Academic Website Setup Guide

This guide walks you through building an academic website exactly like kadengrace.co — a free, professional site hosted on GitHub. It is based on the academicpages Jekyll template and the Jay Rob Williams tutorial, with all the customizations made to this specific site documented step by step.

No prior coding experience is assumed. If you know how to edit a Word document and have ever copied and pasted text, you can do this.

Table of Contents

---

1. What You're Building

You will end up with:

- A **personal academic website** hosted for free at `[yourusername].github.io` (or a custom domain like `kadengrace.co`)
- A **home page** with a short bio and research fields
- A **CV page** that displays your PDF CV directly in the browser
- A **research page** with collapsible abstracts and a scrollable image gallery for graphs
- A **teaching page** with course listings and syllabus links
- Any **custom pages** you want (like a scuba page, a hobbies page, etc.)
- A **dark/light mode toggle** that visitors can use
- **Google Analytics** so you can see how many people visit your site

The site is built from plain text files (called Markdown files) that GitHub automatically turns into a website. You never touch raw HTML except in a few small places.

---

2. Create a GitHub Account

Go to github.com and create a free account.

Important: Your username will become part of your website's URL. For example, the username `kadengrace` produces `kadengrace.github.io`. Pick something professional — your name or a variation of it.

---

3. Fork the Template

"Forking" means making your own copy of someone else's project.

1. Go to: https://github.com/academicpages/academicpages.github.io
2. In the top-right corner, click the **Fork** button.
3. Wait a few seconds. You'll land on your own copy of the repo, which will be named `[yourusername]/academicpages.github.io`.

---

4. Rename Your Repository

Your repo needs to be named `[yourusername].github.io` exactly — this is how GitHub knows to turn it into a website.

1. On your forked repo page, click the **Settings** tab (top right area, has a gear icon).
2. Under the **General** section, find the **Repository name** field.
3. Change it from `academicpages.github.io` to `[yourusername].github.io`.
    - For example: `kadengrace.github.io`
4. Click **Rename**.

---

5. Download Git and Clone the Repo

"Cloning" means downloading the files to your computer in a way that keeps them connected to GitHub.

Install Git

- **Windows:** Download and install Git Bash. This gives you a terminal that works like a Mac/Linux terminal, which is what most online Git instructions assume.
- **Mac:** Run `xcode-select --install` in your terminal. This installs Git plus a Ruby environment you'll need later.
- **Linux:** Git is likely already installed. If not, run `sudo apt install git` or your distro's equivalent.

Clone Your Repo

Open a terminal (Git Bash on Windows, Terminal on Mac/Linux). Navigate to where you want your website folder to live:

```
cd ~/Desktop
```

Then clone the repo:

```
git clone https://github.com/[yourusername]/[yourusername].github.io.git Website
```

Replace `[yourusername]` with your GitHub username. The word `Website` at the end gives the folder a clean name instead of the long default name.

You now have a folder called `Website` on your Desktop with all the template files inside.

---

6. Install Jekyll to Preview Locally

Jekyll is the program that turns your text files into a website. You'll use it to preview changes on your own computer before anyone else can see them.

Install Ruby (Windows only)

Go to rubyinstaller.org and download the recommended version (the one with `(x64)` and a highlighted arrow). Run the installer, accept the defaults, and make sure to check the box to install MSYS2 development tools when prompted.

Install Bundler

Bundler is a tool that installs all of Jekyll's dependencies. In your terminal, run:

```
gem install bundler
```

Install Jekyll's Dependencies

Navigate into your website folder:

```
cd ~/Desktop/Website
```

Then run:

```
bundle install
```

This reads the `Gemfile` in the folder and installs everything needed. It may take a minute.

Preview Your Site

```
bundle exec jekyll serve
```

Open your browser and go to `http://localhost:4000`. You'll see the template site running on your own computer. **No one else can see this** — it's just for you to check your work.

To stop the server, press `Ctrl+C` in the terminal.

> **Tip:** Every time you save a change to a file, Jekyll automatically rebuilds the site. Just refresh your browser to see the update. The one exception is `_config.yml` — if you change that file, you need to stop and restart the server.

