SelfMemo 2.0

Open-Source, Self-Hosted Email Reminder Service

Niklas Lorber, Maria Seiser, and Kilian Weisl

706.041 Information Architecture and Web Usability 3VU WS 2024/2025
Graz University of Technology

04 Feb 2025

Abstract

SelfMemo 2.0 is an open-source, self-hosted, web-based email reminder service. It is designed to help users efficiently schedule and manage email-based notifications and supports both local and cloud-based deployment. The platform was developed to address the discontinuation and limitations of prior solutions, offering a modern and scalable alternative.

SelfMemo 2.0 utilizes state-of-the-art technologies and features, such as role-based authorization, multiple scheduling options, allowing users to create, modify, and manage reminders according to their needs, sending automated warning reminders. Customizable email templates provide users with greater control over the content and format of their reminders.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

SelfMemo 2.0 [Lorber et al. 2025] is an open-source, self-hosted, web-based email reminder service. It designed to assist users in managing email reminders effectively. This project was developed as part of the course 706.041 Information Architecture and Web Usability in WS 2024/2025 at Graz University of Technology [Andrews 2025]. The development of SelfMemo 2.0 was driven by the need for a reliable and user-friendly platform that lets users manage their reminders. The software is freely available on GitHub [Lorber et al. 2025].

Previous services, such as MemoToMe [JJ Zone 2025], provided useful functionality but were discontinued, leaving users without an alternative for managing email-based reminders. This created an opportunity to develop a new solution that incorporates similar functionality while introducing enhanced features for better usability.

Many existing reminder applications do not provide sufficient customization in terms of scheduling, warning notifications, and email template personalization. Furthermore, many reminder services rely on proprietary infrastructure that may not be accessible to all users. SelfMemo 2.0 offers an open-source alternative that ensures reliability, scalability, and ease of deployment, utilizing cloud-based solutions such as Vercel [Vercel 2025c] for deployment and Neon [Neon 2025] for database hosting.

SelfMemo 2.0 aims to achieve the following objectives:

- Integration of administrative role for managing users.
- Integration of user role managing reminders.
- Secure authentication and authorization mechanisms.
- Flexible scheduling options, including customizable reminder intervals and warning notifications.
- Intuitive installation and deployment process for both local and production environments.

Chapter 2

Email Reminder Services

This chapter gives an overview of existing email reminder services, comparing their features and how they address common problems. Particular attention is paid to the two applications that served as a model for the work in this project, namely: MemoToMe and SelfMemo.

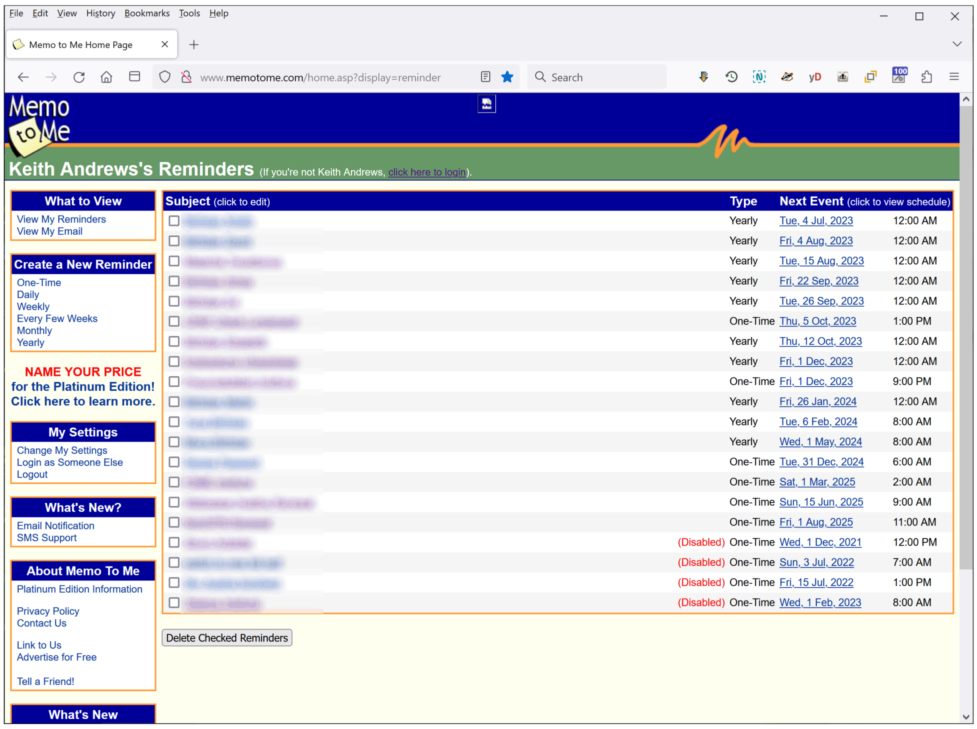
Managing appointments and deadlines efficiently has become increasingly important in today's fast-paced, event-driven society. Reminder services play a crucial role in helping individuals organize their schedules by notifying them of upcoming events. These services come in various forms and implementations. Based on their implementation and primary mode of notification, reminder services can be grouped into several categories:

- *Calendar-Based Reminder Services*: Integrated into digital calendar applications, these services notify users about scheduled events via pop-up alerts, emails, or notifications.
- *Application-Based Reminders*: Standalone applications, such as Apple's Reminders [Apple 2025] or Google Keep [Google 2025], primarily use push notifications to remind users of tasks and events.
- *Email Reminder Services*: These services deliver reminder notifications via email, allowing users to receive alerts directly in their inbox.
- *SMS Reminder Services*: Some services send reminder messages via SMS, ensuring notifications reach users even without internet access.
- *Hybrid Reminder Systems*: A combination of different approaches, these systems integrate email, SMS, and push notifications to maximize reliability and user reach.

These different approaches fit different user needs, whether for personal task management, professional scheduling, or general event notifications.

Email reminder services distinguish themselves by utilizing email as the primary communication channel for delivering notifications. This approach offers several advantages: emails are widely used, ensuring broad accessibility, and they are less intrusive compared to push notifications or SMS messages. Since nearly everyone has an email account, this method provides a convenient and effective way to deliver timely reminders without disrupting the user's daily activities. Email reminder services have several essential features to enhance productivity and ensure no important matters are missed, inlcuding:

- *Scheduled Reminders*: Users can specify exact dates and times when they wish to be reminded about specific events.
- *Recurring Reminders*: This functionality allows users to set reminders that repeat at defined intervals, ideal for ongoing commitments and regular follow-ups or just birthdays.
- *User Management*: Some reminder services can manage users and assign roles.



Keith Andrews's Reminders (If you're not Keith Andrews, click here to login).

Subject (click to edit)		Type	Next Event (click to view schedule)	
[redacted]		Yearly	Tue, 4 Jul, 2023	12:00 AM
[redacted]		Yearly	Fri, 4 Aug, 2023	12:00 AM
[redacted]		Yearly	Tue, 15 Aug, 2023	12:00 AM
[redacted]		Yearly	Fri, 22 Sep, 2023	12:00 AM
[redacted]		Yearly	Tue, 26 Sep, 2023	12:00 AM
[redacted]		One-Time	Thu, 5 Oct, 2023	1:00 PM
[redacted]		Yearly	Thu, 12 Oct, 2023	12:00 AM
[redacted]		Yearly	Fri, 1 Dec, 2023	12:00 AM
[redacted]		One-Time	Fri, 1 Dec, 2023	9:00 PM
[redacted]		Yearly	Fri, 26 Jan, 2024	12:00 AM
[redacted]		Yearly	Tue, 6 Feb, 2024	8:00 AM
[redacted]		Yearly	Wed, 1 May, 2024	8:00 AM
[redacted]		One-Time	Tue, 31 Dec, 2024	6:00 AM
[redacted]		One-Time	Sat, 1 Mar, 2025	2:00 AM
[redacted]		One-Time	Sun, 15 Jun, 2025	9:00 AM
[redacted]		One-Time	Fri, 1 Aug, 2025	11:00 AM
[redacted]	(Disabled)	One-Time	Wed, 1 Dec, 2021	12:00 PM
[redacted]	(Disabled)	One-Time	Sun, 3 Jul, 2022	7:00 AM
[redacted]	(Disabled)	One-Time	Fri, 15 Jul, 2022	1:00 PM
[redacted]	(Disabled)	One-Time	Wed, 1 Feb, 2023	8:00 AM

Figure 2.1: MemoToMe offers a reminder list interface, showing a structured table of scheduled reminders. [Image created by Keith Andrews. Used with kind permission.]

- *Customization Options*: Users can often customize the nature of reminders and how they look, ensuring individuality.
- *Analytics and Reporting*: Advanced services tend to offer analytics features, providing insights into reminder effectiveness, response rates, or overall usage patterns.

