list of hardware components with vendor weblinks is provided in the bill of materials in our GitHub repository (https://github.com/myvivarium/RPi-IoT).

Data visualization and software components

Detailed steps regarding software installation steps can be found on our GitHub repository wiki (https://github.com/myvivarium/RPi-IoT/wiki). Briefly, we installed TigerVNC server (https://tigervnc.org/) on the Pi to enable remote sessions for easy management and data retrieval. We then configured the Pi to be able to detect I2C device interfacing. We also registered a free DuckDNS account to track changes in the IP address of the Pi in the event it restarted and a new IP address was assigned to the Pi. Next, we used IOTstack to run all necessary IoT-related applications on the Pi using a Docker stack (SensorsIOT, 2024). Specifically, we built our stack to include DuckDNS, Grafana, InfluxDB, and Portainer. We then created a database: `multisensors` to store sensor data. We then installed all relevant libraries for the Pi to detect streaming data from the I2C sensors. We wrote a python script (`influxdb-multisensor.py`) to log sensor data to the `multisensors` database as well as store a redundant CSV per day of recording (Sathyanesan, 2024). Intervals at which data can be logged to the InfluxDB database is user-defined, however, relevant for our purposes and also to balance sensor functionality, we used a time interval of 1 minute. After confirming that the database is actively being updated with sensor data input using Grafana, we installed a service file to ensure that the `influxdb-multisensor.py` script runs even if the Pi restarts. We then purchased a low-cost domain name using a domain name registration service (Cloudflare, San Francisco, CA; 1 year domain registration price = $6.65 USD), added this domain to Cloudflare using a free account, and used a Cloudflare tunnel ("Argo Tunnel," 2018; "cloudflare/cloudflared," 2024) to securely expose the local Grafana application running on the Pi. Once the Cloudflare tunnel was active, we were able to access our sensor data directly from the domain. We then obtained the embed URL for individual panels of our dashboard and used

522 these in MyVivarium under the Admin tasks section to directly view near-realtime sensor data.
523 Finally, in order to enable alerting via Grafana and Google chat, we used a webhook app within
524 Google space and added this as a contact point in Grafana's alerting system. We defined alert
525 rules for abnormal illuminance sensor values for both light and dark cycles of the vivarium
526 rooms. We set the evaluation period as 15 minutes with a pending period of 15 minutes, which
527 meant that if the sensor was detecting abnormal light function for at least 30 minutes, then it will
528 send a notification via the webhook app.

529 **Vivarium facility**

530 The University of Dayton vivarium is a modern animal facility specifically designed for rodent
531 (mice and rats) micro-isolator caging with each room that holds animal cages containing
532 Allentown Phantom animal transfer stations. All rooms including those that hold animal cages as
533 well as procedure rooms are independently controlled for air pressure (positive or negative),
534 temperature, and timer-controlled lighting. Lighting in rooms include both white and red
535 dimmable lights. Humidity is controlled centrally by the University of Dayton facilities
536 management.

ACKNOWLEDGEMENTS

We would like to thank the Director of the University of Dayton Vivarium, Dr. Jeffrey Kavanaugh, and our vivarium veterinarian, Dr. Paul A. Stull, DVM, for helpful suggestions during the early phase of this study.

FUNDING

We would like to acknowledge the Graduate Student Summer Fellowship awarded to MAR, A. Sheikh, HO, and AM, the Dean's Summer Fellowship awarded to JP, A. Sharp and CF, and startup funding to AS provided by the College of Arts & Sciences, University of Dayton.

DATA AND CODE AVAILABILITY

All code has been uploaded to public GitHub repositories: https://github.com/myvivarium/MyVivarium and https://github.com/myvivarium/RPi-IoT. All data will be uploaded to a public repository upon manuscript being sent out for peer-review.

REFERENCES

001924 - Ts65Dn Strain Details. n.d. https://www.jax.org/strain/001924

005252 - Ts(17(16))65Dn Strain Details. n.d. https://www.jax.org/strain/005252

Argo Tunnel: A Private Link to the Public Internet. 2018. . *The Cloudflare Blog*. https://blog.cloudflare.com/argo-tunnel

Belson D. 2024. Q2 2024 Internet disruption summary. *The Cloudflare Blog*. https://blog.cloudflare.com/q2-2024-internet-disruption-summary

Build a Google Chat app as a webhook | Google for Developers. 2023. https://web.archive.org/web/20240630230519/https://developers.google.com/workspace/chat/quickstart/webhooks

Chambliss AB, Tolan NV. 2020. Contingency Planning in the Clinical Laboratory: Lessons Learned Amidst COVID-19. *The Journal of Applied Laboratory Medicine* **5**:832–836. doi:10.1093/jalm/jfaa068

Chesler EJ, Wilson SG, Lariviere WR, Rodriguez-Zas SL, Mogil JS. 2002. Influences of laboratory environment on behavior. *Nat Neurosci* **5**:1101–1102. doi:10.1038/nn1102-1101

cloudflare/cloudflared. 2024.

