Tutorial: Setting Up Environment Keys on macOS and Windows

1. Why Use Environment Variables?
- Security: Avoid hard-coding secrets in your code
- Portability: Same scripts work across machines/environments
- Flexibility: Override or extend settings without changing code

2. On macOS
2.1 Identify Your Shell: `echo $SHELL`
2.2 Edit Shell Profile:
  - zsh: edit ~/.zshrc
  - bash: edit ~/.bash_profile or ~/.bashrc
2.3 Add Your Variable:
  export MY_API_KEY="abcd1234"
2.4 Apply Changes: `source ~/.zshrc` or `source ~/.bash_profile`
2.5 Verify: `echo $MY_API_KEY`
Optional for GUI Apps: `launchctl setenv MY_API_KEY "abcd1234"`

3. On Windows
3.1 GUI:
  - Start → Edit the system environment variables → Environment Variables…
  - New User Variable: Name=MY_API_KEY, Value=abcd1234
3.2 Command Line:
  - cmd.exe: `setx MY_API_KEY "abcd1234"`
  - PowerShell:
    [Environment]::SetEnvironmentVariable("MY_API_KEY", "abcd1234", "User")
Verify:
  - cmd.exe: `echo %MY_API_KEY%`
  - PowerShell: `echo $Env:MY_API_KEY`

4. Using .env Files (Cross-Platform)
Create `.env` in project root:
  MY_API_KEY=abcd1234
Python:
  from dotenv import load_dotenv
  load_dotenv()
  import os; os.getenv("MY_API_KEY")
Node.js:
  require('dotenv').config()
  process.env.MY_API_KEY

5. Recap
- macOS: export in shell profile + source
- Windows: GUI or CLI (`setx`/`SetEnvironmentVariable`)
- .env files: cross-platform loader

End of Tutorial