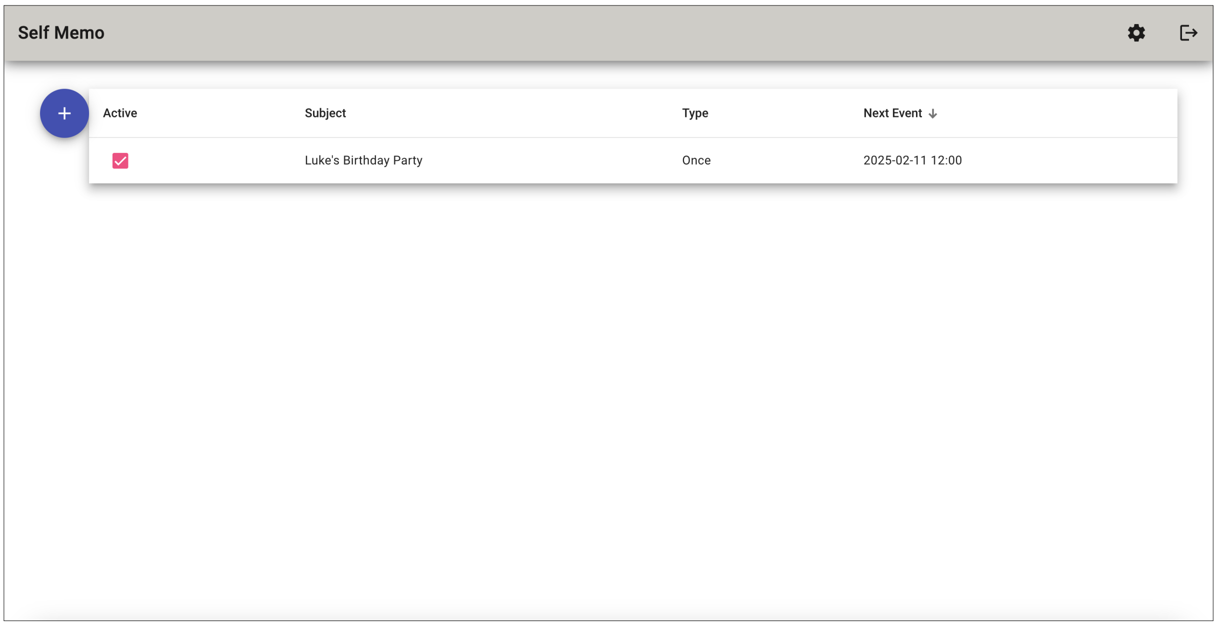
2.1 MemoToMe

MemoToMe [JJ Zone 2025] was a comprehensive web-based email reminder service. The platform allowed for one-time and recurring email reminders, with warning emails at set intervals. Its user interface is shown in Figure 2.1. Unfortunately, although the user interface of the service is still available, it is no longer sending out emails.

The main features of MemoToMe included:

- *Comprehensive Scheduling*: Users can set email reminders for any event, choosing exact times or time intervals.
- *Reminder Overview*: A complete list of all reminders, including the next scheduled occurrence.
- *Various Reminder Types*: Supports multiple reminder intervals, including: one-time, daily, weekly, every few weeks, monthly, and yearly.
- *Warning Reminders*: Users can enable warning reminders at specific intervals before the main reminder occurs.
- *Basic Statistics*: Provides simple usage statistics, helping users analyze their reminder patterns.

Overall, MemoToMe stood out as a robust, feature-rich, free-to-use reminder service that seamlessly combined simplicity and functionality. It offered all the essential tools for managing reminders while



Active	Subject	Type	Next Event ↓
✓	Luke's Birthday Party	Once	2025-02-11 12:00

Figure 2.2: SelfMemo 1.0 offers a simple reminder list interface, showing a simple table with active reminders. [Screenshot created by the author(s) of this report.]

ensuring an intuitive and user-friendly experience. With its discontinuation, a significant gap emerged for a free alternative solution.

2.2 SelfMemo 1.0

SelfMemo 1.0 [Juri et al. 2024] is the direct predecessor to this project work. The web application was developed in 2023 by students participating in the course Information Architecture and Web Usability. Its user interface is shown in Figure 2.2. SelfMemo 1.0 was conceived as an initial response to the discontinuation of the MemoToMe service and successfully implemented some of the key functionality essential to an email reminder service, including:

- *User Management*: An admin user can create and manage users accounts.
- *SMTP Selection*: An admin user can configure the preferred SMTP server for sending reminders.
- *Reminder Overview*: A structured list of all scheduled reminders along with their next occurrences, ensuring clear and efficient tracking.
- *Basic Reminder Types*: Support for the following reminder types: one-time, daily, weekly, monthly, and yearly,

As is evident, SelfMemo 1.0 already provided the fundamental features necessary for a lightweight, simple, and user-friendly mail reminder service. However, the team encountered some challenges in making the platform publicly accessible, reliable, and free to host.

Chapter 3

SelfMemo 2.0

SelfMemo 2.0 is an open-source, self-hosted, web-based email reminder service [Lorber et al. 2025], building on the experience of SelfMemo 1.0 [Juri et al. 2024]. SelfMemo 2.0 is a web application written in TypeScript [Microsoft 2025] and React [React 2025], using Next.js [Vercel 2025b]. This chapter explains the system requirements, system architecture, and the final set of implemented features.

3.1 System Requirements

The system requirements for SelfMemo 2.0 were formulated as follows:

- *Authentication System*: A secure authentication system must be implemented to verify user identities and ensure that each user can access only their designated data. This system will prevent unauthorized access and help distinguish between different user-specific reminders. The authentication mechanism should be username-password verification.

- *Role-Based Authorization System*: An authorization system should be incorporated to differentiate between various user roles. This system must support the creation of roles within the application to distinguish between administrative users and standard users. Administrative users should have full control over the system, including managing all user accounts and performing CRUD (Create, Read, Update, Delete) operations on all reminders across all users. Standard users, on the other hand, should only have permissions to manage their own reminders.

- *Admin Account Setup During Installation*: During the initial setup process of the application, an administrator account should be created. This admin account will serve as the primary access point for managing system-wide settings, user accounts, and application configurations. The setup process should allow the first admin account to be configured with custom login credentials to ensure security.

- *Administrative Privileges for User Management*: Any administrator account should possess the ability to perform CRUD operations on all user accounts. This includes the capability to create new users, update user information, delete users when necessary, and retrieve user details. Such functionality is essential for maintaining user records and ensuring smooth application management.

- *User Permissions for Managing Reminders*: All registered users, regardless of their role, should have the ability to perform CRUD operations on their own reminders. However, administrators have extended privileges that allow them to view and manage all reminders created by any user. Standard users should only be able to access and modify their own reminders.

- *Support for Multiple Reminder Types*: The system must support various types of reminders to accommodate different scheduling needs. The following reminder types should be available:
  - One-Time.

  - Daily.
  - Weekly.
  - N-Weekly (custom interval).
  - Monthly.
  - Yearly.
  - N-Yearly (custom interval).

  These options will provide users with flexible scheduling capabilities for their reminders.

- *Warning Reminders*: To enhance user experience and prevent missed reminders, the system should support warning reminders. These warnings should be configurable to notify users before the actual reminder is due. Users should have the option to define multiple warnings at specific intervals, such as “Send N warnings Y minutes/hours/days apart before the actual reminder.”
- *User Profile Management*: Users should have the ability to manage and update their personal account details. This includes:
  - Changing their name.
  - Updating their email address.
  - Resetting or modifying their password.

  Ensuring users can maintain up-to-date account information is crucial for security and personalization.
- *Minute-Based Reminder Intervals*: The system should allow reminders to be scheduled with precision, including minute-level intervals. This feature will provide greater flexibility in setting up reminders for time-sensitive tasks and activities.
- *Email Notifications for Reminders*: The system must include an automated email notification feature to ensure users receive their reminders at the specified times. The email notifications should include relevant details about the reminder.

3.2 Software Architecture

SelfMemo2 is a web-application based on Next.js [Vercel 2025b] (React [React 2025]), written in TypeScript [Microsoft 2025]. This ensures it has a very simple and basic software architecture of a standard web-application. A visualization of our architecture can be found in figure 3.1.

Both frontend and backend are implemented using Next.js [Vercel 2025b], in order to simplify the installation and setup of the application and make it possible to host the entire application in one repository. In contrast, the predecessor application SelfMemo 1.0 [Juri et al. 2024] has separate repositories for backend and frontend. The frontend communicates with the backend via basic HTTP requests. Common HTTP methods where employed to define the routes of the application, with standard URL design and error message codes. The SelfMemo 2.0 backend communicates via the Prisma Object-Relational-Mapper (ORM) with the database of choice, in this case a cloud-based PostgreSQL [Wikipedia 2025a] database hosted by Neon [Neon 2025]. A cron-job [cron-job.org 2025] sends a request every minute to the SelfMemo 2.0 backend in order to trigger the procedure to check whether any (warning) reminders need to be sent. If so, a connected SMTP server sends the corresponding emails. For final deployement, the hosting provider Vercel [Vercel 2025c] was used.