Corrigan JK, Ramachandran D, He Y, Palmer CJ, Jurczak MJ, Chen R, Li B, Friedline RH, Kim JK, Ramsey JJ, Lantier L, McGuinness OP, Mouse Metabolic Phenotyping Center Energy Balance Working Group, Banks AS. 2020. A big-data approach to understanding metabolic rate and response to obesity in laboratory mice. *eLife* **9**:e53560. doi:10.7554/eLife.53560

Dolgin E. 2018. How to start a lab when funds are tight. *Nature* **559**:291–293. doi:10.1038/d41586-018-05655-3

Donnelly CJ, McFarland M, Ames A, Sundberg B, Springer D, Blauth P, Bult CJ. 2010. JAX Colony Management System (JCMS): an extensible colony and phenotype data management system. *Mamm Genome* **21**:205–215. doi:10.1007/s00335-010-9250-1

Elliott JJ, Miller CT, Hagarman JA, Kelley ST, Tardif SD, Hacker SO, Bettis A. 2018. Management of Research Animal Breeding Colonies In: Weichbrod RH, Thompson GA (Heidbrink), Norton JN, editors. Management of Animal Care and Use Programs in Research, Education, and Testing. Boca Raton (FL): CRC Press/Taylor & Francis.

Ganeshan K, Chawla A. 2017. Warming the mouse to model human diseases. *Nat Rev Endocrinol* **13**:458–465. doi:10.1038/nrendo.2017.48

Gao J, Yin Y, Myers KR, Lakhani KR, Wang D. 2021. Potentially long-lasting effects of the pandemic on scientists. *Nat Commun* **12**:6188. doi:10.1038/s41467-021-26428-z

Graham ML, Prescott MJ. 2015. The multifactorial role of the 3Rs in shifting the harm-benefit analysis in animal models of disease. *European Journal of Pharmacology*, Translational value of animal models **759**:19–29. doi:10.1016/j.ejphar.2015.03.040

Hall W, Tiropanis T. 2012. Web evolution and Web Science. *Computer Networks*, The WEB we live in **56**:3859–3865. doi:10.1016/j.comnet.2012.10.004

Haydar TF, Reeves RH. 2012. Trisomy 21 and early brain development. *Trends Neurosci* **35**:81–91. doi:10.1016/j.tins.2011.11.001

Higgins SG, Nogiwa-Valdez AA, Stevens MM. 2022. Considerations for implementing electronic laboratory notebooks in an academic research environment. *Nat Protoc* **17**:179–189. doi:10.1038/s41596-021-00645-8

Jacobs M, Buck A, Petersen T, Kaim E. 2022. What is Kanban? - Azure DevOps. https://web.archive.org/web/20240305014636/https://learn.microsoft.com/en-us/devops/plan/what-is-kanban

Jaric I, Voelkl B, Clerc M, Schmid MW, Novak J, Rosso M, Rufener R, Kortzfleisch VT von, Richter SH, Buettner M, Bleich A, Amrein I, Wolfer DP, Touma C, Sunagawa S, Würbel H. 2022. The rearing environment persistently modulates mouse phenotypes from the molecular to the behavioural level. *PLOS Biology* **20**:e3001837. doi:10.1371/journal.pbio.3001837

Karanjit A. 2016. MEAN vs. LAMP Stack. *Culminating Projects in Computer Science and Information Technology*.

Kudo E, Song E, Yockey LJ, Rakib T, Wong PW, Homer RJ, Iwasaki A. 2019. Low ambient humidity impairs barrier function and innate resistance against influenza infection. *Proceedings of the National Academy of Sciences* **116**:10905–10910. doi:10.1073/pnas.1902840116

Kunze M. 1998. LAMP: Freeware Web Publishing System with Database Support (Translated from the Original German). *Computertechnik*.

Maier H, Lengger C, Simic B, Fuchs H, Gailus-Durner V, Hrabé de Angelis M. 2008. MausDB: An open source application for phenotype data and mouse colony management in large-scale mouse phenotyping projects. *BMC Bioinformatics* **9**:169. doi:10.1186/1471-2105-9-169

Maier H, Schütt C, Steinkamp R, Hurt A, Schneltzer E, Gormanns P, Lengger C, Griffiths M, Melvin D, Agrawal N, Alcantara R, Evans A, Gannon D, Holroyd S, Kipp C, Raj NP, Richardson D, LeBlanc S, Vasseur L, Masuya H, Kobayashi K, Suzuki T, Tanaka N, Wakana S, Walling A, Clary D, Gallegos J, Fuchs H, de Angelis MH, Gailus-Durner V. 2015. Principles and application of LIMS in mouse clinics. *Mamm Genome* **26**:467–481. doi:10.1007/s00335-015-9586-7

