TECH TIP — BEGINNER/INTERMEDIATE

What a .env File Is and Why You Never Commit It to GitHub

From Nolan Law Firm — nemolegal.com/tools

The Problem It Solves

Every app needs secrets — API keys, database passwords, access tokens. You can't hardcode them in your code (someone will see them). You can't leave them out entirely (the app won't run). The `.env` file is the standard solution: secrets live in one file, separate from your code, that never gets committed to version control.

What It Looks Like

```
# .env file — one variable per line, KEY=value format

DATABASE_URL=postgresql://user:password@localhost:5432/mydb

OPENAI_API_KEY=sk-abc123

STRIPE_SECRET_KEY=sk_live_xyz789

APP_SECRET=somelong randomstring

DEBUG=false
```

How Your Code Reads It

```
# Python — using python-dotenv library:

from dotenv import load_dotenv

import os


load_dotenv() # reads .env and loads variables into environment

db_url = os.getenv("DATABASE_URL")
```

```
api_key = os.getenv("OPENAI_API_KEY")


# Node.js — using dotenv package:

require('dotenv').config()

const dbUrl = process.env.DATABASE_URL
```

The .gitignore Rule — Non-Negotiable

Your project must have a `.gitignore` file with `.env` in it. This tells Git to never track that file. If you don't do this, the first time you push to GitHub your secrets are public.

```
# .gitignore — add these lines:

.env

.env.local

.env.production

*.env
```

The .env.example Pattern

Commit a file called `.env.example` with the variable names but no real values. Anyone setting up the project knows what variables they need without seeing your secrets.

```
# .env.example — safe to commit, shows what's needed

DATABASE_URL=

OPENAI_API_KEY=

STRIPE_SECRET_KEY=
```

If you accidentally committed a .env file: Run git rm --cached .env then add .env to .gitignore and commit. Rotate every secret that was in that file immediately — even if the repo is private. Private repos can become public.

On a Server — Beyond .env Files

In production, secrets often go directly into the environment rather than a file — via systemd unit files, Docker Compose env_file directives, or a secrets manager. The .env file pattern is primarily for local development. On servers, prefer environment variables set at the system or service level.

> **Using AI to Help With This** You don't have to fully understand this to use it. Here are prompts that work:

```
"Set up my [Python/Node] project to use a .env file for secrets. Show me exactly what files to create and what to add to .gitignore."
```

```
"I need to pass secrets to my Docker container without hardcoding them. How do I use an env_file in docker-compose.yml?"
```

```
"I think I committed a .env file to GitHub. Walk me through exactly what to do right now."
```
