**Building CalPal – My Take on Productivity Apps**

**Introduction**

This blog will be a detailed breakdown of why I built CalPal, how I built it, what features I worked on, and the problems I had to face in the process.

**Why did I build CalPal?**

**Personal pain-points and peeves**

It’s always a pain to keep track of everything you want to get done — life’s messy, and remembering things by memory alone just doesn’t cut it.

I wanted a simple place to jot down tasks, organized by day and time, without the clutter of bloated productivity apps.

**Struggles in aligning group schedules**

But beyond personal productivity, I kept running into a deeper problem: planning with other people. You’ve probably been there — trying to organize a group activity, a study session, or a trip, only to realize no one's schedules line up, and everything falls apart.

CalPal was built to solve exactly that.

It’s a clean, well-designed app where you can manage your tasks and align plans with others. Create private groups, see when your friends are free (without oversharing your calendar), and plan your next trip to Goa 😭 without the usual scheduling chaos.

**How did I create CalPal ?**

**Initial broad plan**

While I was brainstorming CalPal, I was also diving into the MERN stack — and it felt like a great fit that could lead to a fun learning experience.

It’s a popular full-stack combo for building dynamic web apps, and with a few adjustments, it can even behave like a PWA (Progressive Web App) — which means CalPal could feel like a native app on mobile devices too.

MERN is also incredibly well-supported, with tons of plugins, tools, and community examples to learn from.

**Frontend plan**

For the frontend, I chose React.js, using ShadCN component libraries for fast development. But instead of using them out of the box (like most people do), I heavily customized the components to match the neo-brutalist aesthetic I had in mind.

Most apps that use ShadCN tend to look the same — I wanted CalPal to feel unique while still benefiting from ShadCN’s speed and accessibility. This balance between style and structure was key to the UI.

**Backend, Database and Authentication**

On the backend, I used Express.js to build my RESTful APIs — it’s lightweight, flexible, and gets out of your way. For the database, I went with MongoDB because of its schema flexibility and ease of use.

For authentication, I integrated Firebase Auth — it cleanly separates login data (emails and passwords) from app-specific user data stored in MongoDB. This approach keeps things modular and secure — no passwords ever touch the main database.

**Design Goal**

In terms of design, I aimed for something that felt clean, sharp, and a little bold. I think the neo-brutalist theme, combined with responsive layout support for both desktop and mobile, gives CalPal a distinct identity without sacrificing usability.

**What are the cool standout features?**

**Privacy-first approach**

One of CalPal’s most thoughtful features is its privacy-first group system. There are two types of groups you can create:

Private groups are completely hidden from other users unless you’re a member. These are perfect for close circles — study groups, travel plans, or personal accountability squads. No one outside the group can even see that it exists.

Invite-only groups are visible to others but can’t be joined freely. When creating an invite-only group, the owner selects members, and each of them receives an invite. They can accept or decline it — no automatic additions, no surprises.

And if someone discovers an invite-only group and wants to join? They must send a join request, which the group owner can either accept or reject. This creates clear social boundaries and avoids awkward group spam or unwanted additions.

But CalPal doesn’t stop at just who can join — it also protects what others see.

Each group has a shared calendar view, where members can see when others are free — but not what they’re doing. No task details are ever shared. You just know when someone is busy, and when they’re free, so you can schedule accordingly — without oversharing.

It's collaborative, but respectful. It’s social, but secure.

**Dynamic App design for all screens**

CalPal is also fully optimized for mobile use — not just responsive, but intentionally tailored for candybar-style phones.

The layout adjusts to different screen sizes with purpose. On desktop, you’ll see a traditional sidebar navigation. But on mobile, the app transforms into a bottom navigation bar, improving reachability, comfort, and flow — without compromising the app’s personality or usability.

Whether you open it on a phone or a laptop, CalPal always feels like it belongs.

**What problems did I face while making the app?**

**Balancing Responsiveness and Complexity**

From the start, I wanted CalPal to feel real-time and responsive — without hammering the backend with excessive API calls. Striking that balance was a fun challenge. I had to carefully design the state management so that updates felt immediate but weren’t constantly querying the server.

One of the trickier architectural decisions was using Firebase for authentication and MongoDB for user data — keeping login credentials and app data separate for better security. But this created a double-handshake problem: the app had to wait for both Firebase and MongoDB to respond before loading user details. That led to a few race conditions and inconsistent states at login.

Eventually, I solved it by introducing controlled loading states and using context APIs to manage global data more intelligently. Context has been a big part of my past projects too — and once again, it proved crucial in keeping the UI synced and clean across pages.

**Deprecated libraries and hosting**

Hosting brought its own headaches. The calendar library I used turned out to be deprecated in React 19, which forced me to rewrite it using the new syntax. The new library had just been released, and the documentation was sparse, so I had to piece things together as I went. Not the most fun I’ve had, but definitely a good learning moment.

Then came deployment realities. The free-tier backend and database were painfully slow at times, especially after cold starts. This taught me how important it is to build thoughtful loading experiences — not just spinners, but real feedback to the user about what’s happening.

There are still a few quirks I’d like to clean up in the future. But for now, I’m proud of how far CalPal has come — and I plan to keep iterating quietly in the background while I explore new projects.

**What’s next for CalPal?**

CalPal isn’t finished — it’s just getting started. I plan to continue expanding it with new features every few months, alongside other projects I work on.

**Personalizing and Gamifying experiences**

One area I’m especially interested in is building more personalized and gamified experiences. Tools that make users feel emotionally connected — not just functionally productive — can go a long way. I’ve been studying how platforms like Duolingo use a mascot to boost user retention and habit-building, and that direction really resonates with me.

So yes — CalPal’s mascot is getting more love, the favicon and app icons already feature him.

I want to introduce a point system where users earn "grow points" by completing tasks, which they can use to customize the mascot. The goal is to make productivity feel fun, not forced.

In the future, I’d love to add mascot animations that react when you complete tasks or hit milestones. Those little celebratory moments can turn dry checklists into meaningful rituals — something that makes the app feel like it’s cheering for you.

**Quality of Life and Clear Indications**

Beyond aesthetics, I’m also focused on backend improvements. I plan to implement rate limiting and better loading state handling to improve responsiveness, especially considering the current limitations of free-tier backend hosting. Reducing unnecessary API calls and giving users proper feedback during interactions will make the experience smoother and less error-prone.

Finally, I want CalPal to be something that grows with its users — and that includes listening. I’ll be open to suggestions and ideas from anyone who uses the app or wants to contribute. You can always reach out to me on my socials.

**Conclusion**

Building CalPal was a deeply rewarding experience. It helped me strengthen my fundamentals in the MERN stack and web development as a whole, while also giving me the freedom to create something I genuinely connect with.

Choosing to build a project that felt personal — rather than just following a tutorial — was one of the best decisions I’ve made as a developer so far.

Thanks for reading and being part of this breakdown of how CalPal came to life. Here’s to more projects, more challenges, and more exciting learning experiences ahead!