STAPLE Documentation

Erin M. Buchanan

September 17, 2025

Table of contents

1 Preface **4**

2 Introduction **6**

3 Core Concepts **9**

3.1 STAPLE Elements 9
3.2 Metadata 9
3.3 Definitions of Terms 10
3.3.1 User 10
3.3.2 Main and Project Level Options 11
3.3.3 Project Level Options 14

4 Installation Guide **16**

4.1 Assumptions 16
4.2 Local Development or Hosting 16
4.3 Installation Steps 17
4.3.1 Web Server Installation 17
4.3.2 Blitz/JS Installation 17
4.3.3 Clone the Repository 18
4.3.4 Database Installation 19
4.3.5 Connect Database to STAPLE App 20
4.3.6 Configure Email 22
4.3.7 Install STAPLE Requirements 23
4.3.8 Starting the App - Local Testing 24
4.3.9 Starting the App - Production 24
4.3.10 Keeping the App Going 25
4.3.11 Setting Up the Proxy (Nginx) 26
4.3.12 Enable HTTPS 27
4.4 Common Issues & Fixes 28
4.4.1 Prisma / Database Errors 28
4.4.2 Build Errors 28
4.4.3 Email Issues 28
4.4.4 Nginx / Proxy Issues 29
4.4.5 Service Issues 29
4.4.6 General Tips 29

5 Glossary **30**

6 References **31**

1 Preface

Scientific research has become increasingly complex, requiring specialized skills, interdisciplinary work, and collaboration among large teams. Managing such projects and tracking data and metadata has become a significant challenge. The need for FAIR (findable, accessible, interoperable, and reusable Wilkinson et al., 2016) open data, materials, and metadata has similarly grown, especially considering recent mandates for required sharing for federally funded projects (Rep. Foxx, 2019; Rep. Ryan, 2019). While user-friendly software tools that help researchers structure metadata exist, few researchers are aware of their necessity and usefulness, the onus is still on the project team to collect and curate this information. Studies using teams of researchers require sophisticated tracking of all project elements, and therefore, there is a need for scientific project management software that is tailored to the management of all manner of science projects. Software that promotes best practices in FAIR data and metadata would increase openness in all reporting, and tracking, and reporting standards.

Currently, the tools and websites designed for researchers are focused on *getting* researchers to share their materials, code, and data (i.e., Open Science Framework, FigShare, Zenodo. These repositories represent a necessary resource for long-term storage of outputs, but do not help researchers organize or track information during the life of a study. Project management software, like Asana, Monday, or ClickUp, are designed from a business perspective that is not tailored to scientific research. While the features that scientific research requires may be found in some individual project management software, no current solution provides all the essential features, such as the ability to assign tasks at different scales (e.g., teams, individuals), integrated metadata, fully transparent access to all information in the project manager, and long-term storage that complies with international data privacy regulations. Project management software is designed to *get things done* rather than *document* the way a project was completed, so attempts to use existing software for these features often involve 'hacking' it to extract necessary records.

To this end, we developed STAPLE (**S**cience **T**racking **A**cross the **P**roject **L**if**E**span), which not only helps with the unique challenges of project management of research but includes open and transparent documentation of data and metadata, as well as providing templates for minimum metadata collection. STAPLE allows users to add project components based on research type, assign timelines, delegate tasks to individuals or groups, and link to long-term storage of research outputs. STAPLE helps track authorship credit and contributor roles via tasks or contributors. STAPLE includes default metadata standards for documenting common research outputs and is fully open-source, enabling community input and adaptation. Designed for global reach, STAPLE supports the transition to open science by including researchers and outputs that are often overlooked, regardless of technical expertise. Through its point-and-click interface, and downloadable documentation outputs, STAPLE makes it easier for all researchers to participate in transparent, inclusive, and reproducible science.

2 Introduction

The STAPLE mission: *Project Management* software that allows you to *document* your research project to improve *transparency.*

As researchers, we must balance attention to the big picture: the overarching goals, timelines, theories, and hypotheses of a project, with the smaller goals and tasks required to realize those ambitions. While most researchers receive training in designing studies, collecting and analyzing data, and publishing findings, formal instruction in project management is rare. Instead, these skills are often acquired informally, modeled within the research lab environment and shaped by a supervisor's style. Each project also introduces unique needs and circumstances, and even the addition of a single collaborator can generate new layers of coordination and complexity.

Evidence indicates that the number of co-authors and collaborators on research projects has increased steadily over time (Wuchty et al., 2007). Advances in digital communication and collaboration tools such as Zoom, GitHub, and Google Workspace, combined with the globalization of research, have enabled teams to work together across great distances and disciplinary boundaries (Cummings & Kiesler, 2005, 2007). Large-scale collaborations are increasingly encouraged, not only to tackle ambitious scientific questions, but also to enhance the diversity of participants, researchers, and cultures represented in data (Adan, 2023; Moshontz et al., 2018; Swartz et al., 2019). The inclusion of multiple collaborators, each embedded in different lab cultures, institutions, geopolitical contexts, and languages, increases the complexity of managing research projects.