The following tools and technologies were used in the final implementation of SelfMemo 2.0:

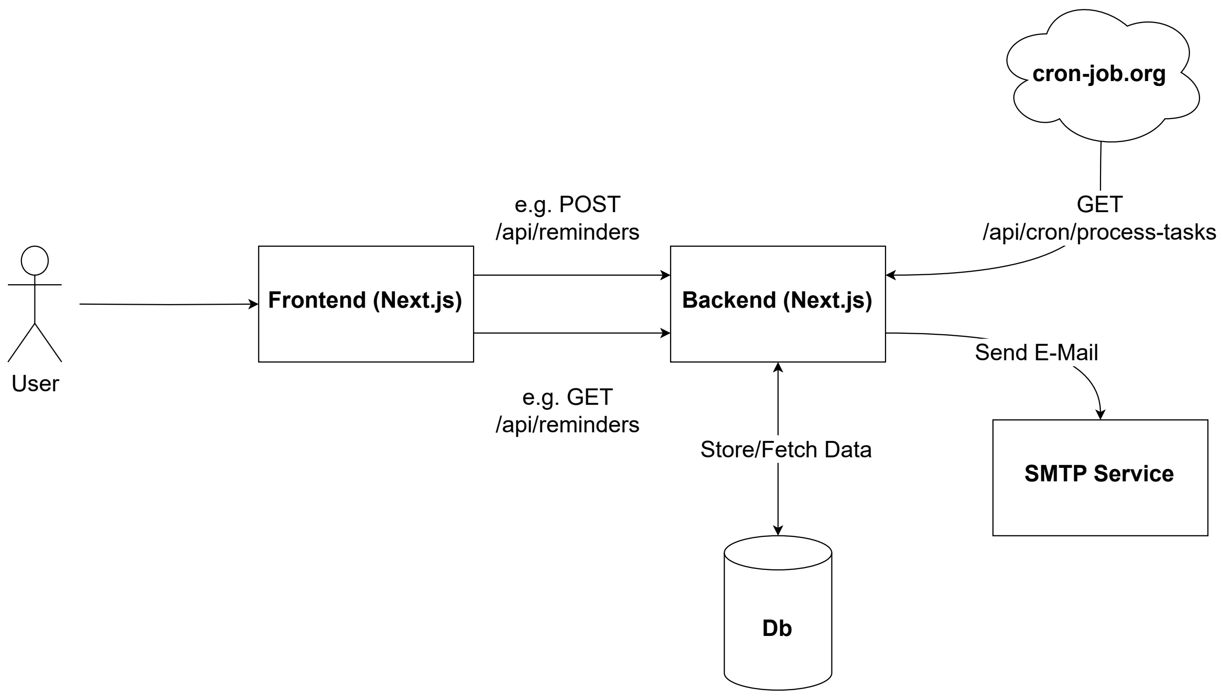


Figure 3.1: SelfMemo 2.0 has a typical web application architecture, consiting of a frontend, backend, database, cron job and SMTP connection. [Diagram Created by the author(s) of this report.]

1. GitHub [GitHub 2025] for code hosting and distribution.
2. Vercel [Vercel 2025c] for app deployment.
3. Neon [Neon 2025] for hosting the PostgreSQL database.
4. cron-job.org [cron-job.org 2025] for triggering the reminder procedure via a cron job.
5. Any kind of SMTP-Service for sending reminder notifications.

In case any of the above tools and technologies are shut down or discontinued, the following alternatives are recommended:

1. Any code hosting platform, e.g. GitLab [GitLab 2025].
2. The application can always be deployed manually. Since SelfMemo 2.0 is a standard Next.js application, the standard procedure of deploying a Next.js application can be followed [Vercel 2025a].
3. Any PostgreSQL database. This can be self-hosted or on an external provider like Supabase [Supabase 2025].
4. A cron job to trigger the reminder procedure. Cron jobs can be self-hosted on a server. GitHub Actions are another possibility, but have a minimum interval of around 5 minutes. Other cron job providers include FastCron [FastCron 2025] and EasyCron [EasyCron 2025].
5. Any SMTP service can be configured for the application to send outgoing emails.

3.3 Implemented Features

SelfMemo 2.0 includes a comprehensive set of features designed to enhance both the user and admin experiences. The following functionality was already provided in the original SelfMemo 1.0 [Juri et al. 2024], and was reimplemented in the new system architecture based on Next.js[Vercel 2025b]:

- Admin Management: Admins can create and manage users.
- User Reminder Management: Users have the ability to create, update, and manage their own reminders.
- Multiple Reminder Types: Various reminder types including one-time, daily, weekly, n-weekly, monthly, yearly and n-yearly.

The following new functionality was added to SelfMemo 2.0 to improve the overall experience and resolve some of issues of SelfMemo 1.0:

- Backend Improvements: The system's backend was optimized with the integration of cron-job.org, ensuring reliable and accurate scheduling of reminders.
- Multiple Admins: Multiple admins can now be created, allowing for better management and distribution of responsibilities.
- Warning Reminders: Users now receive warning reminders before their scheduled reminders, giving them time to adjust or cancel if needed.
- Editable Email Templates: Admins can customize the content of email notifications by editing the email templates, providing flexibility for branding or communication preferences.
- Reminder Statistics Page: A dedicated page has been added to track the statistics of reminders, offering valuable insights into usage patterns.
- Disabling Reminders: Admins and users now have the option to disable reminders, ensuring greater control over the system and its notifications.
- User Credential Management: Users can now change their credentials, including their email, name, and password, improving the overall flexibility and security of the system.

The new SelfMemo 2.0 is designed to provide a more streamlined and customizable experience for both admins and users, with better management capabilities, user-friendly controls, and improved backend stability.

Chapter 4

Setup and Deployment

This chapter explains how to run the local development environment and how to deploy the application. Please note that this software is released under MIT license and is provided as-is, without warranty of any kind.

4.1 Local Development Environment

SelfMemo 2.0 has been developed and tested in macOS Sequoia 15.1, but there should be no issues with developing on Windows or Linux. In order to run the application locally, the following prerequisites are needed:

- GitHub account.
- SMTP account, able to send secure mails via port 465.
- Basic familiarity with TypeScript and Next.js.
- Node and NPM installed.
- PostgreSQL database (either locally or in the cloud).

In summary, a basic TypeScript/JavaScript development environment must be installed and a database is needed. If a local PostgreSQL database is used, please read up on how to connect to a local database with the Prisma ORM. To test the project, Neon [Neon 2025] was used as a hosted PostgreSQL database, and the app was hosted on Vercel [Vercel 2025c].

These are the steps necessary to install a development environment for SelfMemo 2.0:

1. Fork or clone the repository, as shown in Figure 4.1.
2. `cd` into the repository
3. `cp .env.example .env`: Copy the example file.
4. Fill out the necessary environment variables, shown in Table 4.1, in the `.env` file.
5. `npm install`: Install application dependencies.
6. `npx prisma migrate dev`: Create the database schema in the connected database.
7. `npm run dev`: Start the application.
8. Visit `http://localhost:3000/api/seed` once in order to create the initial admin account.
9. The application should now be up and running.

Environment Variable	Description
`NEXTAUTH_URL`	The URL where your application runs, typically `http://localhost:3000` for Next.JS applications.
`AUTH_SECRET`	The secret needed for authentication. Can be generated at `https://generate-secret.vercel.app/32`.
`DATABASE_URL`	The connection string to the PostgreSQL database. Put the connection string inside double quotation marks in the `.env` file. See the Prisma website [Prisma 2025] for more information. In testing, the connection string to Neon [Neon 2025] had the following format: postgres://neo...ner:h...Nep...pooler.eu-central-1.aws.neon.tech/ne...bsslmode=require.
`SMTP_USER`	The SMTP username (e.g., mailUser1).
`SMTP_PASS`	The SMTP password (e.g., 123456789).
`SMTP_MAIL`	The SMTP email address (e.g., mail@example.com).
`SMTP_HOST`	The SMTP host address (e.g., mail.smtp2go.com).
`ADMIN_EMAIL`	The initial admin email address, needed to log in to the application (e.g., user@example.com).
`ADMIN_NAME`	The initial admin name (e.g., "Test User").
`ADMIN_PASSWORD`	The initial admin password (e.g., 123456789).

Table 4.1: The environment variables required to set up SelfMemo 2.0

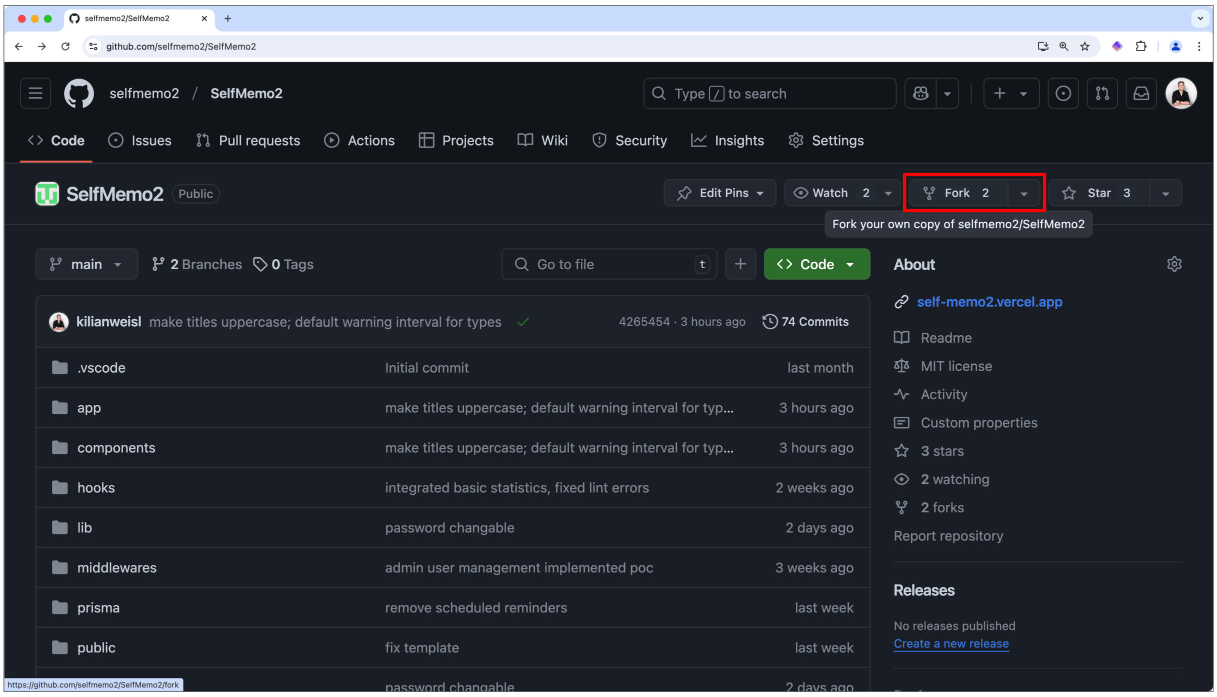
4.2 Hosting a Production Instance with Vercel

There are many alternatives for hosting a web app and an online database. One good choice is to use Vercel for deployment and Neon.tech for database management, ensuring a scalable and efficient infrastructure. This requires:

- GitHub account.
- Vercel account [Vercel 2025c].
- Neon account and database [Neon 2025].
- SMTP account, able to send secure mails via port 465.