---

7. Configure the Site (`_config.yml`)

`_config.yml` is the master settings file for your site. Open it in a text editor (Notepad on Windows, TextEdit on Mac, or a code editor like VS Code).

The file is located at `Website/_config.yml`.

Basic Info

Find and replace the following lines:

```
title                    : "Your Name / Site Title"
```

Change to:

```
title                    : "Your Name"
```

```
name                     : &name "Your Name"
```

Change to:

```
name                     : &name "Your Name"
```

```
description              : &description "personal description"
```

Change to a real description. This shows up in search engine results:

```
description              : &description "Your Name is an Economics PhD candidate
at [University] specializing in [fields]. Research on [topics]."
```

```
url                      : https://academicpages.github.io
```

Change to:

```
url                      : https://[yourusername].github.io
```

(Or your custom domain if you have one — see Step 19.)

```
repository               : "academicpages/academicpages.github.io"
```

Change to:

```
repository               : "[yourusername]/[yourusername].github.io"
```

Author Sidebar

Scroll down to the `author:` section. This controls the information that appears in the sidebar on every page.

```
author:
  avatar           : "Headshot.jpg"       # filename of your photo in /images/
  name             : "Your Name"
  bio              : "Economics PhD Candidate"
  location         : "City, State"
  employer         : "Your University"
  email            : "you@university.edu"
  linkedin         : "your-linkedin-username"
```

Fill in only what you want shown. Leave anything blank (just empty quotes `""` or no value after the colon) and that field's icon won't appear.

Publication Category Labels

By default the template uses "Conferences" as a category name. This site renamed it to "Working Papers":

```
publication_category:
  books:
```

```
    title: 'Books'
  manuscripts:
    title: 'Publications'
  conferences:
    title: 'Working Papers'
```

Social Sharing Thumbnail

These lines set what image and description appear when someone shares a link to your site on social media:

```
og_image                : "/images/Headshot.jpg"
og_description          : "Your Name - [Position] at [University]. Research in
[fields]."
```

Google Analytics

Add your tracking ID here (you get this from Google Analytics — see Step 17):

```
analytics:
  provider              : "google-analytics-4"
  google:
    tracking_id         : "G-XXXXXXXXXX"
```

---

8. Set Up Navigation

The navigation bar at the top of every page is controlled by `_data/navigation.yml`.

Open `Website/_data/navigation.yml`. By default it has many links. This site uses a simplified version:

```
main:
  - title: "Home"
    url: /

  - title: "CV"
    url: /cv/

  - title: "Research"
    url: /research/

  - title: "Teaching"
    url: /teaching/
```

Delete any links you don't want. The order here is the order they appear in the nav bar. You can add new pages later by adding more `- title:` / `url:` entries.

---

9. Edit Your Home Page

The home page lives at `_pages/about.md`. The name is a little misleading — "about" is really just the home page.

Open it and replace the template content with your own introduction. Here is the structure used on this site:

```
---
permalink: /
title: "Your Name, Your Title"
description: "A description for search engines."
author_profile: true
redirect_from:
  - /about/
  - /about.html
---

Hello! I'm [Your Name], a [year]-year [Program] PhD candidate at [University]. My [research](/research/) uses [methods] to answer questions about [fields]. Outside the office, I [hobby].

I will graduate in [Month Year].

You can find my [CV here](/cv/).

You can email me at [you@university.edu](mailto:you@university.edu).

Research Fields
------
* Field 1
* Field 2
* Field 3

Background
------
I earned my [degree] from [University]. [Additional background.]
```

A few things to know about Markdown formatting:

- A `#` at the start of a line makes a big heading. `##` makes a slightly smaller heading, and so on.
- A `*` or `-` at the start of a line makes a bullet point.
- `[link text](url)` creates a clickable link.
- Leaving a blank line between paragraphs creates a new paragraph.
- The section at the very top between `---` lines is called "front matter." Don't delete it — it tells Jekyll how to build the page.

---

10. Build Your CV Page

The CV page lives at `_pages/cv.md`.