Meur J-YL, Tarocco N. 2019. The obsolescence of Information and Information Systems CERN Digital Memory project. *EPJ Web Conf* **214**:09003. doi:10.1051/epjconf/201921409003

Mishra J. 2020. Use your lab website to make a compelling first impression. *Nature*. doi:10.1038/d41586-020-01298-5

MMRRC - Introduction. n.d. https://www.mmrrc.org/about/generalInfo.php

Mouse Room Conditions | The Jackson Laboratory. 2023. https://web.archive.org/web/20231206081828/https://www.jax.org/jax-mice-and-services/customer-support/technical-support/breeding-and-husbandry-support/mouse-room-conditions

Nano E. 2023. Digital obsolescence In: CEUR-WS, editor. Doctoral Symposium of ICT4S 2023, The International Conference on Information and Communications Technology for Sustainability. Rennes, France. pp. 36–47.

National Research Council. 2011. Guide for the Care and Use of Laboratory Animals: Eighth Edition. Washington, DC: The National Academies Press. doi:10.17226/12910

Pallidus Free Cloud. n.d. . *Pallidus*. https://pallidus.io/products/pallidus-free-cloud

Pallidus [@PallidusSensing]. 2024. Want to monitor the environment of your housing room? We have an affordable solution for monitoring temperature, humidity, light, noise levels, and human activity! Get in touch to learn more or see a live demo! https://t.co/XpxL0m2dqh. *Twitter*.

Pawar S, Batra H. 2018. RodentSQL: a software suite for colony management of animal protocols. *Future Science OA* **4**:FSO330. doi:10.4155/fsoa-2018-0054

Per Diem Animal Housing Rates | Boston University Office of Research. 2016. https://web.archive.org/web/20231206134704/https://www.bu.edu/research/ethics-compliance/animal-subjects/housing/per-diem-housing-rates/

Renschen A, Matsunaga A, Oksenberg JR, Santaniello A, Didonna A. 2020. TopoDB: a novel multifunctional management system for laboratory animal colonies. *Database* **2020**:baaa098. doi:10.1093/database/baaa098

Richter SH, Garner JP, Zipser B, Lewejohann L, Sachser N, Touma C, Schindler B, Chourbaji S, Brandwein C, Gass P, Stipdonk N van, Harst J van der, Spruijt B, Võikar V, Wolfer DP, Würbel H. 2011. Effect of Population Heterogenization on the Reproducibility of Mouse Behavior: A Multi-Laboratory Study. *PLOS ONE* **6**:e16461. doi:10.1371/journal.pone.0016461

Rothenberg J. 1999. Avoiding Technological Quicksand: Finding a Viable Technical Foundation for Digital Preservation • CLIR. *CLIR*. https://www.clir.org/pubs/reports/rothenberg/

RRID | Resource Report (RRID: IMSR_JAX:001924). n.d. https://rrid.site/data/record/nlx_154697-1/IMSR_JAX:001924/resolver?q=001924&l=001924&i=rrid::imsr_jax:001924

RRID | Resource Report (RRID: IMSR_JAX:005252). n.d. https://rrid.site/data/record/nlx_154697-1/IMSR_JAX:005252/resolver?q=IMSR_JAX%3A005252&l=IMSR_JAX%3A005252&i=rrid::imsr_jax:005252

Sathyanesan A. 2024. myvivarium/RPi-IoT: Open-source IoT for MyVivarium web apps. https://github.com/myvivarium/RPi-IoT

SensorsIOT. 2024. SensorsIot/IOTstack.

Sundberg B, Simons A. 2020. jaxcs/JCMS-root: The Jackson Laboratory Colony Management System. https://github.com/jaxcs/JCMS-root

Unger CA, Hope MC, Aladhami AK, Velázquez KT, Enos RT. 2023. How stable is your vivarium's temperature? Fluctuations in vivarium temperature significantly impact metabolism and behavior impeding scientific reproducibility. *Physiology & Behavior* **258**:114029. doi:10.1016/j.physbeh.2022.114029

What is a LAMP Stack? - LAMP Stack Explained - AWS. 2024. https://web.archive.org/web/20240717231209/https://aws.amazon.com/what-is/lamp-stack/

Wood D, King M, Landis D, Courtney W, Wang R, Kelly R, Turner JA, Calhoun VD. 2014. Harnessing modern web application technology to create intuitive and efficient data visualization and sharing tools. *Front Neuroinform* **8**. doi:10.3389/fninf.2014.00071

Yang K, Heidi E, Abid E. 2024. How To Install Linux, Apache, MySQL, PHP (LAMP) Stack on Ubuntu | DigitalOcean. https://web.archive.org/web/20240704012542/https://www.digitalocean.com/community/tutorials/how-to-install-lamp-stack-on-ubuntu