So why do we need software?

General project management platforms, such as Asana, Monday, ClickUp, and Trello, are widely used in business settings, but they were designed around business-oriented workflows and templates. Research projects, however, present unique challenges that do not map neatly onto these models. Scientific projects often require coordination of specialized tasks such as research design, ethical approval, data collection across multiple sites, metadata documentation, open data sharing, and publication requirements (Borghi & Gulick, 2021; Fecher & Friesike, 2014).

Large-scale collaborations have experimented with adapting existing structures, sometimes treating multi-team projects like classrooms or leveraging educational technologies, but these solutions are partial fits at best. What researchers need is a system that recognizes the scientific lifecycle: from ideas and preregistration, to data collection, to analysis, dissemination, and long-term archiving. STAPLE is designed for scientists, by scientists. It was built with input not only from researchers but also from key stakeholders in the broader research ecosystem, including funders, librarians, and journals, ensuring that the software supports transparency, reproducibility, and effective project management across diverse scientific contexts.

So what's the deal with documentation?

In tandem with the encouragement for interdisciplinary, diverse teams, researchers have begun to focus on the improvement of science through transparency and sharing. The Transparency and Openness guidelines establish ideals for projects to be reproducible, open, and transparency through data sharing, code/analysis sharing, materials sharing, pre-registration, and replication (Grant et al., 2025). Journals require different levels of sharing for publication, which has pushed researchers to begin to implement these practices. However, the simple sharing of data, code, and materials does not make them FAIR: Findable, Accessible, Interoperable, and Reusable (Wilkinson et al., 2016). Simple sharing of data does not necessarily allow someone to find that data in relation to the research project, or allow a researcher to know what `V1` means in the data, or even reuse part of the data.

Creating adequate documentation for the reuse of any output from a project can be difficult. There are often no clear standards, the work can be time consuming, and is often left to the end of a project when important details may have been forgotten. STAPLE is designed to encourage you to document information along the way and to provide guidance with what you should include with each document. As you complete specific task, such as create your materials for a study, you will be able to link your project to those materials and include information to interpret and understand those documents. We provide a set of minimum documentation standards for multiple types of documents - and the ability for you to include extra information if your field has set standards.

Ok, so that's transparent?

STAPLE allows you to track and run your project - and then include research outputs from that project. Each step of the way, STAPLE collects data about who did what and when. If you are the only person working on a project, it's simple: you will export a final project timeline that includes information about each step of the project, each output, and the documentation needed to make those outputs useful. If you work with other people, STAPLE will allow you to assign tasks to those individuals (or teams of individuals) to complete as part of the project management functionality. At the same time, STAPLE tracks their work and creates a project timeline that shows that they completed a task for the project.

This component of the software was designed to align with contributorship models of research - instead of "authorship", each person receives credit for the components they contributed to the project (Allen et al., 2014, 2019). Each field has different standards for authorship, and many individual's work may not be acknowledged due to those cultures. Using STAPLE, their work would be credited, even if they did not earn final "authorship" on a publication. This output is also useful for defining contributions into systems such as CRediT - where each person's role is binned into categories. By linking each role to specific tasks, we can transparently show what each role means to a specific project.

3 Core Concepts

In this section, we introduce the core terminology for STAPLE elements to help users understand their purpose and function. The following section will demonstrate how these elements appear in practice, with screenshots and step-by-step guides for using them within STAPLE.

3.1 STAPLE Elements

Research projects are made up of many moving parts: overarching goals, specific hypotheses, tasks to carry out, and the people who contribute along the way. To manage this complexity, STAPLE organizes projects into a small set of core elements. These elements provide a shared structure that scientists and collaborators can understand quickly, regardless of discipline.

At the highest level, a Project represents the scientific effort being undertaken: its aims, scope, and timeline. Within each project, researchers define Tasks, which capture the individual units of work needed to achieve those aims. Teams and Contributor/Members provide the human side of the system, reflecting roles, privileges, and collaborations across labs and institutions. Every action taken on a task, whether assigning it, updating its status, or marking it as complete, is recorded in Task Logs, creating a transparent history of progress. Finally, Tags allow projects to be categorized, searched, and documented in ways that make sense for science.

Together, these elements form the backbone of STAPLE, giving research teams a clear, consistent framework for planning and documenting their work.

3.2 Metadata

In scientific research, data alone is never enough, what makes data meaningful is the metadata that describes it. Metadata captures the critical context behind research activities: the methods used, the instruments applied, the variables measured, and the conditions under which data were collected. Without metadata, results can be difficult to reproduce, interpret, or build upon.

