Mac Admin Project Plan: [Project Name]

> **Planning Philosophy**: Think through the design before writing code. Make decisions explicit, not implicit.

Author: [Your Name]
Date: [YYYY-MM-DD]
Version: 0.1.0
Status: Draft | Active | Complete

What & Why

What it does: [One sentence description]

Problem it solves: [What pain point does this address?]

Who uses it: [IT admins, end-users, both?]

Key benefit: [Main value this provides]

Scope

Will Do

1. [Primary feature/capability]
2. [Secondary feature/capability]
3. [Additional feature/capability]

Won't Do

1. [What you're explicitly NOT building]
2. [Features you're intentionally excluding]

Success Looks Like

- [ ] [Measurable outcome 1]
- [ ] [Measurable outcome 2]
- [ ] [Measurable outcome 3]

How It Works

Main Use Case

User scenario: [Describe the typical usage in 2-3 sentences]

Steps:

1. [What happens first]
2. [What happens next]
3. [Final outcome]

Technical Requirements

Must have:

- [Required OS version, tool, dependency]
- [Required permission level]
- [Required integration]

Assumes:

- [Environmental assumption]
- [User capability assumption]

Design

Components

[Component/Function Name]

- Purpose: [What it does]
- Input: [What it needs]
- Output: [What it produces]

[Component/Function Name]

- Purpose: [What it does]
- Input: [What it needs]
- Output: [What it produces]

Data Flow

1. [Data/trigger enters from...]
2. [Gets processed by...]
3. [Results in...]

Configuration

- `settingName` - [default] - [what it controls]
- `settingName` - [default] - [what it controls]

User Interface

Type: [CLI, GUI, Self Service, silent, etc.]

What the user sees:

```
[Example output or interaction]
```

Error handling:

- [Error type] → [How user sees it]
- [Error type] → [How user sees it]

Implementation

Language/Tools: [Zsh, Swift, Python, etc.]

Key libraries/dependencies:

- [Tool/library] - [why needed]

File structure:

```
project-root/
├── [main-script]
├── [config/resources]
└── README.md
```

Core logic:

1. [High-level step 1]
2. [High-level step 2]
3. [High-level step 3]

Edge cases to handle:

- [Edge case] → [solution]
- [Edge case] → [solution]

Error Handling

Fatal errors (exit):

- [Condition] → [message/action]

Warnings (continue):

- [Condition] → [message/action]

Logging:

- Location: [path]
- What gets logged: [events to track]

Testing

Test in:

- ☐ Development mode: [how to test safely]
- ☐ Single test Mac
- ☐ Pilot group
- ☐ Production rollout

Validation checklist:

- ☐ [Key thing to verify]
- ☐ [Key thing to verify]
- ☐ [Key thing to verify]

Deployment

Prerequisites:

- OS: [minimum version]
- Dependencies: [what must be installed first]
- Permissions: [required privileges]

Installation:

1. [Step 1]
2. [Step 2]
3. [Step 3]

Verification:

- ☐ [How to confirm it worked]

Maintenance

Version scheme: [e.g., semantic versioning]

Update process: [How updates are distributed]

Deprecation: [How old features are retired]

Documentation

- [ ] README with quickstart
- [ ] Configuration reference
- [ ] Troubleshooting guide

Notes

Open Questions

- [Question to resolve before implementation]

Key Decisions

Decision	Why	Date
[Choice made]	[Rationale]	[Date]

Known Limitations

1. [Limitation] - [impact/workaround]

Future Ideas

1. [Enhancement idea]
2. [Enhancement idea]

Implementation Log

> Track progress and changes here

- [Date]: [What was done]
- [Date]: [Change made and why]

End of Plan