This site's CV page embeds the PDF directly so visitors can read it in their browser without downloading it, plus provides a download button. Here is the full structure:

```
---
layout: archive
title: "CV - Your Name"
description: "Curriculum Vitae of Your Name."
permalink: /cv/
author_profile: true
redirect_from:
  - /resume
---

[Download my CV (PDF)](/files/CV - Your Name.pdf){: .btn .btn--primary .btn--
large}

<embed src="/files/CV - Your Name.pdf" type="application/pdf" width="100%"
height="800px" />

---

## Education

**Ph.D. [Field]** (Expected [Year])
*[University]*

**B.S. [Field]** ([Year])
*[University]*

---

## Research Fields

* Field 1
* Field 2

---

## [Teaching Experience](/teaching/)

**Instructor of Record**
*[University]*
* [Course name], [Semester Year]

---

## Fellowships & Awards

* Award name, Year
* Fellowship name, Year
```

The `<embed>` tag is a small piece of HTML that displays the PDF inline. The `{: .btn .btn--primary .btn--large}` part after the download link is a special instruction that styles it as a button.

Put your actual PDF file in the `files/` folder inside your Website folder (see Step 14).

---

11. Build Your Research Page

The research page lives at `_pages/research.md`.

This site's research page has two features beyond plain text: **collapsible abstracts** and a **scrollable image gallery**. Those require some extra setup covered in Step 15. Here is the base structure without those features first, so you can get the page working:

```
---
layout: archive
title: "Research - Your Name"
description: "Your Name's research in [fields]."
permalink: /research/
author_profile: true
---
<br>
# Working Papers

## Paper Title [(Click to view latest draft)](/files/yourpaper.pdf)
{:target="_blank"}

*Abstract:* Your abstract text here.

---

# Papers in Progress

## Another Paper Title

*(with [Coauthor Name](https://their-website.com){:target="_blank"})*

Brief description or abstract.
```

`{:target="_blank"}` makes the link open in a new tab rather than navigating away from your site. This is a good practice for any external links or PDF links.

---

12. Build Your Teaching Page

The teaching page lives at `_pages/teaching.md`.

```
---
layout: archive
title: "Teaching - Your Name"
```

```
description: "Your Name's teaching experience."
permalink: /teaching/
author_profile: true
---

## Instructor of Record

*[University]*

<br>
**[Course Number: Course Name](link-to-course-catalog){:target="_blank"}**

*[Semester Year]*

*[Syllabus](/files/Syllabus.pdf){:target="_blank"}*

Brief description of the course and your role.

---

## Teaching Assistant

*[University]*
<br>
**[Course Number: Course Name]**

*[Semester Year]*

Description of your responsibilities.
```

`<br>` is HTML for a line break — it adds a little extra space between elements. The template doesn't always add enough space between items naturally, so sprinkling `<br>` tags in is a simple fix.

---

13. Add a Custom Page (e.g., Scuba)

You can add any page you want that isn't part of the default template. To add a page like the scuba/hobbies page on this site:

1. Create a new file in the `_pages/` folder. Name it whatever you want, e.g., `scuba.md`.
2. Give it this front matter at the top:

```
---
layout: archive
title: "Page Title"
description: "A description for search engines."
permalink: /scuba/
author_profile: true
---
```

3. Write your content below the front matter using normal Markdown.
4. Add a photo with a caption like this:

```
<figure>
  <img src="/images/yourphoto.jpg" alt="Description of photo" style="border-
radius: 10px;">
  <figcaption>Caption text here</figcaption>
</figure>
```

5. Finally, add the page to your navigation in `_data/navigation.yml` if you want it in the top nav bar:

```
  - title: "Scuba"
    url: /scuba/
```

Or you can simply link to it from another page (like the home page) without it being in the nav bar:

```
Outside the office, I spend a lot of time [underwater](/scuba/).
```

---

14. Add PDFs and Images to Your Site

PDFs

Put any PDF files (CV, papers, syllabi) in the `files/` folder inside your Website directory. Then link to them in your pages like this:

```
[My CV](/files/CV - Your Name.pdf)
```

The exact filename in the link must match the exact filename of the file you placed in `files/`. Capitalization and spaces matter.

Images

Put image files (`.jpg`, `.png`, `.jpeg`) in the `images/` folder inside your Website directory.

