Contents

User Rules System (Memory Rules) **1**
- Overview . . . 2
- Key Concepts . . . 2
  - Rule Types . . . 2
  - Rule Sources . . . 2
- Think Tank UI . . . 2
  - My Rules Tab . . . 2
  - Add from Presets Tab . . . 2
- Pre-seeded Preset Rules . . . 3
  - Privacy & Safety . . . 3
  - Sources & Citations . . . 3
  - Response Format . . . 3
  - Tone & Style . . . 3
  - Advanced . . . 3
- How Rules Are Applied . . . 4
  - Application Flow . . . 4
  - Prompt Injection Format . . . 4
  - Priority . . . 4
- Database Schema . . . 4
  - user_memory_rules . . . 4
  - preset_user_rules . . . 5
- API Endpoints . . . 5
  - User Rules . . . 5
  - Presets . . . 5
  - Internal (Service Layer) . . . 5
- Service Integration . . . 5
  - user-rules.service.ts . . . 5
  - preprompt-learning.service.ts Integration . . . 6
- Best Practices . . . 6
  - Writing Effective Rules . . . 6
  - Rule Limits . . . 6
  - When to Use Presets vs Custom . . . 6
- Memory Categories . . . 7
  - Category Hierarchy . . . 7
  - Category Codes . . . 7
  - Database Schema: memory_categories . . . 8
  - API Methods . . . 8
  - Future Expansion . . . 8
- Related Documentation . . . 8

User Rules System (Memory Rules)

Version: 4.18.3
Last Updated: 2024-12-28

Overview

The User Rules System allows Think Tank users to set persistent personal preferences that govern how the AI responds to them. Similar to Windsurf policies but for end users, these rules are automatically applied to every AI interaction.

Key Concepts

Rule Types

Type	Description	Example
restriction	Things the AI must NOT do	"Do not discuss religion"
preference	Things the AI SHOULD do	"Acknowledge uncertainty"
format	How responses should be structured	"Use bullet points"
source	Citation requirements	"Always cite sources"
tone	Communication style	"Be concise"
topic	Topic-specific rules	"Add health disclaimers"
privacy	Personal data handling	"Protect my privacy"
accessibility	Readability preferences	"Use simple language"

Rule Sources

- **user_created**: User typed the rule manually
- **preset_added**: Added from the preset library
- **ai_suggested**: AI suggested based on feedback patterns
- **imported**: Imported from another source

Think Tank UI

Location: Think Tank → My Rules
URL: `/thinktank/my-rules`

My Rules Tab

View and manage your personal rules:

- **Toggle**: Enable/disable individual rules
- **Edit**: Modify rule text
- **Delete**: Remove rules
- **Stats**: See how many times each rule was applied

Add from Presets Tab

Browse and add pre-seeded rules:

Popular Rules - Most commonly used rules by Think Tank users

Categories: - Privacy & Safety - Sources & Citations - Response Format - Tone & Style - Accessibility - Topic Preferences - Advanced

Pre-seeded Preset Rules

Privacy & Safety

Rule	Description
Protect my privacy	Prevents personal references or assumptions
No religious content	Filters out religious discussions
No political content	Keeps responses politically neutral

Sources & Citations

Rule	Description
Always cite sources	Includes verifiable sources for facts
Prefer academic sources	Prioritizes peer-reviewed content
Include source dates	Adds publication dates for recency

Response Format

Rule	Description
Be concise	Produces shorter, focused responses
Use lists for clarity	Organizes with bullets/numbers
Use headings	Adds section headers to long content
Comment code	Documents code examples

Tone & Style

Rule	Description
Professional tone	Business-appropriate style
Casual tone	Relaxed, conversational style
Simple explanations	Accessible without oversimplifying

Advanced

Rule	Description
Acknowledge uncertainty	States limitations honestly
Clarify before answering	Confirms question understanding
Show multiple viewpoints	Balanced coverage of debates

How Rules Are Applied

Application Flow

1. User sends message to Think Tank
2. AGI Brain generates plan with pre-prompt selection
3. `prepromptLearningService.selectPreprompt()` is called
4. Service fetches user rules via `userRulesService.getRulesForPrompt()`
5. Rules are formatted and appended to the system prompt
6. Rule application is logged for tracking

Prompt Injection Format

Rules are injected into the system prompt in categorized sections:

```
## User Preferences
The user has set the following rules for how you should respond:

**Restrictions (Must Follow):**
- Do not discuss religious topics...

**Source Requirements:**
- Always provide sources and citations...

**Format Preferences:**
- Keep responses concise...
```

Priority

- Restrictions are always enforced first (highest priority)
- Higher priority numbers (0-100) take precedence
- Conflicting rules resolved by priority

Database Schema

user_memory_rules

Column	Type	Description
`id`	UUID	Primary key
`tenant_id`	UUID	Multi-tenant isolation
`user_id`	UUID	Rule owner
`rule_text`	TEXT	Full rule content
`rule_summary`	VARCHAR	Short display text
`rule_type`	VARCHAR	restriction, preference, etc.
`priority`	INTEGER	0-100, higher = more important
`source`	VARCHAR	user_created, preset_added, etc.
`is_active`	BOOLEAN	Enable/disable
`apply_to_preprompts`	BOOLEAN	Apply to system prompts

