📚 GISTer Documentation Strategy

Multi-Agent Collaboration Guide

🎯 Purpose

This document establishes the documentation strategy for GISTer, designed for **multiple AI agents and human developers** working simultaneously on the codebase. The goal is to prevent documentation drift, maintain consistency, and ensure all team members have access to accurate, up-to-date information.

📁 Documentation Structure

```
/gist_list/
├── docs/                          # Living documents (UPDATED, not replaced)
│   ├── CHANGELOG.md               # What changed (conventional commits format)
│   ├── FEATURES.md                # Current features & status
│   ├── ARCHITECTURE.md            # System design & tech stack
│   ├── API.md                     # API endpoints & contracts
│   ├── DATABASE.md                # Schema, migrations, relationships
│   └── ROADMAP.md                 # Planned features & priorities
├── sessions/                      # Session summaries (APPEND-ONLY)
│   ├── 2025-10-14_ui-ux-fixes.md
│   ├── 2025-10-15_telemetry.md
│   └── ...
├── AGENT_INSTRUCTIONS.md          # Agent-specific guidelines (READ THIS FIRST)
└── DOCUMENTATION_STRATEGY.md      # This file
```

📖 Document Types

1. Living Documents (`/docs/`)

- **Purpose**: Single source of truth, continuously updated
- **Update Frequency**: Every code change that affects the document
- **Ownership**: Shared (any agent/developer can update)
- **Format**: Structured sections with clear headings

2. Session Summaries (`/sessions/`)

- **Purpose**: Record of what was done, why, and by whom
- **Update Frequency**: Once per work session
- **Ownership**: Created by the agent/developer who did the work
- **Format**: Date-prefixed markdown files
- **Naming Convention**: `YYYY-MM-DD_short-description.md`

3. Code-Level Documentation

- **Purpose**: Inline explanations for complex logic
- **Location**: Within source files
- **Format**: JSDoc/TSDoc for functions, inline comments for business logic
- **Update Frequency**: When code changes

🔄 Documentation Workflow

Before Starting Work:

1. **Read AGENT_INSTRUCTIONS.md** (mandatory for new agents)
2. **Review latest CHANGELOG.md** to understand recent changes
3. **Check ROADMAP.md** to align work with priorities
4. **Scan recent session summaries** in `/sessions/` for context

During Work:

1. **Update living documents** as you make changes (don't wait until the end)
2. **Use conventional commits** (`feat:` , `fix:` , `docs:` , `refactor:`)
3. **Add inline comments** for complex business logic

Before Creating a Checkpoint:

1. ✅ **Update CHANGELOG.md** with your changes (following conventional format)
2. ✅ **Update FEATURES.md** if you added/modified features
3. ✅ **Update API.md** if you changed endpoints
4. ✅ **Update DATABASE.md** if you modified the schema
5. ✅ **Update ARCHITECTURE.md** if you changed system design
6. ✅ **Create session summary** in `/sessions/YYYY-MM-DD_description.md`
7. ✅ **Run tests** to verify nothing broke
8. ✅ **Create checkpoint** with descriptive message

📝 Documentation Standards

CHANGELOG.md Format

Use Conventional Commits (https://www.conventionalcommits.org/) format:

```
## [Date] - Agent Name

### Added
- feat: New feature description
- feat(api): New API endpoint for X

### Changed
- refactor: Improved performance of Y
- style: Updated UI component Z

### Fixed
- fix: Resolved bug in A
- fix(auth): Fixed session expiration issue

### Removed
- Deprecated feature B
```

FEATURES.md Format

```
## Feature Name
**Status**: ✅ Complete | 🚧 In Progress | 📋 Planned | ❌ Deprecated

**Description**: Brief description of the feature

**Key Components**:
- Component A (`path/to/file.ts`)
- Component B (`path/to/file.ts`)

**Dependencies**: List of external services/APIs

**Last Updated**: YYYY-MM-DD
```

API.md Format

```
## POST /api/endpoint
**Auth Required**: Yes/No
**Premium Only**: Yes/No

**Request Body**:
```json
{
  "field": "type"
}
```
```

Response:

```
{
  "result": "success"
}
```

Errors:
- `400` : Description
- `401` : Description
```
```

Session Summary Format

See `/sessions/2025-10-14_ui-ux-fixes.md` for reference.