Your **profile photo** goes here too. Name it something simple like `Headshot.jpg` and reference it in `_config.yml`:

```
avatar           : "Headshot.jpg"
```

Link to images in your pages like this:

```
![Alt text description](/images/yourimagefile.png)
```

Or with more control over sizing and style (using HTML):

```
<img src="/images/yourimagefile.png" alt="Description" style="border-radius:
10px;">
```

---

15. Add Collapsible Abstracts and an Image Scroller

This is the most complex customization on the site. The research page uses:

- **Collapsible abstracts** — clicking "Abstract" expands/collapses the text
- **Image scroller** — a horizontal scroll gallery with arrow buttons for graphs

These require three new files and two small edits to existing template files.

Step 1: Create `assets/css/collapse.css`

Create a new file at `Website/assets/css/collapse.css` and paste in the following:

```
/* Abstract Accordion */
.abstract-accordion {
    margin: 15px 0;
    border: none;
}

.abstract-header {
    background-color: #3a3a3a;
    color: #ffffff;
    padding: 12px 15px;
    cursor: pointer;
    font-weight: 600;
    border-radius: 5px;
    border: 1px solid #4a4a4a;
    transition: background-color 0.3s ease, border-color 0.3s ease;
    display: flex;
    justify-content: space-between;
    align-items: center;
    user-select: none;
}

.abstract-header:hover {
    background-color: #4a4a4a;
}

.abstract-header::after {
    content: "▼";
```

```
    color: #ffffff;
    font-size: 12px;
    transition: transform 0.3s ease;
}

.abstract-header.active::after {
    transform: rotate(-180deg);
}

.abstract-content {
    display: none;
    padding: 15px;
    margin-top: 10px;
    font-style: italic;
    line-height: 1.6;
    background-color: #3a3a3a;
    color: #ffffff;
    border-radius: 5px;
}

/* Image Scroller */
.image-scroller-container {
    margin: 20px 0;
    overflow: hidden;
    position: relative;
}

.scroller-arrow {
    position: absolute;
    top: 50%;
    transform: translateY(-50%);
    background-color: rgba(0, 0, 0, 0.6);
    color: white;
    border: none;
    width: 50px;
    height: 50px;
    border-radius: 50%;
    cursor: pointer;
    font-size: 24px;
    display: flex;
    align-items: center;
    justify-content: center;
    z-index: 10;
    transition: background-color 0.3s ease;
    user-select: none;
}

.scroller-arrow:hover {
    background-color: rgba(0, 0, 0, 0.8);
}

.scroller-arrow.left { left: 10px; }
.scroller-arrow.right { right: 10px; }
```

```
.image-scroller {
    display: flex;
    gap: 15px;
    overflow-x: auto;
    scroll-behavior: smooth;
    padding: 10px 0;
}

.image-tile {
    flex: 0 0 auto;
    width: 400px;
    max-width: 90vw;
}

.image-tile img {
    width: 100%;
    height: auto;
    border-radius: 8px;
    box-shadow: 0 2px 8px rgba(0,0,0,0.1);
    cursor: pointer;
}

.image-tile img:hover {
    transform: scale(1.02);
    box-shadow: 0 4px 16px rgba(0,0,0,0.2);
}

@media (max-width: 768px) {
    .image-tile { width: 300px; }
}
```

Step 2: Create `assets/js/collapse.js`

Create a new file at `Website/assets/js/collapse.js` and paste in:

```
document.addEventListener('DOMContentLoaded', function() {
    // Collapsible abstracts
    const headers = document.querySelectorAll('.abstract-header');
    headers.forEach(function(header) {
        header.addEventListener('click', function() {
            const content = this.nextElementSibling;
            const isActive = this.classList.contains('active');
            headers.forEach(function(h) {
                h.classList.remove('active');
                if (h.nextElementSibling) h.nextElementSibling.style.display =
'none';
            });
            if (!isActive) {
                this.classList.add('active');
                content.style.display = 'block';
            }
```

```
        });
    });

    // Image scroller arrows
    document.querySelectorAll('.image-scroller-
container').forEach(function(container) {
        const scroller = container.querySelector('.image-scroller');
        const leftBtn = container.querySelector('.scroller-arrow.left');
        const rightBtn = container.querySelector('.scroller-arrow.right');
        if (leftBtn) leftBtn.addEventListener('click', function() {
            scroller.scrollBy({ left: -420, behavior: 'smooth' });
        });
        if (rightBtn) rightBtn.addEventListener('click', function() {
            scroller.scrollBy({ left: 420, behavior: 'smooth' });
        });
    });
});
```