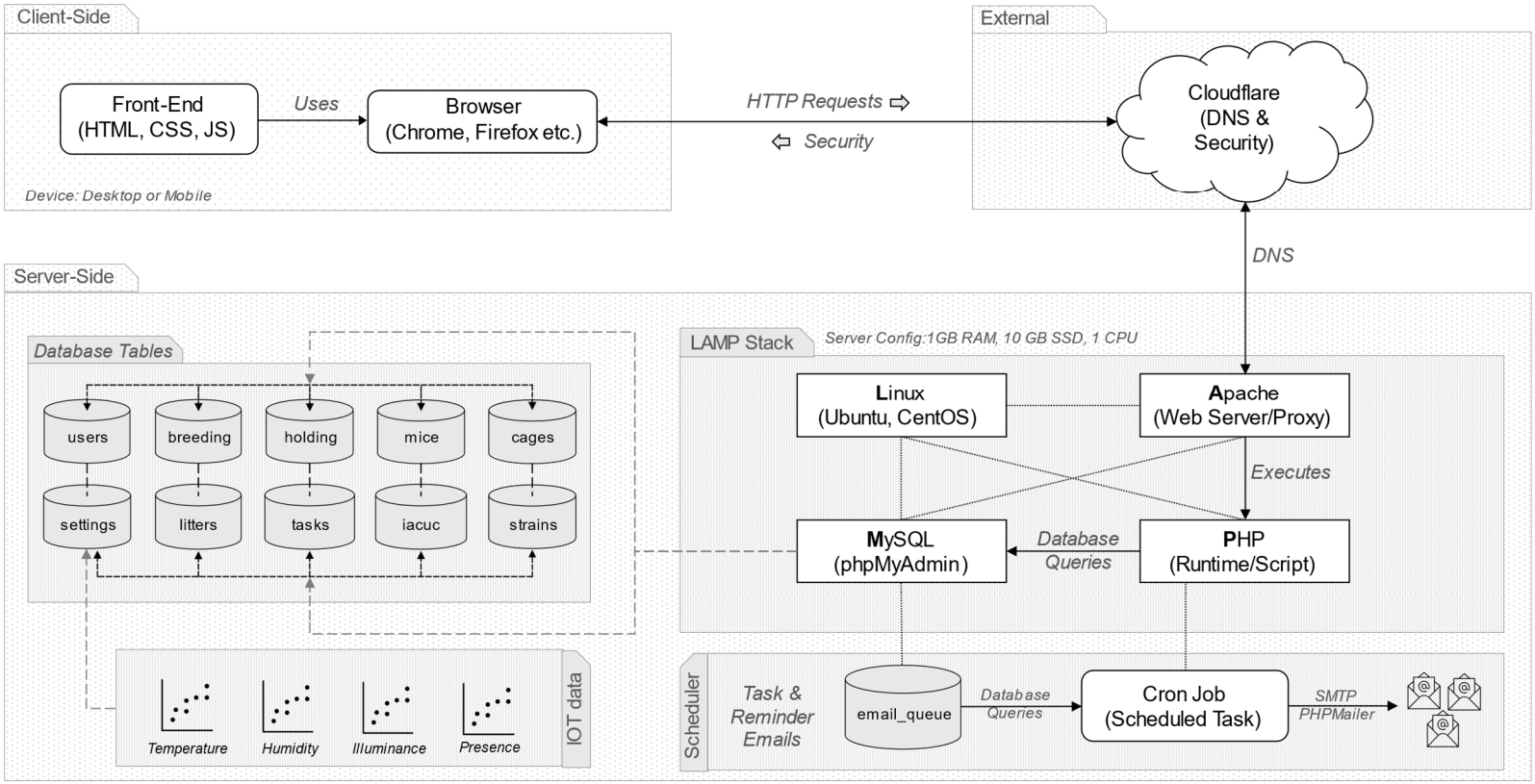


Figure 1. MyVivarium System Architecture Overview. MyVivarium system architecture consists of three main components: the client side, the server side, and external services. Each component interacts with the others to provide a robust and secure web application for managing vivarium operations. *Client-side:* MyVivarium's front-end is built using standard web technologies, including HTML, CSS, and JavaScript, ensuring that the user interface is interactive and responsive. Users can interact with the application through web browsers such as Chrome and Firefox with cross-device compatibility. *External services:* Cloudflare provides DNS services and security features, including DDoS protection, to enhance the reliability and security of the MyVivarium application. *Server-Side*: The backend is built on a LAMP stack, consisting of Linux (operating system), Apache (webserver), MySQL (database), and PHP (scripting language). The database includes tables such as `users`, `breeding`, `holding`, `mouse`, `files`, `settings`, `litter`, `tasks`, `notes`, and `strains`, which store various types of data required for the application's functionality. *IoT Data*: The system integrates IoT sensor data to monitor ambient parameters such as temperature, humidity, illuminance, and presence. *Scheduler:* A scheduler (cron job) handles tasks such as sending reminder emails utilizing the `email_queue` for processing. Cron jobs automate database queries and sending emails through SMTP using PHPMailer.