STAPLE treats metadata as a first-class feature, not an afterthought. The software also automatically records who did what, and when, embedding provenance metadata directly within projects. Project information, such as data, materials, and analysis code, can be linked within STAPLE and enriched with metadata. These details are captured using the Forms feature, which provides structured templates tailored to research needs. Completion of metadata can be assigned to contributors via Tasks, making documentation part of the normal project workflow rather than an extra burden.

By embedding metadata capture directly into everyday research activities, STAPLE makes it easier for scientists to document their work transparently, meet the expectations of journals and funders, and create records that future collaborators, or even their own future selves, can interpret and reuse. All metadata can be exported as machine-readable JSON for computational use and as human-readable interactive Project Summary reports for sharing and communication.

3.3 Definitions of Terms

3.3.1 User

A user in STAPLE is anyone with an account who has accepted the platform's terms and conditions. Users can create projects and related elements within their own accounts, and they can also participate in projects owned by others. Users can log in and out, update their personal information (e.g., username, password, email, and profile details), and delete their account. They may add and manage contributors, organize teams within their projects, and assign tasks.

Note: Deleting your STAPLE account *does not* remove your past contributions to other projects. You will lose editing access, but only a project administrator can fully remove you from their project.

3.3.1.1 Project Administrator

- The user who creates a project is automatically assigned as its project administrator.
- Additional administrators can be added to a project as needed.
- Every project must have at least one administrator at all times.
- Administrators have the highest level of control, including managing contributors, editing project settings, and assigning roles.

3.3.1.2 Contributor

- A contributor is a user added to a project by an administrator.
- Contributors generally have limited access: they can view project information and complete tasks assigned to them but cannot change project settings or manage other users.
- This role allows project teams to include collaborators without granting full administrative privileges.

3.3.1.3 Teams

- Teams are a special type of contributor within a project.
- A user can belong to multiple teams in the same project.
- Teams are useful when a task requires collaboration by a group rather than an individual.

For example, a data collection team working with human participants may need to complete an ethics application. Only one person needs to submit the form within STAPLE, but the entire team receives credit for completing the task. Similarly, a translation team might be responsible for converting study materials from French into Spanish. The collective goal is to provide the final Spanish materials, but all members of the team are credited for the work.

3.3.2 Main and Project Level Options

3.3.2.1 Dashboard

The Dashboard is your personalized home page in STAPLE. It provides a high-level overview of all the projects you're involved in, along with recent updates and tasks assigned to you. From here, you can quickly jump into active projects, track progress at a glance, and see what needs your attention today. Think of it as your command center for research project management.

In addition to this main dashboard, each project also has its own Project Dashboard. While the main dashboard gives you a cross-project snapshot, the project dashboard focuses on the details of a single project, its contributors, tasks, metadata, milestones, and outputs. Together, the two levels of dashboards help you move easily between the big picture and the project-specific details.

3.3.2.2 Projects

Projects are the central containers for scientific work. Each project includes its own tasks, teams, metadata, and outputs. On the Projects page, you can:

- Create a new project.
- View and manage existing projects.
- Go into the project level dashboard.

Projects provide the big-picture structure that keeps research activities organized and transparent.

3.3.2.3 Invitations

Invitations are how collaborators join projects in STAPLE. Project administrators send invitations that specify the role a user will have once they accept.

- On the main dashboard, you can view all of your pending invitations across projects and choose to accept or decline them.
- On a project dashboard, only project administrators can see the invitations they have sent for that specific project.

A contributor cannot be assigned tasks or added to teams until they have accepted their invitation. This system ensures that every collaborator's involvement is explicit, documented, and consented to before work begins.

3.3.2.4 Tasks

Tasks break a project down into actionable steps and are the backbone of daily research activity in STAPLE.

- On the main dashboard, the Tasks page shows a consolidated view of all tasks assigned to you across every project, so nothing slips through the cracks.
- On a project dashboard, project administrators can view and manage all tasks within that project, while contributors see only the tasks assigned to them.

Within a project, you can:

- Create new tasks for yourself or others.
- Assign tasks to contributors or teams.
- Update task status (to-do, in progress, complete).
- Add labels, metadata, and logs to track progress.

This two-level view ensures that contributors can stay focused on their own responsibilities while project managers maintain oversight of the entire project's workflow.

3.3.2.5 Notifications

Notifications keep you informed about project activity without needing to check each page manually. These alerts let you know when:

- You've been assigned a new task.
- A task you're involved in has been updated, completed, or approved.
- You've received a project invitation.

On the main dashboard, notifications show updates from all of your projects in one place. On a project dashboard, notifications are filtered to that project only, helping you focus on what matters most in the current context. The Notifications page ensures you never miss important updates in collaborative projects.