Step 3: Load the CSS in `_includes/head.html`

Open `_includes/head.html`. Find the closing `</head>` tag near the bottom and add one line just before it:

```
<link rel="stylesheet" href="/assets/css/collapse.css">
</head>
```

Step 4: Load the JS in `_includes/scripts.html`

Open `_includes/scripts.html`. Find the closing `</body>` tag (or the end of the file) and add:

```
<script src="/assets/js/collapse.js"></script>
```

Step 5: Use the Components in `_pages/research.md`

Now you can use these components anywhere in your research page.

Collapsible abstract:

```
<div class="abstract-accordion">
  <div class="abstract-header">Abstract</div>
  <div class="abstract-content">
    Your abstract text goes here.
  </div>
</div>
```

Image scroller (put this after a paper's abstract):

```
<div class="image-scroller-container">
  <button class="scroller-arrow left">‹</button>
  <button class="scroller-arrow right">›</button>
  <div class="image-scroller">
    <div class="image-tile">
      <img src="/images/graph1.png" alt="Description of graph 1">
    </div>
    <div class="image-tile">
      <img src="/images/graph2.png" alt="Description of graph 2">
    </div>
  </div>
</div>
```

Add as many `<div class="image-tile">` blocks as you have graphs.

---

16. Customize the Dark Mode Colors

The site uses a darker, higher-contrast dark mode than the template default. These changes are in `_sass/theme/_default_dark.scss`.

Open that file and find the variable declarations at the top. Change:

```
$background                 : #474747;
$background-light           : mix(#fff, $background, 80%);
$background-lighter         : mix(#fff, $background, 90%);
$link                       : #0ea1c5;
```

To:

```
$background                 : #2a2a2a;
$background-light           : #3a3a3a;
$background-lighter         : #4a4a4a;
$link                       : #52adc8;
```

Then, inside the `html[data-theme="dark"]` block further down in the same file, make these changes:

Variable	Old value	New value
`--global-footer-bg-color`	`#{$background}`	`#1a1a1a`
`--global-border-color`	`#{$light-gray}`	`#{$background-lighter}`
`--global-dark-border-color`	`#{$background-light}`	`#5a5a5a`
`--global-code-background-color`	`#fafafa`	`#1a1a1a`
`--global-code-text-color`	`#{$darker-gray}`	`#{$text}`

Variable	Old value	New value
`--global-fig-caption-color`	`#{$light-gray}`	`#{$text}`
`--global-link-color-hover`	`#{$link-dark}`	`#{$link-light}`
`--global-masthead-link-color-hover`	`#{$background-light}`	`#{$link}`
`--global-text-color-light`	`#f3f3f3`	`#{$text}`
`--global-thead-color`	`#{$background-lighter}`	`#{$background-light}`

These changes make the dark mode background darker (easier on the eyes), give code blocks proper contrast, and make links more visible.

---

17. Set Up Google Analytics

Google Analytics lets you see how many people visit your site, which pages they read, and where they came from.

1. Go to analytics.google.com and sign in with a Google account.
2. Click **Start measuring** and follow the prompts to create an account and a property for your website.
3. When asked for your website URL, enter `https://[yourusername].github.io` (or your custom domain).
4. Choose **Web** as the platform.
5. After setup, you'll receive a **Measurement ID** that looks like `G-XXXXXXXXXX`.

Paste that ID into `_config.yml`:

```
analytics:
  provider               : "google-analytics-4"
  google:
    tracking_id          : "G-XXXXXXXXXX"
```

Analytics data won't show up until your site is live and people start visiting it.

---

18. Go Live on GitHub Pages

Now it's time to push all your changes to GitHub and turn the repo into a live website.

