Care Notes

Most practical care starts **relationally**:

- a small group notices
- a serving team steps in
- a few people act informally

But sometimes:

- the need **outgrows the relationship**
- the load becomes **too heavy for a few people**
- coordination matters
- equity and clarity matter

That's when the **church body** gets involved.
So the system must:

- respect **local, relational care**
- without blocking **wider mobilization**

That's the tension you're holding — and it's the right one.

The key modeling decision

Care ≠ Support

A **Care Case** tracks *why we are paying attention*.
Supports (meals, money, rides, childcare) are:

- expressions of care
- scoped responses
- optional
- escalatable

They should **attach to a case**, not replace it.

The crucial concept: *Scope of Support*

Instead of asking:
"Is this a small group thing or a church thing?"
Ask:
"Who is this visible to and recruitable from?"
That gives you a clean axis.

Support Scope (recommended)

- **Private / Relational**
  - small group
  - serving team
  - friends/family
  - informal coordination
- **Church-Wide**
  - broader volunteer pool
  - announcements

    - signups
    - centralized oversight

This lets care *grow* without being re-created.

How this plays out in WorshipOS

Example: Meal Train

1. **Starts relationally**
    - Small group organizes meals
    - Logged as a support under a Care Case
    - Scope: *Private*
2. **Need increases**
    - More meals
    - Longer duration
    - Burnout risk
3. **Escalation**
    - Scope changes to *Church-Wide*
    - Visibility expands
    - More people can help
    - Leadership awareness increases

Same care case.
Same meal train.
Wider reach.

Example: Financial Help

1. **Initial support**
    - Deacon or group covers a bill
    - Scope: *Private*
2. **Ongoing or large need**
    - Benevolence fund involved
    - Approval workflow
    - Scope: *Church-Wide* (or Staff-Only)

Again: no new case required.

Why this is important theologically and pastorally

This design:

- honors **subsidiarity** (care handled as close as possible)
- avoids unnecessary institutionalization
- prevents burnout
- allows dignity and privacy
- scales *only when needed*

That aligns extremely well with Vineyard DNA and healthy pastoral practice.

Concrete data model suggestion (lightweight)

You don't need much:

Care Case

- Type (Celebration, Crisis, etc.)
- Status
- People involved
- Notes

Support (child record)

- Kind: meal | financial | transport | childcare | other
- Scope: private | church
- Owner / coordinator
- Active dates
- Notes
- Optional signup link / task list

That's it. No explosion of tables.

UX language that keeps this human

Avoid:

- "Broadcast to church"
- "Escalate case"

Prefer:

- **"Invite more people to help"**
- **"Expand support beyond this group"**
- **"Open to the wider church"**

That tone matters.

How this fits with "Tend"

"Tend" is the *reason we keep watching* as scope changes.
You're not "handing it off to the church."
You're **tending together**.
That's philosophically clean and pastorally grounded.

Strong recommendation (summary)

- Keep **meal train / financial help / transport / childcare** as *Supports*
- Attach them to **Care Cases**
- Give each support a **Scope**
- Allow scope to expand without recreating anything
- Default to *Private*, escalate only when needed

This keeps WorshipOS:

- relational-first
- non-clinical
- scalable
- faithful to how churches actually work

If you want, next we can:

- Draft the **exact UI flow** for “Expand support”
- Decide **who is allowed to change scope**
- Map this to **roles/permissions**
- Or sanity-check this against Planning Center / competitors to ensure differentiation

should we build a workflow?

You’re thinking about the **right future**, and your instinct about **not forcing it into MVP** is correct.
This is a classic WorshipOS moment:
the idea is *pastorally true*, but the implementation cost is high if you rush it.
Below is a **clear trajectory** that preserves the vision without overbuilding.

First: name the thing clearly

What you’re describing is not a workflow engine.
It’s a **Care Journey**:
the story of how a situation unfolds over time
That framing keeps it human and avoids BPMN hell.

The core tension

- Pastoral reality is **non-linear**
- Software loves **linear workflows**

So the system must:

- allow movement
- without enforcing steps
- and without implying moral progress (“you leveled up!”)

That rules out strict state machines in MVP.

MVP stance (strong recommendation)

MVP: Single Case, Many Moments

- One Care Case
- Case Type can change
- Status can change
- Timeline captures the story

No explicit “journey” object yet.
This already allows:

- Prayer → Crisis → Celebration
- Crisis → Ongoing → Celebration
- Check-in → Prayer → Resolved
- Celebration → Follow-up → Closed

Without formalizing paths.

What to ship now (MVP)

1. Case Type is editable

Not immutable.
That alone unlocks the future.
Example:

- Created as *Prayer Request*
- Later changed to *Crisis*
- Later changed to *Celebration*

Log the change in the timeline. Don't dramatize it.

