Future Features - Backend & Agent Changes Required

This document outlines proposed features that require backend modifications and/or agent-level changes.

UI + Minor Backend Changes

1. Model Selector

Purpose: Allow users to switch between Claude models (Opus/Sonnet/Haiku) during a conversation.

Backend Changes:

- Add model field to session or message schema
- Create API endpoint: PATCH /api/sessions/:name/model
- Pass model preference to ai-client.ts query function

UI Components:

- Dropdown selector in chat header
- Display current model badge

Telegram: Add /model <opus|sonnet|haiku> command

2. Token/Cost Display

Purpose: Show users their token usage and estimated API costs per session.

Backend Changes:

- Store token counts from Claude API response in message table
- Create API endpoint: GET /api/sessions/:name/usage
- Aggregate input/output tokens per session

UI Components:

- Usage badge in session list
- Detailed breakdown in session settings

3. Settings Page

Purpose: Allow users to customise their experience and manage preferences.

Backend Changes:

- Create user_settings table for preferences
- API endpoints for CRUD operations on settings
- Store: default model, theme preference, notification settings

UI Components:

- Settings page/modal with sections
- Theme toggle (already have via next-themes)
- Model preference selector

4. Pin Sessions

Purpose: Let users pin important sessions to the top of their list.

Backend Changes:

- Add pinned boolean field to sessions table
- Update session query to sort pinned first
- API endpoint: PATCH /api/sessions/:name with { pinned: true/false }

UI Components:

- Pin icon button on session items
- Visual differentiation for pinned sessions

UI + Backend + Agent Changes

5. Stop Generation

Priority: High

Purpose: Allow users to interrupt long-running agent responses.

Backend Changes:

- Implement abort controller in ai-client.ts
- WebSocket message type: { type: "abort", sessionName: string }
- Graceful cleanup of partial responses

Agent Changes:

- Handle abort signals in streaming responses
- Store partial messages appropriately

UI Components:

- Stop button that replaces send during generation
- Visual feedback on cancellation

Telegram: Add /stop command to interrupt generation

6. File Upload via Web

Priority: High

Purpose: Enable users to upload files through the web interface (similar to Telegram).

Backend Changes:

- File upload endpoint: POST /api/upload
- Store files in configurable location
- Generate file URLs for agent access
- Integrate with existing server/telegram/utils.ts file handling

Agent Changes:

- Include file paths in agent context
- Support common file types (images, PDFs, documents)

UI Components:

- Drag-and-drop zone in chat input
- File attachment button with preview
- Upload progress indicator

7. Session Templates

Purpose: Pre-configure sessions with system prompts and settings for specific use cases.

Backend Changes:

- Templates table with system prompts
- Endpoint: GET/POST /api/templates
- Apply template on session creation

Agent Changes:

- Load template system prompt
- Template-specific model preferences

UI Components:

- Template selector in new session dialog
- Template management page

Telegram: Add /template <name> command

8. Voice Input

Purpose: Speech-to-text for hands-free message input.

Backend Changes:

- Audio transcription endpoint using Whisper or similar
- Endpoint: POST /api/transcribe
- Stream audio handling

UI Components:

- Microphone button in input area
- Recording indicator
- Transcription preview

Note: Telegram already supports voice messages - backend transcription can be shared.

9. Agent Workflow Builder

Priority: Future

Purpose: Visual interface for creating multi-step agent workflows.

Backend Changes:

- Workflows table and step definitions
- Workflow execution engine
- Endpoint: CRUD for workflows

Agent Changes:

- Execute workflow steps sequentially
- Handle step outputs as inputs to next steps

UI Components:

- Visual workflow canvas
- Step configuration panels
- Workflow library

Implementation Priority

Phase 1 (Quick Wins):

1. Pin Sessions
2. Stop Generation
3. Model Selector

Phase 2 (Core Enhancements):

1. File Upload via Web
2. Token/Cost Display
3. Settings Page

Phase 3 (Advanced Features):

1. Session Templates
2. Voice Input
3. Agent Workflow Builder