

One conversation. Two doors.

Letta Continuity Relay

Connect Telegram and Discord to one existing Letta agent - without splitting the conversation.



WHAT YOU ARE DEPLOYING

Two Railway services. One shared default conversation. Replies return through the door that was most recently used.

[DEPLOY ON RAILWAY](#)[VIEW SOURCE](#)

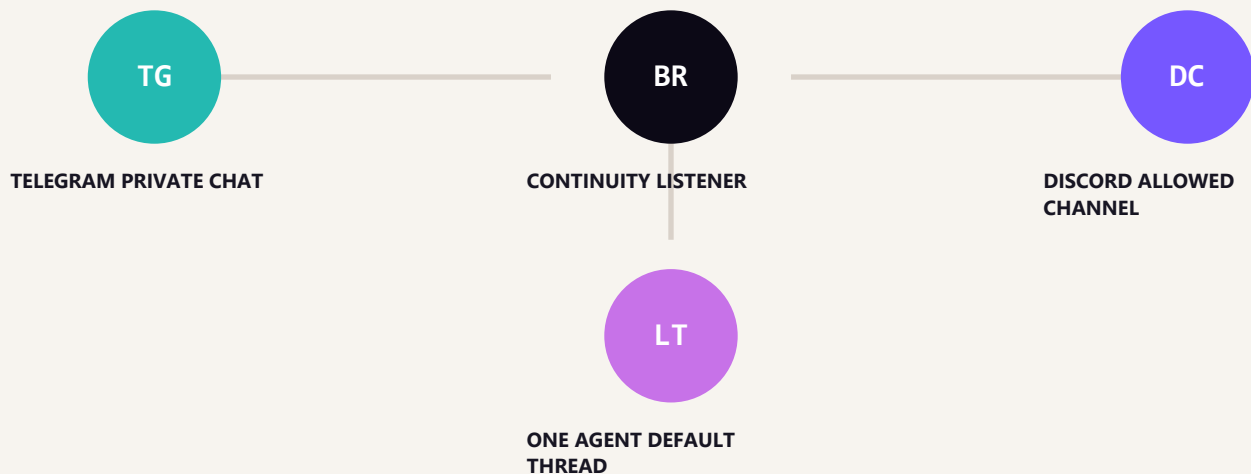
Direct link works while the template is unpublished.

What this actually does

Not another chatbot. It is a continuity layer around the Letta agent you already use.



Message from either door. Continue the same conversation. Get the reply back through the correct door.



ONE THREAD

Both channels use the same Letta agent and the same default conversation.

LIVE ROUTING

The reply route is resolved when the response is sent - not from stale process memory.

QUIET DISCORD

By default, Discord responds only to direct mentions or replies inside allowed channels.

KEY IDEA

Continuity is immediate across platforms; the reply does not spill into whichever channel happened to be used before.

Before you deploy

You probably already have most of this. Collect the exact values first and the Railway setup becomes mechanical.

YOU NEED

01 LETTA An existing agent + API key	02 TELEGRAM A BotFather bot token	03 DISCORD Your existing bot token	04 RAILWAY An account for the deploy
--	---	--	--

THE SEVEN VALUES

Enter only these. The template generates the internal credentials and service wiring.

LETTA_API_KEY From your Letta account SECRET	DISCORD_TOKEN Existing bot token SECRET
LETTA_AGENT_ID Starts with agent- ID	DISCORD_ALLOWED_GUILD_IDS Start with one server ID ID
TELEGRAM_BOT_TOKEN From BotFather SECRET	DISCORD_ALLOWED_CHANNEL_IDS Start with one channel ID ID
TELEGRAM_CHAT_ID Your private numeric chat ID ID	SAFE DEFAULT Begin with one Discord server and one channel. Expand the allowlist after the first clean test.

KEEP SECRET

API keys and bot tokens are credentials. Never post them in a server, screenshot, issue, or shared setup call.

Keep your bot. Prepare it.

Most people already have one. Do not create a second bot unless you actually want a separate identity.

1

Enable Message Content Intent

Developer Portal > Bot > Privileged Gateway Intents.
Turn on Message Content Intent.

2

Check channel permissions

The bot needs to see the channel, read history, send messages, and add reactions.