Then, follow these steps:

1. Fork the repository on GitHub. See Figure 4.1.
2. Login (or create an account) on Vercel and import a repository. See Figure 4.2.
3. Connect your GitHub account to Vercel by clicking on Add GitHub Account. See Figure 4.3.
4. A small installation window will appear. Either grant Vercel access to all your repositories or only selected ones. See Figure 4.4.
5. Select your SelfMemo 2.0 fork and click on Import. See Figure 4.5.
6. Choose all default settings and hit Deploy. See Figure 4.6.
7. Click on Continue to Dashboard. See Figure 4.7.



kilianweisl make titles uppercase; default warning interval for types 4265454 · 3 hours ago 74 Commits

.vscode	Initial commit	last month
app	make titles uppercase; default warning interval for typ...	3 hours ago
components	make titles uppercase; default warning interval for typ...	3 hours ago
hooks	integrated basic statistics, fixed lint errors	2 weeks ago
lib	password changable	2 days ago
middlewares	admin user management implemented poc	3 weeks ago
prisma	remove scheduled reminders	last week
public	fix template	last week
	password changable	2 days ago

Figure 4.1: Step 1: Fork the repository on GitHub. [Created by the author(s) of this report.]

8. Visit Neon [Neon 2025] and create an account or connect your Vercel account. Then create a new project. See Figure 4.8.

9. You should see a popup that says “Your database is live”. Now select "Prisma" and the ".env" tab and click on Copy code. Paste this code into a text editor and copy everything inside the quotation marks from DATABASE_URL="...". See Figure 4.9.

10. Go back to Vercel $\longrightarrow$ Settings $\longrightarrow$ Environment Variables and insert the copied connection string. See Figure 4.10.

11. Create all other environment variables shown in Table 4.1. Make sure that your `NEXTAUTH_URL` is set to the production URL, which can be found on the project dashboard under Production Deployment under Domains. See Figure 4.11.

12. Go to Settings $\longrightarrow$ General and insert the build command:
`prisma generate && prisma migrate deploy && next build`. Don’t forget to save the changes. See Figure 4.12.

13. Finally, go to Deployments and redeploy your application. Your application should be set up correctly. You can now make a GET request to `https://your-application-url.com/api/seed` and then go to `https://your-application-url.com/login` to login. See Figure 4.13.

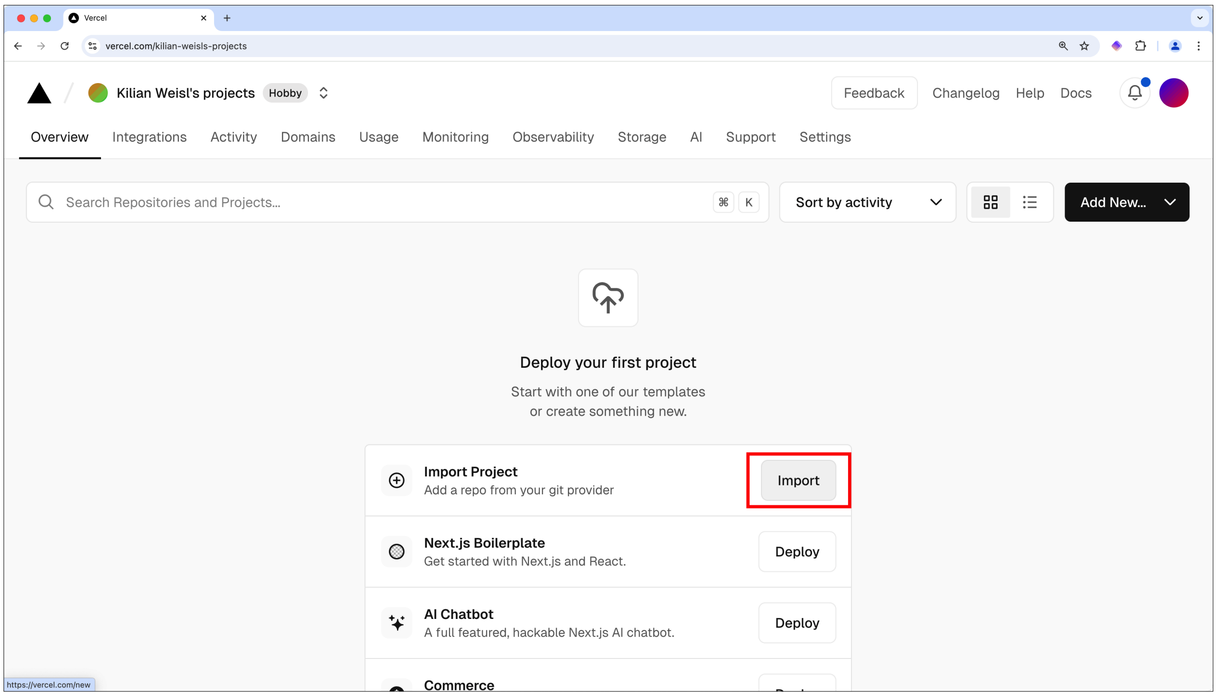

vercel.com/kilian-weisls-projects
Kilian Weisl's projects Hobby
Feedback Changelog Help Docs
Overview Integrations Activity Domains Usage Monitoring Observability Storage AI Support Settings
Search Repositories and Projects...
Sort by activity
Add New...
Deploy your first project
Start with one of our templates or create something new.
Import Project
Add a repo from your git provider
Import
Next.js Boilerplate
Get started with Next.js and React.
Deploy
AI Chatbot
A full featured, hackable Next.js AI chatbot.
Deploy
Commerce

Figure 4.2: Step 2: Import the forked repository on Vercel. [Created by the author(s) of this report.]

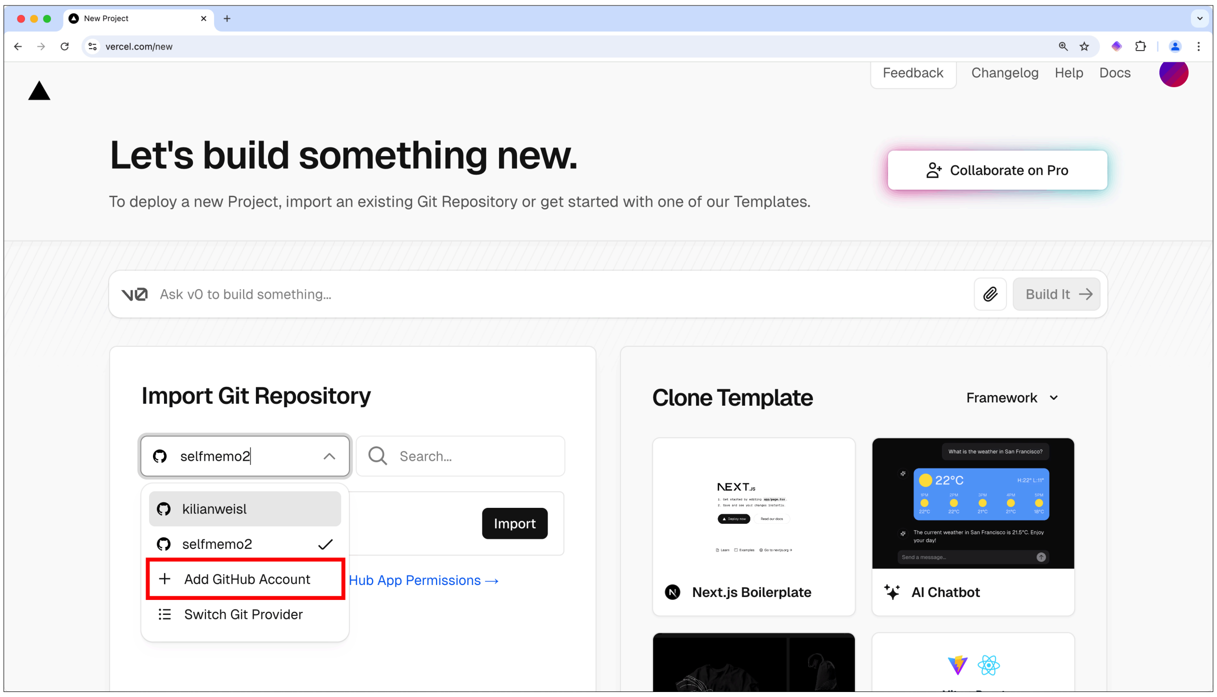
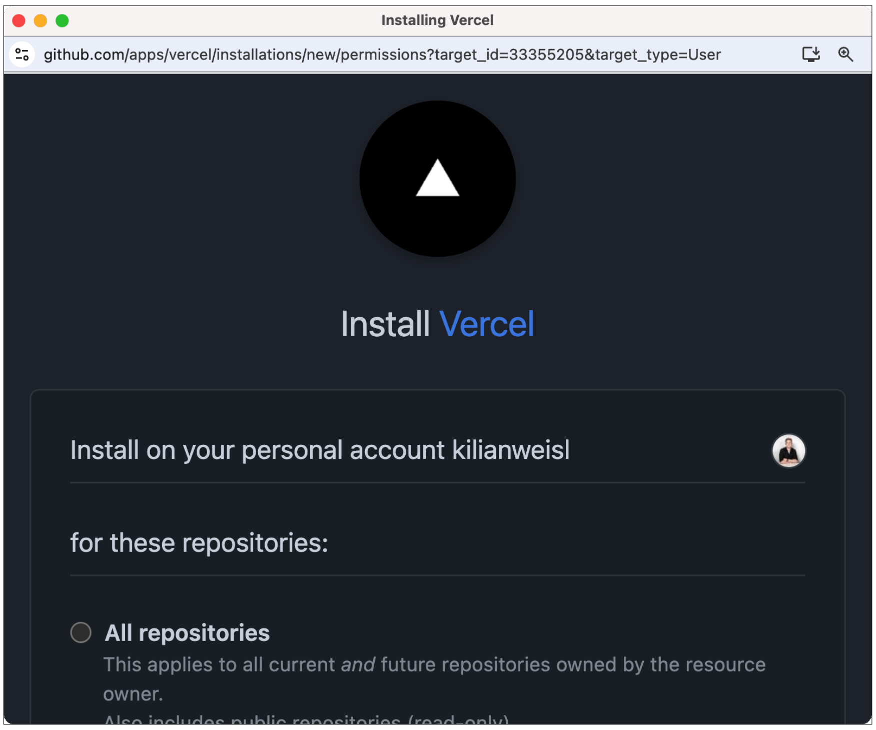