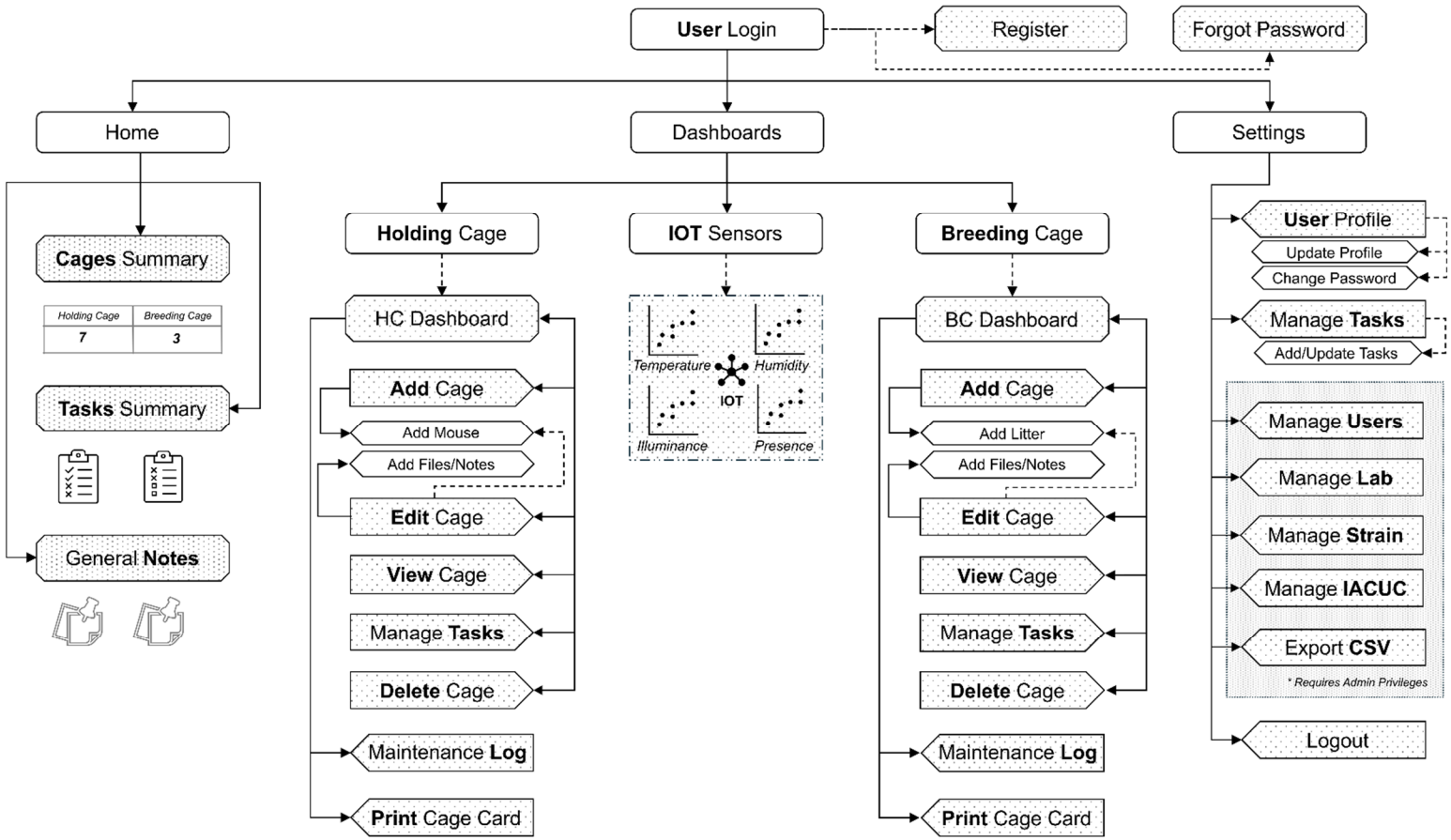
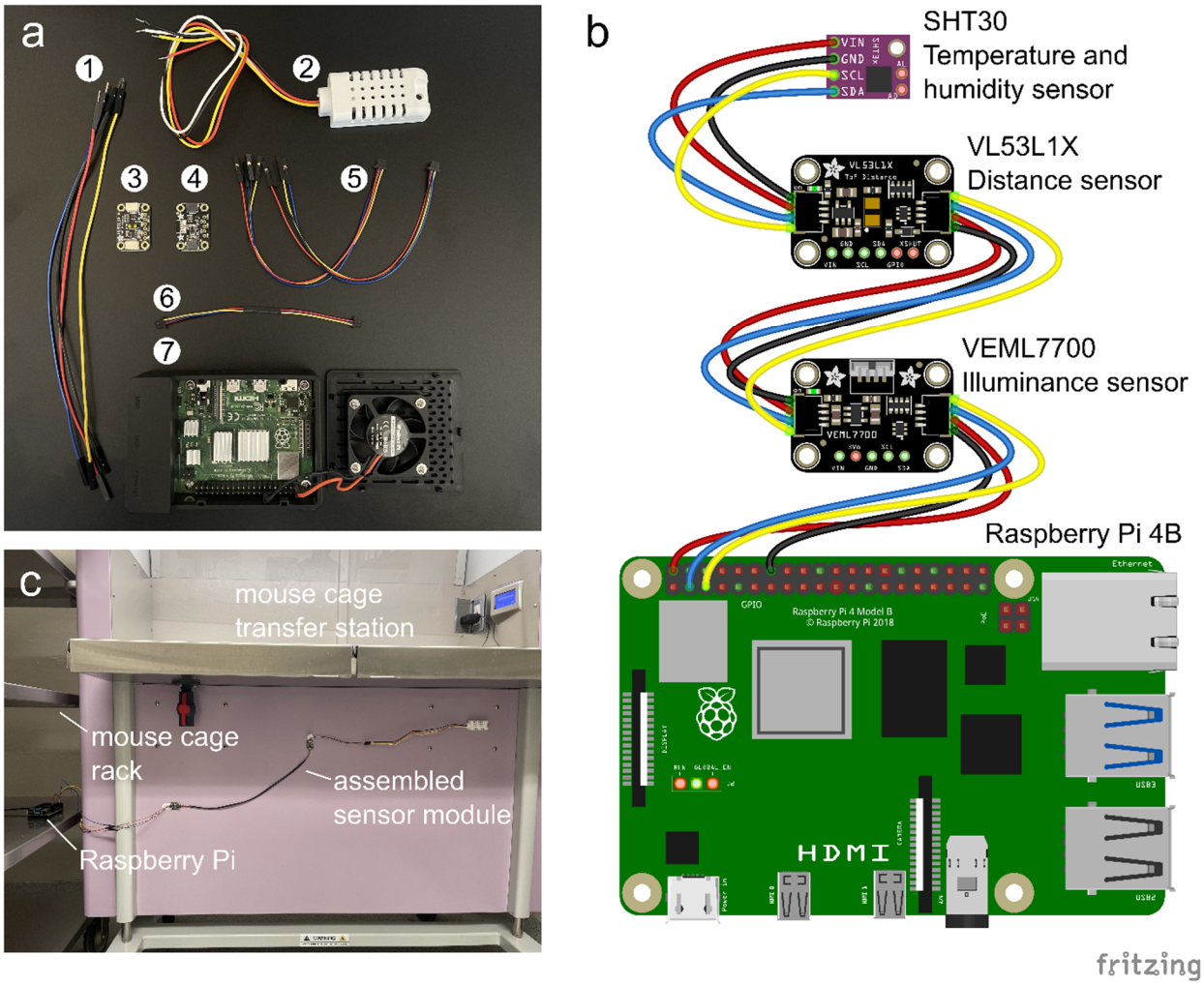