Required sections:
1. **Session Overview** (checkpoint name, date, agent)
2. **Changes Made** (detailed list)
3. **Files Modified** (paths)
4. **Testing Notes** (build status, tests run)
5. **Next Steps** (if any)
6. **Known Issues** (if any)

🤖 Multi-Agent Coordination

Preventing Conflicts:

1. **Always pull latest changes** before starting work
2. **Check recent session summaries** to see what other agents did
3. **Update documentation incrementally** (don't batch updates)
4. **Use descriptive commit messages** to communicate intent

When Two Agents Modify the Same File:

1. **Document wins over code** - if docs say X but code does Y, fix the code or update the docs
2. **Newer session summary wins** - if conflict, check which agent worked most recently
3. **Ask for clarification** if truly unclear

Communication Channels:

- **Session summaries**: Async communication between agents
- **CHANGELOG.md**: What changed
- **Inline comments**: Why it changed
- **Commit messages**: How it changed

🚨 Critical Rules

❌ Never Do This:

- ❌ Skip updating living documents after code changes
- ❌ Create a checkpoint without updating CHANGELOG.md
- ❌ Overwrite someone else's session summary
- ❌ Remove documentation because "it's outdated" (update it instead)
- ❌ Push code without running tests

✅ Always Do This:

- ✅ Update docs in the **same commit** as the code change
- ✅ Use conventional commit messages
- ✅ Create session summaries before finishing work
- ✅ Read AGENT_INSTRUCTIONS.md if you're a new agent

- ✅ Test before checkpointing

🔍 Finding Information

"Where is feature X implemented?"

→ Check `FEATURES.md` for file paths

"What changed recently?"

→ Check `CHANGELOG.md` for recent entries

"Why was this decision made?"

→ Check session summaries in `/sessions/`

"What's planned next?"

→ Check `ROADMAP.md`

"How does system Y work?"

→ Check `ARCHITECTURE.md`

"What's the schema for model Z?"

→ Check `DATABASE.md`

"What does endpoint /api/X do?"

→ Check `API.md`

🎓 Onboarding New Agents

First 5 Minutes:

1. Read **AGENT_INSTRUCTIONS.md** (mandatory)
2. Read **this file** (DOCUMENTATION_STRATEGY.md)
3. Skim **CHANGELOG.md** (last 2-3 entries)
4. Review **FEATURES.md** (understand what exists)
5. Check **ROADMAP.md** (understand priorities)

Before First Commit:

1. Read relevant sections of `ARCHITECTURE.md`
2. Review `DATABASE.md` if touching data models
3. Check `API.md` if working with endpoints
4. Scan recent `/sessions/` summaries

📊 Documentation Health Metrics

Green (Healthy):

- ✅ All living documents updated in last 7 days
- ✅ Session summaries exist for all checkpoints
- ✅ No TODOs older than 30 days in docs
- ✅ All features in FEATURES.md have status

Yellow (Needs Attention):

- ⚠️ Living documents not updated in 7-14 days
- ⚠️ Missing session summaries for some checkpoints
- ⚠️ Some TODOs older than 30 days
- ⚠️ Some features missing status

Red (Critical):

- 🚨 Living documents not updated in 14+ days
- 🚨 No session summaries for multiple checkpoints
- 🚨 TODOs older than 60 days
- 🚨 Major features undocumented

🛠️ Maintenance

Weekly:

- Review documentation health metrics
- Archive old session summaries (>90 days) to `/sessions/archive/`
- Update ROADMAP.md with new priorities

Monthly:

- Audit all living documents for accuracy
- Consolidate repetitive information
- Update ARCHITECTURE.md if system evolved significantly

Quarterly:

- Review and update this DOCUMENTATION_STRATEGY.md
- Gather feedback from all agents/developers
- Improve processes based on learnings

📞 Questions or Issues?

If this documentation strategy isn't working:
1. Create a session summary explaining the issue
2. Propose changes to this file
3. Discuss with other agents via commit messages
4. Update this file once consensus is reached

Last Updated: 2025-10-14
Version: 1.0
Maintained By: All agents and developers working on GISTer