2. Timeline / Activity Feed

This is the secret MVP workhorse.
Each entry might be:

- prayer logged
- visit happened
- meal train started
- support expanded
- case type changed
- case resolved

This *becomes* the journey without naming it yet.

3. Supports are additive

Supports come and go:

- meals start/stop
- visits scheduled
- funds approved
- help expands/contracts

No "phase" enforcement.

4. Soft close, reopenable

Cases can be closed, but never sealed forever.
That matches pastoral reality.

MVP deliberately *not* included

Do **not** ship these yet:

- predefined transitions
- required steps
- automation rules
- "next suggested action"
- success metrics

Those are all V2+ features.

Phase 2 (after real usage): "Care Trajectories (light)"

Once you have data, *then* you can add:

1. Case Origin + Case History

- Origin: prayer card, email, metric, hallway
- History: list of prior types this case has held

This enables insight without forcing behavior.

2. Suggested (not enforced) next steps

Language matters here:

- "Often after a crisis, teams choose to…"
- "You may want to consider…"
- "Common next steps include…"

This is pastoral assist, not workflow tyranny.

3. Journey tagging (optional)

Instead of workflows:

- `journey: prayer→crisis`
- `journey: crisis→celebration`
- `journey: ongoing care`

These are descriptive, not prescriptive.

Phase 3 (much later): "Care Patterns"

Only if churches ask for it.
This is where you *analyze* trajectories, not enforce them.
Examples:

- "Prayer requests that become crises often need X within 7 days"
- "Celebrations tend to close quickly — should we add follow-up reminders?"
- "Which supports burn out volunteers?"

This is insight, not control.

A clean roadmap view

MVP

- Editable case type
- Status
- Timeline
- Supports
- Manual changes only

Next

- Origin tracking
- History of type changes
- Gentle suggestions
- Optional tagging

Later

- Patterns
- Insights
- Capacity signals
- Preventative nudges

No hard workflows at any stage.

Why this aligns with Vineyard DNA

- The Spirit leads, not the software
- People are not tickets
- Stories matter more than states
- Presence over process

Your roadmap preserves that.

Bottom line (clear answer)

Yes — a trajectory from **Prayer → Crisis → Celebration** is *absolutely* worth supporting.
No — it should *not* be formalized as a workflow in MVP.
The MVP should record the story, not dictate it.

Care vs Tend

Right now, "care" exists as *notes*.
"Tend" should exist as *attention over time*.
So the Tend module is not "more notes."
It's the layer that **notices, carries, and remembers**.
I'll walk this in four parts:

1. What you already have (and why it's good)
2. What Tend *adds* conceptually
3. What Tend looks like at MVP
4. What it grows into later (without rework)

I'll stay concrete.

1. What you already have (baseline)

From the screenshots, your current system has:

- **Care Notes**
  - Category (General Check-in, Prayer, Visitation, Crisis, Celebration)
  - Freeform pastoral text
  - Restricted visibility
  - Attached to a person
- These notes are:
  - atomic
  - point-in-time
  - descriptive

This is good.
It mirrors how pastoral notes actually start.
But notes alone cannot:

- track responsibility
- track follow-up
- show unresolved care
- surface patterns
- survive staff transitions

That's where **Tend** comes in.

2. What "Tend" adds (the missing layer)

Think of Tend as answering three questions **notes cannot**:

1. *Is someone paying attention to this right now?*

2. *Who is responsible?*

3. *What still needs to happen (if anything)?*

So:

- **Care Note** = "Something was said or noticed"
- **Care Case (Tend)** = "This is being carried"

Tend doesn't replace notes.
It **wraps them**.

3. What a Tend module looks like (MVP, grounded in what you have)

A. Tend is a first-class area (not hidden in People)

You want a **Tend workspace** that shows:

Tend Dashboard (MVP)

- Open care cases
- Recently updated
- Waiting / follow-up needed
- Assigned to me
- Unassigned

No analytics yet. Just visibility.

B. A Care Case is born *from* a Care Note

Your current "Add Care Note" modal is the perfect intake.

MVP behavior change:

- When saving a Care Note:
  - Optionally: **"Carry this as ongoing care"**
  - Default off (important)
- If enabled:
  - Create a **Care Case**
  - Link the note as the origin

This preserves the low-friction intake you already have.

C. The Care Case becomes the Tend object

A **Care Case** minimally tracks:

- Person (or household)
- Case Type
  (Prayer, Crisis, Celebration, etc — editable)
- Status
  (New, Active, Waiting, Resolved, Closed)
- Assigned caregiver(s)
- Last activity date
- Linked notes
- Linked supports (later)

This is *not* a workflow engine.
It's a folder with awareness.

D. Timeline, not workflow

Inside a Care Case, you'd see:

- Origin note

- Additional care notes
- Status changes
- Case type changes
- Supports added / ended
- “Expanded to church-wide support” (later)

This **is the journey**, without enforcing one.