Figure 2. MyVivarium web application navigation overview. The MyVivarium web application navigation is intuitive, structured into “Home”, “Dashboards”, and “Settings”. The “Home” section includes “Cages Summary”, “Tasks Summary”, and “General Notes”. The “User Login” provides links for registration and password recovery. “Dashboards” gives access to holding cage and breeding cage management, including adding, editing, viewing, and deleting cages, printing cage cards, maintenance logs, and near-realtime ambient sensing via the IoT sensor datastream. “Settings” includes user profile management, task management, and admin options such as managing users, lab details, mouse strains, IACUC protocols, and data export in CSV format. The Logout option allows users to log out securely. This structure ensures easy access to all functionalities, with specific options available based on user roles and permissions.



701 **Figure 3. Design, assembly, and deployment of MyVivarium Pi-IoT sensor system**. **a.**
702 Components of Pi-IoT sensor system include 1: Jumper wires (to extend reach) 2: SHT30
703 Temperature and humidity sensor with plastic shell casing 3: VL53L1X distance or time-of-flight
704 sensor which can measure distance to a person (vivarium-worker) 4: VEML7700 illuminance
705 sensor which can detect lights ON and OFF 5 and 6: STEMMA-QT connectors (for Raspberry Pi
706 GPIO pins to sensors and between sensors) 7: Raspberry Pi 4B (2 GB) in a case with a cooling
707 fan. **b.** Fritzing sketch of serial connections between I^2C sensors and Raspberry Pi GPIO pins
708 based on a validated sensor module build. Wire color designations are as follows – red: power
709 (+ve), black: ground (-ve), yellow: SCL (serial clock), blue: SDA (serial data). Note that in a, the
710 color of SDA (data) is white. **c.** Assembled sensor module deployed in our vivarium room.
711 Sensors have been affixed to the lower part of the mouse cage transfer station. The Raspberry
712 Pi has been placed in a corner of a mouse cage rack.