Figure 4.3: Step 3: Connect GitHub to Vercel. [Created by the author(s) of this report.]


Installing Vercel
github.com/apps/vercel/installations/new/permissions?target_id=33355205&target_type=User

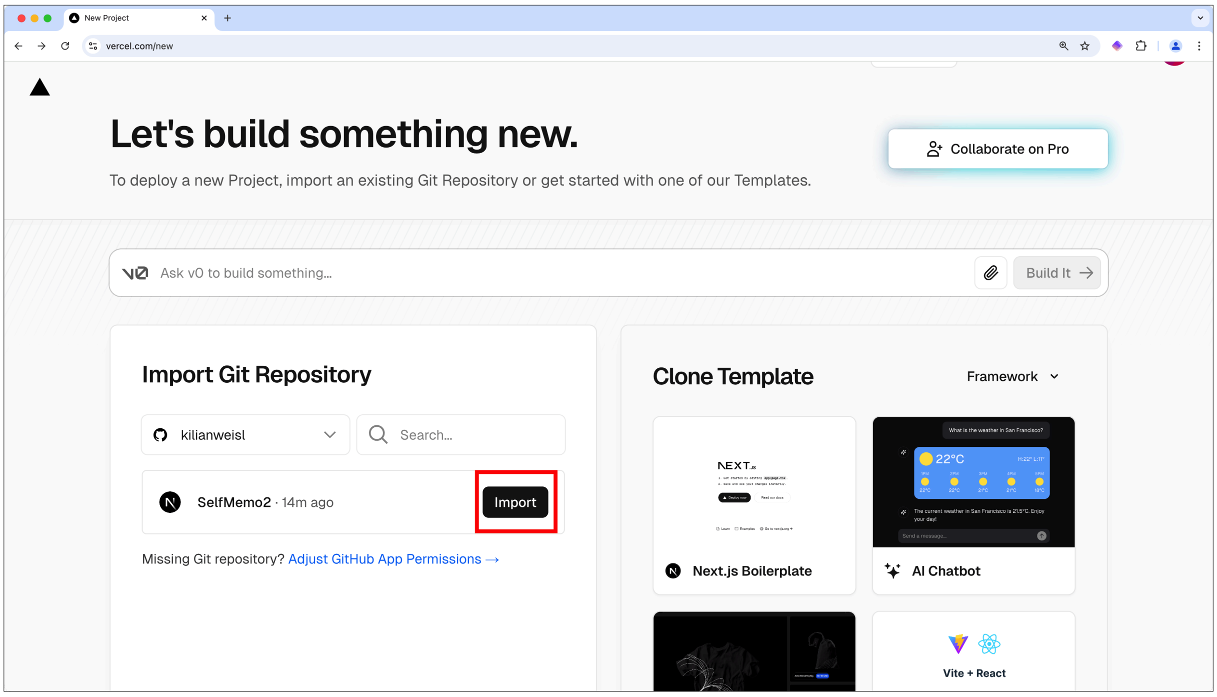
Install Vercel

Install on your personal account kilianweisl

for these repositories:

All repositories
This applies to all current *and* future repositories owned by the resource owner.

Figure 4.4: Step 4: Confirm connection from GitHub to Vercel. [Created by the author(s) of this report.]



Let's build something new.

To deploy a new Project, import an existing Git Repository or get started with one of our Templates.

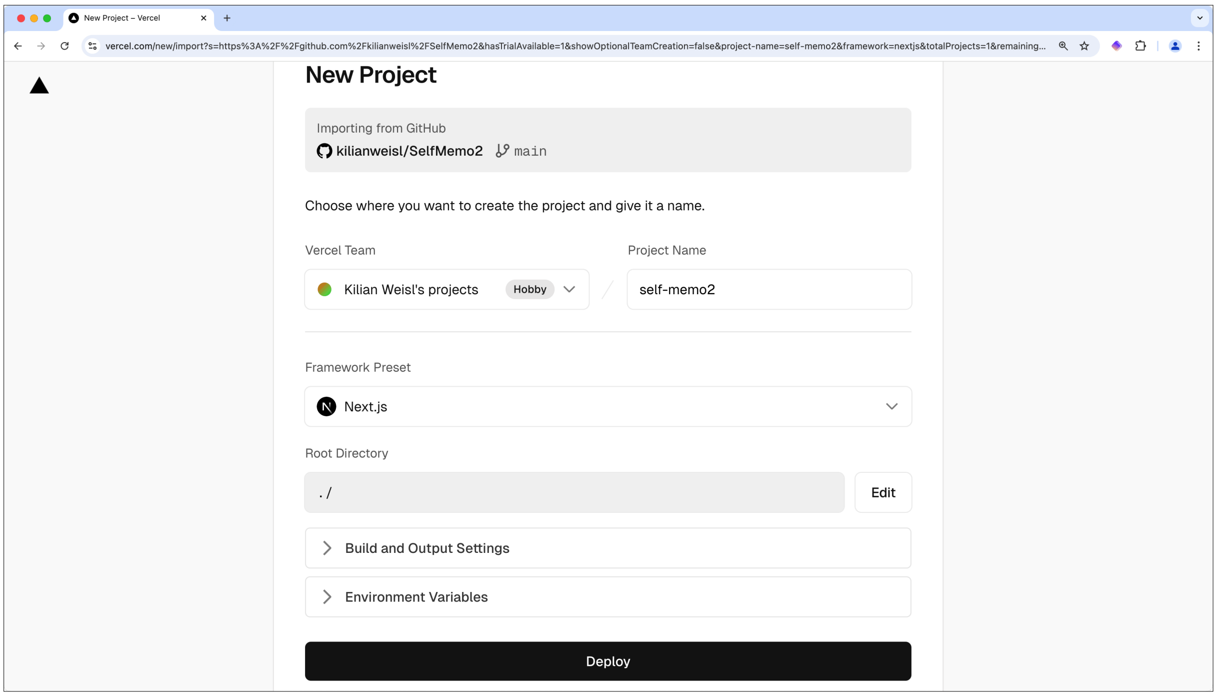
Import Git Repository

SelfMemo2 · 14m ago

Missing Git repository? Adjust GitHub App Permissions →

Clone Template

Figure 4.5: Step 5: Select SelfMemo 2.0 fork and click on Import. [Created by the author(s) of this report.]



New Project – Vercel

vercel.com/new/import?s=https%3A%2F%2Fgithub.com%2Fkilianweisl%2FSelfMemo2&hasTrialAvailable=1&showOptionalTeamCreation=false&project-name=self-memo2&framework=nextjs&totalProjects=1&remaining...

New Project

Importing from GitHub
kilianweisl/SelfMemo2 main

Choose where you want to create the project and give it a name.