3.3.2.6 Forms

Forms are structured templates for collecting metadata at the project or task level. They make sure essential details, such as ethics approvals, instrument calibration, dataset versions, or preregistration links, are captured consistently.

STAPLE includes a library of form templates, giving you a starting point if you're not sure what metadata to collect. You can also design your own forms tailored to your project's needs. Forms must be created before they can be used in a project, ensuring that the structure is in place before contributors start filling them out.

Once in use, forms become a powerful way to collect metadata directly within your workflow. You can:

- Fill out forms as part of project tasks.
- Assign forms to contributors or teams to ensure documentation responsibilities are shared.
- Reuse templates across multiple projects for consistency.

By embedding metadata capture directly into the research process, forms help projects meet transparency and reproducibility standards while reducing the burden of after-the-fact documentation.

3.3.2.7 Roles

Roles in STAPLE describe the kinds of contributions people make within a project. While they draw inspiration from frameworks like the CRediT taxonomy, they are not limited to authorship or publication contexts. Instead, roles provide a flexible way to document responsibilities and ensure that contributions are visible across the lifespan of a project.

In STAPLE, roles can be:

- Assigned to people → to show what responsibilities each contributor holds in a project.
- Assigned to tasks → to clarify what kind of contribution a task represents and how it fits into the project.

Roles provide a flexible system for understanding who did what and for standardizing contributions across projects. Role templates are avaliable to add to your account, which you can use across all projects. They must be created on the main dashboard before you can use them.

3.3.3 Project Level Options

3.3.3.1 Tags

Tags are flexible labels that help organize and filter milestones, tasks, and contributors. Tags make it easier to search and group related work across large projects. You can view customized tracking by using tags across these parts of the project.

3.3.3.2 Milestones

Milestones mark major points of progress within a project and help teams track deadlines and achievements. In STAPLE, milestones can also group related tasks together, providing a clear structure for managing phases of a project. Each milestone is tied to a timeline, and STAPLE automatically visualizes them in a Gantt chart, making it easier to see how tasks and deadlines align. This helps teams monitor progress, anticipate bottlenecks, and keep projects moving forward on schedule.

3.3.3.3 Notes

Notes provide a space for free-text documentation within a project. Unlike structured metadata captured through Forms, Notes allow contributors to record context, reminders, or reflections in their own words.

Notes can be:

- Private – visible only to the author.
- Shared with contributors – if shared by a project administrator, contributors can view them.
- Restricted to administrators – for communication only among project managers.

This flexibility makes Notes useful for everything from personal reminders to collaborative discussions among administrators.

3.3.3.4 Form Data

Form Data is the collected information from metadata Forms within a project. It provides a centralized view of all structured metadata entries—such as instrument details, ethics protocols, or dataset versions—that have been completed by contributors. Form Data can be reviewed, exported, and updated if errors have occurred.

3.3.3.5 Summary

The Summary page generates a shareable overview of the project. It consolidates contributors, tasks, roles, metadata, milestones, and outputs into one report. Project Summaries can be exported in both machine-readable formats (JSON) and human-readable interactive reports, making them useful for collaboration, archiving, and compliance with funder or journal requirements.

3.3.3.6 Settings

The Settings page allows administrators to configure project-level metadata. This includes details such as the project title, description, timeline, and high-level information about the project's scope.

4 Installation Guide

STAPLE is available as a hosted service at https://app.staple.science, which is the simplest way to get started.

This guide is for individuals or organizations who would like to host their own instance of STAPLE. Running your own version gives you full control over your data, configuration, and integration with your existing infrastructure.

4.1 Assumptions

This guide assumes you meet the following prerequisites:

- You are working on a Linux server. We use Ubuntu 20.04 in this guide, but the steps should be consistent for most Linux distributions.
- You have shell/terminal access to the server.
- You have worked with the command line before.
- You have `sudo` privileges to install packages and manage services.
- You understand what hidden files are (and how to view them).
- You have a basic understanding of SQL and databases (familiarity with PostgreSQL is not required).
- You know what Apache and/or Nginx servers are and how to edit their configuration files.
- You can imagine yourself using git (even if you're new to it).

4.2 Local Development or Hosting

You can also run STAPLE locally on your own computer, with a few limitations.

- If you are using a Linux system, you can follow the same instructions described in this guide from Blitz/JS Installation through Starting the App – Production.
- On Windows, you can run STAPLE as well, but you'll need to use a shell environment (such as PowerShell, Git Bash, or WSL2) to follow the same command-line instructions.
- On macOS, the setup is nearly identical to Linux, and you can use the standard Terminal application to run the commands.

When running locally, you will be the only user able to enter data. The application is not hosted on the web, so others will not be able to connect.