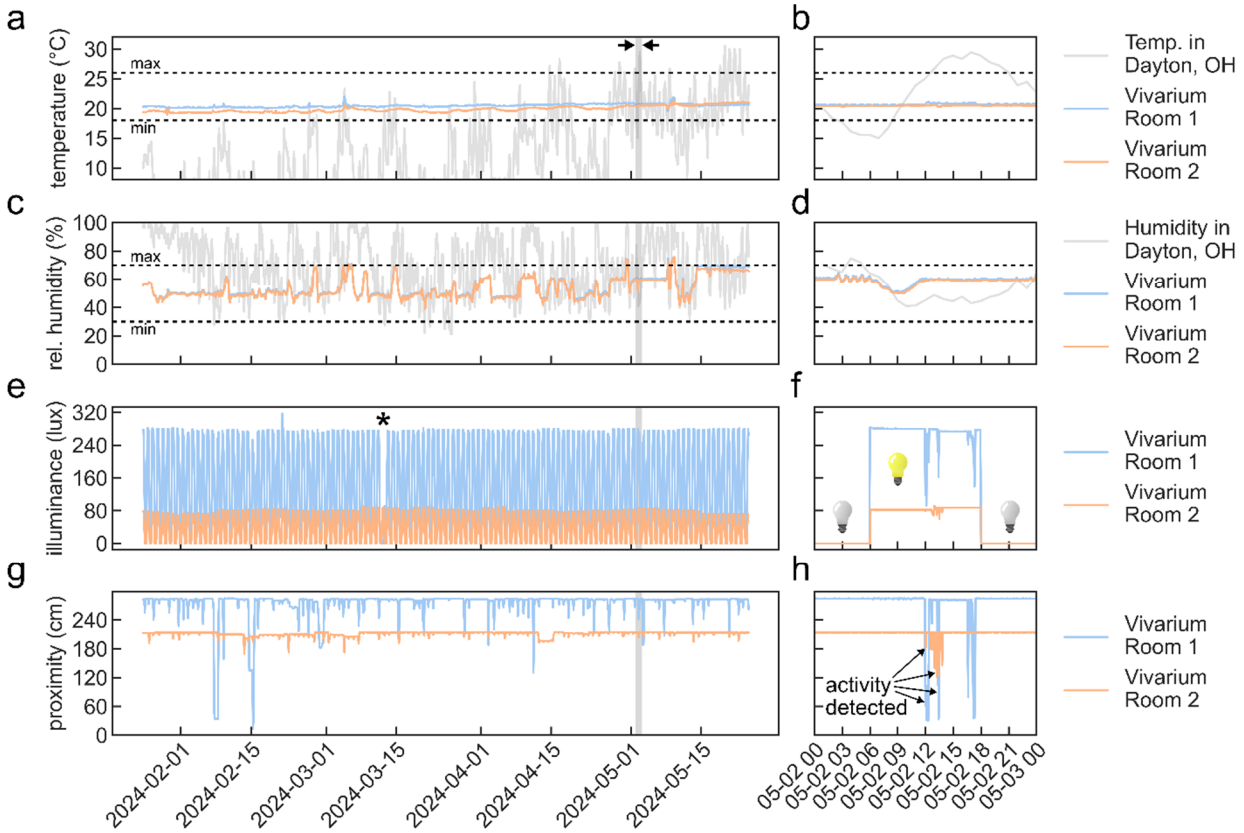


Figure 4. Continuous near-realtime ambient factor recording over a period of months using MyVivarium Pi-IoT sensor system. a. Temperature (ºC) recorded over four months in vivarium rooms 1 (blue) and 2 (orange). Temperature in Dayton, OH is plotted as reference (grey). Dotted lines represent minimum ("min" = 18 ºC) and maximum ("max" = 26 ºC) recommended temperatures. Time-window (grey shaded rectangle) between arrows (12 midnight May 2nd, 2024 – 12 midnight May 3rd, 2024) is shown in greater detail in **b. c.** Relative humidity (%) recorded over four months. Humidity in Dayton, OH is plotted as reference (grey). Dotted lines represent minimum ("min" = 30%) and maximum ("max" = 70%). Time-window (grey shaded rectangle) is shown in greater detail in **d**. **e.** Illuminance sensor values recorded over months. Asterisk indicates an abnormal event detected by the sensor due to an erroneous manual override of the automated light cycle **f.** Sensor value change over the course of 24 hours shows vivarium lights OFF (light bulb turned off) and lights ON (light bulb turned on) in rooms 1 and 2 set to a 6 AM – 6 PM light cycle. **g.** Proximity sensor values for rooms 1 and 2 recorded for months **h.** Vivarium worker presence or activity is detected as negative deflections. Minimum and maximum temperature and humidity thresholds are based on suggested recommendations by *The Guide* (Office of Laboratory Animal Welfare, National Institutes of Health) and the Jackson Laboratory ("Mouse Room Conditions | The Jackson Laboratory," 2023; National Research Council, 2011).