Column	Type	Description
apply_to_synthesis	BOOLEAN	Apply during synthesis
times_applied	INTEGER	Usage counter

preset_user_rules

Column	Type	Description
id	UUID	Primary key
rule_text	TEXT	Full rule content
rule_summary	VARCHAR	Short display text
description	TEXT	User-facing explanation
rule_type	VARCHAR	Rule category
category	VARCHAR	UI grouping
icon	VARCHAR	Lucide icon name
is_popular	BOOLEAN	Show in popular section
min_tier	INTEGER	Subscription tier requirement

API Endpoints

User Rules

```
GET    /api/thinktank/user-rules           - Get user's rules
POST   /api/thinktank/user-rules           - Create new rule
PATCH  /api/thinktank/user-rules/:id       - Update rule
DELETE /api/thinktank/user-rules/:id       - Delete rule
PATCH  /api/thinktank/user-rules/:id/toggle - Enable/disable rule
```

Presets

```
GET  /api/thinktank/user-rules/presets     - Get preset categories
POST /api/thinktank/user-rules/add-preset  - Add preset to user rules
```

Internal (Service Layer)

```
getRulesForPrompt(tenantId, userId, domainId?, mode?)
  → Returns formatted rules for prompt injection
```

Service Integration

user-rules.service.ts

```
// Get rules formatted for prompt injection
const rules = await userRulesService.getRulesForPrompt(
  tenantId,
```

```
  userId,
  domainId,      // Optional: filter by domain
  mode           // Optional: filter by orchestration mode
);

// Returns:
{
  rules: UserMemoryRule[],
  formattedForPrompt: string,
  ruleCount: number,
  hasRestrictions: boolean,
  hasSourceRequirements: boolean
}
```

preprompt-learning.service.ts Integration

The preprompt service automatically fetches and applies user rules:

```
// In selectPreprompt()
const userRules = await userRulesService.getRulesForPrompt(
  request.tenantId,
  request.userId,
  request.detectedDomainId,
  request.orchestrationMode
);

// Append to rendered preprompt
rendered.full = rendered.full + userRules.formattedForPrompt;
```

Best Practices

Writing Effective Rules

1. **Be Specific**: "Always cite sources with URLs" vs "cite sources"
2. **Use Positive Framing**: "Use bullet points" vs "Don't write paragraphs"
3. **One Rule Per Preference**: Easier to toggle and track

Rule Limits

- Maximum 50 rules per user
- Maximum 1000 characters per rule text
- Inactive rules don't count toward limits

When to Use Presets vs Custom

- **Presets**: Common preferences with proven effectiveness
- **Custom**: Unique personal requirements

Memory Categories

Each memory/rule is categorized by **what it IS**, enabling better organization and future expansion.

Category Hierarchy

Top-Level	Sub-Categories	Description
Instruction	format, tone, source	Direct instructions for AI behavior
Preference	style, detail	Preferences that guide (not mandate) behavior
Context	personal, work, project	Background information about the user
Knowledge	fact, definition, procedure	Facts and information to remember
Constraint	topic, privacy, safety	Hard limits that must be followed
Goal	learning, productivity	User objectives and desired outcomes

Category Codes

```
instruction              # Direct instructions
  instruction.format     # How to structure responses
  instruction.tone       # Communication style
  instruction.source     # Citation requirements

preference               # Preferences
  preference.style       # Writing style preferences
  preference.detail      # Detail level preferences

context                  # User context
  context.personal       # Personal information
  context.work           # Professional context
  context.project        # Project-specific info

knowledge                # Knowledge to remember
  knowledge.fact         # Specific facts
  knowledge.definition   # Terms and meanings
  knowledge.procedure    # How to do things

constraint               # Hard limits
  constraint.topic       # Topics to avoid
  constraint.privacy     # Privacy rules
  constraint.safety      # Safety limitations
```

```
goal                    # User goals
  goal.learning         # Learning objectives
  goal.productivity     # Efficiency goals
```

Database Schema: memory_categories

Column	Type	Description
id	UUID	Primary key
code	VARCHAR	Unique category code (e.g., 'instruction.format')
name	VARCHAR	Display name
parent_id	UUID	Parent category for hierarchy
level	INTEGER	1=top-level, 2=sub-category
path	VARCHAR	Materialized path (e.g., 'instruction.format')
icon	VARCHAR	Lucide icon name
color	VARCHAR	Tailwind color class
is_system	BOOLEAN	System categories cannot be deleted
is_expandable	BOOLEAN	Can users add sub-categories?

API Methods

```
// Get category tree
const tree = await userRulesService.getMemoryCategories();
// Returns: { categories, topLevel, byCode }

// Get memories grouped by category
const grouped = await userRulesService.getMemoriesByCategory(
  tenantId,
  userId,
  categoryCode  // Optional: filter to specific category
);
```

Future Expansion

The category system is designed for expansion: - **Custom Categories**: Users can create sub-categories under expandable parents - **Category Inheritance**: Rules can inherit from parent categories - **Category-Specific Behavior**: Different application logic per category - **Cross-Category Rules**: Rules that span multiple categories

---

Related Documentation

- Pre-Prompt Learning System
- Think Tank Documentation
- AGI Brain Plan System