Despite this limitation, you will still have access to the full functionality of project management and metadata entry/output. This makes local installation useful for:

- Testing the system before deploying to a server
- Exploring functionality on your own
- Managing personal projects without a shared server

4.3 Installation Steps

4.3.1 Web Server Installation

STAPLE requires a web server to handle requests. You can choose between **Apache** or **Nginx**:

- **Apache** – Installation Instructions
- **Nginx** – Installation Instructions

  Important: Only install one. Apache and Nginx don't play nicely together if they're both running on the same server.

We recommend using **Nginx**, and our production server is configured with Nginx. This step is only required for people who want to host the software online. You do not need it for local development or use.

4.3.2 Blitz/JS Installation

STAPLE is built with **Blitz.js**, which runs on Node.js. To install the required tools:

- Install Node.js and npm
  - Follow the official installation instructions: Node.js Downloads
  - Required versions:
    - **Node.js**: v18+
    - **npm**: v9+
  - Verify installation with:

```
node -v
npm -v
```

- (Optional) Yarn: You may use Yarn if you prefer, but this guide assumes **npm**.
- Install Blitz.js
  - Run:

```
npm install -g blitz
```

- Verify installation (version **v2+** required):

```
blitz -v
```

> Note: Depending on your server setup, you may need to prepend commands with `sudo`.

4.3.3 Clone the Repository

Next, you'll need to copy the STAPLE source code onto your server.

- Install Git (if not already installed):

```
sudo apt-get update
sudo apt-get install git
```

- Navigate to your web directory (commonly `/var/www/html/` or `/var/www/`):
- *Note*: you can use any folder on your computer if you are not hosting the software online or want to do local development.

```
cd /var/www/html/
```

- Clone the repository:
  - Follow the GitHub guide on cloning repositories.
    - Or simply run:

```
git clone https://github.com/STAPLE-verse/STAPLE.git
```

- This will create a new folder named STAPLE containing all the files you need.
- Move into the project folder:

```
cd STAPLE
```

4.3.4 Database Installation

STAPLE requires a PostgreSQL database.

- Install PostgreSQL
  - Ensure that you have a local PostgreSQL service running.
  - Follow this guide for installation: Postgres Setup Instructions.
  - During installation:
    * Write down the superuser credentials (important on non-Linux systems).
    * Other databases may work, but you would need to modify the STAPLE codebase to support them.
- Create Databases
  - Open a terminal and run:

```
# Switch into PostgreSQL as the postgres superuser
sudo -u postgres psql

# Create databases for STAPLE
CREATE DATABASE staple;
CREATE DATABASE staple_test;
```

- Create a User
  - It's best not to use the default postgres user for STAPLE. Instead, create a new user:
  - Replace username and password with your desired values (keep the quotes).

```
CREATE USER username WITH PASSWORD 'password';
ALTER ROLE username CREATEDB;
```

- Check Databases
  - Inside the PostgreSQL prompt, you can verify:

```
\l   -- lists databases
\c staple   -- connect to the STAPLE database
\dn  -- lists schemas
```

- You should see something like the following using those commands:

```
postgres=# \c staple
You are now connected to database "staple" as user "postgres".
staple=#
```

```
staple=# \dn
 List of schemas
  Name  | Owner
--------+--------
 public | staple
(1 row)

staple=#
```

```
postgres=# \l
                                     List of databases
    Name     |  Owner   | Encoding |   Collate   |    Ctype    |     Access privileges
-------------+----------+----------+-------------+-------------+---------------------------
 postgres    | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 staple      | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =Tc/postgres             +
             |          |          |             |             | postgres=CTc/postgres    +
             |          |          |             |             | staple=CTc/postgres      +
             |          |          |             |             | staple_admin=CTc/postgres
 staple_test | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 template0   | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres              +
             |          |          |             |             | postgres=CTc/postgres
 template1   | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres              +
             |          |          |             |             | postgres=CTc/postgres
(5 rows)
```

- Grant Privileges
  - Finally, grant permissions for your new user to manage the staple schema:

```
GRANT ALL ON SCHEMA public TO username;
```

- Exit PostgreSQL using `\q`

4.3.5 Connect Database to STAPLE App

- Now we'll connect the PostgreSQL database you created to the STAPLE application.
- Navigate to your STAPLE project folder

```
cd /var/www/html/STAPLE
ls -al
```

- You should see a file named `.env` along with other project files.
- You may need to enable hidden files to see .env files.
- Example:

```
erin@staple:/var/www/html/STAPLE$ ls -al
total 764
drwxr-xr-x  13 root root   4096 Oct 24 06:17 .
drwxr-xr-x   6 root root   4096 Oct 24 05:02 ..
drwxr-xr-x   3 root root   4096 Oct 24 04:27 db
-rw-r--r--   1 root root    175 Oct 24 04:27 .editorconfig
-rw-r--r--   1 root root    494 Oct 24 06:17 .env
```