Vercel Team: Kilian Weisl's projects Hobby

Project Name: self-memo2

Framework Preset: Next.js

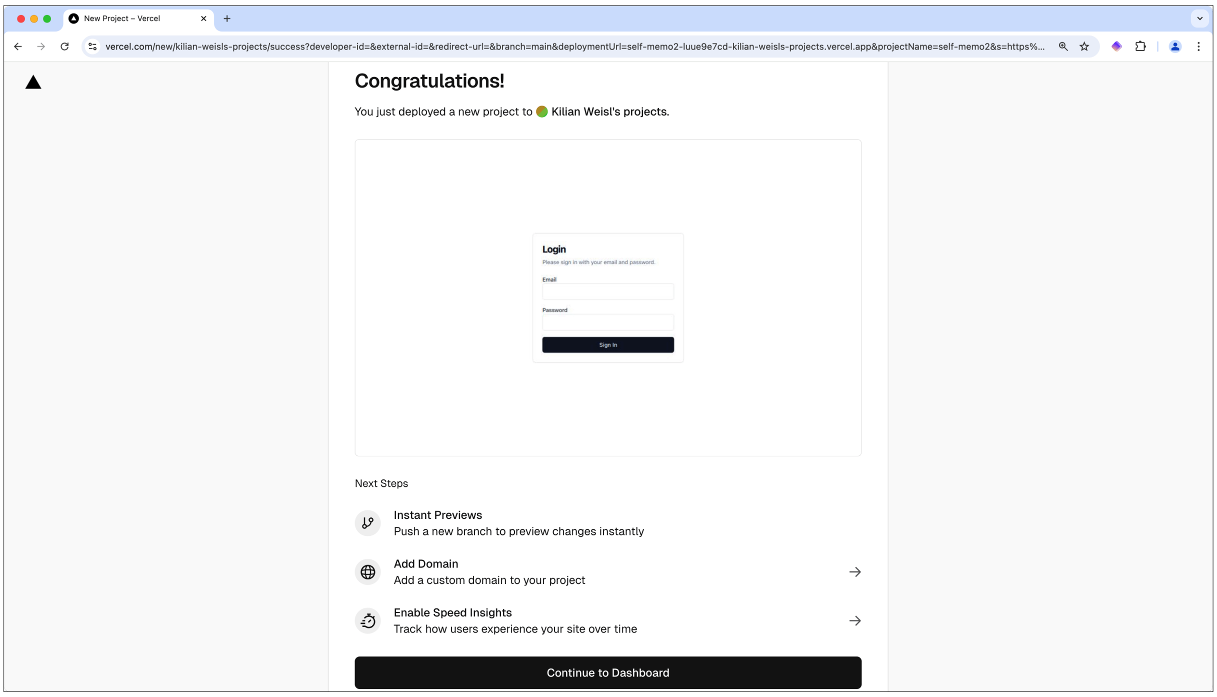
Root Directory: ./ Edit

Build and Output Settings

Environment Variables

Deploy

Figure 4.6: Step 6: Do not change default settings and deploy. [Created by the author(s) of this report.]



New Project – Vercel

vercel.com/new/kilian-weisls-projects/success?developer-id=&external-id=&redirect-url=&branch=main&deploymentUrl=self-memo2-luue9e7cd-kilian-weisls-projects.vercel.app&projectName=self-memo2&s=https%...

Congratulations!

You just deployed a new project to Kilian Weisl's projects.

Login
Please sign in with your email and password.
Email
Password
Sign in

Next Steps

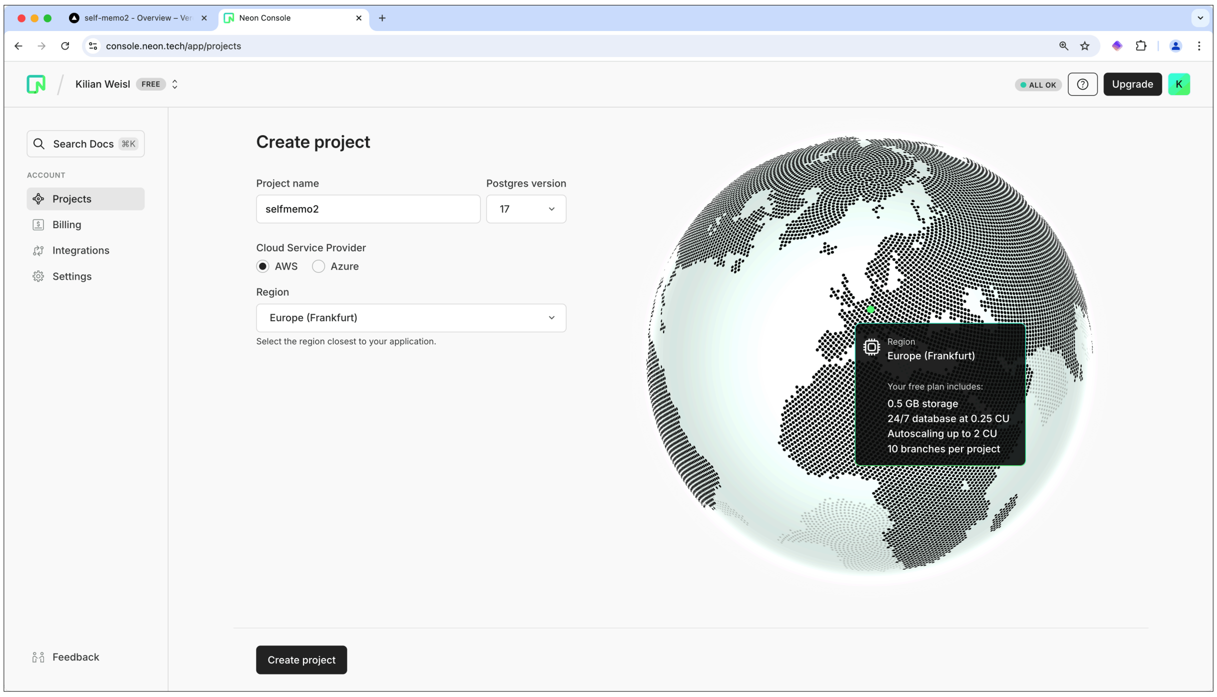
Instant Previews
Push a new branch to preview changes instantly

Add Domain
Add a custom domain to your project

Enable Speed Insights
Track how users experience your site over time

Continue to Dashboard

Figure 4.7: Step 7: Click on Continue to Dashboard. [Created by the author(s) of this report.]



Create project

Project name: selfmemo2

Postgres version: 17

Cloud Service Provider: [● AWS] ○ Azure

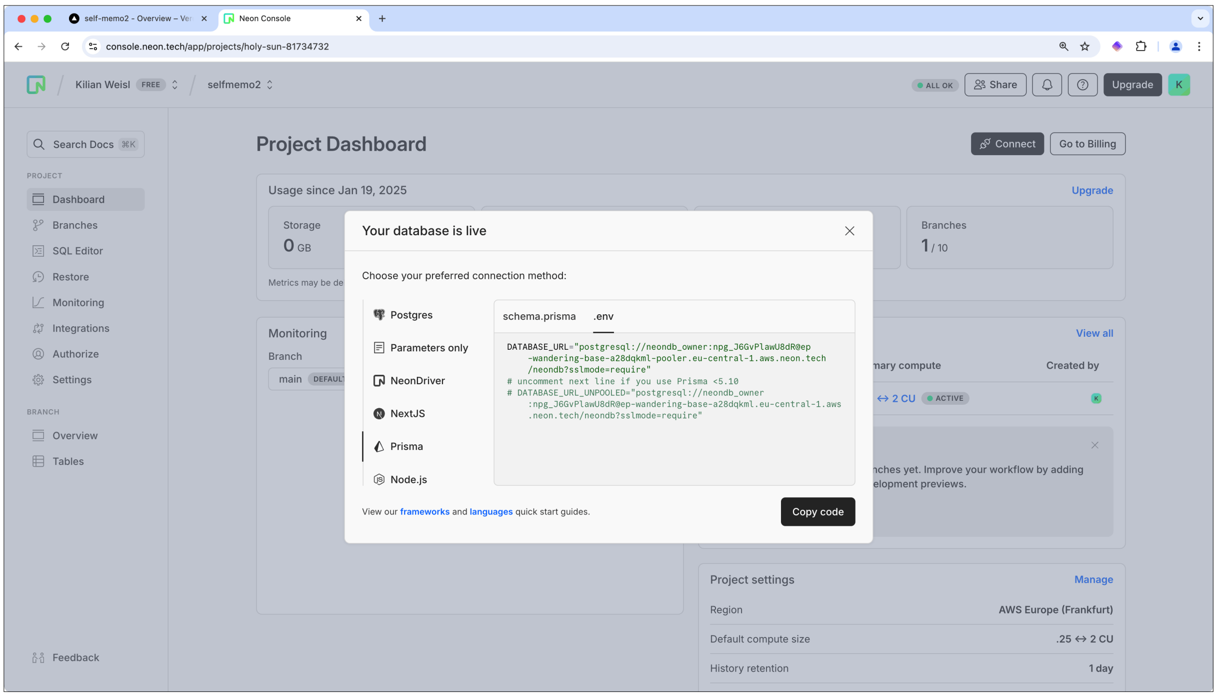
Region: Europe (Frankfurt)

Select the region closest to your application.

Region
Europe (Frankfurt)

Your free plan includes:
0.5 GB storage
24/7 database at 0.25 CU
Autoscaling up to 2 CU
10 branches per project

Figure 4.8: Step 8: Create a new Neon project. [Created by the author(s) of this report.]



Usage since Jan 19, 2025

Storage 0 GB

Branches 1 / 10

Metrics may be de

Monitoring

Branch: main DEFAULT

mary compute: ↔ 2 CU ACTIVE

Created by

nches yet. Improve your workflow by adding
elopment previews.