E. Supports live under the case (not globally)

As discussed earlier:

- Meal train
- Financial help
- Transportation
- Childcare
- Visits

These are **Supports**, attached to a Care Case.
Each support tracks:

- Type
- Scope (private / church)
- Coordinator
- Start / end
- Notes

You don’t need to ship supports in MVP —
but the case model should expect them.

4. How Tend changes the *meaning* of your categories

Right now your category dropdown is doing double duty:

- classification
- urgency
- intent

In Tend, that becomes cleaner.

Category becomes Case Type

- Why this exists

Status becomes Care State

- What’s happening now

That’s a big clarity win.

5. What you explicitly should *not* do yet

For MVP, resist:

- forced transitions
- required next steps
- automation
- reminders
- metrics
- success scoring

Those all come later *if* churches ask.
Your instinct is right: **record the story first**.

6. What Tend enables later (without redesign)

Because you modeled this cleanly, later you can add:

Phase 2

- Suggested next steps
- Gentle nudges ("no activity in 14 days")
- Visibility controls (staff vs care team)

Phase 3

- Patterns ("Prayer → Crisis" happens often)
- Load awareness ("too many active crises per pastor")
- Preventative check-ins

None of that requires changing the core model.

7. A simple litmus test (keep this)

Whenever you're unsure if something belongs in Tend, ask:
"Is this about *remembering and carrying* a person over time?"
If yes → Tend
If no → Notes, People, Groups, or Flow

Bottom line (clear answer)

Tend in WorshipOS is not a feature.
It's a posture made visible.

- Care Notes capture moments
- Care Cases carry responsibility
- Supports express care
- Tend keeps attention from slipping

What you've built already is the *front door*.
Tend is the **room behind it**.

Season Awareness/Protection

Tending well sometimes means *doing less*, not more.
Let's ground this.

For MVP:

- ✔️ A season/capacity note is enough
- ✔️ Optional date = good
- ❌ Hard reminders, alerts, or auto-tasks = too much (for now)

The key is **intentional restraint with a gentle hook for the future**.

What "Season Protection" really is

What you've designed here is *not* a care case.
It's closer to:

- a **pastoral boundary**
- a **capacity signal**
- a **protective context**

Examples:

- "Stepping back, family needs"
- "Grieving — not ready to serve"
- "Burnout risk"
- "New baby season"

This is not:

- something to resolve
- something to escalate
- something to taskify

So it should **not** behave like a Care Case.

Where Season Protection fits in the Tend model

Think in layers:

1. Care Notes

Momentary, descriptive

2. Care Cases

Ongoing situations that require attention

3. Season / Capacity Context ← *this*

Protective, contextual, advisory
Season Protection lives in layer 3.
That's why your placement near **Current Season / Capacity Note** feels right.

The right MVP behavior (recommendation)

✅ Add an optional date field

But with **very soft semantics**.
Not:

- "Follow up on this date"
- "Reminder"
- "Task due"

Instead:

- **"Check back after"**
- **"Revisit no earlier than"**
- **"Protect until"**

Your current copy is already heading this direction — keep that tone.

How the date should behave (MVP)

- Optional
- Stored
- Visible in the UI
- **No automatic alerts**
- **No forced actions**

At most, later:

- A subtle visual cue like "Season protection active until Feb 1"

That's it.

Why this is still "tending well"

Because tending is not just *care given* — it's *care withheld wisely*.
Season protection:

- prevents over-scheduling
- avoids volunteer guilt
- honors real life
- protects relationships
- communicates trust

In pastoral terms:
You are guarding the field, not harvesting it.
That's Tend.

What *not* to do yet (important)

Avoid in MVP:

- auto-generated follow-up tasks
- reminders to "check in"
- escalation flows
- status changes tied to dates

Those turn a humane boundary into a compliance system.
Churches will *ask* for reminders later if they want them.

A clean roadmap for Season Protection

MVP (now)

- Capacity / Season note
- Optional "check back after" date
- Passive visibility only

Next (if requested)

- Optional personal reminder ("remind *me*")
- Gentle dashboard hint ("some seasons have passed their date")

Much later

- Team-level signals (aggregate burnout risk)
- Planning safeguards ("don't auto-schedule during this season")

None of this breaks your current design.

One small language tweak (suggestion)

If you want to tighten the concept without adding features:
Instead of:
Capacity Note
Consider:

- **Current Season**
- **Season Note**
- **Season & Capacity**

You're already halfway there visually.

Final framing (keep this)

Care Cases ask: "Who needs attention?"
Season Protection asks: "What needs guarding?"
Both are Tend.
They just serve different pastoral instincts.

Resist the urge to automate that which should remain relational.
That restraint is part of WorshipOS's integrity.