- Create a local environment file
    - Copy the .env file and rename it to .env.local:

```
cp .env .env.local
```

```
- Open it for editing:
```

```
nano .env.local
```

- Add required environment variables
    - At minimum, your .env.local file needs these:

```
# This env file should NOT be checked into source control
# Use this file for local overrides

# Replace username:password with your Postgres user and password
DATABASE_URL=postgresql://username:password@localhost:5432/staple

# Generate a random session key
SESSION_SECRET_KEY=your_generated_session_key

# Where the app will live
APP_ORIGIN="http://localhost:3000/"
```

- To generate a SESSION_SECRET_KEY:

  - Copy the output and paste it in place of your_generated_session_key.

```
openssl rand -hex 16
```

- Set up a test environment
  - Copy .env.test to .env.test.local:

```
cp .env.test .env.test.local
```

- Edit it to point to your test database:

```
DATABASE_URL=postgresql://username:password@localhost:5432/staple_test
```

4.3.6 Configure Email

STAPLE requires an email provider to send system emails (e.g., password resets, project invitations). You can use other options here, but we've configured three of them in our current code.

4.3.6.1 Secret Keys

- You must configure at least one of the following options:
- Gmail (App Passwords)
  - Create a Gmail App Password: Google Guide
  - Add to .env.local:

```
EMAIL_PASS="your_app_password"
```

- Amazon SES
  - Set up Amazon Simple Email Service (SES): Amazon SES Docs
  - Add to .env.local:

```
ACCESS_KEY="amazonACCESS"
SECRET_ACCESS_KEY="amazonSECRET"
```

- Resend
  - Get an API key from Resend.
  - Add to .env.local:

```
RESEND_API_KEY="resendAPI"
```

4.3.6.2 Edit the Mailer Configuration

In addition to adding one of the above environment variables, you'll need to edit the mailer configuration file so STAPLE knows which provider to use.

- Open the mailer configuration file:

```
nano integrations/mailer.js
```

- Delete non-used mailers: there are three chunks of code in this file - one for each type of email. You should delete the types you are not using to ensure the app runs. You can also insert your own mail options in this section.
- Edit the forgot password mailer:

```
nano mailers/forgotPasswordMailer.ts
```

- Change these sections to the mailer you want to use:

```
// import { Mailer } from "integrations/mailer"
import { createForgotPasswordMsg } from "integrations/emails"
// import { Amazon } from "integrations/mailer"
import { ResendMsg } from "integrations/mailer"


//send the email
await ResendMsg(createForgotPasswordMsg(to, resetUrl))
// await Amazon(createForgotPasswordMsg(to, resetUrl)) # or amazon
// await Mailer(createForgotPasswordMsg(to, resetUrl)) # or gmail
```

- Edit the emails.tsx file if you want to customize the emails that are sent with your own email address and logos.

```
nano integrations/emails.tsx
```

4.3.7 Install STAPLE Requirements

- Make sure you are in the STAPLE main folder before running these commands.
- Use the following (not the `#` line, these are notes) to install packages, tailwind, and daisyui.

```
# Install all project dependencies
npm install

# Install Tailwind CSS
blitz install tailwind

# Install DaisyUI
npm i -D daisyui@latest
```

- Set up the database schema:

```
# Generate Prisma client
blitz prisma generate

# Run migrations to create database structure
blitz prisma migrate dev
```

4.3.8 Starting the App - Local Testing

- Open a terminal window and navigate to the STAPLE folder you cloned:

```
cd /var/www/html/STAPLE
```

- Start the development server:

```
blitz dev
```

- Once the server is running, open your browser and go to: http://localhost:3000 (or whatever address is shown in your terminal output).

4.3.9 Starting the App - Production

- Open a terminal window and navigate to the STAPLE folder you cloned:

```
cd /var/www/html/STAPLE
```

- Build the application:

```
blitz build
```

Note: If the build process produces errors, you'll need to fix these before you can continue. Common issues include missing dependencies or mismatched Prisma client versions. Check the terminal output for details. We do not update our dev or main branches without fixing these, but it may be that your edits for your server caused hiccups.

- Once the build completes successfully, start the app in production mode:

```
blitz start
```

4.3.10 Keeping the App Going

When running in production, you'll want STAPLE to automatically restart after crashes or reboots. The recommended way to do this on Linux is by creating a systemd service.