Your database is live

Choose your preferred connection method:

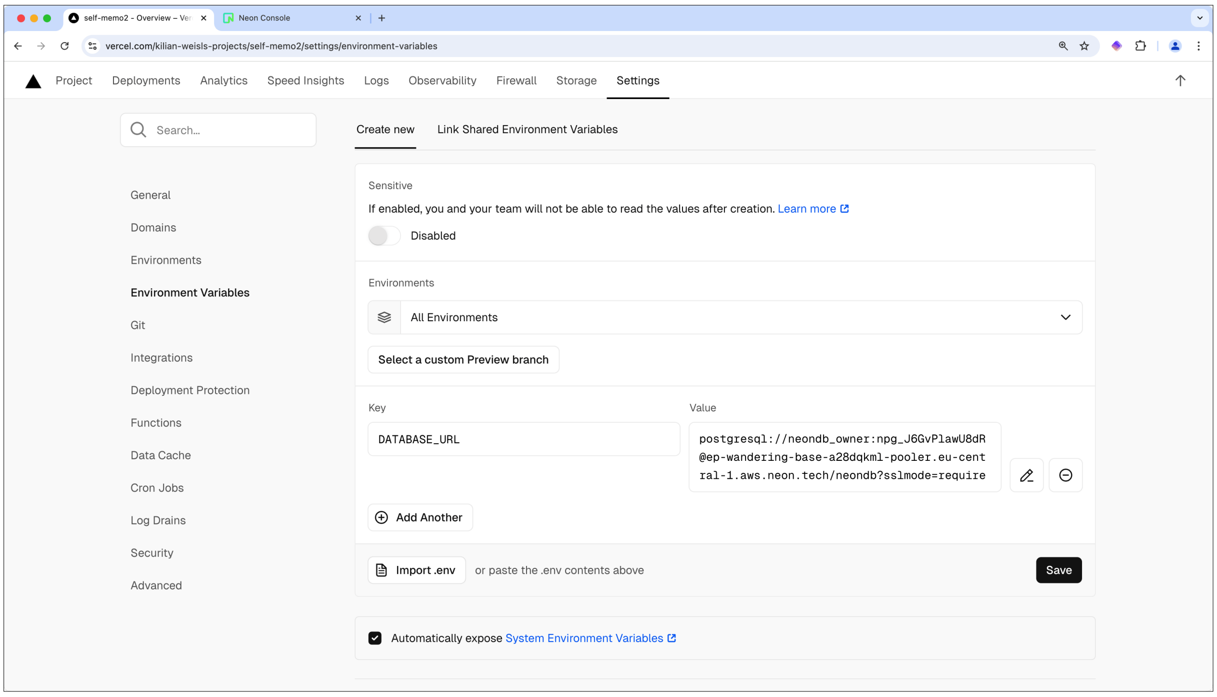
```
DATABASE_URL="postgresql://neondb_owner:npg_J6GvPlawU8dR@ep
    -wandering-base-a28dqkml-pooler.eu-central-1.aws.neon.tech
    /neondb?sslmode=require"
# uncomment next line if you use Prisma <5.10
# DATABASE_URL_UNPOOLED="postgresql://neondb_owner
    :npg_J6GvPlawU8dR@ep-wandering-base-a28dqkml.eu-central-1.aws
    .neon.tech/neondb?sslmode=require"
```

View our frameworks and languages quick start guides.

Project settings

Region	AWS Europe (Frankfurt)
Default compute size	.25 ↔ 2 CU
History retention	1 day

Figure 4.9: Step 9: Copy the database URL. [Created by the author(s) of this report.]



Sensitive

If enabled, you and your team will not be able to read the values after creation. Learn more

Disabled

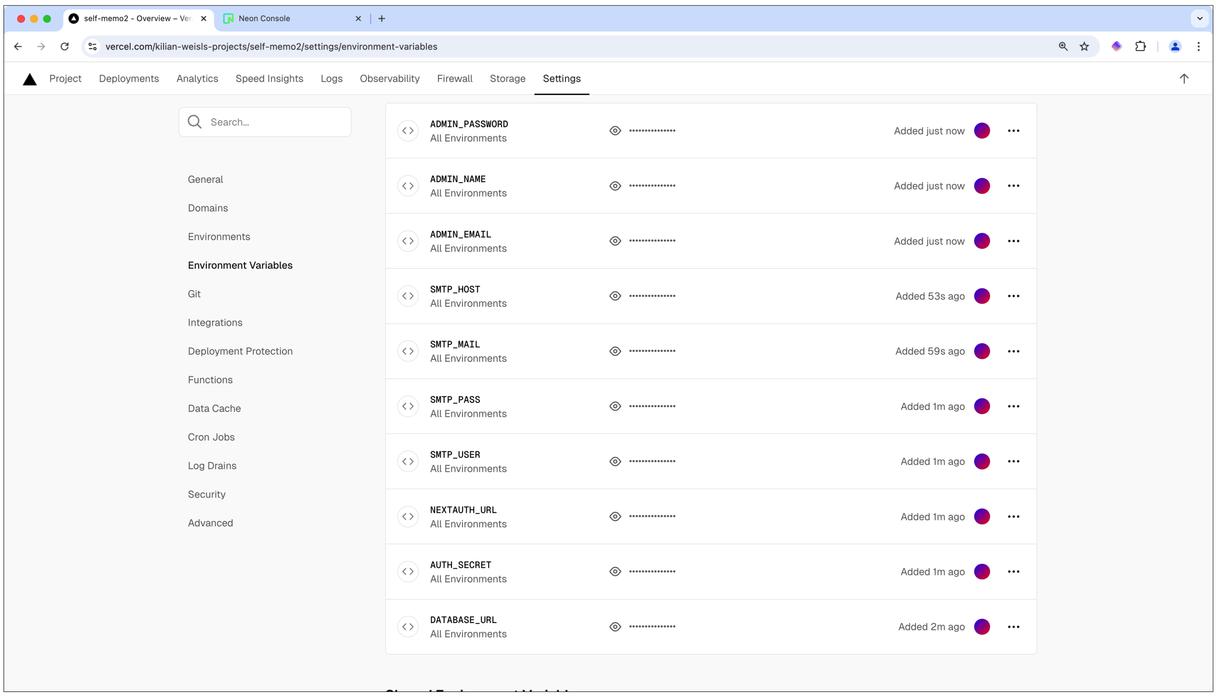
Environments: All Environments

Key	Value
DATABASE_URL	postgresql://neondb_owner:npg_J6GvPlawU8dR@ep-wandering-base-a28dqkml-pooler.eu-central-1.aws.neon.tech/neondb?sslmode=require

or paste the .env contents above

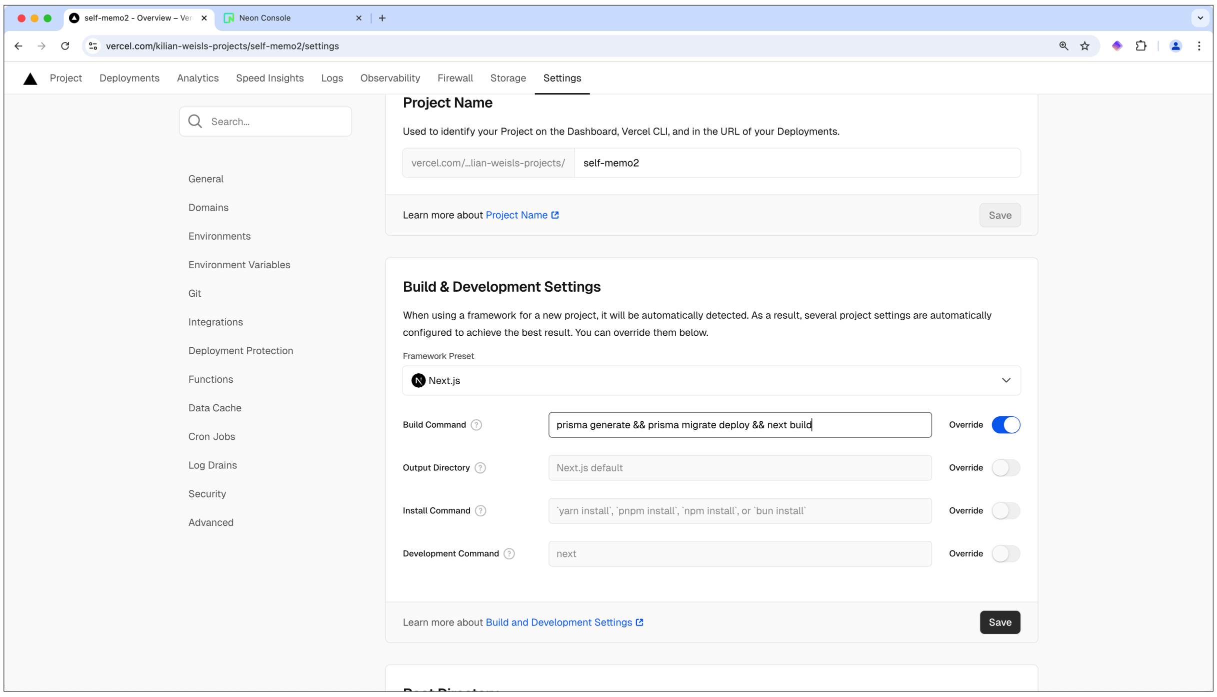
✓ Automatically expose System Environment Variables

Figure 4.10: Step 10: Create environment variable. [Created by the author(s) of this report.]



