# Why Agile?

Honestly, Agile just makes sense for this kind of project. Stuff changes fast—sometimes a tool doesn’t work as expected, or you find a better API halfway through. With Agile, you don’t have to stick to a strict plan from day one. You can build things step by step, test as you go, and fix problems right away instead of waiting till the end. Plus, it’s way easier to get feedback and improve things each week. Basically, Agile keeps things flexible and helps avoid last-minute surprises.

# Week-wise Plan

## Week 3 & 4: Tool Hunt & Research

* Check out open-source recon tools (Censys, Shodan, Amass, Sublist3r, Nmap, etc.)
* Note what each tool can do: subdomains, open ports, running services, tech stack, server info, etc.
* Explore APIs for each tool—how to access, what data is available, any setup needed
* Make a shortlist of the best tools and gather API docs
* Testing: Try out basic API calls with sample inputs to see what kind of data comes back

## Week 5 & 6: Basic App Skeleton

* Build a simple frontend for entering a URL or IP
* Set up backend logic to receive and validate input
* Organize project folders and push everything to GitHub
* Aim for a basic prototype where submitting a URL/IP doesn’t break anything
* Testing: Test form input, backend response, and error handling with different types of URLs/IPs

## Week 7 & 8: Input Prep

* Add DNS resolution, reverse DNS, and any tweaks needed for tool input formats
* Make sure backend can handle and prep any user input for the APIs
* Testing: Run multiple test cases to check if input is correctly transformed for each tool

## Week 9 & 10: Output Mashup

* Collect results from all integrated tools
* Write scripts to combine outputs, remove duplicates, and standardize data
* End up with a single, clean data structure for all recon info
* Testing: Compare outputs from different tools, check for missing or duplicate data, validate merged results

## Week 11: Info Sorting

* Classify gathered info into high, medium, and low impact (asset classification)
* Use rules or logic for sorting, and document how stuff is being classified
* Testing: Review classification logic with sample data, check if assets are being sorted correctly

## Week 12: Results & Downloads

* Make results easy to read—tables, maybe charts if there’s time
* Add options to download results as .csv or .pdf
* Testing: Test download functionality, file formatting, and make sure exported data matches what’s shown

## Week 13: Project Presentation

* Prepare slides covering the whole project — intro, tools used, how it works, challenges, and final output
* Add screenshots or short demos of the tool in action
* Practice explaining each part clearly (especially tool integration and classification logic)
* Get feedback from friends or classmates before the final presentation
* Testing: Do a full run of the tool and demo to make sure everything works smoothly before presenting

## Extras

* Keep notes and screenshots for the report
* Test each part with sample data
* Regular commits and backups on GitHub

Note: This plan has been enhanced and rewritten with the help of GPT-5, but all the core planning, ideas, and development work belong to the project team. This is not the final version—changes may still happen after any week if things don’t go as expected.