- Create a service file
  - Navigate to the systemd directory and create a new file (we'll call it `blitz.service`):

```
cd /etc/systemd/system/
sudo nano blitz.service
```

- Example service file:
  - Replace `WorkingDirectory` with the path to your STAPLE installation.
  - You can also set specific users or production modes for security.

```
[Unit]
Description=Starts the Blitz service.
After=network.target

[Service]
Type=simple
WorkingDirectory=/var/www/html/STAPLE
ExecStart=/usr/local/bin/blitz start
Restart=always

[Install]
WantedBy=default.target
```

- Enable and control the service

  - Once your file is saved, you can manage it with:

```
# Start the service
sudo systemctl start blitz

# Check status (press "q" to quit)
sudo systemctl status blitz

# Restart the service
sudo systemctl restart blitz

# Stop the service
sudo systemctl stop blitz

# Enable service on boot
sudo systemctl enable blitz

# Disable service
sudo systemctl disable blitz

# Reset if failed
sudo systemctl reset-failed blitz
```

- Tutorial reference
  - If you're new to systemd services, see this helpful guide: Service Installation Instructions

4.3.11 Setting Up the Proxy (Nginx)

Now that STAPLE is running as a background service, you'll want to expose it through your web domain with Nginx.

- Navigate to your Nginx configuration folder

```
cd /etc/nginx/sites-enabled
```

- Create a new site config file
  - Name it after your site, e.g.:

```
sudo nano app.staple.science
```

- Add the server configuration
  - Replace app.staple.science with your domain and update the log path if needed:

```
server {
    listen 80;
    server_name app.staple.science;

    error_log /var/www/html/YOURFOLDER/logs/web-server.log;

    location / {
        proxy_pass http://localhost:3000/;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }
}
```

Important: Make sure the server_name matches your domain name, and the proxy_pass points to the port where Blitz is running (localhost:3000 by default).

- Test your config:

```
sudo nginx -t
```

- If you see syntax is ok and test is successful, reload Nginx:

```
sudo systemctl reload nginx
```

4.3.12 Enable HTTPS

- Use Let's Encrypt to add SSL by changing the location of your app in the following:

```
sudo apt install certbot python3-certbot-nginx
sudo certbot --nginx -d app.staple.science
```

- This will automatically configure HTTPS and renew your certificate.
- See Installation Instructions.

4.4 Common Issues & Fixes

4.4.1 Prisma / Database Errors

- Error: Error: P1000: Authentication failed against database server
  - Fix: Double-check your DATABASE_URL in .env.local. Make sure the username/password match the Postgres user you created.
- Error: Error: P3014: The underlying table does not exist
  - Fix: Run migrations again:

```
blitz prisma migrate dev
```

- Permission denied when running migrations
  - Fix: Ensure the Postgres user has privileges:

```
ALTER ROLE your_username CREATEDB;
GRANT ALL ON SCHEMA public TO your_username;
```

4.4.2 Build Errors

- Error: blitz build fails with missing dependencies
  - Fix: Run npm install again to ensure all dependencies are installed.
- Prisma client mismatch
  - Fix: Run:

```
blitz prisma generate
```

4.4.3 Email Issues

- No emails are being sent
  - Fix: Check that you kept only one mailer provider in mailer.ts.
  - Ensure the correct environment variables (EMAIL_PASS, ACCESS_KEY, SECRET_ACCESS_KEY, or RESEND_API_KEY) are set in .env.local.
  - For Amazon SES, make sure your MAIL_FROM address is verified.
  - For Gmail, use an App Password, not your regular password.

4.4.4 Nginx / Proxy Issues

- Nginx won't start
  - Fix: Run `sudo nginx -t` to test configuration. Correct any syntax errors before restarting.
- Site loads, but app doesn't respond
  - Fix: Make sure blitz is running (check with `sudo systemctl status blitz`).
  - Check that the `proxy_pass` in your Nginx config matches the Blitz port (localhost:3000 by default).
- SSL certificate not working
  - Fix: Ensure port 80/443 are open on your firewall.
  - Re-run certbot:

```
sudo certbot --nginx -d yourdomain.com
```

4.4.5 Service Issues

- Blitz doesn't restart after reboot
  - Fix: Make sure the service is enabled:

```
sudo systemctl enable blitz
```

- Service crashes with permission errors
  - Fix: Ensure the User in `blitz.service` has read/write access to the STAPLE folder.

4.4.6 General Tips

- After editing .env.local, .env.test.local, or blitz.service, restart the app:

```
sudo systemctl restart blitz
```

- Check logs:

```
journalctl -u blitz -f
```

Get help: contact staple.helpdesk at gmail.com.