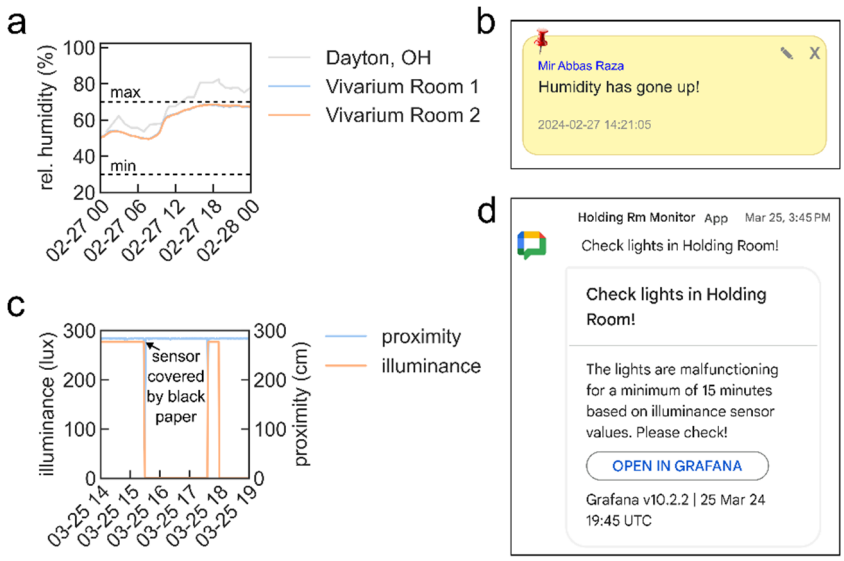


Figure 5. User notification of sensor readings and automated alerts. a. Increased humidity in vivarium rooms 1 (blue) and 2 (orange) on February 27th, 2024. **b.** General user notification using the "Sticky notes" on MyVivarium. **c.** Abnormal illuminance event and automated alert testing (arrow indicating when black unreflective paper was used to block the illuminance sensor from light exposure) shows reduced illuminance (orange) at 3:30 PM on March 25th **d.** Alert message triggered by Grafana and relayed via a Google chat webhook app to a dedicated Google space.

738 **Table 1. Comparison of MyVivarium to other available open-source and commercial colony management tools**

Name	Open-source or commercial/proprietary	Cost	Admin management	File upload	Tasks and Reminders	Smart-device compatible	Tool-linked cage card functionality	IoT ambient sensing and alerts
MyVivarium	Open-source https://github.com/myvivarium/MyVivarium	Low	Yes	Yes	Yes	Yes	Yes; QR-code	Yes
TopoDB	Open-source (Renschen et al., 2020) https://github.com/UCSF-MS-DCC/TopoDB	Low	Yes	Yes	No	Not mentioned, unavailable, or not tested	Not mentioned or unavailable	No
RodentSQL	Open-source (Pawar and Batra, 2018) https://github.com/spawar2/Java_SQL_MiceExperiment_Record_Tracking_Software	Low	Yes	No	Unclear	Not mentioned, unavailable, or not tested	Yes; barcode	No
MausDB	Open-source (Maier et al., 2008) https://github.com/MouseGenomics/mausdb	Low	No	Yes	Yes	No	Yes; barcode	No
mLIMS	Commercial/proprietary https://bioinforx.com/web/product_mlims.php	Very high	Yes	Yes	Yes	Yes	Yes; QR code	No
SoftMouse (2 users)	Commercial/proprietary https://www.softmouse.net/login.jsp	High	Yes	Yes	Yes	Yes	Yes; QR code	No
PyRAT	Commercial/proprietary https://www.scionics.com/pyrat.html	High	Yes	Yes	Yes	Yes	Yes; QR code	No

739

740 **Table 2. Comparison of MyVivarium RPi-IoT ambient sensing tool compared to other**
741 **tools**

Name	Open-source or commercial/proprietary	Cost of components (in USD)	
		Software-related	Hardware-related
MyVivarium RPi-IoT system	Open-source https://github.com/myvivarium/RPi-IoT	Grafana Open source: $0; Domain name registration: varies, but as low as $6 per year	$110.08
Pallidus MR1 sensor and Pallidus cloud software (Pallidus Sensing LLC)	Commercial/ proprietary https://pallidus.io/products/mr1-sensor	Pallidus Free: $0 (only access to last 30 days of data); Pallidus professional cloud: $2,990.00	$399.00 per sensor, however can only be added to an existing installation; Pallidus starter kit (8 sensors): $4,499.00 (includes only one year subscription to cloud)
REM Module and Guardian software (excludes software maintenance, training, and installation) (Techniplast, USA)	Commercial/ proprietary https://digitalcage-tecniplast.com/en/products/rem.html	> $800.00 license fee	> $2000.00