Push Your Changes

In your terminal, navigate to your Website folder if you aren't already there:

```
cd ~/Desktop/Website
```

Then run these three commands in order:

```
git add -A
git commit -m "Initial setup with my info"
git push
```

`git add -A` stages all changed files. `git commit -m "..."` saves a snapshot of your changes with a description. `git push` uploads everything to GitHub.

Git will prompt you for your GitHub username and password the first time. If it asks you to authenticate, follow the prompts — GitHub may ask you to use a personal access token instead of your password (you can create one at github.com → Settings → Developer settings → Personal access tokens).

Enable GitHub Pages

1. Go to your repository on GitHub (`github.com/[yourusername]/[yourusername].github.io`).
2. Click the **Settings** tab.
3. In the left sidebar, click **Pages**.
4. Under **Source**, select `Deploy from a branch`.
5. Under **Branch**, choose `master` (or `main`) and click **Save**.

After a minute or two, your site will be live at `https://[yourusername].github.io`.

> **Tip:** The first build sometimes takes 5–10 minutes. If the site isn't there yet, wait a bit and refresh.

---

19. Connect a Custom Domain

A custom domain like `kadengrace.co` looks more professional than `kadengrace.github.io`. Domains cost around $10–15/year and can be purchased from registrars like Namecheap or Google Domains.

Step 1: Create a CNAME File

In your Website folder, create a plain text file named exactly `CNAME` (no extension, all caps). Its entire contents should be just your domain:

```
yourdomain.com
```

For example, this site's CNAME file contains:

```
kadengrace.co
```

Add, commit, and push this file:

```
git add CNAME
git commit -m "Add custom domain"
```

```
git push
```

Step 2: Configure DNS at Your Registrar

Log into wherever you bought your domain and find the DNS settings. You need to add the following records:

A records (point your domain to GitHub's servers):

Type	Host	Value
A	@	185.199.108.153
A	@	185.199.109.153
A	@	185.199.110.153
A	@	185.199.111.153

CNAME record (for the `www` version of your domain):

Type	Host	Value
CNAME	www	[yourusername].github.io

Step 3: Update `_config.yml`

Change the `url` line to your custom domain:

```
url                      : https://yourdomain.com
```

Step 4: Enable HTTPS in GitHub Pages Settings

Back on your GitHub repo's Settings → Pages page, after the DNS propagates (can take up to 24 hours), check the **Enforce HTTPS** box.

---

20. The Git Workflow: Saving and Uploading Changes

Every time you update your site — changing a page, adding a PDF, tweaking the config — you need to go through the same three steps to get those changes live:

```
git add -A
git commit -m "Brief description of what you changed"
git push
```

What These Commands Do

- `git add -A` — Tells Git to pay attention to all the files you've changed. Think of it as selecting everything to save.
- `git commit -m "..."` — Saves a snapshot of those changes with a short description. This is your save point. The message is just for your own records — write something that reminds you what you did.
- `git push` — Uploads your saved snapshot to GitHub. GitHub then rebuilds your live website, which takes about a minute.

Tips

- To see what files you've changed before committing, run `git status`.
- To see exactly what changed in a file, run `git diff _pages/about.md` (replace with any filename).
- If you added new files (like a new PDF or image), `git add -A` handles those automatically.
- Your live site updates within about 60 seconds of a push. Hard-refresh your browser (`Ctrl+Shift+R` on Windows, `Cmd+Shift+R` on Mac) to see the latest version.

File Structure Reference

Here are the files you'll edit most often:

File	What it controls
`_config.yml`	Site-wide settings: name, URL, author info, analytics
`_data/navigation.yml`	Which links appear in the top nav bar
`_pages/about.md`	Home page content
`_pages/cv.md`	CV page
`_pages/research.md`	Research page
`_pages/teaching.md`	Teaching page
`_pages/[name].md`	Any custom page you create
`files/`	PDFs (CV, papers, syllabi)
`images/`	Photos and graphs
`assets/css/collapse.css`	Styles for accordion and image scroller
`assets/js/collapse.js`	JavaScript for accordion and image scroller
`_sass/theme/_default_dark.scss`	Dark mode color scheme
`CNAME`	Custom domain name

Built on the academicpages template. Guide written May 2026.