5 Glossary

term	definition
authorship	Being listed as an author on a research paper, usually to recognize someone's c
Contributor/Member	An individual who actively participates in a project. Contributors may take on
contributorship	An alternative to traditional authorship that provides a more transparent acco
CRediT	A standardized system for describing the specific roles individuals play in resea
FAIR	FAIR stands for Findable, Accessible, Interoperable, and Reusable, a set of gui
Forms	Structured templates for capturing important project metadata, such as study
metadata	Metadata is information that describes other data. It helps explain what a dat
Milestones	Markers of significant progress within a project, such as completing data collec
Project	A container for a scientific effort, capturing the overall scope, goals, timeline, a
Project Summary	An automatically generated overview that consolidates key project information
Roles	Defined levels of access and responsibility within a project (e.g., project manag
Tags	Flexible labels used to categorize and filter tasks or projects. Tags help organiz
Task	A discrete unit of work within a project, such as designing a survey, analyzing
Task Logs	A chronological record of all activity on a task, including assignments, updates
Team	The group of contributors working together on a project. Teams may span labs
Transparency and Openness	A set of community-developed standards created to improve the transparency,
User	Any person with access to STAPLE. A user may belong to multiple projects ar

6 References

Adan, C. (2023). The importance of diversity in clinical research. *British Journal of Nursing (Mark Allen Publishing)*, *32*(18), 898–901. https://doi.org/10.12968/bjon.2023.32.18.898

Allen, L., O'Connell, A., & Kiermer, V. (2019). How can we ensure visibility and diversity in research contributions? How the Contributor Role Taxonomy (CRediT) is helping the shift from authorship to contributorship. *Learned Publishing*, *32*(1), 71–74. https://doi.org/10.1002/leap.1210

Allen, L., Scott, J., Brand, A., Hlava, M., & Altman, M. (2014). Publishing: Credit where credit is due. *Nature*, *508*(7496), 312–313. https://doi.org/10.1038/508312a

Borghi, J. A., & Gulick, A. E. V. (2021). Data management and sharing: Practices and perceptions of psychology researchers. *PLOS ONE*, *16*(5), e0252047. https://doi.org/10.1371/journal.pone.0252047

Cummings, J. N., & Kiesler, S. (2005). Collaborative Research Across Disciplinary and Organizational Boundaries. *Social Studies of Science*, *35*(5), 703–722. https://doi.org/10.1177/0306312705055535

Cummings, J. N., & Kiesler, S. (2007). Coordination costs and project outcomes in multi-university collaborations. *Research Policy*, *36*(10), 1620–1634. https://doi.org/10.1016/j.respol.2007.09.001

Fecher, B., & Friesike, S. (2014). *Open Science: One Term, Five Schools of Thought* (S. Bartling & S. Friesike, Eds.; pp. 17–47). Springer International Publishing. https://doi.org/10.1007/978-3-319-00026-8_2

Grant, S., Corker, K., Mellor, D., Stewart, S., Cashin, A., Lagisz, M., Mayo-Wilson, E., Moher, D., Umpierre, D., Barbour, V., Buck, S., Collins, G., Hazlett, H., Hrynaszkiewicz, I., Lee, C., Parker, T., Rethlefsen, M., Toomey, E., & Nosek, B. (2025). *TOP 2025: An update to the transparency and openness promotion guidelines.* https://doi.org/10.31222/osf.io/nmfs6_v2

Moshontz, H., Campbell, L., Ebersole, C. R., IJzerman, H., Urry, H. L., Forscher, P. S., Grahe, J. E., McCarthy, R. J., Musser, E. D., Antfolk, J., Castille, C. M., Evans, T. R., Fiedler, S., Flake, J. K., Forero, D. A., Janssen, S. M. J., Keene, J. R., Protzko, J., Aczel, B., … Chartier, C. R. (2018). The Psychological Science Accelerator: Advancing Psychology Through a Distributed Collaborative Network. *Advances in Methods and Practices in Psychological Science*, *1*(4), 501–515. https://doi.org/10.1177/2515245918797607

Rep. Foxx, V. [R.-N. (2019). *H.R.150 - 116th Congress (2019-2020): GREAT Act.* https://www.congress.gov/bill/116th-congress/house-bill/150

Rep. Ryan, P. D. [R.-W. (2019). *H.R.4174 - 115th Congress (2017-2018): Foundations for Evidence-Based Policymaking Act of 2018.* https://www.congress.gov/bill/115th-congress/

house-bill/4174

Swartz, T. H., Palermo, A.-G. S., Masur, S. K., & Aberg, J. A. (2019). The Science and Value of Diversity: Closing the Gaps in Our Understanding of Inclusion and Diversity. *The Journal of Infectious Diseases*, *220*(220 Suppl 2), S33–S41. https://doi.org/10.1093/infdis/jiz174

Wilkinson, M. D., Dumontier, M., Aalbersberg, Ij. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., Silva Santos, L. B. da, Bourne, P. E., Bouwman, J., Brookes, A. J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C. T., Finkers, R., … Mons, B. (2016). The FAIR Guiding Principles for scientific data management and stewardship. *Scientific Data*, *3*(1), 160018. https://doi.org/10.1038/sdata.2016.18

Wuchty, S., Jones, B. F., & Uzzi, B. (2007). The Increasing Dominance of Teams in Production of Knowledge. *Science*, *316*(5827), 1036–1039. https://doi.org/10.1126/science.1136099