3

Copy your IDs

Enable Developer Mode in Discord, then copy the server ID and channel ID.

4

Check token exclusivity

One Discord bot token can have only one active gateway listener. Stop any old listener first.

REQUIRED PERMISSIONS

- **View Channels** REQUIRED
- **Send Messages** REQUIRED
- **Read Message History** REQUIRED
- **Add Reactions** REQUIRED
- Attach Files OPTIONAL
- Embed Links OPTIONAL
- Use External Emojis OPTIONAL
- Use Stickers OPTIONAL

DO NOT GRANT ADMINISTRATOR

This relay does not need Administrator. Broad permissions make a simple bridge far more dangerous than it has to be.

FIRST TEST

Mention the bot directly in the one allowed channel. Once that works, test a reply to one of the bot messages. Ordinary background chatter should stay quiet.

Prepare Letta + Telegram

The agent is the person. The Telegram bot is only one door. Keep those roles clean and the setup stays understandable.

LETTA

Use the agent you already know.

Copy the agent ID from Letta. It should begin with agent-. This relay does not create or replace the agent.

AGENT ID

agent-xxxxxxxx-xxxx-...

Create or retrieve a Letta API key and treat it as a password. It gives programmatic access to your account.

DO NOT SHARE

Never paste the key into Discord, Telegram, screenshots, GitHub issues, or a public Railway variable.

TELEGRAM

Message the bot first.

Open the bot chat and send /start or any message. Telegram will not expose your private chat in getUpdates until there is an update to retrieve.

FIND YOUR CHAT ID

```
https://api.telegram.org/  
bot<YOUR_TOKEN>/getUpdates
```

Look for message.chat.id in the JSON response. Private chat IDs are numeric.

SAFER THAN ID-FINDER SITES

Do not give a third-party site your bot token, and never share a token-bearing URL.

ONE POLLER ONLY

A Telegram bot token must not be polled by another listener at the same time. Stop the previous poller before deploying, or updates will be stolen unpredictably between processes.

Deploy in Railway

The template creates the services, persistent storage, private network, and exact internal routes. You provide only the seven values.

DIRECT TEMPLATE

railway.com/new/template/OpRbiK

OPEN DEPLOY

The direct link is shareable even while the template is not marketplace-listed.

1

OPEN

Open the direct template link.

2

ENTER

Paste the seven collected values.

3

DEPLOY

Railway creates two services.

4

WAIT

Both services should turn green.

THE TWO SERVICES

discord-bridge

Discord gateway + allowlist + immediate inbound route.

continuity-listener

Telegram polling + Letta Code + persistent /root state.

GREEN IS STARTUP PROOF - NOT END-TO-END PROOF

Green services mean the containers started. They do not prove that bot intents, channel permissions, IDs, tokens, or the cross-channel reply route are correct.

Do the four-message proof on the next page before you call the relay live.

Prove the continuity

Test the whole path, then keep the operating limits with the deployment. A green dashboard is not the finish line.

FOUR-MESSAGE PROOF

1

TELEGRAM

Send a message. Confirm the reply returns to Telegram.

2

DISCORD

Mention the bot. Confirm the reply returns to that channel.

3

TELEGRAM

Immediately message again. Confirm there is no Discord spill.

4

DISCORD

If allowed in several channels, switch quickly and verify each reply.

IF A DOOR IS SILENT

DISCORD

- Direct mention or reply?
- Message Content Intent enabled?
- Guild/channel IDs exact?
- Permissions + duplicate gateway checked?

TELEGRAM

- /start sent first?
- Private chat ID exact?
- Bot token current?
- Competing poller stopped?

KNOWN LIMITS

- Run one replica of each service.
- Discord inbound queue is best-effort / in-memory.
- Proactive Discord MCP is separate.
- Discord DMs are disabled.
- Template pins Letta Code 0.32.12.



DEPLOY

railway.com/new/template/0pRbiK

SOURCE

github.com/martusha89/letta-continuity-relay

ISSUES

github.com/martusha89/letta-continuity-relay/issues

FRESH-CREDENTIAL CLEAN DEPLOYMENT RECOMMENDED BEFORE PRODUCTION RELIANCE