ADMIN_PASSWORD	All Environments	•••••••••••••••	Added just now
ADMIN_NAME	All Environments	•••••••••••••••	Added just now
ADMIN_EMAIL	All Environments	•••••••••••••••	Added just now
SMTP_HOST	All Environments	•••••••••••••••	Added 53s ago
SMTP_MAIL	All Environments	•••••••••••••••	Added 59s ago
SMTP_PASS	All Environments	•••••••••••••••	Added 1m ago
SMTP_USER	All Environments	•••••••••••••••	Added 1m ago
NEXTAUTH_URL	All Environments	•••••••••••••••	Added 1m ago
AUTH_SECRET	All Environments	•••••••••••••••	Added 1m ago
DATABASE_URL	All Environments	•••••••••••••••	Added 2m ago

Figure 4.11: Step 11: Create other environment variables. [Created by the author(s) of this report.]



vercel.com/kilian-weisls-projects/self-memo2/settings

Project Name

Used to identify your Project on the Dashboard, Vercel CLI, and in the URL of your Deployments.

vercel.com/...lian-weisls-projects/ self-memo2

Learn more about Project Name

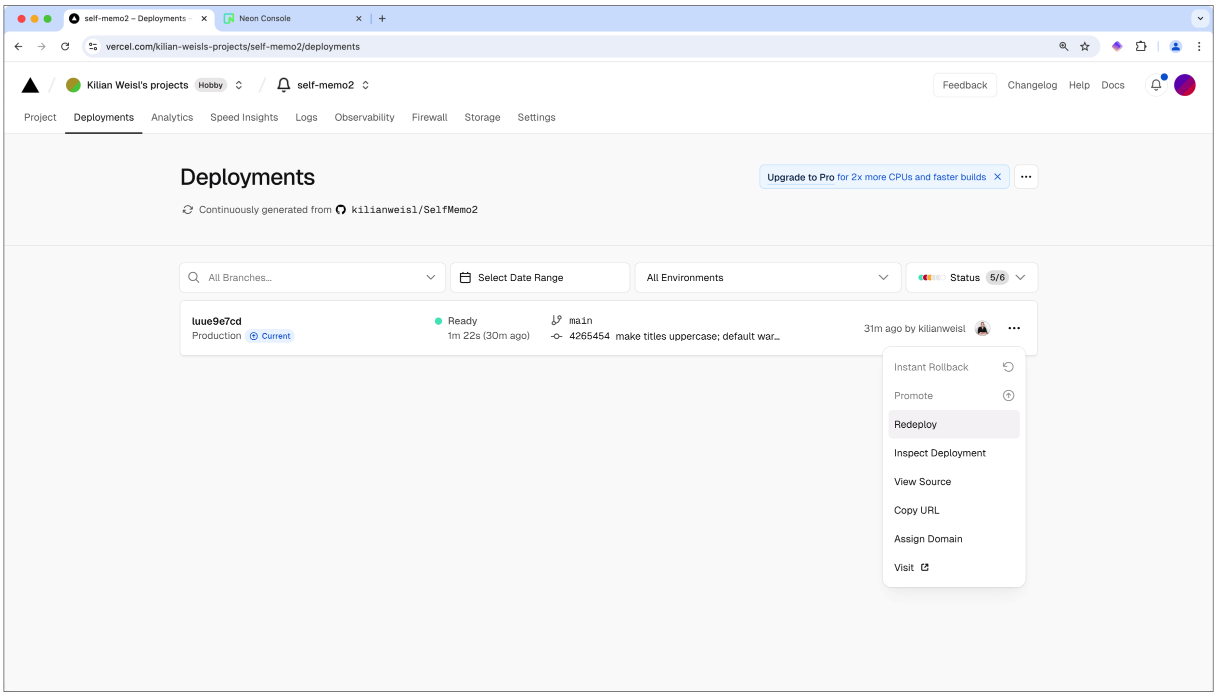
Build & Development Settings

When using a framework for a new project, it will be automatically detected. As a result, several project settings are automatically configured to achieve the best result. You can override them below.

Framework Preset	Next.js	
Build Command	prisma generate && prisma migrate deploy && next build	Override: on
Output Directory	Next.js default	Override: off
Install Command	`yarn install`, `pnpm install`, `npm install`, or `bun install`	Override: off
Development Command	next	Override: off

Learn more about Build and Development Settings

Figure 4.12: Step 12: Update the build command in Vercel. [Created by the author(s) of this report.]



vercel.com/kilian-weisls-projects/self-memo2/deployments

Deployments

Continuously generated from kilianweisl/SelfMemo2

luue9e7cd Production · Current	Ready 1m 22s (30m ago)	main 4265454 make titles uppercase; default war...	31m ago by kilianweisl

Figure 4.13: Step 13: Redeploy the application. [Created by the author(s) of this report.]

Chapter 5

Concluding Remarks

This report described SelfMemo 2.0, an open-source, self-hosted, web-based email reminder service. The software is freely available on GitHub [Lorber et al. 2025].

SelfMemo 2.0 is functional, but could be improved or extended in numerous ways:

- *Import and Export for Reminders*: One major improvement would be to add the ability for users to export and import reminders in common formats like CSV or JSON. This would enable easier data transfer between platforms and allow users to back up or migrate their reminders. Additionally, providing a customizable field mapping feature during imports would help ensure that data is imported correctly, particularly if users are moving from a different reminder system.

- *Visualize Reminders in Calendar*: A significant feature improvement would be the introduction of a visual calendar view for reminders. This would offer a more intuitive way to track upcoming events. Integration with external calendars, such as Google Calendar, Outlook, and Apple Calendar, would further enhance usability by enabling users to sync their SelfMemo 2.0 reminders with their existing scheduling tools. Additionally, implementing drag-and-drop functionality within the calendar view would let users easily reschedule reminders.

- *Customizable Email Notification Templates*: Currently, email reminders follow a generic format only adaptable by the system admin. To improve user experience, allowing users to customize the email notification templates would be a useful feature. This could include modifications to the subject line, body content, and style of the email, using a simple WYSIWYG (What You See Is What You Get) editor [Wikipedia 2025b]. For those who prefer not to create their own templates, predefined options for different styles could be provided.

- *Adapting Reminders to Current Timezone*: Time zone management is crucial for users in different locations. Automatically detecting a user's local time zone and adjusting reminder times accordingly would greatly improve accuracy. Users should also be able to manually set their time zone if needed. For frequent travellers, enabling the ability to convert reminder times when crossing time zones would ensure they never miss an important task. Additionally, ensuring reminders are displayed in the correct time zone for all recipients, especially in group settings, would reduce confusion. Finally, email notifications should clearly display the time zone in which a reminder is set to ensure transparency.

Bibliography

Andrews, Keith [2025]. *Information Architecture and Web Usability*. 25 Feb 2025. https://courses.isds.tugraz.at/iaweb (cited on page 1).

Apple [2025]. *Reminders*. 01 Feb 2025. https://icloud.com/reminders (cited on page 3).

cron-job.org [2025]. *Scheduled Execution of Your Websites and Scripts*. 31 Jan 2025. https://cron-job.org/ (cited on pages 8–9).

EasyCron [2025]. *EasyCron*. 03 Feb 2025. https://easycron.com/ (cited on page 9).

FastCron [2025]. *FastCron*. 03 Feb 2025. https://fastcron.com/ (cited on page 9).

GitHub [2025]. *GitHub*. 03 Feb 2025. https://docs.github.com/en/get-started/using-git/about-git (cited on page 9).

GitLab [2025]. *GitLab*. 03 Feb 2025. https://about.gitlab.com/ (cited on page 9).

Google [2025]. *Keep*. 01 Feb 2025. https://keep.google.com/ (cited on page 3).

JJ Zone [2025]. *MemoToMe*. 01 Feb 2025. http://memotome.com/ (cited on pages 1, 4).

Juri, Joseph, Corentin Lebleu, Sanja Marinkovic and Luka Serdar [2024]. *SelfMemo*. 06 Dec 2024. https://github.com/Self-Memo (cited on pages 5, 7–8, 10).

Lorber, Niklas, Maria Seiser and Kilian Weisl [2025]. *SelfMemo2*. 03 Feb 2025. https://github.com/selfmemo2/SelfMemo2 (cited on pages 1, 7, 21).

Microsoft [2025]. *TypeScript*. 31 Jan 2025. https://typescriptlang.org/ (cited on pages 7–8).

Neon [2025]. *Neon DB*. 31 Jan 2025. https://neon.tech/ (cited on pages 1, 8–9, 11–13).

Prisma [2025]. *Prisma ORM - PostgreSQL Connection URL*. 03 Feb 2025. https://prisma.io/docs/orm/overview/databases/postgresql#connection-details (cited on page 12).

React [2025]. *React*. 31 Jan 2025. https://react.dev/ (cited on pages 7–8).

Supabase [2025]. *Supabase*. 03 Feb 2025. https://supabase.com/ (cited on page 9).

Vercel [2025a]. *Deploying Next.js Applications*. 2025. https://nextjs.org/docs/pages/building-your-application/deploying (cited on page 9).

Vercel [2025b]. *Next.js*. 31 Jan 2025. https://nextjs.org/ (cited on pages 7–8, 10).

Vercel [2025c]. *Vercel*. 31 Jan 2025. https://vercel.com/ (cited on pages 1, 8–9, 11–12).

Wikipedia [2025a]. *PostgreSQL*. 31 Jan 2025. https://wikipedia.org/wiki/PostgreSQL (cited on page 8).

Wikipedia [2025b]. *WYSIWYG*. 01 Feb 2025. https://wikipedia.org/wiki/WYSIWYG (cited on page 21).